Towards a resource based approximation theory of programs

Davide Barbarossa

UNIVERSITÉ PARIS XIII - SORBONNE PARIS NORD
UNIVERSITÀ DEGLI STUDI ROMA TRE
École Doctorale Sciences, Technologies, Santé Galilée

Vers une théorie de l'approximation des programmes basée sur la notion de ressources

Towards a resource based approximation theory of programs

THÈSE DE DOCTORAT en cotutelle / TESI DI DOTTORATO in cotutela
présentée par / presentata da

Davide BARBAROSSA
Laboratoire d'Informatique Paris Nord
Dipartimento di Matematica e Fisica

pour l'obtention du grade de / per ottenere il grado di
DOCTEUR EN INFORMATIQUE
DOTTORE IN MATEMATICA

soutenue le 10/12/2021 devant le jury d'examen composé de :

LAURENT Olivier	Directeur de recherche, CNRS	Rapporteur
MCCUSKER Guy	Full professor, University of Bath	Rapporteur
GUERRINI Stefano	Professeur, Université Sorbonne Paris Nord	Examinateur
MARTINI Simone	Professore ordinario, Università di Bologna	Examinateur
VAN RAAMSDONK Femke	Assistant Professor, Vrije Universiteit Amsterdam	Examinatrice
RONCHI DELLA ROCCA Simona	Professore emerita, Università di Torino	Examinatrice
MANZONETTO Giulio	Maître de Conférences, Université Sorbonne Paris Nord	Directeur de thèse
TORTORA DE FALCO Lorenzo	Professore associato, Università Roma Tre	Directeur de thèse

“Je voudrais que ma vie et ma mort servent à quelque chose”

Aurait-il-dit Prof. Samuel Paty, décapité près de Paris en 2020 par un fondamentaliste religieux, pour avoir enseigné la liberté de conscience à ses élèves.

Résumé

Il y a un peu moins de 20 ans, Ehrhard et Regnier, inspirés par la sémantique de la logique linéaire, ont découvert la possibilité d'effectuer un développement de Taylor d'un programme dans le cadre du λ-calcul. Même en oubliant les coefficients rationnels, les termes qui apparaissent dans le développement de Taylor d'un programme contiennent de l'information quantitative sur le programme lui-même. Si l'on considère ces termes sensibles aux ressources, on obtient une théorie de l'approximation qui est plus simple que l'originelle et qui est encore intéressante. Or, en mathématiques, une notion d'approximation est d'habitude vue comme un *outil* pour prouver des propriétés des objets que l'on approxime: ce sera également notre approche dans ce manuscrit.

Dans la première partie de la thèse, on définit cette théorie de l'approximation par ressources et on l'applique à des résultats, qui sont déjà connus mais fondamentaux, du λ-calcul. On montre notamment que, dans une certaine mesure, cela subsume la théorie classique de l'approximation qui est basée sur les arbres de Böhm et sur la continuité, et on le fait en fournissant des nouvelles preuves de ces résultats importants. Nos techniques de preuves utilisent toutes les propriétés cruciales de telles notions d'approximation, notamment la linéarité, la normalisation forte et la confluence. On obtient ainsi une nouvelle formulation d'une partie importante du λ-calcul, qui en plus apporte deux avantages: premièrement, même si nos preuves ne sont pas du tout triviales, elles sont tout de même plus "basiques" que celles traditionnelles, dans le sens où elles sont basées sur l'induction (à la base de la définition du développement de Taylor) au lieu de la coinduction (à la base de la définition d'arbre de Böhm); deuxièmement, on peut voir de cette façon que tous ces résultats, qui sont d'habitude prouvés avec des techniques *ad-hoc* et dépendants les uns des autres, sont en réalité indépendants et proviennent tous d'une seule technique: l'approximation par ressources (et la commutation d'Ehrhard et Regnier entre la normalisation et le développement).

Cette notion d'approximation, et en général la formule du développement de Taylor, a un troisième avantage: elle est facilement transportable à d'autres langages de programmation, et cette direction de recherche a été souvent empruntée dans les années récentes. Dans une deuxième partie de la thèse, on se tourne justement vers cette optique, et on adapte l'approximation par ressources au $\lambda\mu$-calcul, l'un des habituels langages de programmation fonctionnels (impurs) qui sont en correspondance de Curry-Howard avec la logique classique. Même si plusieurs définitions et propriétés sont simples à adapter dans ce nouveau cadre, les preuves de normalisation forte et de confluence du calcul avec ressources associé ne sont pas immédiates. Enfin, on s'attaque à la question des applications de ces outils; on montre comment reproduire, dans le $\lambda\mu$-calcul, les preuves données avant pour le λ-calcul pour établir la propriété de Stabilité et celle des Lignes Perpendiculaires. Ce sont deux nouveaux résultats sur la structure mathématique du $\lambda\mu$-calcul. On remarquera aussi que la preuve de la propriété de Stabilité peut être également adaptée dans le cadre du λ-calcul avec appel par valeur.

On conclut le manuscrit avec des réflexions qui ont pour but de lier des méthodes d'homotopie avec la sémantique dénotationnelle de la logique linéaire, mais aussi avec des réflexions méthodologiques et philosophiques sur le statut de notre discipline de recherche.

Mot-clés: Lambda-calcul, devéloppement de Taylor, lambda-mu-calcul, propriétés syntaxiques, arbres de Böhm, logique linéaire, topologie, appel par valeur, fondements des mathématiques.

Riassunto

Poco meno di 20 anni fa Ehrhard e Regnier, grazie alla semantica della logica lineare, hanno scoperto come sia possibile operare una espansione di Taylor di un programma scritto λ-calcolo. Anche tralasciando i coefficienti razionali, i termini che popolano tale espansione contengono dell'informazione quantitativa sul programma stesso. Considerando tali approssimanti per risorse, si ottiene una teoria dell'approssimazione più semplice di quella orginaria, ma ancora valida. In matematica, una nozione di approssimazione è tipicamente concepita come uno *strumento* per dimostrare delle proprietà degli ogetti approssimati: tale sarà anche il nostro approccio.

Nella prima parte della tesi, definiamo la teoria della approssimazione per risorse e la applichiamo a dei risultati, già bene noti ma fondamentali, del λ-calcolo. In particolare mostriamo che, in un certo senso, questa sussume la classica teoria dell'approssimazione, che è basata sugli alberi di Böhm e sulla continuità, e lo facciamo fornendo nuove dimostrazioni di questi risultati. Utilizzeremo tutte gli aspetti cruciali di tale nozione di approssimazione, ovvero la linearità, la normalizzazione forte e la confluenza. Così facendo, otteniamo una riformulazione di una parte importante del λ-calcolo, guadagnando due vantaggi: innanzitutto, anche se non triviali, le nostre prove sono più "basilari" delle vecchie, nel senso che si basano sull'induzione (che è alla base della espansione di Taylor) invece che sulla coinduzione (che è alla base degli alberi di Böhm); inoltre, ci si accorge che tutti questi risultati, che di solito vengono ottenuti con delle tecniche più *ad-hoc* e dipendenti l'una dall'altra, sono in realtà indipendenti tra loro, e derivano tutti un'unica tecnica: l'approssimazione per risorse (utilizzando anche la commutazione di Ehrhard e Regnier tra normalizzazione ed espansione).

Questa nozione di approssimazione, ed in generale la formula dello sviluppo di Taylor, ha anche un terzo vantaggio: è facilmente adattabile ad altri linguaggi di programmazione, come molte ricerche recenti hanno mostrato. Nella seconda parte della tesi ci muoviamo proprio in questa direzione, adattando la teoria della approssimazione con risorse al $\lambda\mu$-calcolo, uno dei più tipici linguaggi di programmazione funzionali (impuri) in corrispondenza di Curry-Howard con la logica classica. Molte definizioni e proprietà sono facilmente trasportabili in questo nuovo contesto, ma la prova della normalizzazione forte e della confluenza del calcolo con risorse associato non sono immediate. Infine, ci poniamo la questione della applicazione di tali strumenti; mostriamo come adattare, in $\lambda\mu$-calcolo, le dimostrazioni fornite per il λ-calcolo di due dei risultati precedentemente menzionati, la proprietà di stabilità e della proprietà delle linee perpendicolari. Si tratta di due nuovi risultati sulla struttura matematica del $\lambda\mu$-calcolo. Inoltre, osserviamo come la proprietà di stabilità possa essere facilmente adattata anche al caso del λ-calcolo "call-by-value".

Infine, concludiamo la tesi presentando delle idee in corso di studio, che puntano ad applicare metodi omotopici alla semantica denotazionale della logica lineare, ma anche presentando delle considerazioni a carattere metodologico e filosofico sullo statuto della nostra disciplina di ricerca.

Parole chiave: Lambda-calcolo, espansione di Taylor, lambda-mu-calolo, proprietà sintattiche, alberi di Böhm, logica lineare, topologia, call-by-value, fondamenti della matematica.

Abstract

Almost 20 years ago Ehrhard and Regnier, inspired by the semantics of linear logic, discovered the possibility of performing the Taylor expansion of a program in the realm of λ-calculus. Even forgetting the rational coefficients, the terms populating the Taylor expansion of a program contain quantitative information about the program itself. Simply collecting such resource sensitive approximants allows to define an approximation theory simpler than the original one, but still meaningful. Now, in mathematics, a notion of approximation is usually conceived as a *tool* for inferring properties of the approximated objects: it is precisely the approach we adopt in this manuscript.

In the first part of the thesis, we define this resource approximation theory and apply it to the investigation of some already known, but fundamental, properties of λ-calculus. More specifically, we show that in some sense it subsumes the classic theory of approximation, based on Böhm trees and continuity, by providing new proofs of these important results. Our proof-techniques employ all the crucial aspects of this notion of approximation, i.e., linearity, strong normalisation and confluence. We obtain in this way a new formulation of an important part of λ-calculus, presenting two advantages: firstly, even though they are not trivial, our new proofs are more "basic" than the old ones, in the sense that they are based on induction (at the base of Taylor expansion) rather than coinduction (at the base of Böhm trees); secondly, we are able to see that in fact all these results, usually based on *ad-hoc*, interrelated techniques are actually independent from each another, and only originate from a unique technique: resource approximation (exploiting Ehrhard and Regnier's commutation between normalisation and expansion).

This notion of approximation and, more generally, the Taylor expansion formula, has a third advantage: it scales to other programming languages, and this direction has been often explored in the recent years. In the second part of the thesis, we add another brick to these works by adapting the resource approximation to the $\lambda\mu$-calculus, one of the most typical (impure) functional programming language in Curry-Howard correspondence with classical logic. Although most definitions and properties are straightforward to obtain in the new setting, the proof of strong normalisation and confluence of the corresponding resource calculus are not immediate. Finally, we consider the question of the application of the developed tools; we show how one can easily reproduce, in the $\lambda\mu$-calculus, the proofs given for λ-calculus of two results, namely the Stability Property and the Perpendicular Lines Property. Those constitute two original results on the mathematical foundations of $\lambda\mu$-calculus. We also remark that the Stability Property can be easily adapted to the case of call-by-value λ-calculus.

We conclude the thesis by presenting some work in progress, aiming at relating homotopy with denotational semantics of linear logic, as well as some methodological and philosophical considerations about our discipline of research.

Keywords: Lambda-calculus, Taylor expansion, lambda-mu-calculus, syntactical properties, Böhm trees, linear logic, topology, call-by-value, foundations of mathematics.

Remerciements

Je tiens d'abord à remercier mes encadrants Giulio et Lorenzo. En particulier:

merci à Giulio, pour m'avoir enseigné, entre autre, le λ-calcul. Si ce n'était pas pour son aide de toute sorte, mathématique, administrative et tant d'autres, je n'aurais jamais pu arriver au bout de cette thèse. Je lui suis particulièrement reconnaissant de sa disponibilité, qui a toujours été absolue, et nos discussions ont toujours été éclairantes;

merci à Lorenzo, pour m'avoir enseigné, entre autre, la logique. Lorenzo suit mes études depuis mon mémoire de Licence, que ce soit le master italien ou celui de Marseille. Si ce n'était pas pour lui, je ne serais même pas en France maintenant. Nos discussions, depuis désormais cinq années, des plus mathématiques aux plus philosophiques, ont toujours eu une importance fondamentale pour moi.

Merci à Olivier Laurent et à Guy McCusker pour avoir accepté d'être les rapporteurs de ce manuscrit. Merci ègalement à Stefano Guerrini, Simone Martini, Femke Van Raamsdonk et Simona Ronchi della Rocca pour être dans mon jury.

Merci au LIPN, et en particulier aux membres de l'axe "Lo" de l'équipe LoVe. Malheureusement la pandémie de Covid19, qui a marqué toute la deuxième partie de mon expérience de thèse, nous a volé tellement de choses, parmi lesquelles la possibilité de bénéficier pleinement de la présence d'une si agréable équipe de travail. C'est dommage.

Merci aux membres de l'IRN Linear Logic et à tous les chercheurs que j'ai rencontré pendant ces années. Ils ont toujours bien accueilli les jeunes étudiants, même quand en 2016 j'ai participé à ma première rencontre de recherche, en ne connaissant rien du tout de l'informatique théorique. Je remercie en particulier Thomas Ehrhard et Lionel Vaux pour les intéressantes discussions, et Laurent Regnier, qui avais encadré mon stage M2 et avec qui je me suis trouvé vachement bien.

Merci à ceux qui ont été disponibles pour m'apporter de l'aide de tout type lors de mes enseignements dans un établissement que je ne connaissais pas, avec des matières qui n'étaient pas forcement celles de ma formation, et pendant une pandémie. Merci notamment à Micaela Mayero et à Giulio. Ces enseignements m'ont apporté beaucoup et je suis content et fier de les avoir menés.

Un grand merci à tous mes collègues les doctorants du LIPN et aux stagiaires, avec qui on a pas mal rigolé, ainsi qu'à toutes les personnes avec qui j'ai partagé de jolis moment parisiens. Notamment (en ordre casuelle): Zeinab, Guillaume, Elena, Teresa, Matteo, Paolo, Daniel, Carole, Gwen, Clarisse, Nour, J-J, Alex, Axel, Boris, Ugo, Florent, Dina, Nick, Sarah, Khautar, Tiffaine, Aloÿs, Emna, Yousra, Massinissa, Rimez, Rabeb, Ikram, Hiba, Linda, Antoine, Giulia, Maria-Luisa, Emiliano, Enrico et les autres que j'aurai oublié de mentionner. Encore une fois, la Covid nous a malheureusement empêché de nous voir autant que l'on aurait souhaité. C'est dommage.

Un dernier merci tout particulier va: à Mehdi, pour son ouverture mentale, pour nos infinies discussions sur tous les sujets jusqu'à pas d'heures, et pour tous les plus beaux rires que l'on a eu ensemble; à Jawher, pour tous les moments que l'on a passé ensemble, nos infinies balades dans Paris et son constant aide informatique qu'il a toujours su m'apporter; à Tito, pour les discussions sur les sujets les plus variés, qui ont contribué à me faire comprendre plusieurs choses (pas seulement dans le milieu scientifique); à Federico, dont notre amitié – toute sur le sol français – et nos longues conversations quotidiennes ont joué un rôle très important pour moi.

Je n'aurais pas à préciser mes remerciements à Pierpaolo, Riccardo et Scandi. De même, à papà, maman, Luca&Valentina et mes petits neveux Emanuele, Gabriele et Beatrice.

Contents

Chapter 1

Introduction

1.1 Brief outline of the thesis

This manuscript is organized in 5 chapters, which can be in turn organized in the three main parts which follow below. Each chapter, as well as most of the sections, begins with a "plan" where we detail its content and our contributions; in addition to that, Chapter 3 and Chapter 4 end with some conclusive comments and the main possible future works. We refer the reader to those explanations for the details. In particular:

- Part 1: Chapter 1 and Chapter 2. These first two chapters are introductory: in the present Chapter 1 we introduce the main topics which constitute the general framework of this thesis, and in Chapter 2 we quickly give some basic details about λ-calculus and Böhm trees. The content of this part is explicitly high level and less formal.

- Part 2: Chapter 3 and Chapter 4. It constitutes the main part of the thesis. In Chapter 3, whose detailed description can be found at Section 3.1, we first consider the resource approximation theory for λ-calculus and then we study its relation with the approximation theory based on Böhm trees. Details about this latter section are given at the beginning of Section 3.5. We refer to Section 3.6 for the conclusive comments. We also quickly reproduce some results in the call-by-value framework; we refer to the beginning of Section 3.4 for the details.

  In Chapter 4, we introduce the resource approximation theory for $\lambda\mu$-calculus and show how one can reproduce some important results of the previous chapter in this framework. A detailed plan is given at Section 4.1. We also refer to Section 4.5 for the conclusive comments.

- Part 3: Chapter 5. This chapter is not about approximating programs; instead, it collects a scratch of a possible research direction (Section 5.2.2), as well as our "meditations" about some methodological and philosophical issues related to our discipline of research (Section 5.2.1 and Section 5.3). Its detailed plan is given at Section 5.1.

1.2 The λ-calculus

The λ-calculus was introduced by Alonzo Church around 1930 as a "formal calculus" for manipulating functions in their generality, with the idea of founding mathematics taking the notion of function as primitive, instead of the more common approach considering sets. The Kleene-Rosser paradox [KR35], and Curry's paradox involving its famous "`Y` fixed point combinator" [Cur41], show that, from a logical point of view, the theory does not allow to provide a sound

foundations of mathematics[1]. But, for a great part thanks to Kleene, it was understood that this calculus could be used to model the intuitive notion of computable function on integers. This same notion was being formalized in the same years by Turing with his Turing-machines, and by Herbrand-Gödel, Kleene and others with the "partial recursive functions". Gödel was initially convinced that any intuitively computable function is Turing-computable, but was not so convinced about the same for partial recursive or lambda-definable ones (see [Dav82], pag. 12). Turing-machines are so convincingly close to the intuitive notion of computation, that they become the unity of measure for the notion of computable functions, and one calls Turing-complete any system which is as expressive as Turing-machines. But if Turing-machines are so intuitive, they are also, mathematically speaking, quite inelegant objects: at the end of the day, a Turing-machine essentially reduces a program to a mere sequence of instructions on some hardware[2]. At the same time, it is exactly thanks to this point of view that in this formalism the notion of *step of computation* is clear and *elementary*. That is why this approach is extremely fertile if one wants to take into account the temporal and spatial resources that a computation needs.

But if one cares about developing a real mathematical theory of algorithms (and not of what is computable), since this means to find in some way the "essence" of what an algorithm is, one should abstract as much as possible from implementation details, and turn to other formalisms. One could think that a valid alternative for achieving this goal could be the theory of *partial recursive functions*: in fact, they are Turing-complete. The big advantage of that approach is to completely depart from implementation details, and to have a nice mathematical definition[3]. But expressing an algorithm as such a particular kind of functions from $\mathbb{N}$ to $\mathbb{N}$ means - as in all mathematics - reducing it to sets. This is not quite satisfying, as Scott in [Sco93] would explain better than us:

> *For the purposes of understanding computation, however, set-theoretical formalism is not too helpful in any direct way. In the first place, too much of set theory concerns the transfinite, and ordinary computation has rather to do with finite processes. In the second place the axioms of set theory are meant to capture something essential of the idea of an arbitrary subset, while computation theory is more interested in the notion of an algorithmically defined subset (or function). Of course, one can define in set theory such notions as that of a general recursive function, but such definitions do not emphasize enough what is special about algorithms. Nor is it generally clear when a defined function is recursive. So what we want is a "restricted" system that is specially designed for algorithms.*

What is so special about algorithms is... the fact that they compute! By reducing an algorithm to a static set of pairs, one makes the crucial notion of step of computation disappear. So recursive function have abstracted too many details. Hence, one wants to depart from having the idea of set as the basic notion, and depart from being static. Quoting again [Sco93]:

> *[...] The reason is that for algorithms it is more natural to consider functions rather than sets. We can reduce the notion of function to that of set, but it is not convenient to do so. A much better plan is to treat sets (and relations) as special functions (truth-valued) as Church does.*

Since in a computation the time is discrete - it is just counting the successive steps of computation - one ends up with a rewriting system of "functions", which is the lambda-calculus. This is a great advantage with respect to the aforementioned two models of computations: it is

[1] More than one century after the birth of mathematical logic this is of course not a surprise...

[2] We can say that it modelize computation via the *imperative paradigm*.

[3] Partial recursive functions model computation via the *functional paradigm*.

a "dynamical system", as Turing-machines, and it abstracts from implementation details[4], as recursive functions, while still having a "clear" notion of step of computation, the *β-reduction*. The fact that the integer functions that are representable inside λ-calculus are exactly the same as the ones computable with a Turing-machine provides (together with the equivalence with partial recursive functions and other "models of computation") the major support for the so-called *Church-Turing thesis*: the intuitive notion of computable function from integers to integers is exactly given by any of those equivalent models of computation. The conceptual price to pay when considering λ-calculus is that we are no longer dealing with algorithms for themselves but only with one chosen *representation*[5] of them (they will be words respecting some syntactic constraints); that is, we are dealing with (functional) *programs*.

It is only many decades after the introduction of λ-calculus, around the 60's and the 70's, that this formalism got much more attention in the academic community, from mathematicians but in particular from the just born community of (theoretical) computer scientists. Today, we would say that:

- for a mathematician, λ-calculus is the mathematical theory of functions as algorithms, which is rich and very different from mainstream mathematics;

- for a computer scientist, λ-calculus provides a mathematical theory of functional programming; something that does not exist – at least not in that mathematical strength – for the other programming paradigms;

- for a logician, λ-calculus is the core of the best suited programming language for studying the several manifestations of the Curry-Howard correspondence.

Not only the mathematics of λ-calculus is non-trivial and interesting but, if one considers it together with the whole area of research in "computer science logic", it has crucial applications in real life: the birth itself of the mathematical investigation of computation, as well as of the practical idea of computers, was mainly due to mathematical logic in the first half of the XX^{th} century, and the abstract study of programming languages (for safety reasons but not only) is nowadays done, for a great part, via λ-calculus and/or proof theory; consequently, this places those branches of mathematics among the ones with the most outstanding applications, contrary to what many usually think.

Let us clarify the point of "functions as algorithms", which is clear to computer scientists but less explicit in mathematics, and is at the core of λ-calculus. Consider the following examples:

1. define functions $F_1, F_2 : \mathbb{N}^4 \to \mathbb{N}$ setting, $\forall x, y, z, n \in \mathbb{N}$, $F_1(x, y, z, n) := 0$ and

$$F_2(x, y, z, n) := \begin{cases} 0 & \textit{if } xyzn = 0 \textit{ or } n = 1 \textit{ or } n = 2 \textit{ or } x^n + y^n \neq z^n \\ 1 & \textit{otherwise} \end{cases}$$

   Thanks to A. Wiles we know that $F_1 = F_2$. However, it is clear that the way we defined them is radically different, independently from the truth of Fermat's last theorem.

2. let Bool := $\{0, 1\}$ and let `leftOr` : $\text{Bool}^2 \to$ Bool, with `leftOr`(x, y) defined via the following algorithm: *if* $x = 1$ *then* 1 *else* y, and let `rightOr` : $\text{Bool}^2 \to$ Bool, with `rightOr`(x, y) defined via the following algorithm: *if* $y = 1$ *then* 1 *else* x. As functions, we have: `leftOr` = `rightOr`. But, seen as algorithms, then they become different; not only syntactically, but more interestingly they satisfy different properties: indeed, we can take

[4] Again, it models computation via the functional paradigm.

[5] The quest towards a direct definition of an abstract notion of algorithm which is mathematically satisfying is exciting but not clear.

as x an argument which loops – call it $\perp$ – and obtain that $\texttt{leftOr}(\perp, 1)$ is undefined while $\texttt{rightOr}(\perp, 1) = 1$. Since the years '30 we know that in order to really deal with the notion of algorithms, one has to consider *partial* functions. Considered as partial functions from $(\text{Bool} \cup \{\perp\})^2$ to Bool, `leftOr` and `rightOr` are different.

These examples show that there is an aspect, to which one usually refers by the word *intentionality*, which is consciously neglected in the mathematical notion of functions; this aspect is *the way* a function is given, i.e. the "expression" that defines it. Often, in order to reach the needed level of abstraction one deals with in mathematics, one has to employ a more abstract way of defining functions and free themselves from the *actual* way a function is given – typically by some "law"; it only counts the relation "input/output". Whence, the notion of function as a relation. This way of understanding functions is usually called *extensional*, since from this point of view two functions are equal iff the relations defining them have exactly the same points, i.e. they are equal as *sets* - and in set theory this property is called *axiom of extensionality*.
The point of λ-calculus (and of theoretical computer science, really), is *not* to refuse the way mathematics treats functions: the point is, on the contrary, to actually render the abstract notion of "intensional function" the object of a *mathematical study*.

1.3 Mathematical logic: intuitionistic vs classic

The word Logic comes from the Greek *logos*, which mainly meant "reason" and "language". In fact, Logic was born in Greece by Aristotle, and developed in two main topics: the (philosophical) study of (human) reason and (human) language. After Aristotle, Scolastics used it in relation to theology, then it went transcendental with Kant and dialectical with Hegel. But the real breakthrough came at the end of the XIX^{th}, when it became (probably forever?) mathematical. The adjective mathematical is added at two different layers: first, we do not study all human reason and language anymore, but only the mathematical ones; second, we realize this study using mathematical tools and from a mathematical point of view. In other words, one is interested in the mathematical study of mathematics itself, reason for which the discipline is sometimes called "metamathematics". Quoting Bourbaki's introduction to the volume "Théorie des ensembles" of his "Élements de mathématiques": *Depuis les Grecs qui dit mathématique dit démonstration*. It is clear then that the mathematical reasoning is expressed by the proofs. For what concerns the mathematical language, we mean more or less "the rest", namely axioms and in general mathematical statements. Mathematical logic can be thus divided into two main areas, which are called respectively "proof theory" and "model theory". For historical reasons[6] the second area is the one which developed faster. Starting from the years '60 proof theory knew an incredible development, thanks to computer science.

Historically speaking, mathematical logic was born in the second half of the XIX^{th} mainly due to the following situation: mathematics was becoming more and more abstract (also in the Hilbertian sense of "infinitary"), and it was natural to think about an essential unity of all the discipline, together with some justification of its solid foundation, in particular the absence of contradictions in it. However, several antinomies - that is, real contradictions - appeared. The following are the most famous:

1. Berry's paradox[7]: consider the number b defined as "the smallest integer not definable in under than 12 English words". Does b exist? Either way, we have a contradiction.

[6] We would say that one of the main reasons is the fact that in order to really understand what proof theory is about, one needs at the very least a clear notion of "computable", which was not there until 1936. Furthermore, a deeper comprehension of the discipline is possible only after the notion of programming was introduced, so at least after the 50's.

[7] Often named as/confused with Richard's paradox. It was written by Russel but credited to some Berry.

2. Russel paradox[8]: consider the set R of all sets that do not belong to themselves. Does R belong to R? Either way, we have a contradiction.

3. Burali-Forti paradox: consider the set O of all ordinals. It is easily seen that O is an ordinal. Therefore so is $O+1$, i.e. $O+1 \leq O$. So $O < O+1 \leq O$, contradiction.

4. Cantor's paradox: consider the set U of all sets. So $\mathscr{P}(U) \subseteq U$ and thus $\mathrm{Card}(\mathscr{P}(U)) \leq \mathrm{Card}(U)$. This contradicts Cantor's cardinality theorem.

We know that, in any case, no contradiction will ever appear in "practical mathematics", and we can sleep more than reassured; but there is something we could do: investigate this phenomenon, maybe coming out with some arguments telling us which parts of mathematics are certainly free from contradictions, which would be very nice and reassuring. To attack this questions, one should have a "mathematical copy" of a given part of mathematics to mathematically study, eventually aiming to construct the "bigger" system possible. This is precisely how mathematical logic was born, under the influence of three of the major father founders of the discipline: Frege, Russel and Hilbert.

Even if this manuscript, strictly speaking, is not about mathematical logic, its topics are tightly related with proof theory, so let us mention some basic ideas of the discipline. One can begin with the simple remark that every mathematical statement has one of the following shapes: either it is an "atomic" statement directly expressing a property of some entity, or is constructed from other statements by means of a *conjunction*, or *disjunction*, or *implication*, or *quantification* (*existential* or *universal*). In order to distinguish a formal statement from a mathematical one, one first fixes a series of symbols associated with the properties one wants to talk about, and then composes formal statements using the symbols $\wedge, \vee, \to, \exists, \forall$ instead of the above italics words. Formal statements thus built are known as *formulas*. The next step is to decide how to represent proofs as *interesting* mathematical objects. This is far from being trivial[9]. The most common way of representing formal proofs is through *sequent calculus* or *natural deduction*, both introduced by Gentzen in the same years. The idea is to represent a proof as a tree, called a "derivation", whose nodes are pairs (Γ, Δ) – written "$\Gamma \vdash \Delta$" and called *sequent* or *judgment* – of finite multiset of formulas. Derivations are then constructed inductively by means of rules merging a finite number of inductively defined derivations within a new one, and they are defined in order to be *logically correct*: this means that, if we read a sequent as the implication $\bigwedge \Gamma \to \bigvee \Delta$[10], then assuming all the hypotheses (the leaves of the derivation tree) the conclusion (the root of) holds as well. There are two main classes of formal systems one can construct in this way: intuitionistic and classical ones. The difference is reflected in two layers: in an intuitionistic framework, the second component of sequents is restricted to having *at most* one formula; moreover, one does not include the rule representing the reasoning by contradiction. The two systems so defined are very different and the intuitionistic one is strictly included in the classical one, meaning that the former is a subsystem of the latter which cannot simulate it. Let us give the main rules for a very basic system, known as "propositional intuitionistic logic" (we only give here the "implicative" framework):

$$\frac{\vdash A \to B \qquad \vdash A}{\vdash B}\ \textit{Destructor of an implication} \qquad\qquad \frac{A \vdash B}{\vdash A \to B}\ \textit{Constructor of an implication}$$

[8] Apparently Zermelo had found in the same year (1902) the same paradox, but did not publish it.

[9] The first proposal was made by Hilbert but, mathematically speaking, his systems are not rich. Modern proposals, much more interesting for a number of reasons, can be found especially in Girard's proof-nets [Gir96], as well as in "deep inference's systems" [TS19] and "combinatorial proofs" [Hug06].

[10] This is the reason why one usually explains that the "intended" meaning of a sequent is as such an implication. However, as Girard points out, a more interesting look on sequents can be found reading them though a distinction "explicit/implicit", on which we will come back in Section 5.3.2.

to which one adds an "axiom rule" saying that the one leaf tree $\Gamma, A \vdash A$ is always a derivation. In order to pass from this system to "minimal propositional classical logic", one can add the following rule:

$$\frac{}{\vdash ((A \to B) \to A) \to A}\, \textit{Peirce law}$$

Equivalently, one can add a rule representing the reasoning by contradiction or a rule representing the tertium non datur. The equivalence means that adding any of such rules to minimal propositional intuitionistic logic, allows to prove the other two rules.

We have to mention another crucial point in proof theory: cut-elimination. Suppose to have a derivation π_1 of $\vdash A$, and consider the following derivation of $\vdash A$:

$$\pi_2 := \dfrac{\dfrac{\dfrac{}{A \vdash A}}{\vdash A \to A} \qquad \pi_1}{\vdash A}$$

It is clear that π_2 is redundant, since it already contains the "simpler" π_1. It is important to notice that, in order for π_2 to be redundant, it has to introduce the implication $A \to A$ and, immediately after, consume it by applying it to π_1. Such a configuration is the prototypical case of a "cut"[11] in a derivation. Motivated by coherence-proofs, in the '30 Gentzen was led to study the structure of derivations. He made the fundamental discovery of a series of basic manipulations, each of them transforming a derivation of a sequent in another derivation of the same sequent, which can be consecutively applied to any starting derivation admitting some cuts in order to eventually end on a derivation (thus, a derivation of the same sequent as the first one) which do not contain cuts anymore. This result – called *cut-elimination* – endows the set of derivations with a dynamics, and sets the beginning of modern proof theory. In the case of intuitionistic logic, each rule of cut-elimination defines a *function* which transforms a given derivation into exactly another one. That is, intuitionistic derivations obey to a *deterministic* cut-elimination, which is in addition strongly normalising, confluent and admits non-trivial denotational semantics. The case of classical logic is more subtle: in sequent calculus[12], there are derivations such as:

$$\dfrac{\dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A \vee A \vdash A, A}}{A \vee A \vdash A} \qquad \dfrac{\dfrac{A \vdash A \qquad A \vdash A}{A, A \vdash A \wedge A}}{A \vdash A \wedge A}}{A \vee A \vdash A \wedge A}$$

which give rise to an infinite cut-elimination reduction sequence, thus making cut-elimination not strongly normalising. The same derivation can also produce different normal forms. This aspect is better seen in the following derivation:

$$\pi \;=\; \dfrac{\dfrac{\pi_1}{\Gamma \vdash \Delta, A} \qquad \dfrac{\pi_2}{A, \Gamma \vdash \Delta}}{\Gamma \vdash \Delta}$$

where π_1 and π_2 are both derivations of the sequent $\Gamma \vdash \Delta$. Such a proof π rewrites, by cut-elimination, both in π_1 and in π_2, which can be normal and distinct, thus making cut-elimination not confluent. This example is known as Lafont's critical pair [GLT89]. Furthermore, this means that the equivalence induced by cut-elimination identifies all proofs, and thus any denotational semantics must be trivial. For a long time it was thought that this was it, about cut-elimination for classical logic. But, mainly thanks to Griffin's [Gri90], Girard's LC [Gir91], Parigot's FD [Par91] and CD [Par92], and Danos, Joinet, Shellinx's analysis in [DJS97], it was understood where the "problems" of classical cut-elimination are, and in which sense one can "constructivize" classical logic by means of better syntaxes.

[11] Most often called a *detour* in this precise setting.

[12] The same phenomenon can be reproduced in natural deduction.

1.4 The Curry-Howard correspondence: intuitionistic vs classic

Let us go back to λ-calculus, and consider a restriction of it – the simply-typed λ-calculus –, introduced again by Church, and proved to be consistent by Rosser. From the point of view of computability this restriction is too strong, because the algorithms representable in it are a proper class of the always terminating ones, but it is still interesting when related with weaker notions of computability which are no more Turing-complete. We could say that the main point in the simply-typed λ-calculus is to "see" a λ-term as an ordinary mathematical function. This is easily done: if we have a function $F : A \to B$ and an element $a : A$, we can apply F to a and get an element $F(a) : B$. Let us write it as:

$$\frac{\vdash F : A \to B \qquad \vdash a : A}{\vdash F(a) : B}$$

Also, let us say we have an expression $M = M\{x\}$ depending on a variable x; suppose that when we substitute x in M for an element $a : A$, we get an $M(a) : B$. Then we can "see" the expression M as a *function of* x, going from A to B. Such a function is denoted by $\lambda x.M$. Let us write this fact as:

$$\frac{x : A \vdash M : B}{\vdash \lambda x.M : A \to B}$$

In order to compute the application $F(a)$ of a function $F = \lambda x.M$ on an element a of the domain of F, one has to substitute in the expression M all the occurrences of x by a; this substitution is written $M\{a/x\}$, and the result belongs to the codomain of F. So we have:

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

The system we just defined by those "typing rules" is the simply-typed λ-calculus (and by forgetting the information about types one gets the untyped λ-calculus). One of the main interests in this restricted calculus is the striking similarity with elementary logical rules. In particular, if we forget the λ-terms, we recognise the already mentioned rule for introducing an implication, and the rule for eliminating it. This is the first layer of the so-called *Curry-Howard correspondence*, which is also called "formulas-as-types correspondence" for the reason just explained. Also, one can see that – once we have fixed types for the variables – the λ-terms in a typing derivation carry all the information of the types one is using; actually, a typing derivation becomes, when forgetting about the λ-terms, the proof of the formula which is the type of the final λ-term, and that vice-vera any proof can be written as a typed λ-term in a given typing-context (corresponding to the hypotheses). This is why one also calls the correspondence "proofs-as-programs". Mathematically speaking, all this is quite elementary. The second layer of the Curry-Howard correspondence is much more interesting, because it says that this correspondences are not just happy coincidences. The already mentioned cut-elimination algorithm – discovered by Gentzen years before the birth of programming and with completely different motivations – corresponds to execution of programs in the following sense: if a proof π of a sequent $\Gamma \vdash A$ rewrites in one step of cut-elimination to a proof π' (necessarly of the same sequent $\Gamma \vdash A$), then the λ-term M associated with π in the simply-typed λ-calculus[13] β-rewrites in one step to the term N corresponding, in the same sense, to the proof π'. Thus the correspondence lifts to the dynamics, and this is why it is also known as "Curry-Howard isomorphism". This result is, mathematically speaking, less trivial.

If this correspondence is, at a first look, perhaps surprising, one can argue that from a logical point of view it is very restricted: the logical system that we used only treats implication and, although being implication at the heart of logic, one cannot prove much in this system; it is the

[13]The term M s.t. $\vec{x} : \Gamma \vdash M : A$, where $\vec{x}$ are the variables of M.

already mentioned minimal propositional intuitionistic logic. Now, something definitely relevant in the investigation of the mathematical nature of proofs, is that the same correspondence lifts to much more complicated systems of interest. We will not dwell on how important, from the point of view of mathematics, computer science and philosophy, this phenomenon is, but let us just say that, in a sense, it is the result which motivates the existence itself of a unique discipline of proofs and programs, which constitutes an important part of theoretical computer science[14].

We will consider now a more specific question: extending the correspondence from minimal propositional intuitionistic logic to classical one, whose possibility was not even clear. The situation changed at the beginning of the 90's thanks to Griffin [Gri90], who proposed to type control operators, such as Scheme's *callcc* or Felleisen's *C*, with Peirce law. This retrieves classical logic since we already mentioned that, in intuitionistic logic, adding Peirce's law is equivalent to adding the law of excluded middle or the law of contradiction. From this intuition, several ways of extending the correspondence to classical logic appeared. From our point of view, the most interesting ones are $\lambda\mu$-calculus [Par92] – that will occupy all Chapter 3 – and, only for this introduction, Krivine's classical realizability [Kri09]. In Krivine's realizability, one extends the simply typed λ-calculus with a new constant, called `callcc` (*call-with-current-continuation*) having the type of Peirce law, following Griffin. Then, one defines a finer relation than typing ($\Gamma \vdash M : A$), which is the "realizability relation" $\Gamma \Vdash M : A$, to be thought of as the statement that the program M "computationally justifies" the formula A. Another way of seeing it is that the program M corresponds to the extraction of the computational content of the formula A. In this sense – and only in this sense, as we are going to see – one can say that we are still in the Curry-Howard perspective. Now, the deep difference between this calculus (called "λ_c-calculus") and the usual one is, apart from `callcc` and the realizability relation, the way one models the notion of computation: in Krivine's realizability, programs are executed together with a *stack*[15] – to be thought of as the execution stack – playing the role of the *environment*[16]. This is the real difference between intuitionistic and classical: the computational content of the former lies in the internal properties of the programs, while the one of the latter lies in their interaction with the environment. In fact, the control operators are precisely instructions dealing with this aspect. The main reason why we are mentioning Krivine's realizability here is because in this setting one clearly sees what is the computational behaviour of `callcc` and, thus, the one of classical logic: the use of `callcc` consists in applying it on a term M and launching it in front of a stack; this "process" executes by, first, freezing the execution stack as a data in the memory of M, and then by launching M on the same stack; if at a certain moment the execution will need to make a decision (left/right in Figure 1.4), it can "guess" the right one (first guessing left in the figure); if this guess was wrong (the star in the figure) – say, it raises an error – the process can restore the memorized stack, thus *backtracking* at the moment it was first launched; now, when it will have to make the same decision again, it knows that the correct choice is the other one (on the right, in the figure), and it can continue the execution with (hopefully) no errors[17]. This is exactly the behaviour of the `callcc` in "real" programming languages. We will find the same behaviour also in $\lambda\mu$-calculus, as we will see in Section 4.2. This backtracking technique can be also stated in the form of the story of the "devil realising the excluded middle" ([Wad03, Chapter 4]), as well well as in the proof of the famous "drunk man formula":

$$\exists x\,(D\{x\} \to \forall y D\{y\}).$$

[14] We refer here to the vague distinction in "Volume A/Volume B" of theoretical computer science: "Volume B", or "European" TCS, mainly deals with programs/verification, while the other half, "Volume A", or "North American" TCS, is concerned essentially with algorithmics/complexity. Discussions on it can be found in [Var15].

[15] And not more "by themselves", as in usual λ-calculus.

[16] This notion of computation is also closer to reality.

[17] This mechanism is also essentially the same as the *exception handling* in languages like Java.

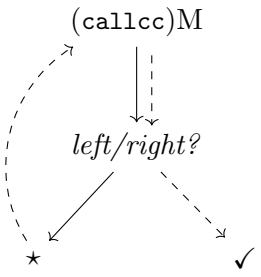


Figure 1.1: The typical behaviour of callcc

This is a classically provable formula, but *not* intuitionnistically provable. It says that if we place ourselves in the domain of non-empty pubs, and we read $D\{x\}$ as "x is drunk", then in any non-empty pub we can always find someone (x) which is s.t. if he is drunk, then all the other clients of the pub are drunk too. It may seem a strange property, but the proof is actually very simple: take one person x_0 in the pub (there is at least one since it is non-empty); either declaring $x := x_0$ works, in which case we are done, or it does not. If it does not, it means that the environment (the "opponent", if we think of a proof as a dialogue) has provided us with a person x_1 which is *not* drunk while our x_0 is. Thus, we change our mind and declare $x := x_1$. Now this choice necessarily works, thanks to the information (x_1 is not drunk) the environment itself gave us.

Anyways, there is a crucial point of fracture between the Curry-Howard perspective and Krivine's one: the realizability relation is not defined via typing derivations anymore, but in a more complex and indirect way; so now one is able to prove that some formulas can be realized[18] even when one does not have a proof in classical logic for that formula. This is a huge improvement, because now one can extract computational content from *axioms*, and thus consider much bigger parts of mathematics, such as those formalizable in ZFC – apparently Krivine has recently realized the full axiom of choice [Kri20] – which were not in the scope of the "traditional" Curry-Howard perspective simply because axioms do not have proofs, and hence cannot be typed! The price to pay is, however, that now the correspondence is not anymore an isomorphism, in the sense that the very notion of cut-elimination and/or normalization does not play any role. The notion of computation is only considered with respect to a certain set of "observable behaviours" one fixes at the beginning – playing the role of a parameter of the whole construction – and the realizability relation makes the correspondence more between formulas and programs than between proofs and programs[19].

If one wants to stay in a "purely Curry-Howard" perspective, that is, linking cut-elimination and program execution, the most "basic" framework is maybe Parigot's $\lambda\mu$-calculus. Of course, the first problem to deal with in classical logic, is the intrinsic non-determinism of cut-elimination and the presence of critical pairs, as we already discussed. The solution proposed by Parigot consists in forcing cut-elimination to be deterministic, by *a priori* choosing the way to eliminate the cuts. In order to do so, he modifies the formal system in use to write classical proofs, and introduces its *classical natural deduction* (which we will briefly present in Chapter 3) – in turn a particular case of its more general *free deduction*. The main difference with sequent calculus is that now one has two kinds of formulas at the right side of the "$\vdash$"": at most one *active*

[18] When one is lucky, even to exhibit such a realizer; and when one is extra lucky, to actually describe its programming behaviour – it is what Krivine calls the *specification problem*.

[19] But of course proofs still give raise to programs, as the crucial "adequacy" theorem states: any simply-typed λ-term realizes its type.

formula, and finitely many *passive* formulas. The new system enjoys a strongly normalizing cut-elimination. Once properly defined the typed programming language classical natural deduction derivations describe, one can extract the underlying untyped language simply forgetting types. The problem of course is which program to associate with the new rule of contradiction, and how to handle the notions of active and passive formulas. Parigot simply adds a new constructor to the ones of λ-calculus, involving a new binding operator called the μ-abstraction, which operates on a new set of variables usually called *names*. In this way, one can prove that a Curry-Howard correspondence (or, better said, isomorphism) holds between the $\lambda\mu$-calculus and the classical natural deduction. Hence, in the same way one can say that the (untyped) λ-calculus is the Turing-complete programming language in which intuitionistic proofs express their computational content, one can say as well that the (untyped) $\lambda\mu$-calculus is the Turing-complete programming language in which classical proofs express their computational content. The reader can consult [Bon07] for an interesting philosophically-oriented discussion about the Curry-Howard correspondence for classical logic.

1.5 Linear logic

Let $\texttt{id} : x \in \text{Bool} \to x \in \text{Bool}$ and let $\texttt{redundantId} : \text{Bool} \mapsto \text{Bool}$ be defined via the algorithm: *if* $x = 1$ *then* x *else* x. Of course $\texttt{id} = \texttt{redundantId}$ but while `id` uses its argument exactly once, `redundantId` uses it exactly twice in the execution: this is a fundamental difference, from both the point of view of computer science and that of logic. To understand why this is important from a computational view point, it is enough to see that λ-calculus has a kind of source of unsatisfactoriness: the notion of step of computation is clear, but it is not really elementary. In fact in a computation from $(\lambda x.M)N$ to $M\{x/N\}$, one has to duplicate the whole term N – independently from its size – the exact number of times (possibly zero, in which case N is erased) that x occurs in M. So it is not realistic to consider this step of computation to take place in constant time. A more "elementary version" of it would substitute N for *exactly one* occurrence of x at a time, but in λ-calculus there is no such thing.

To see that the point of duplication is important also from a logical point, let us mention that Girard found[20] that two rules that one usually considers for building derivations[21], which were usually considered "harmless", are in fact crucial. The rules are the following:

$$\frac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\, Contraction \qquad\qquad \frac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\, Weakening$$

and the analogous rules for the case at the left of the "$\vdash$"[22]. Abolishing these two rules one observes a surprising phenomenon: both the connectives $\wedge$ and $\vee$ split[23] in two different connectives: the multiplicative conjunction $\otimes$ and the additive one &, and the multiplicative disjunction $\parr$ and the additive one $\oplus$. This "hidden structure" of logical connectives is not ad hoc or chaotic, but is well-structured and deserves its own place as a well-behaved logical system, which is "linear logic without exponentials". Inside it, one finds two main subsystems, called the *additive* and the *multiplicative* system, which are again well-behaved systems. When we say "well-behaved" we mean that they satisfy all the properties a theory of proofs should have: namely a semantics and, most importantly, a (terminating and "natural") cut-elimination

[20] We are not following an historical order here.

[21] We did not write them in the previous section.

[22] Note that in intuitionistic logic the rule of contraction that we wrote is not possible since we cannot have two formulas at the right of the "$\vdash$", and weakening is only possible when Δ is empty – that is, it represents a contradiction, and so weakening "at the right of the "$\vdash$" is simply the principle "ex falso quodlibet".

[23] Linear logic is often referred to as a "microscope", or as a "micrologic" – not in the sense of being "small", but in the sense of looking at the internal structure of usual logic.

procedure. A crucial aspect is that one can define a duality relation $(.)^{\perp}$, playing the role of negation, which satisfies non-trivial properties (namely, it is involutive and yet constructive). This allows to define the *linear implication* $A \multimap B := A^{\perp} \parr B$. Furthermore, and most importantly for us, this system allows to decompose the implication "$\to$" – which corresponds to λ-abstraction under the Curry-Howard correspondence – in two more elementary and intuitive operations: in order to produce an output of type B from an input of type A – this corresponds to the specification "$A \to B$" – one, first, duplicates a certain number of times the input of type A – this has type $!A$ in linear logic – and then uses exactly once this duplicated inputs in order to produce the output of type B. Putting things together, we have the *fundamental decomposition*: $A \to B = (!A) \multimap B$. From a logical point of view, the new modality "!", together with its dual "?", correspond to a rehabilitation of the structural rules – weakening and contraction – in the system, but only in a controlled way. These two fundamental modalities are called *exponentials* because they allow to pass from the additive world to the multiplicative one – just like an exponential function – and *vice-versa*[24]. When added to the previous system, one obtains what is usually called "full linear logic". Since we will not deal with linear logic by itself, let us just mention that Girard's discoveries have literally created a whole new discipline, bringing a new insight in the foundations of logic, on which we will come back in Chapter 5, as well as new important mathematics in the realm of proof theory, and have found in theoretical computer science – especially in the abstract theory of programming languages – considerable applications. The situation of structural rules is a typical example of how the point of view of semantical preservation of truth is blind with respect to deeper considerations, that require a finer point of view.

1.6 The Taylor expansion of a program

If the most naive interpretation of a (simply-typed) λ-term is as a function, one may wonder if it is possible, as it is often the case for functions, to "approximate" it in some sense. The answer is positive, and the historically first notion of approximation has not much to do with functions: it is given by Böhm approximants[25]. This notion is related with the partial information that a program, from time to time, may output during its execution; so it is mainly "computer science driven". One may thus still wonder if a more "mathematically driven" notion of approximation exists. The answer is again positive, and was discovered by Ehrhard and Regnier in [ER03]. In this pioneering article, they introduce a *derivative operator* in the higher-order functional setting, defining a *differential λ-calculus*. This new syntax is not *ad-hoc*, as it finds its root in the Ehrhard's work in the semantics of linear logic [Ehr02]. Thus the differential λ-calculus is another great example of the interesting methodological phenomenon which consists in discovering a notable property in a semantical setting, and then internalising this property via a new syntax. The most exciting achievement of the differential λ-calculus is the possibility of defining a *Taylor expansion* of a λ-term, which has striking similarities with the usual one of analysis; the similarities are first in the shape: in the same way the Taylor expansion $\Theta(F)$ of a real function F, near 0, is the series of functions:

$$\Theta(F)(x) = \sum_n \frac{1}{n!}(\mathtt{D}^{(n)}F \cdot x^n)(0)$$

[24] Also, they correspond to the exponential power series in the differential λ-calculus, as we will mention in Section 3.2.

[25] Also called *partial* or *finite* approximants.

where $\mathtt{D}^{(n)}F \cdot a$ is the function $y \mapsto F^{(n)}(y) \cdot a$, the Taylor expansion $\Theta(Fx)$ of a term Fx is the formal series:

$$\Theta(Fx) = \sum_n \frac{1}{n!}(\mathtt{D}^{(n)}F \bullet x^n)0.$$

where $\mathtt{D}^{(n)}F \bullet N^n$ is a term in the syntax. But, more deeply, the similarities are in the role of this expansion: as each polynomial appearing in the series $\Theta(F)(x)$ approximates the function $F(x)$ (near 0), the "differential terms" occurring in $\Theta(Fx)$ provide a valid notion of approximation for Fx. Furthermore, in the same way a polynomial of degree n is a "simple" function which uses its argument exactly n times (in the sense of analysis[26].), the approximants of Fx given in $\Theta(Fx)$ are "simple" functions that use their arguments exactly n times (in the sense of computer science). The main intuition behind this notion of approximation is thus the tight connection which appears between the notion of linearity in analysis and that in programming (to use its argument exactly once during the computation). The paradigm shift initiated with the work of Ehrhard and Regnier has originated several different – but interrelated – axes of research, involving for instance an axiomatic description of differentiation in the categorical setting (see for instance [BCS09, Ehr16, CL19]) or the production of resource sensitive type systems and cost models for several functional programming languages (see for instance [Acc17, dC18]), as well as the adaptation of this kind of approximation to other functional languages (see for instance [BEM12, LZ12, LL19a, Vau07b, KMP20]).

The two notions of approximation – Böhm's and Taylor's – are strictly related by Ehrhard and Regnier's fundamental *commutation property*[27]:

$$\mathrm{NF}(\Theta(M)) = \Theta(\mathrm{BT}(M)).$$

One could even say that they give raise to "the same" approximation.

Actually, approximating via "differential terms" is not the only possibility given by the work of Ehrhard and Regnier. It turns out that even if we drop the rational factorial coefficients, and move to a much simpler syntax, we still get a valid approximation for λ-calculus. We will call this approximation the *resource approximation*, since the terms obtained via the mentioned restriction of differential λ-terms are called *resource terms*. The resource approximation can be also introduced independently from the differential λ-calculus. This kind of approximation is exactly the object of study of this manuscript.

[26] We mean that they use n the argument x in order to compute $x^n = x \cdot \cdots \cdot x$

[27] The word "commutation" here is appropriate because the Böhm tree $\mathrm{BT}(M)$ of M is a generalized normal form.

Chapter 2

Preliminaries

We first recall the few and very simple general mathematical notions we need in this manuscript. Then, we recall the basic notions about λ-calculus we will use. In particular, even if well known, we believe it is interesting to give some details about the construction of Böhm trees, as well as the conceptual place such an approximation occupies, because an important part of the thesis is precisely about the relation between this kind of approximation and the resource one.

2.1 Notations, terminologies and basic facts

Natural numbers The set $\mathbb{N}$ of natural numbers start from 0. The power set of a set X is denoted by $\mathscr{P}(X)$; we denote by $\mathscr{P}^*(X)$ the set $\mathscr{P}(X) - \emptyset$; we denote by $\mathscr{P}_{\text{fin}}(X)$ the finite parts of X, and analogously for the other uses of the subscript "fin". The cardinality of X is denoted by $\text{Card}(X)$.

Multisets and orders A *multiset* over a set X is a map from X to $\mathbb{N}$. The *support* of a multiset $A : X \to \mathbb{N}$ is the set $X - A^{-1}(0)$. A multiset is *finite* when its support is finite. We denote with $!X$ the set of finite multisets over X. Let A be a finite multiset with support $\{a_1, \dots, a_k\}$. As usual, we write such an A as $[a_1, \overset{(A(a_1))}{\dots}, a_1, \dots, a_k, \overset{(A(a_k))}{\dots}, a_k]$. In this manuscript, when we do not need to specify the multiplicities $A(a_i)$, we will shorten this notation to $[a_1, \dots, a_k]$, meaning that some of the a_i might be equal. Moreover, if an operation $n \pm a_i$ is defined for all $i = 1, \dots, k$, we will sometimes write $n \pm A$ for the multiset $[n \pm a_1, \dots, n \pm a_k]$.

We use a multiplicative notation for multisets: this means that the empty multiset is denoted with 1 and the union of two multisets A, B is denoted with $A * B$. The set of multisets over a set X (as well as $!X$) is a commutative monoid w.r.t. $*$, with neutral element 1.

It is well-known that, if X is a well-founded ordered set, then $!X$ is well-founded as well, w.r.t. to an order, called the *multiset order* on X, which is defined by setting: $A < B$ in $!X$ iff A can be obtained from B by replacing at least one occurrence of an element b of B with an arbitrary finite number (even 0, which corresponds to erasing b) of elements $a_1, \dots, a_n < b$.

Let $X_1, \dots, X_n$ be ordered sets (let us simply call $\leq$ the order in each of them). It is well known that one can define an order on the product $X_1 \times \cdots \times X_n$, different from the product order and called the *lexicographic order*, by setting: $(x_1, \dots, x_n) \leq (y_1, \dots, y_n)$ iff there is $1 \leq k \leq n$ s.t. $x_i = y_i$ in X_i for $i = 1, \dots, k-1$ and $x_k \leq y_k$ in X_k. The lexicographic order has the property that if each X_i is well-founded, then the product (of fixed length) is well-founded.

Syntax We will essentially deal with words on alphabets, so we will work in the *free monoid* of some alphabets. In particular, the words of our interest will always raise from *inductive*

constructions, and so one calls them *terms* instead. As usual, we make extensive use of "Backus-Naur-like notations" and do not specify the underlying alphabet, which is inessential and clear from the context: the "real" way of looking at terms is as (higher-order) abstract-syntax trees.

Rewriting systems Our terms are not just static pieces of information: they will represent *programs*, whose principal feature is of course to evaluate. This is well-modelled by appropriate *rewriting systems*, that is, simply appropriate sets X equipped with a binary relation $\to \subseteq X \times X$.

- The reflexive closure of $\to$ is the relation $\to \cup \{(x,x) \mid x \in X\}$, usually denoted by $\to^{=}$.

- The transitive closure of $\to$ is the smallest transitive binary relation on X containing $\to$. I.e., it is the relation $\{(x,y) \mid x \to x_1 \to \cdots \to x_n \to y \text{ for some } x_1, \ldots, x_n \in X, n \geq 0\}$.

- The reflexive-transitive closure (i.e. the union of the reflexive and of the transitive closure) of $\to$ is denoted with $\twoheadrightarrow$. It is the smallest transitive and reflexive binary relation on X containing $\to$.

- The symmetric closure of $\to$ is the relation $\to \cup \{(y,x) \mid x \to y\}$.

- The symmetric closure of $\twoheadrightarrow$ is the smallest equivalence on X containing $\to$, and is denoted with $\simeq$ in this preliminary section. That is, $x \simeq y$ iff there exist $x_1, \ldots, x_n \in X$ s.t. $x \twoheadrightarrow x_1 \twoheadleftarrow x_2 \twoheadrightarrow \cdots \twoheadleftarrow x_{n-1} \twoheadrightarrow x_n \twoheadleftarrow y$.

An element $x \in X$ is *reducible* iff there exist $y \in X$ s.t. $x \to y$. Otherwise it is said to be *normal*. An element $y \in X$ is a *normal form of* $x \in X$ iff y is normal and $x \twoheadrightarrow y$. An element $x \in X$ is said to be *normalizable* if it admits a normal form. A *reduction sequence* starting from an $x \in X$ is a (possibly infinite) sequence $x \to x_1 \to x_2 \to \cdots$ in X.

A rewriting system $(X, \to)$ is said to be:

- *strongly normalising* iff the are no elements of X admitting an infinite reduction sequence.

- *diamond* iff for all $x, v, w \in X$ s.t. $v \leftarrow x \to w$, there exist $y \in X$ s.t. $v \to y \leftarrow w$.

- *locally confluent* iff for all $x, v, w \in X$ s.t. $v \leftarrow x \to w$, there exist $y \in X$ s.t. $v \twoheadrightarrow y \twoheadleftarrow w$.

- *confluent* iff $(X, \twoheadrightarrow)$ is diamond.

- *Church-Rosser* iff for all $v, w \in X$ s.t. $v \simeq w$, there exist $y \in X$ s.t. $v \twoheadrightarrow y \twoheadleftarrow w$.

It is well-known that the confluence and the Church-Rosser are equivalent properties. In a confluent rewriting system $(X, \to)$ every $x \in X$ admits at most one normal form.

We recall the following general and well-known result:

Lemma 2.1.1 (Newman's lemma)**.** *A strongly normalising rewriting system is confluent iff it is locally confluent.*

We will also mention to the following well-known notion:

Definition 2.1.2 (Commutation of reductions)**.** *Two binary relations $\to_1$, $\to_2$ on X are said to* commute *iff for all $x, v, w \in X$ s.t. $v \ {}_2\!\leftarrow x \to_1 w$, there exist $y \in X$ s.t. $v \to_1 y \ {}_2\!\leftarrow w$.*

Trees We will also encounter finitely branching rooted trees with an order defined on the set of subtrees of each node. From now on, "tree" is synonymous of such trees. If B is tree, having R as root and immediate subtrees $B_0, \dots, B_k$ ordered following the indices, we represent it via the writing $B = RB_0 \cdots B_k$.

We mention this kind of trees because Böhm trees (discussed in Section 2.2.2 and in Section 3.5) are a special case of them; in this setting there is a one-leaf tree $\bot$ (which can thus appear only as a leaf). We add from now on this constraint in our definition of tree. Moreover, in the setting of Böhm trees one can decide to "obscure" an entire subtree by contracting it to $\bot$; this is why we give the following:

Definition 2.1.3. *We define the order $\sqsubseteq$ on the set of trees*[1] *by: $B \sqsubseteq B'$ iff B can be obtained from B' by contracting any number of subtrees of B' to $\bot$.*

Formal sums We describe below a construction which will be constantly used. Consider a set $\mathcal{L}$ (it will be some "resource sensitive language"). We call $2\langle\mathcal{L}\rangle$ the free module generated by $\mathcal{L}$ over the boolean semiring[2], which simply means the quotient set of the *formal sums* of finitely many elements of $\mathcal{L}$, quotiented by commutativity, idempotency and associativity of $+$. An element of $2\langle\mathcal{L}\rangle$ will be called a *sum*. We denote with 0 the empty sum, which is the neutral element for $+$. Said differently, a sum $\mathbb{T}$ is just a finite subset of $\mathcal{L}$, and the "$+$" is the union[3].

Every operation defined on $\mathcal{L}$, can be extended to $2\langle\mathcal{L}\rangle$ by linearity, but for the seek of clarity we will specify that at each time. Of course 0 is also the annihilating element for those linearly-extended operations.

Definition 2.1.4. *If some relation $\rightsquigarrow \subseteq \mathcal{L} \times 2\langle\mathcal{L}\rangle$ is defined, we extend it to all $2\langle\mathcal{L}\rangle \times 2\langle\mathcal{L}\rangle$ by setting:*

$$\rightsquigarrow := \{(t + \mathbb{S}, \mathbb{T} + \mathbb{S}) \mid \mathbb{S}, \mathbb{T} \in 2\langle\mathcal{L}\rangle \text{ with } t \notin \mathbb{S} \text{ and } t \rightsquigarrow \mathbb{T}\}.$$

Observe that the condition $t \notin \mathbb{S}$ is crucial if we want to hope for a strongly normalising reduction $\rightsquigarrow$. Indeed, if we defined the extension to sums without bothering for that condition, we would immediately have: $t = t + t \rightsquigarrow \mathbb{T} + t = \mathbb{T} + t + t \rightsquigarrow \mathbb{T} + \mathbb{T} + t = \mathbb{T} + t$ and thus we have created a cycle in the reduction, yielding an infinite $\rightsquigarrow$-reduction sequence. With the condition $t \notin \mathbb{S}$, both the previous $\rightsquigarrow$-reductions are not allowed: in the first, (the term) t belongs to (the single-element sum) t, and in the second $t \in t + \mathbb{T}$.

However, this condition forces us to pay particular attention to successive reductions. For instance, if $t \rightsquigarrow u \rightsquigarrow v$, then in order to obtain $t + u \twoheadrightarrow u + v$ (for lack of symbol, let us use the symbol $\twoheadrightarrow$ for the reflexive transitive closure of $\rightsquigarrow$) one *cannot* reduce first t, otherwise we would have $t + u \rightsquigarrow u + u = u$. In this case reducing first u, and only then t, yields to the desired result. But in general, because of the definition of reduction on sums (Definition 2.1.4), consecutive reductions – and in particular reducing a sum to the sum of the reduct of each addend – are, in general, not necessary possible.

2.2 The (untyped) λ-calculus

2.2.1 Basic notions and facts

The set of *pre-λ-terms* is defined inductively as:

$$M ::= x \mid \lambda x.M \mid MM$$

[1] We mean the just given definition of tree, thus with the added constraint on $\bot$.

[2] We mean the booleans with the equation $1 + 1 = 1$.

[3] We keep the "sum" notation because it is manageable and reminiscent of the fact that the general case of coefficients in a ring different from $\mathbb{Z}_2$ is important in the setting of $\lambda/\lambda\mu$-calculus, even if we will not need it.

where $x \in \mathrm{Var}$ – a countably infinite set which we fix now and for all the thesis. The sets $\mathrm{FV}(M)$ and $\mathrm{BV}(M)$ of *free variables* and of *bound variables* of a pre-λ-term M are defined as always. We also assume Barendregt's convention[4] – that is, all the bound variables in a pre-λ-term are different. The α-equivalence is the binary relation on pre-λ-terms which equates M and N whenever there are bound variables $x_1, \dots, x_n$ ($n \geq 0$) of M and variables $y_1, \dots, y_n$ not occurring free in N s.t. N can be obtained from M by replacing all the occurrences of x_i by y_i. It is clearly an equivalence relation.

The set Λ of λ-*terms* is the set of pre-λ-terms quotiented by α-equivalence.

Remark that the notion of free variable still makes sense for a λ-term (but not that of bound variable). We will thus just say that a variable "occurs in a λ-term M", or that it "does not occur" (hence omitting the adjective "free"). Except if explicitly stated, when we write a pre-λ-term we will always mean the associated λ-term, that is, its α-equivalence class. The following are famous λ-terms which we will happen to use in this manuscript without recalling them:

$$\mathsf{I} := \lambda x.x, \quad \Delta := \lambda x.xx, \quad \Omega := \Delta\Delta, \quad \texttt{True} := \lambda xy.x, \quad \texttt{False} := \lambda xy.y.$$

A k-*context* (also called a *multi-hole context* if we do not need to specify k) is a function $C : \overbrace{\Lambda \times \cdots \times \Lambda}^{k} \to \Lambda$ inductively defined as:

$$C ::= \xi_1 \mid \cdots \mid \xi_k \;\mid\; x \;\mid \lambda x.C \mid CC,$$

where ξ_i is the i-th projection (of arity k) and x is the constantly equal to $x \in \mathrm{Var}$ function (of arity k). The functions ξ_i's occurring in the inductive definition of C are traditionally called *holes*. We denote by $C(\!|\vec{M}|\!) \in \Lambda$ the image of $\vec{M}$ under C. Remark that, by definition of contexts as functions, free occurrences of variables in M may become bound by a "λ" in $C(\!|M|\!)$, like as in $C(\!|x|\!) = \mathsf{I}$, where $C = \lambda x.\xi$. Thus, contrary to terms, contexts do *not* satisfy α-equivalence; for instance $\lambda x.\xi \neq \lambda y.\xi$, since $(\lambda x.\xi)(\!|x|\!) = \mathsf{I} \neq \lambda y.x = (\lambda y.\xi)(\!|x|\!)$.

A *single-hole context* is a 1-context with exactly one occurrence of its hole (usually denoted ξ instead of ξ_1). The *contextual closure* of a binary relation $\mathcal{R}$ on λ-terms is the binary relation given by set $\{(C(\!|M|\!), C(\!|N|\!)) \mid M\mathcal{R}N,\ C \text{ single-hole context}\}$. Thinking as contexts as defined on pre-λ-terms (instead of terms), we can give the same definition for a relation of pre-λ-terms.

We denote by $M\{N/x\}$ the result of replacing the term N for all the free occurrences of x in M. One properly defines this operation through pre-λ-terms, performing the replacement only after the usual renaming of bound variables in the pre-λ-term M, to avoid capture of free variables in N. The α-equivalence coincides then with the relation on pre-λ-terms which is the contextual closure of the relation equating a pre-λ-term of shape $\lambda x.M$ with a pre-λ-term of shape $\lambda y.M\{y/x\}$, whenever $y \notin \mathrm{FV}(M)$.

Lambda-terms are meant to represent programs, the notion of execution being as usual β-reduction, which we will call instead λ-*reduction* and denote by $\to_\lambda$. It is the contextual closure of the relation $(\lambda x.M)N \to_{\text{base}} M\{N/x\}$. The reflexive, symmetric and transitive closure of $\to_\lambda$ is called λ-*equivalence* and is denoted by $=_{\boldsymbol{\lambda}}$.

One might be interested in equating λ-terms via an equivalence relation different from $=_\lambda$. In this case the following are the standard notions to consider: a *congruence* is a contextual equivalence[5] on Λ; a λ-*theory* is a congruence containing $=_{\boldsymbol{\lambda}}$; the *term algebra*[6] of a λ-theory $\mathcal{R}$ is the quotient $\Lambda/\mathcal{R}$ (endowed with the induced constructors of Λ); a λ-theory $\mathcal{R}$ is said to be *non-trivial* when its term algebra is not a singleton.

[4] Appartently, Barendregt in turn attributes it to Ottmann – see the Addenda for the sixth imprinting of [Bar84].

[5] An equivalence is said to be contextual when it is the contextual closure of some relation.

[6] Here the word "algebra" is intended in the sense of universal algebra.

2.2.2 Böhm approximation

Lambda-terms can be organized in two classes, depending on their behaviour: the solvable ones, and the unsolvable ones. This is a fundamental distinction and plays a crucial role in Böhm approximation, so let us give some details. We will use as "running example" the algorithm known as the "Babylonian method" for the computation of square roots.

A λ-term M is called *solvable* if there exists a context C of shape[7] $C = (\lambda x_1 \dots x_n.\xi)N_1 \cdots N_k$ such that $C(\!|M|\!) =_{\boldsymbol{\lambda}} \mathtt{I}$; it is *unsolvable* otherwise. A solvable term can be sent (mod $=_{\boldsymbol{\lambda}}$) to any other term by means of some appropriate context.

An *a priori* unrelated notion is the one of head normal form: a λ-term M is called a *head normal form* (*hnf* for short) if it is of shape $M = \lambda x_1 \dots x_n.\, x\, M_1 \cdots M_k$ for some $n, k \geq 0$, and the (occurrence of the) variable x (which can be equal to some of the x_i's) is said to be in *head-position* for M. The terms that are not hnf's are necessarily of shape $\lambda x_1 \dots x_n.\, (\lambda y.P)Q\, M_1 \cdots M_k$, that is, in their head-position there is a redex $(\lambda y.P)Q$ instead of a variable.

One says that a term M *has an hnf* iff $M =_{\boldsymbol{\lambda}} N$ with N hnf. The *head reduction* is the partial function which is undefined exactly on the hnf's and, when it is defined, gives the term obtained by reducing the head-redex of the term it is applied to. We write $\to_{\mathrm{h}}$ for its graph relation.

It is well-known that one has the following fundamental characterization: M is solvable iff M has a hnf iff the head-reduction on M terminates.

The paradigmatic example of an unsolvable term is Ω.

Once a hnf $N = \lambda x_1 \dots x_n.\, y\, M_1 \cdots M_k$ of a term M is reached during λ-reduction, pursuing the reduction will only affect $M_1, \dots, M_k$ but not the "head part" $\lambda x_1 \dots x_n.y$ of M, nor the number k of applications of the head variable y. The information given by the pair "(head part, number of applications)" is thus a partial output of the computation, and the term will never return working on it. The existence of terms with a hnf – i.e. solvable – but that are not normalizable appears now very natural: it corresponds to algorithms that, from time to time, output a part of an ideal "fully computed" output, providing thus an increasing approximation of it, although never reaching it once and for all. The ancient Babylonians and the ancient Indians already knew[8] the power of such algorithms: they knew a procedure computing each consecutive digit of $\sqrt{2}$. Compare it to Ω which, in turn, really produces no kind of information, as it is only capable of reducing to itself.

As first understood by Barendregt, the right way of modelling the notion of "computationally meaningful" is, in λ-calculus, exactly given by "producing partial information", that is, solvable terms - and not by normalizable ones as Church originally thought. This is supported by the "Genericity Property", on which we will come back at time (Theorem 3.5.29).

In the remaining of this Section, we are going to first give some detail about the approximation based on the notion of solvable term.

If M is normalising, then the question "what does M compute?" is easily answered: it computes $\mathrm{nf}_\lambda(M)$. But what do solvable terms compute? The Babylonian algorithm producing $1.41421\dots$ is computing the ideal object $\sqrt{2}$, which can be thought of as the infinite sequence of its digits. The idea is then to do the same for solvable terms: we register all outputted information returned by M when reaching a hnf. The natural way of collecting it is to organize

[7] Wlog, we can restrict the abstracted variables $x_1, \dots, x_n$ in C to the free variables of M.

[8] Apparently there is no certain historical evidence of the fact that those cultures knew exactly what we now call the "Babylonian method", but there are reasons to think that they did.

it as a tree, called the *Böhm tree* of M. So one gets the following:

Definition 2.2.1 (Böhm trees)**.** *The Böhm tree* $\mathrm{BT}(M)$ *of a* λ*-term* M *is defined coinductively*[9] *as follows:*

If M *is unsolvable, then* $\mathrm{BT}(M)$ *is the single leaf* $\bot$ *tree;*

If M *is solvable with* $M \twoheadrightarrow_h \lambda x_1 \dots x_n.yM_1 \cdots M_k$*, then* $\mathrm{BT}(M)$ *is the finitely branching rooted tree with*[10]*:*

- $\lambda x_1 \dots x_n.y$ *as root*
- $\mathrm{BT}(M_1), \dots, \mathrm{BT}(M_k)$ *as immediate subtrees, ordered following the indices.*

We denote by $\mathfrak{B}_\Lambda$ *the set of all the Böhm trees of some* λ*-term, and by* $=_{\mathcal{B}}$ *the equivalence on* Λ *given by the equality of Böhm trees.*

If M is normalizable then $\mathrm{BT}(M) = \mathrm{nf}_\lambda(M)$, where the equality means that $\mathrm{BT}(M)$ is just the tree representation of the word $\mathrm{nf}_\lambda(M)$. Thus, Böhm trees are "generalized normal forms", defined for all terms, and they answer to the question "what does M compute?". So, Böhm trees form a semantics for λ-terms (if $M =_{\boldsymbol{\lambda}} N$ then $\mathrm{BT}(M) = \mathrm{BT}(N)$), in the same way one can say that $\sqrt{2}$ is the "semantics" of the Babylonian method – in the sense that no matter what implementation has the algorithm, the computational meaning will always be to compute $\sqrt{2}$.

The analogy with the Babylonian method goes further. This method produces rational numbers ("truncations" of real numbers, if one thinks them as infinite sequences of digits) that are approximations of an irrational one. This is expressed by the fact that $\mathbb{Q}$ is dense in $\mathbb{R}$ (with their natural topologies) and by the fact that the rationals produced by the Babylonian method tend to $\sqrt{2}$ – in the topological sense of limit. Now, the same happens in λ-calculus, as any "truncation" of $\mathrm{BT}(M)$ – what we call a "Böhm approximant" (Definition 2.2.3) – turns out to be an approximation of $\mathrm{BT}(M)$ in the following sense: the set $\mathfrak{B}_\Lambda$ of Böhm trees carries a canonical topology (its Scott-topology as cpo), with respect to which its subset $\mathcal{A}pp$ is dense into. Also, the Böhm approximants of M tend to $\mathrm{BT}(M)$, in the order-theoretic sense corresponding to a limit, that is, by taking a sup (Theorem 2.2.9).

As a last parallelism, let us say that in the same way the ideal object $\sqrt{2}$ is of a different "nature" of its rational approximations – the former is conceived as an infinite object while the latters are finite – also Böhm approximants are finite while Böhm trees are infinite.

Historically this is the first notion of approximation that has been discovered. Let us give some details about it.

Definition 2.2.2. *The set* $\mathfrak{B}$ *of all the* Böhm-like trees *is the set of trees*[11] *with nodes in the set* $\{\bot\} \cup \{\lambda x_1 \dots x_n.y \mid x_1, \dots, x_n, y \in \mathrm{Var}\}$ *and s.t. the* $\bot$*'s can only appear as leaves. Böhm-like trees are endowed with the partial order* $\sqsubseteq$ *defined in Section 2.1.*

Of course Böhm trees are (possibly infinite) Böhm-like trees.

Definition 2.2.3 (Böhm approximants)**.** *The set* $\mathcal{A}pp$ *of* Böhm approximants[12] *is the set of* finite *Böhm-like trees. It can be given the following inductive characterization:*

$$\mathcal{A}pp: \qquad P ::= \bot \;\mid\; \lambda x_1 \dots x_n.y\overbrace{P \cdots P}^{k} \quad \textit{(for } n, k \geq 0\textit{).}$$

[9] The interested reader is invited to consult [JR12] for an introduction to coinduction, and [Las99] for a discussion of the coinduction principles behind this definition.

[10] The definition makes sense because, as we mentioned, the head part and the number of applications are the same for all hnf's of a same term.

[11] We mean the notion of tree given in Section 2.1.

[12] Also called "finite approximants" or also "partial approximants".

Böhm approximants can be equivalently defined as follows:

Definition 2.2.4. *The set $\Lambda_\perp$ of* $\lambda\perp$-terms *is the set of λ-terms built with an additional constant $\perp$, and quotiented by the equations:*

$$\lambda x.\perp = \perp \qquad\qquad \perp M = \perp.$$

The $\lambda\perp$-terms are endowed with usual λ-reduction.

Now the set of Böhm approximants can be identified with the set of λ-normal forms and, under this identification, the preorder $\sqsubseteq$ they inherit from Böhm-like trees coincides with the contextual partial preorder on $\Lambda_\perp$ generated by the clause: $\perp \sqsubseteq M$ for all $M \in \Lambda_\perp$.

One extends Böhm trees to $\Lambda_\perp$ trivially setting $\mathrm{BT}(\perp) := \perp$, so that $P = \mathrm{BT}(P)$ for all Böhm approximant P.

The map $\mathrm{BT} : M \in \Lambda_\perp \mapsto \mathrm{BT}(M) \in \mathfrak{B}_\Lambda$ endows a preorder (it is not antisymmetric) $\sqsubseteq$ on $\Lambda_\perp$ (so, in particular, on Λ), defined by $M \sqsubseteq N$ iff $\mathrm{BT}(M) \sqsubseteq \mathrm{BT}(N)$. It is clear that $=_\mathcal{B}$ is the equivalence induced by the preorder $\sqsubseteq$ on $\Lambda_\perp$ (that is, $M =_\mathcal{B} N$ iff $M \sqsubseteq N \sqsubseteq M$).

We can now give the following:

Definition 2.2.5 (Böhm approximants of a λ-term)**.** *Given a λ-term M, the set $\mathcal{A}(M)$ of* Böhm approximants of M *is defined by:*

$$\mathcal{A}(M) = \{P \in \mathcal{A}pp \mid M \twoheadrightarrow_\lambda N \sqsupseteq P \text{ for some } N \in \Lambda\}.$$

Definition 2.2.6. *1. One says that $P_1, P_2 \in \mathcal{A}pp$ are* compatible *if they share a common upper bound in $\Lambda_\perp$ w.r.t. $\sqsubseteq$. It is trivial to check that if $P_2 \neq \perp$, then P_1, P_2 are compatible iff $P_1 = \lambda x_1 \dots x_n.yP_{11} \cdots P_{1k}$ and $P_2 = \lambda x_1 \dots x_n.yP_{21} \cdots P_{2k}$ for (the same) $n, k \geq 0$, with P_{1j}, P_{2j} compatible for all $1 \leq j \leq k$, and $\perp$ being compatible with any term.*

2. Given two compatible $P_1, P_2 \in \mathcal{A}pp$, one can easily see (for example by induction on P_1) that there exist a sup *$P_1 \sqcup P_2 \in \mathcal{A}pp$, which can be inductively constructed as:*

$$P_1 \sqcup P_2 = \begin{cases} \lambda\vec{x}.y(P_{11} \sqcup P_{21}) \cdots (P_{1k} \sqcup P_{2k}), & \text{if } P_1 = \lambda\vec{x}.yP_{11} \cdots P_{1k} \text{ and} \\ & \quad P_2 = \lambda\vec{x}.yP_{21} \cdots P_{2k}, \\ P_i, & \text{if } P_{3-i} = \perp \ (i = 1, 2\ ). \end{cases}$$

It is easy to see that two upper bounds of a same term must be compatible. It is also easily seen (for example by induction on $P_1 \in \Lambda_\perp$) that $\sqcup$ is associative, and thus one defines as expected a term $P_1 \sqcup \cdots \sqcup P_k \in \Lambda_\perp$ whenever the finitely many P_i's are pairwise compatible, and this is again the sup of the set $\{P_1, \dots, P_k\}$ in $\Lambda_\perp$.

Proposition 2.2.7. *For $M \in \Lambda$, the set $\mathcal{A}(M)$ is an ideal in $(\mathfrak{B}, \sqsubseteq)$, i.e. non-empty, downward closed and directed* [13]*.*

Remark 2.2.8. *It is possible to define an operation $\bigsqcup_i P_i := \lim_{k\to\infty} \bigsqcup_{i=1}^{k} P_i \in \mathfrak{B}$ for countably many pairwise compatible P_i's forming a directed set. This operation informally takes the potentially infinite tree obtained by extending $\bigsqcup_{i=1}^{k} P_i$ with the next P_{k+1} for all k. One has that $\bigsqcup_i P_i = \sup_i P_i$ in $(\mathfrak{B}, \sqsubseteq)$, and it does not depend on the chosen enumeration of the P_i's.*

In particular, there always exists $\sup_{P\in\mathcal{A}(M)} P = \bigsqcup_{P\in\mathcal{A}(M)} P$. Actually, this sup is the Böhm tree of M, as the following known fundamental theorem states:

[13] A set is *directed w.r.t.* a preorder whenever any two of its elements share a common upper bound.

Theorem 2.2.9 (Approximation theorem)**.** *Let $M \in \Lambda$. One has:*

$$\mathrm{BT}(M) = \bigsqcup_{P \in \mathcal{A}(M)} P.$$

One can also see that: $\mathcal{A}(M) \subseteq \mathcal{A}(N)$ iff $\mathrm{BT}(M) \sqsubseteq \mathrm{BT}(N)$ iff $M \sqsubseteq N$ as well (for $M, N \in \lambda\bot$).

An issue intrinsic to the definition of Böhm tree and Böhm approximants is the difficulty of characterising $\mathcal{A}(MN)$ in terms of $\mathcal{A}(M)$ and $\mathcal{A}(N)$. Indeed, not only $\mathcal{A}(\cdot)$ is not closed under application, but applying PQ can give rise to a term without a $\lambda\bot$-normal form, like Ω. The same problem arises for the computation of a Böhm tree. At the heart of those difficulties one can say that there is the fact that Böhm-trees are a coinductive definition, and Böhm-approximants are defined via an existential quantification on λ-reductions. Thanks to the approximation theorem however one can, in a sense, take the best from both worlds: Böhm trees are generally easy to compute (because one can follow the head reduction), while Böhm approximants are generally better suited for proving properties.

Chapter 3

The resource approximation for λ-calculus

> Understanding the relation between the term and its full Taylor expansion might be the starting point of a renewing of the theory of approximations (usually based on Böhm trees).
>
> *Thomas Ehrhard, Laurent Regnier –[ER03]*

3.1 Plan of the Chapter

In this chapter we study the resource approximation for the λ-calculus, and in particular how one can use it to produce results about the latter. The main parts of it, Section 3.3 and Section 3.5, are based on our work [BM20]; not only we give all the needed details, but also we reorganize the matter of the paper, in order to show which are the real dependences between the results and in order to make it a real alternative formulation of, essentially, Chapter 14 of [Bar84].

Even if resource approximation could be introduced independently from the differential λ-calculus, it finds its roots in it and so we find it useful to begin with an introduction to the ideas of the differential λ-calculus Section 3.2). However, we do it in a different way from how this topic is usually introduced: we do not pretend to be rigorous, but instead we prefer to explain the ideas with an accent to the relation with mathematical analysis, trying to make it clear what is the real connection between this "differential calculus" and the usual one in mathematical analysis. The aim is to give a *heuristics* of the topic, in the same spirit one often motivates or derives some modelizations in applyied mathematics and similar fields.

After that, we restrict our attention to the world of the resource approximants: Λ^{r}. We first of all study the properties of the (qualitative[1]) Taylor expansion map $\mathcal{T}$ (Section 3.3). In particular, in order to prove some properties, we will have to consider the *rigid terms* (Definition 3.3.28) – resource terms in which there are lists instead of bags – and prove some basic results about them (Section 3.3.1). At that moment we will already be able to prove (Section 3.3.2) two main results: Stability property (Theorem 3.3.37, for which we need the rigid terms) and Perpendicular lines property (Theorem 3.3.40). We express and prove them inside $\Lambda/_{=_\tau}$ (the

[1] We will sometimes omit this adjective since we will only talk about the qualitative version, and not about the quantitative one.

term algebra of the λ-theory of the Taylor normal form) and not in the usual $\Lambda/_{=_{\mathcal{B}}}$, but in the next sections we will translate them inside the latter.

In Section 3.4 we provide a quick "interlude" about the call-by-value λ-calculus (Definition 3.4.2): in fact, a nice value of our new proof of the Stability is that it is highly "syntax independent" and it can be easily adapted to this framework[2]. The results we need, in order to make sure the arguments in the proof do hold, are basically all taken by [KMP20].

In Section 3.5 we finally show in which sense, and how, the resource approximation "subsumes" Böhm's one. The crucial ingredient here is the renowed *commutation formula* (Theorem 3.5.6), for which we provide a "simpler" proof: by simpler we mean that it basically uses properties which one has already found when developing the resource approximation. With the addition of some other quite natural properties (Lemma 3.5.9 and related ones), we can prove that the λ-theories induced by Taylor and Böhm approximation coincide (Corollary 3.5.13). The "moral" is that, at least in the qualitative case (no coefficients), this non-trivial result is however easier to prove than it could appear looking at the original proof in [ER08]. The commutation formula means that the λ-theories $=_{\tau}$ and $=_{\mathcal{B}}$ actually coincide (Corollary 3.5.13): this provides an alternative proof of the Contextuality of Böhm trees, which is essentially reduced to the Monotonicity of Taylor normal form.

We can do much more: in fact we can already translate the Perpendicular lines and the Stability property from the realm of Taylor expansion to the one of Böhm trees. For the former, it is straightforward (Theorem 3.5.14); for the latter (Theorem 3.3.37), we need to prove some results about intersection of Taylor normal forms (Lemma 3.5.27). As it is well known, these two results (which express some forms of weak sequentiality) in fact have as immediate consequence that λ-calculus *cannot* implement parallel computations; we achieve therefore this important result entirely "syntactically" (the usual proofs being semantic). Finally, we turn the attention to two other fundamental results of λ-calculus: the Genericity property (Theorem 3.5.29) – which motivates the identification "unsolvables = meaningless" – and the Continuity lemma (Lemma 3.5.33) – which is the crucial ingredient of the proof of the fact that all λ-definable functions are continuous. As for the Stability and for the Perpendicular Lines property, our proof technique strongly relies on the linearity of resource terms (if something is erased in the computation, it did not appear from the beginning), but also on the relation between Böhm approximants and linear/affine resource terms that we will introduce at time.

We conclude the chapter with some comments (Section 3.6) mainly about the relation between our techniques and the usual one used in order to prove the above results.

3.2 Interlude: an heuristic for the differential λ-calculus

First intuitions Let us consider a real valued function $y := f(x)$ with $x \in \mathbb{R}^n$. Denote with $\mathbb{R}^n \multimap \mathbb{R}$ the dual space of $\mathbb{R}^n$. Suppose y is differentiable everywhere. This means, by definition, that there exists a function (the differential of y, which has no reason to be linear) $dy : \mathbb{R}^n \to (\mathbb{R}^n \multimap \mathbb{R})$ s.t. $y(x+u) = y(x) + dy(x)(u) + o(\|u\|)$ for all x and for $\|u\|$ small. Moreover, the differential operator $d : y \in \{\text{differentiable functions}\} \mapsto dy \in (\mathbb{R}^n \multimap \mathbb{R})^{\mathbb{R}^n}$ is linear (sum of functions is defined pointwise). It a basic known fact that $d(y)(x)(u) = \mathtt{D}y(x) \bullet u = \dfrac{\partial y}{\partial u}(x)$ (where $\bullet$ is the scalar product in $\mathbb{R}^n$). Here $\mathtt{D}y(x)$ is the gradient of y in x, and we used this notation instead of the more common one $\nabla y(x)$ because it is the one one finds in differential λ-calculus. Let us denote with $\mathtt{D}(.) \bullet (.)$ the function:

$$\mathtt{D}(.) \bullet (.) : (y, u) \in \{\text{differentiable functions}\} \times \mathbb{R}^n \mapsto \frac{\partial y}{\partial u} \in \mathbb{R}^{(\mathbb{R}^n)}$$

[2] In the next chapter we will see that actually it adapts also to the case of λμ-calculus, which supports the thesis of a "scalable proof".

which is linear in both its arguments.

Since we want to put some λ-calculus into those considerations, it is helpful to remember that, by *currying*, we can restrict to the case where all functions have arity 1; so let us consider the case $n = 1$. The gradient $\mathtt{D}y$ of y becomes to the derivative $\frac{d}{dx}y$ of y and $\bullet$ the product $\cdot$ in $\mathbb{R}$. Now, remember that y is defined via the "law" $y = f(x)$, so if one wanted to explicit f in the expression "$\mathtt{D}y \bullet u$", one would usually write "$\mathtt{D}f(x) \bullet u$". In λ-calculus notation one would instead write "$\mathtt{D}(\lambda x.f)\bullet u$", since the declaration "$y = f(x)$" is written "$y = \lambda x.f$". Furthermore, since the explicit dependence of $\mathtt{D}y \bullet u$ w.r.t. the independent variable x is given by the "law" $\frac{d}{dx}f(x)\cdot u$, one would write this fact in λ-calculus notation by "$\mathtt{D}y\bullet u = \lambda x.\left(\frac{d}{dx}f \cdot u\right)$". Putting this two things together, the fact that "the function $\mathtt{D}y \bullet u$, with $y = f(x)$, is given by the law $\frac{d}{dx}f \cdot u$", takes the form:

$$\mathtt{D}(\lambda x.f) \bullet u \to \lambda x.\left(\frac{d}{dx}f \cdot u\right). \tag{3.1}$$

However, what we just did is only some abuse of λ-calculus notation, and has, *a priori*, no serious content.

Keeping on with the heuristics The following considerations in fact provide some serious motivation for continuing the investigation. Taylor expanding $f(u)$ near x_0 we get:

$$f(u) = f(x_0) + \frac{d}{dx}f(x_0) \cdot (u - x_0) + \sum_{n\geq 2} \frac{1}{n!}\frac{d^n}{dx^n}f(x_0) \cdot (u - x_0)^n.$$

Let us read this as saying that if we "force" f to use its argument u *exactly once*, then (for u close to x_0) $f(u)$ behaves like $f(x_0)+\frac{d}{dx}f(x_0)\cdot(u-x_0)$: in fact, in the other factors of the expansion, f needs "u" at least twice in order to compute $(u-x_0)^n$. But $f(x_0)+\frac{d}{dx}f(x_0)\cdot(u-x_0)$ is just a vertical and horizontal shift of $\frac{d}{dx}f(x_0)\cdot u$, so that (for u close to x_0) we get the following "moral equation":

$$f \text{ forced to use its argument } u \text{ exactly once } = \frac{d}{dx}f(x_0)\cdot u.$$

Of course, mathematically speaking, we are saying nothing more than the basic fact that the derivative of a function gives its best linear approximation; what we are adding here is the reference to the fact of "being forced" to use its argument exactly once, i.e. the notion of linearity in computer science.

Rembering equation (3.1), we are tempted to read the "term" $\mathtt{D}(\lambda x.M) \bullet N$ precisely as the "name" of an expression of x, such expression being $\frac{d}{dx}M \cdot N$, which behaves as $\lambda x.M$ when it is applied to N under the constraint of using N exactly once. Let us say that $\mathtt{D}(\lambda x.M) \bullet N$ is the name of the function that "linearly applies" $\lambda x.M$ to N, call it the *linear application*.

On the other hand, to force $\lambda x.M$ to use N exactly once during *all* the computation, means to make sure N goes replacing exactly one occurrence (instead of all the occurrences) of x in M and also to make sure it will be never duplicated nor erased. Thus $\frac{d}{dx}M \cdot N$ can be thought of as some sort of substitution; call it *linear substitution*.

Consider this new linear substitution. It is clear that not all the occurrences of x in M can be used for substituting: take for instance $M := (\lambda y.yy)x$; then $(\lambda x.M)N$ moves N in exactly one occurrence of x in M (the only one) but doing so one obtains $(\lambda y.yy)N$ which, during

the computation, duplicates N. In a sense, this occurrence of x in this M is the prototypical "non linear" one. Let us call *linear* the good occurrences of a variable x in M we can use for the linear substitution we are looking for, that is, the occurrences of x in M which cannot be duplicated nor erased no matter what use of M one does. Call a constructor $C(.)$ of λ-calculus *linear* when for all term S and for all variable x, the linear occurrences of x in $C(S)$ are those linear in S. One sees that:

1. $\lambda x.(\cdot)$ is linear
2. $(\cdot)Q$ is linear but $P(\cdot)$ need *not* be linear
3. $\mathtt{D}(\cdot) \bullet (\cdot)$ is linear in both arguments.

Remark now a crucial fact:

Remark 3.2.1. *The previous constructors in 1,2,3 are linear in the sense of λ-calculus exactly when, read as a functions, they are linear in the usual sense of commutating with sums. In fact:*

1. *in the case $\lambda y.(\cdot)$ this simply follows from the fact that summing functions is done pointwise, and in particular from the* definition *of function sum: the definition of the function $P + Q$ is, reading the following λ-terms as real functions, precisely $\lambda x.(P + Q) := \lambda x.P + \lambda x.Q$*
2. *in the case $C = (\cdot)Q$ this simply follows again from the fact that summing functions is done pointwise, and in particular from the* expression *defining the sum function: the expression defining $(P+Q)(x)$ is, reading the following λ-terms as real functions, precisely $(P + Q)x := Px + Qx$. On the contrary, a function $P(\cdot)$ need not be linear in the sense of commuting with sums.*
3. *in the case $\mathtt{D}(\cdot) \bullet (\cdot)$, this is just the already mentioned linearity in both arguments of $\mathtt{D}(\cdot) \bullet (\cdot)$ seen as a real function.*

If we remember that the notion of linearity in programming (and proof theory) is – *a priori* – unrelated with the usual notion of analysis, all the coincidences expresses by Remark 3.2.1 are surprising.

Differential λ-calculus in a nutshell Before quickly presenting the actual syntax of the differential λ-calculus, let us see how, in addition to the previous considerations, $\frac{d}{dx}M \cdot N$ can be defined inductively on M.

The "directional derivative" $\frac{d}{dx}M \cdot N$ of the program $M = M\{x\}$ along N should represent the linear substitution of N in M for x, that is something which behaves as $(\lambda x.M)N$ where $\lambda x.M$ is forced to use N exactly once. So the linear application $\mathtt{D}(\lambda x.M) \bullet N$ and the linear substitution $\frac{d}{dx}M \cdot N$ are respectively the name and the concrete expression of the algorithm which non-deterministically chooses exactly one linear occurrence of x in M, declares it not duplicable nor erasable, and then substitutes N for that occurrence. In order to perform such a linear substitution we certainly have to substitute N for exactly one linear occurrence x_0 of x in M (such an occurrence is guessed non-deterministically); call S the maximal strict subterm of M containing this chosen occurrence x_0. If $M = C(S)$ and $C(.)$ is a linear constructor, we say that S is in *linear position* for M. If S is in linear position for M, we are sure that N will

not be duplicated nor erased in the future. This happens for sure in the cases $1, 3$ above, so for this cases we have:

$$\begin{gathered}\frac{d}{dx}x \cdot N := N \text{ and } \frac{d}{dx}y \cdot N := 0 \,(\text{for } y \neq x) \\ \frac{d}{dx}(\lambda y.P) \cdot N := \lambda y.\left(\frac{d}{dx}P \cdot N\right) \\ \frac{d}{dx}\left(\mathtt{D}P \bullet Q\right) \cdot N := \mathtt{D}\left(\frac{d}{dx}P \cdot N\right) \bullet Q + \mathtt{D}P \bullet \left(\frac{d}{dx}Q \cdot N\right)\end{gathered} \tag{3.2}$$

where we added the case for the variables, which is clear. The sum in the case of linear application is there to non-deterministically guess if S is P or Q.

The only remaining case is case 2 above, i.e. $M = PQ$ and $S = P$ or $S = Q$. If $S = P$ then S is in linear position for M, so we are already sure the computation will be linear. But if $S = Q$ the situation is more subtle: our term $\frac{d}{dx}M \cdot N$ has to behave similarly to $P(Q\{N/x_0\})$ with the only difference that, among all the times that P calls its argument, exactly one has to be linearly treated while the others are treated as P commands. Now, by definition of the linear application $\mathtt{D}(.) \bullet (.)$, this effect is precisely obtained by linearly applying P to $\frac{d}{dx}Q \cdot N$ and then apply the result to Q. That is, via the term $\left(\mathtt{D}P \bullet \left(\frac{d}{dx}Q \cdot N\right)\right) Q$. Using again formal sums to represent the non-deterministic guess of S being P or Q, for case 2 we obtain:

$$\frac{d}{dx}(PQ) \cdot N := \left(\frac{d}{dx}P \cdot N\right) Q + \left(\mathtt{D}P \bullet \left(\frac{d}{dx}Q \cdot N\right)\right) Q. \tag{3.3}$$

We can finally concretized all those considerations in the following:

Definition 3.2.2. *Fix a commutative semiring $\mathcal{R}$. The set of differential terms is the $\mathcal{R}$-module given by quotienting the words (modulo α-equivalence) inductively defined as:*

$$M ::= 0 \mid x \mid \lambda x.M \mid MM \mid (\mathtt{D}M) \bullet M \mid aM + bM \qquad (\text{for } x \text{ variable and } a, b \in \mathcal{R})$$

under the equations:

i. commutativity and associativity of $+$

ii. 0 neutral element for $+$ and annihilating element for all constructors except $+$

iii. linearity (w.r.t. $+$) of the constructors[3] $\lambda x.(.)$, $(.)M$ and $\mathtt{D}(.) \bullet (.)$

iv. usual module equations[4].

Differential terms are endowed with the reduction $\to$ given by the union[5] of the usual λ-reduction and the relation:

$$\mathtt{D}(\lambda x.M) \bullet N \to_\partial \lambda x.\left(\frac{d}{dx}M \cdot N\right)$$

where $\frac{d}{dx}M \cdot N$ is the differential λ-term inductively defined by equations (3.2) and (3.3)[6].

[3] Remark that since an application is not linear in the argument, we didn't add linearity of $M(.)$.

[4] That is: $a(M + N) = aM + aN$, $(a + b)M = aM + bM$, $(ab)M = a(bM)$ and $1M = M$.

[5] One has to be careful to define context closure in differential λ-calculus. But we are not concerned with technical details here.

[6] Plus the case $\frac{d}{dx}(aP + bQ) \cdot N := a\frac{d}{dx}P \cdot N + b\frac{d}{dx}Q \cdot N$ which comes from the fact that the linear occurrences of x in $aP + bQ$ are clearly those linear in P and those linear in Q. This corresponds again to the fact that summing functions is done pointwise.

The following three lemmas are a confirmation that we are, indeed, doing something related to analysis.

Lemma 3.2.3. *One can easily prove that if x does not occur in M then $\frac{d}{dx}M \cdot N = 0$. From the perspective of analysis this read as the fact that the derivative of a constant is null.*

Lemma 3.2.4. *If x does not occur in $\lambda y.P$, the definition of linear substitution gives:*

$$\frac{d}{dx}((\lambda y.P)Q) \cdot N = \left(\mathtt{D}(\lambda y.P) \bullet \left(\frac{d}{dx}Q \cdot N\right)\right) Q.$$

From the perspective of analysis, since the fact that x does not occur in $\lambda y.P$ entails that $(\lambda y.P)Q$ can be seen as the composition function "$(\lambda y.P) \circ Q$" of the independent variable x, this reads as:

$$\frac{d}{dx}((\lambda y.P) \circ Q) \cdot N = \frac{d}{dy}P\Big|_{y=Q} \cdot \left(\frac{d}{dx}Q \cdot N\right).$$

It is exactly the chain rule[7] *for the directional derivative of the composition of two functions*[8]*.*

Lemma 3.2.5. *If y does not occur in Q, one has:*

$$\frac{d}{dx}\left(\frac{d}{dy}M \cdot P\right) \cdot Q = \frac{d}{dy}\left(\frac{d}{dx}M \cdot Q\right) \cdot P + \frac{d}{dy}M \cdot \left(\frac{d}{dx}P \cdot Q\right)$$

In particular, if moreover the second summand of the previous equality is zero, a non trivial case for it to happen being if x does not occur in P [9]*, one has:*

$$\frac{d}{dx}\left(\frac{d}{dy}M \cdot P\right) \cdot Q = \frac{d}{dy}\left(\frac{d}{dx}M \cdot Q\right) \cdot P$$

From the perspective of analysis, the hypotheses say that P is independent from x and Q is independent from y, so we get $\frac{d}{dx}\left(\frac{d}{dy}M\right) \cdot P \cdot Q = \frac{d}{dy}\left(\frac{d}{dx}M\right) \cdot Q \cdot P$. Cancelling P and Q, we recover Schwarz's theorem on the symmetry of second derivatives.

Ehrhard and Regnier found in [ER03] that all this considerations do in fact make sense. Of course history is always different from the way one can present the topic years later, and they were not at all thinking of the above heuristics; they realized that the notion of derivative present in some denotational models of linear logic, based on vector spaces and built by Ehrhard some time before [Ehr02], could be internalized in the syntax. Their differential constructions are more general than the λ-calculus framework we quickly presented here, giving life to *differential linear logic* [Ehr16] and *differential interaction nets* [ER06b]. This calculus is confluent, strongly normalising on typable terms and admits interesting denotational models.

Taylor expansion Differential λ-calculus is mainly interesting because it provides the crucial tool of *Taylor expansion* of a program, which we describe below by carrying on our heuristic.

Put, for brevity, $\mathtt{D}^n M \bullet (N_1, \ldots, N_n) := \mathtt{D}(\mathtt{D} \cdots (\mathtt{D}M \bullet N_1) \bullet \cdots) \bullet N_{n-1}) \bullet N_n$ and also $\mathtt{D}^n M \bullet N^n := \mathtt{D}^n M \bullet (N, \ldots, N)$. So, in $\mathtt{D}^n M \bullet (N_1, \ldots, N_n)$, exactly n terms (the N_i's) are linearly passed to M, and the whole term is still ready to receive other arguments. Remark that, by definition of linear application, one has:

[7] With the notation of the first page: $d(f \circ g)(a) = df(g(a)) \circ dg(a)$.

[8] Or equivalently, but more simply, just cancel N and obtain the usual chain rule for 1-dimensional argument functions.

[9] The case y not free in the M is trivial because the equality would become $0 = 0 + 0$.

- if $n > \deg_x M$ then $\mathtt{D}^n(\lambda x.M) \bullet (N_1, \ldots, N_n) \to 0$;

- if $n < \deg_x M$ then $(\mathtt{D}^n(\lambda x.M) \bullet (N_1, \ldots, N_n))P$ will make $\deg_x M - n$ copies of P and put those copies in M at the place of the remaining $\deg_x M - n$ occurrences of x;

- if $\deg_x M = n$ then $\mathtt{D}^n(\lambda x.M) \bullet (N_1, \ldots, N_n)$ applied on P will just throw it away.

So if we wanted, for some reason, to force $\lambda x.M$ to use, linearly, exactly $\deg_x M$ times an argument N, we could simply consider $(\mathtt{D}^n(\lambda x.M) \bullet N^n)0$: the lines above precisely say that, in order for it to not annihilate to 0, the only possibility is that $n = \deg_x M$. Remember that we started our heuristics remarking that, in analysis, this is precisely the behaviour produced by Taylor expansion: it decomposes a function as the sum of what is obtained by forcing it to use exactly a fixed number n of times its argument, for all possible n.

The same should hold here: if we knew that in an ordinary application MN, the term M would use its argument N *exactly* n times, we could in advance provide n copies of N to M and now force it to treat them linearly, obtaining necessary the same behaviour of MN. On the other hand, an M applied to an N will certainly use N a *certain* number $n \in \mathbb{N}$ of times, so it should be the case that the behaviour of MN is a superposition of behaviours where M uses N exactly n times, for all the possible $n \in \mathbb{N}$. Moreover, from the point of view of analysis – say that M is a function of the independent variable x – we remark that $(\mathtt{D}^n M \bullet N^n)0$ reads as $\dfrac{d^n}{dx^n} M\Big|_{x=0} \cdot N^n$, which is just, modulo the factorial coefficient $\frac{1}{n!}$, the n-th summand of the Taylor expansion of $M(N)$ in 0.

In conclusion, if all this heuristics really makes sense, we should have:

$$MN = \sum_{n \in \mathbb{N}} \frac{1}{n!} (\mathtt{D}^n M \bullet N^n)0. \tag{3.4}$$

A first step in this direction in given in [ER03] where, after having properly defined the framework of differential λ-calculus, it is proved that:

Proposition 3.2.6. *If MN λ-reduces to a variable, then equation* (3.4) *holds (for $=_{\lambda\partial}$).*

What one would now really hope for, would be on one hand to eliminate the constraint of MN of reducing to a variable and, on the other hand, to be able to Taylor expand *all* the applications in a term: indeed, if the above formula holds for any application one could nest the expansions in order to get, at the end of the day, only linear applications and proving thus that any application can be really obtained via a sum of *purely differential* terms, just like in analysis. These terms could now be really seen as a kind of approximation.

That is, we are looking for a function Θ – the *quantitative* (or *full*) Taylor expansion – mapping λ-terms in a still to be specified space s.t. $\Theta(x) = x$, $\Theta(\lambda x.M) = \lambda x.\Theta(M)$ and:

$$\Theta(MN) = \sum_{n \in \mathbb{N}} \frac{1}{n!} (\mathtt{D}^n \Theta(M) \bullet \Theta(N)^n)0.$$

The difficulty is that since Θ must thus be a (possibly infinite) sum, we have to make the codomain space able to sum series, multiplicate them, admit the same language's constructors etc. In [ER08], Ehrhard and Regnier find such a space to be the module $\mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$ of formal linear combinations, with scalars in a semiring $\mathcal{R}$, of some terms in a language Λ^{r} similar to ordinary λ-calculus but with stronger properties, called *resource calculus*. It is a very simple confluent and strongly normalising calculus (also in the untyped case) – with a lot of similarities to Boudol's λ-calculus with multiplicities [Bou93] – in which resource consumption is strictly controlled and the vast majority of its "resource-terms" annihilates for lack or abundance of resources to be consumed. So the previous recursive equations for Θ do indeed define a function

$\Theta : \Lambda \to \mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$, the (qualitative) *Taylor expansion* and moreover this function satisfies a direct characterization:

$$\Theta(M) = \sum_{t\in\mathcal{T}(M)} \frac{1}{m(t)} t.$$

Here $m(t) \in \mathbb{N}$ is a certain number which has to do with the combinatorics of some groups handling permutation of resources, and $\mathcal{T} : \Lambda \to \mathcal{P}(\Lambda^{\mathrm{r}})$ is an easily inductively defined function, called the *qualitative Taylor expansion*. This function associates a λ-term with the set of resource-terms which "try to act as it" but in the resource-sensitive world of Λ^{r}, and will be the major object of study in this thesis.

Let us conclude with two final remarks.

First, $\mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$ is endowed with a structure of algebra over $\mathcal{R}$ by means of a product which is the natural notion of multiplication of series; one defines then the constructor $(\mathtt{D}^n(.) \bullet (.)^n)0$ in $\mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$ as: $(\mathtt{D}^n\vartheta_1 \bullet \vartheta_2^n)0 := \vartheta_1\vartheta_2^n$ (we mean that the former ϑ_2^n is part of the *name* of the constructor we are defining, while the latter is the n-th power of ϑ_2 via the product of $\mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$). Therefore now one has:

$$\Theta(MN) = \sum_{n\in\mathbb{N}} \frac{1}{n!}\Theta(M)\Theta(N)^n.$$

Since $\Theta(M)$ in the above sum does not depend on n, we define the map $(.)^! : \mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty \to \mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$ given by $\vartheta^! := \sum_{n\in\mathbb{N}} \frac{\vartheta^n}{n!}$ and obtain:

$$\Theta(MN) = \Theta(M)\Theta(N)^!. \tag{3.5}$$

The map $(.)^!$ does not only look like the exponential power series, but it really is an exponential, as it satisfies in $\mathcal{R}\langle\Lambda^{\mathrm{r}}\rangle_\infty$ the characteristic equations: $0^! = 1$ and $(\vartheta_1 + \vartheta_2)^! = \vartheta_1^!\vartheta_2^!$. Reading equation (3.5) from a logical point of view, one recovers the fundamental decomposition of linear logic $A \to B = (!A) \multimap B$. In fact, the map $(.)^!$ corresponds to the *promotion rule* of linear logic.

Finally, let us remark that in [ER06a] it is proven a *commutation formula*:

$$\mathrm{NF}(\Theta(M)) = \Theta(\mathrm{BT}(M))$$

for a natural definition of the normal form of the full Taylor expansion and the full Taylor expansion of Böhm trees. The easier "qualitative" version (i.e. without coefficients) of this formula still holds:

$$\mathrm{NF}(\mathcal{T}(M)) = \mathcal{T}(\mathrm{BT}(M)).$$

It will be a central tool in Section 3.5 and we will give an alternative proof of it (Theorem 3.5.6).

3.3 The qualitative Taylor expansion for λ-calculus

Notation 3.3.1. *From now on, we will simply write $M \to N$ instead of $M \to_\lambda N$. We will as well omit the λ in $\twoheadrightarrow_\lambda$. On the contrary, for any other reduction rule we* will *precise it, as for instance in the head-reduction $M \to_{\mathrm{h}} N$.*

Definition 3.3.2 (Resource λ-terms)**.** *The set Λ^{r} of* resource λ-terms *is defined by induction as:*

$$t ::= x \mid \lambda x.t \mid t[\overbrace{t,\dots,t}^{n}] \qquad \text{for } n \in \mathbb{N}.$$

Resource terms are subject to all the usual syntactical precautions, namely α-equivalence and Barendregt convention for the writing. We call bag *a multiset of resource terms. We call*

$\deg_x(t)$ *the* degree *of x in t, that is, the number of (free) occurrences of x in t. We define multi-hole resource contexts as expected.*

Remark 3.3.3. *In the following it will happen to use the size* $\mathbf{sz}(t) \in \mathbb{N}_{\geq 1}$ *of a resource λ-term t, which is inductively defined as:* $\mathbf{sz}(x) := 1$, $\mathbf{sz}(\lambda x.t) := 1 + \mathbf{sz}(t)$, $\mathbf{sz}(t_0[t_1, \ldots, t_k]) := 1 + k + \sum_{i=0}^{k} \mathbf{sz}(t_i)$.

We consider the extension of this syntax to sums, by means of the free $\mathbb{Z}_2$-module $2\langle\Lambda^{\mathrm{r}}\rangle$ constructed as described in Section 2.1. The application and the λ-abstraction constructors are extended, by linearity, on all sums. In particular, multihole-resource-contexts are always linear when seen as functions on $2\langle\Lambda^{\mathrm{r}}\rangle$.

Definition 3.3.4. *1. For $t, t_1, \ldots, t_k \in \Lambda^{\mathrm{r}}$, we define the sum $t\langle[t_1, \ldots, t_k]/x\rangle \in 2\langle\Lambda^{\mathrm{r}}\rangle$, called the* linear substitution*, as:*

$$t\langle[t_1, \ldots, t_k]/x\rangle := \begin{cases} \sum_{\sigma \in \mathfrak{S}_k} t\{t_{\sigma(1)}/x^1, \ldots, t_{\sigma(k)}/x^k\} & \text{if } k = \deg_x(t) \\ 0 & \text{otherwise} \end{cases}$$

where $\mathfrak{S}_n$ is the symmetric group over n elements, and with x^i we denote the i^{th} occurrence of x in t. This operation is well-defined as the sum makes it independent from the choice of the enumeration of the occurrences of x in t. An equivalent inductive way of defining the operation is given by Figure 3.1[10]*.*

2. This calculus is endowed with the resource λ-reduction $\to_{\mathrm{r}} \subseteq \Lambda^{\mathrm{r}} \times 2\langle\Lambda^{\mathrm{r}}\rangle$ *defined as the contextual closure of the relation:*

$$(\lambda x.t)[\vec{u}] \to_{\mathrm{r}} t\langle[\vec{u}]/x\rangle.$$

This relation is extended to a relation $\to_{\mathrm{r}} \subseteq 2\langle\Lambda^{\mathrm{r}}\rangle \times 2\langle\Lambda^{\mathrm{r}}\rangle$ as described in the preliminaries.

$$\begin{array}{rcl} x\langle[v]/x\rangle = v & y\langle 1/x\rangle = y & \\ x\langle 1/x\rangle = x\langle[v, w, \vec{u}]/x\rangle = 0 & y\langle[v, \vec{u}]/x\rangle = 0 & (\lambda y.t)\langle[\vec{u}]/x\rangle = \lambda y.t\langle[\vec{u}]/x\rangle \\ (t[v_1, \ldots, v_n])\langle[\vec{u}]/x\rangle = & \sum_{([\vec{w}^{\,0}], \ldots, [\vec{w}^{\,n}])\ \text{w.c. of } [\vec{u}]} & t\langle[\vec{w}^{\,0}]/x\rangle \left[\, v_1\langle[\vec{w}^{\,1}]/x\rangle, \ldots, v_n\langle[\vec{w}^{\,n}]/x\rangle \,\right] \end{array}$$

Figure 3.1: Inductive characterisation of linear substitution

The application of function to a bag, as in $(\lambda x.s)[t_1, \ldots, t_k]$, should be understood as follows: the program s during its execution must perform n "calls" to its argument x, one for each occurrence of x in s, each time receiving as value a different t_i non-deterministically chosen from the bag (in the reduction we consider a formal sum of all possibilities). In case of a mismatch $n \neq k$, s "raises an exception" annihilating the whole term, so the outcome of the computation is the empty sum 0. This is precisely the meaning of the following reduction $\to_{\mathrm{r}}$.

It is well known that:

[10] In the figure we refer to the notion of "weak composition" ("w.c." in the figure) of a bag; such notion is defined in Definition 4.4.3.

Proposition 3.3.5. *The reduction* $\to_{\mathrm{r}} \subseteq 2\langle\Lambda^{\mathrm{r}}\rangle \times 2\langle\Lambda^{\mathrm{r}}\rangle$ *is confluent and strongly normalising. In particular, every sum* $\mathbb{T} \in 2\langle\Lambda^{\mathrm{r}}\rangle$ *admits exactly one normal form* $\mathrm{nf}_{\mathrm{r}}(\mathbb{T}) \in 2\langle\Lambda^{\mathrm{r}}\rangle$.

The strong normalization is easy, since as there is no duplication and contracting a redex eliminates exactly one abstraction, the size $\mathbf{sz}(\cdot)$ is a strictly decreasing measure. Moreover, one can check that the resource calculus is actually locally confluent, thus confluence follows from strong normalization by Newman's Lemma.

Remark 3.3.6. *Resource λ-terms are very similar, in the syntax, to usual λ-terms. They also have all shape* $\lambda\vec{x}.s[\vec{u}^{\,1}]\cdots[\vec{u}^{\,n}]$, *with* s *either a* r*-redex or a variable. In particular, analogously to λ-normal λ-terms, the* r*-normal resource terms obey to the following inductive grammar:*

$$\lambda\vec{x}.y \;\mid\; \lambda\vec{x}.y[\vec{u}^{1}]\cdots[\vec{u}^{n}] \qquad \text{where the } u^{i}_{j}\text{'s are r-normal.}$$

It will happen (in particular in Section 3.5), to "go by induction on the r*-normal structure of a resource term" (which we already know to be* r*-normal). By that we simply mean that we are going by induction on the above inductive grammar of* r*-normal resource terms. We will denote the set of all the* r*-normal resource terms by* $\mathrm{nf}_{\mathrm{r}}(\Lambda^{\mathrm{r}})$.

Notation 3.3.7. *We will often use the notation above for indices on bags:* $[\vec{u}^{\,i}]$ *denotes some bag whose elements are denoted by* $u^{i}_{1},\ldots,u^{i}_{k}$ *(for some* k*).*

Remember that the reduction on sums is not "well-behaved" with respect to $+$ (see last lines of Section 2.1). The following is a special case in which we can reduce in all the addends, eliminating the terms which "go to zero".

Remark 3.3.8. *Suppose that* $\mathbb{T},\mathbb{S} \in 2\langle\Lambda^{\mathrm{r}}\rangle$ *are s.t.* $\mathrm{nf}_{\mathrm{r}}(s) = 0$ *for all* $s \in \mathbb{S}$ *and* $\mathrm{nf}_{\mathrm{r}}(t) \neq 0$ *for all* $t \in \mathbb{T}$. *Then we have:* $\mathbb{T}+\mathbb{S} \twoheadrightarrow_{\mathrm{r}} \mathbb{T}$.

In fact, say $\mathbb{S} = s_0 + \mathbb{S}_0$ *(if* $\mathbb{S} = 0$ *there is nothing to prove). Say* $s_0 \to_{\mathrm{r}} \mathbb{S}'_0$. *Then we are allowed to reduce* s_0 *in* $\mathbb{T}+\mathbb{S}$, *because* s_0 *is different from all the other addends of the sum (it is not in* $\mathbb{S}$ *by definition, and it is not in* $\mathbb{T}$ *because* $\mathrm{nf}_{\to_{\mathrm{r}}}(s_0) = 0$*). So* $\mathbb{T}+\mathbb{S} \to_{\mathrm{r}} \mathbb{T}+\mathbb{S}'_0+\mathbb{S}_0 = \mathbb{T}+\widetilde{\mathbb{S}}'_0+\mathbb{S}_0$, *where* $\widetilde{\mathbb{S}}'_0 \subseteq \mathbb{S}'_0$ *and* $\widetilde{\mathbb{S}}'_0+\mathbb{S}_0$ *does* not *contain duplicates. This is due to the fact that in* $\mathbb{S}'_0$ *we may have created elements already in* $\mathbb{S}_0$, *so* $\widetilde{\mathbb{S}}'_0$ *is taking into account the erasing of such duplicates. Now we have two cases:*

1- either $\widetilde{\mathbb{S}}'_0 = 0$, *in which case we have* $\mathbb{T}+s_0+\mathbb{S} \to_{\mathrm{r}} \mathbb{T}+\mathbb{S}_0$;

2- or $\widetilde{\mathbb{S}}'_0 \neq 0$, *say* $\widetilde{\mathbb{S}}'_0 = s_1 + \mathbb{S}_1$. *In this last case we remark that the size of* s_1 *is strictly smaller of that of* s, *and that* s_1 *is different from all the elements of* $\mathbb{T}+\mathbb{S}_1+\mathbb{S}_0$. *Thus, if* $s_1 \to_{\mathrm{r}} \mathbb{S}'_1$, *we can reduce* $\mathbb{T}+\widetilde{\mathbb{S}}'_0+\mathbb{S}_0 = \mathbb{T}+s_1+\mathbb{S}_1+\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{T}+\mathbb{S}'_1+\mathbb{S}_1+\mathbb{S}_0$. *Now we can apply the same argument as before (ereasing the duplicates) and we have again the two cases 1 or 2. However, we cannot stay forever in case 2, because the (well-founded) size is stricly decreasing. Thus at a certain iteration we must fall into case 1. But if we fall into case 1, then we are done, since we have "eliminated* s_0*" from* $\mathbb{S}$ *and we can start the proof again*[11] *knowing that there is only a finite number of elements in* $\mathbb{S}$.

The following lemma gives another case in which we can consecutively reduce sums.

Lemma 3.3.9. *1. Let* c *be a single-hole context.*

(a) Let $\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{S}_1$. *Then:* $c(\!|\mathbb{S}_0|\!) \to_{\mathrm{r}} c(\!|\mathbb{S}_1|\!)$.

(b) Let $\mathbb{S}_0 \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_1$. *Then:* $c(\!|\mathbb{S}_0|\!) \twoheadrightarrow_{\mathrm{r}} c(\!|\mathbb{S}_1|\!)$.

2. Let $c = c(\!|\xi_1,\xi_2|\!)$ *be a 2-context s.t.* $\deg_{\xi_1}(c) = \deg_{\xi_2}(c) = 1$, *and* $\mathbb{T}$ *be a sum.*

[11] A more rigorous proof would require an induction on the number of elements of $\mathbb{S}$ and on the size of a sum, but we preferred to leave the argument "non-inductive".

(a) Let $\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{S}_1$. *Then:* $c(\!|\mathbb{T}, \mathbb{S}_0|\!) \twoheadrightarrow_{\mathrm{r}} c(\!|\mathbb{T}, \mathbb{S}_1|\!)$ *(and the same for* $c(\!|\mathbb{S}_0, \mathbb{T}|\!)$*).*

(b) Let $\mathbb{S}_0 \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_1$. *Then:* $c(\!|\mathbb{T}, \mathbb{S}_0|\!) \twoheadrightarrow_{\mathrm{r}} c(\!|\mathbb{T}, \mathbb{S}_1|\!)$ *(and the same for* $c(\!|\mathbb{S}_0, \mathbb{T}|\!)$*).*

(c) Let $\mathbb{S}_0 \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_1$ *and* $\mathbb{T} \twoheadrightarrow_{\mathrm{r}} \mathbb{T}_1$. *Then:* $c(\!|\mathbb{T}, \mathbb{S}_0|\!) \twoheadrightarrow_{\mathrm{r}} c(\!|\mathbb{T}_1, \mathbb{S}_1|\!)$.
It is easy to lift this property to the case of more than two holes (but always with degree 1*).*

Proof. (1a). If $\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{S}_1$ then $\mathbb{S}_0 = s + \widetilde{\mathbb{S}}$, $\mathbb{S}_1 = \mathbb{S}' + \widetilde{\mathbb{S}}$, with $s \to_{\mathrm{r}} \mathbb{S}'$ and $s \notin \widetilde{\mathbb{S}}$. So $c(\!|s|\!) \notin c(\!|\widetilde{\mathbb{S}}|\!)$. Remark that there may be duplicates in $\mathbb{S}' + \widetilde{\mathbb{S}}$. Now, since the reduction is contextual, we have $c(\!|s|\!) \to_{\mathrm{r}} c(\!|\mathbb{S}'|\!)$. So $c(\!|\mathbb{S}_0|\!) = c(\!|s|\!) + c(\!|\widetilde{\mathbb{S}}|\!) \to_{\mathrm{r}} c(\!|\mathbb{S}'|\!) + c(\!|\widetilde{\mathbb{S}}|\!) = c(\!|\mathbb{S}_1|\!)$, where the last equality is easily seen to hold.

(1b). Straightforward induction on the number $n \in \mathbb{N}$ s.t. $\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{S}'_1 \to_{\mathrm{r}} \cdots \to_{\mathrm{r}} \mathbb{S}'_{n-2} \to_{\mathrm{r}} \mathbb{S}_1$, using point (1)(a).

(2a). Since $c(\!|t, \xi|\!)$ is a single-hole context, if $\mathbb{S}_0 \to_{\mathrm{r}} \mathbb{S}_1$ then by point (1a) we have $c(\!|t, \mathbb{S}_0|\!) \to_{\mathrm{r}} c(\!|t, \mathbb{S}_1|\!)$ for all $t \in \mathbb{T}$. Let $\mathbb{T} =: \sum_{i=0}^{k} t_i$. Since $c(\!|t_0, \mathbb{S}_0|\!) \neq c(\!|t_i, \mathbb{S}_0|\!)$ for all $i \neq 0$, we have: $c(\!|\mathbb{T}, \mathbb{S}_0|\!) = \sum_{i=0}^{k} c(\!|t_i, \mathbb{S}_0|\!) \to_{\mathrm{r}} c(\!|t_0, \mathbb{S}_1|\!) + \sum_{i=1}^{k} c(\!|t_i, \mathbb{S}_0|\!)$. But $c(\!|t_0, \mathbb{S}_1|\!) \neq c(\!|t_1, \mathbb{S}_0|\!) \neq c(\!|t_i, \mathbb{S}_0|\!)$ for all $i \neq 1$, so we have: $c(\!|t_0, \mathbb{S}_1|\!) + \sum_{i=1}^{k} c(\!|t_i, \mathbb{S}_0|\!) \to_{\mathrm{r}} c(\!|t_0, \mathbb{S}_1|\!) + c(\!|t_1, \mathbb{S}_1|\!) + \sum_{i=2}^{k} c(\!|t_i, \mathbb{S}_0|\!)$. We can keep reducing in this way all the k elements of $\mathbb{T}$, and we obtain as final sum $\sum_{i=0}^{k} c(\!|t_i, \mathbb{S}_1|\!) = c(\!|\mathbb{T}, \mathbb{S}_1|\!)$, so we are done.

(2b). As in point (1)(b).

(2c). Immediate by point (2b).

□

We will not explicit the use of the previous lemma.

Definition 3.3.10 (Qualitative Taylor expansion)**.** *The* (qualitative) Taylor expansion *of a term is the map* $\mathcal{T} : \Lambda \to \mathscr{P}^*(\Lambda^{\mathrm{r}})$ *inductively defined as:*

$$\mathcal{T}(x) := \{x\}$$

$$\mathcal{T}(\lambda x.M) := \{\lambda x.t \mid t \in \mathcal{T}(M)\}$$

$$\mathcal{T}(MN) := \{t[\vec{u}] \mid t \in \mathcal{T}(M) \text{ and } [\vec{u}] \in !\,\mathcal{T}(N)\}.$$

For $M \in \Lambda$ *one defines the* (qualitative) Taylor normal form *as the possibly empty set:*

$$\mathrm{NF}\mathcal{T}(M) := \bigcup_{t \in \mathcal{T}(M)} \mathrm{nf}_{\mathrm{r}}(t) \subseteq \mathscr{P}(\Lambda^{\mathrm{r}}).$$

The inclusion of Taylor normal forms endows Λ *with a* partial preorder $\leq$*, whose induced equivalence* $=_\tau$ *on* Λ *is given by the equality of Taylor normal forms.*

Example 3.3.11. *It is well known that* $\mathrm{NF}\mathcal{T}(\Omega) = \emptyset$. *Let us compute it as an example.*

For $n \in \mathbb{N}$, *let us set, in this example,* $\Delta_n := \lambda x.x[\overbrace{x, \ldots, x}^{n}]$. *Now we have:* $\mathcal{T}(\Omega) = \{\Delta_{n_0}[\Delta_{n_1}, \ldots, \Delta_{n_k}] \mid k, n_0, \ldots, n_k \in \mathbb{N}\}$, *and it is immediate to show, by induction on* $k \in \mathbb{N}$, *that* $\mathrm{nf}_{\mathrm{r}}(\Delta_{n_0}[\Delta_{n_1}, \ldots, \Delta_{n_k}]) = 0$ *for any generic such term. In fact, by definition of* Δ_{n_0}, *such element reduces to* 0 *whenever* $k \neq n_0 + 1$, *and otherwise, that is if* $k = n_0 + 1 \geq 1$, *it reduces to* $\Delta_{n_{\sigma(1)}}[\Delta_{n_{\sigma(2)}}, \ldots, \Delta_{n_{\sigma(k)}}]$, *for some permutation* σ *on* k *elements. But now we are done by the inductive hypothesis on this latter term, which has* $k - 1 < k$ *elements in the bag.*

Lemma 3.3.12. *If* $\lambda x_1 \dots \lambda x_n.xM_1 \cdots M_k =_\tau \lambda x_1 \dots \lambda x_{n'}.yN_1 \cdots N_{k'}$ *then* $n = n'$, $k = k'$, $x = y$ *and* $M_i =_\tau N_i$ *for all* $i = 1, \dots, k$.

Proof. This easily follows from the definitions. Indeed, since:

$$\mathrm{NF}\mathcal{T}(\lambda x_1 \dots \lambda x_n.xM_1 \cdots M_k) = \lambda x_1 \dots \lambda x_n.x\,!\,\mathrm{NF}\mathcal{T}(M_1) \cdots !\,\mathrm{NF}\mathcal{T}(M_k)$$

and the same for $\lambda x_1 \dots \lambda x_{n'}.yN_1 \cdots N_{k'}$, one has $n = n'$, $k = k'$ and $x = y$. In order to show $M_i =_\tau N_i$ it suffices to show $M_i \leq N_i$, the result being symmetrical. To that end, take $t \in \mathrm{NF}\mathcal{T}(M_i)$, i.e. $[t] \in !\,\mathrm{NF}\mathcal{T}(M_i)$. Thus:

$$\begin{array}{rcl} \lambda\vec{x}.x1 \cdots 1[t]1 \cdots 1 \in \mathrm{NF}\mathcal{T}(\lambda\vec{x}.xM_1 \cdots M_k) & = & \mathrm{NF}\mathcal{T}(\lambda\vec{x}.xN_1 \cdots N_k) \\ & = & \lambda\vec{x}.x\,!\,\mathrm{NF}\mathcal{T}(N_1) \cdots !\,\mathrm{NF}\mathcal{T}(N_k), \end{array}$$

whence $[t] \in !\,\mathrm{NF}\mathcal{T}(N_i)$, i.e. $t \in \mathrm{NF}\mathcal{T}(N_i)$. □

The first important result is the following:

Theorem 3.3.13 (Monotonicity property)**.** *Any context* C *is monotone w.r.t.* $\leq$.

Proof. We have to prove that for all $M, N \in \Lambda$ s.t. $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N)$, one has $\mathrm{NF}\mathcal{T}(C(\!|M|\!)) \subseteq \mathrm{NF}\mathcal{T}(C(\!|N|\!))$. We go by induction on C.

If $C = \xi$ then $\mathrm{NF}\mathcal{T}(C(\!|M|\!)) = \mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(C(\!|N|\!))$.

If $C = x$ then $\mathrm{NF}\mathcal{T}(C(\!|M|\!)) = \{x\} = \mathrm{NF}\mathcal{T}(C(\!|N|\!))$.

If $C = \lambda x.C'$ then:

$$\mathrm{NF}\mathcal{T}(C(\!|M|\!)) = \{\lambda x.t \mid t \in \mathrm{NF}\mathcal{T}(C'(\!|M|\!))\} \subseteq \{\lambda x.t \mid t \in \mathrm{NF}\mathcal{T}(C'(\!|N|\!))\} = \mathrm{NF}\mathcal{T}(C(\!|N|\!)).$$

If $C = C'C''$ then:

$$\begin{array}{rcl} \mathrm{NF}\mathcal{T}(C(\!|M|\!)) & = & \displaystyle\bigcup_{t \in \mathrm{NF}\mathcal{T}(C'(\!|M|\!))} \bigcup_{[\vec{u}] \in !\,\mathrm{NF}\mathcal{T}(C''(\!|M|\!))} \mathrm{nf_r}(t[\vec{u}]) \\ & \subseteq & \displaystyle\bigcup_{t \in \mathrm{NF}\mathcal{T}(C'(\!|N|\!))} \bigcup_{[\vec{u}] \in !\,\mathrm{NF}\mathcal{T}(C''(\!|N|\!))} \mathrm{nf_r}(t[\vec{u}]) \\ & = & \mathrm{NF}\mathcal{T}(C(\!|N|\!)). \end{array}$$

□

Monotonicity also holds for multi-hole contexts. That is: if $M_1 \leq N_1, \dots, M_n \leq N_n$ then $C(\!|M_1, \dots, M_n|\!) \leq C(\!|N_1, \dots, N_n|\!)$ for all k-context C. This can be directly proved simply by considering a k-context C and carrying on the exact same proof as above. Or also, by considering the contexts $C_i(\!|\xi|\!) := C(\!|N_1, \dots, N_{i-1}, \xi, M_{i+1}, \dots, M_n|\!)$ (for $i = 1, \dots, n$), and remarking that, by Monotonicity on contexts, one has: $C(\!|M_1, \dots, M_n|\!) \leq C(\!|N_1, M_2, \dots, M_n|\!) \leq \cdots \leq C(\!|N_1, \dots, N_{n-1}, M_n|\!) \leq C(\!|N_1, \dots, N_n|\!)$.

Monotonicity immediately implies that $C(\!|U|\!)$ is a minimal element w.r.t $\leq$, for any $U =_\tau \Omega$. But we can say more:

Corollary 3.3.14. *1. The equivalence* $=_\tau$ *is contextual. That is, multi-hole contexts are well defined functions on* $\Lambda/=_\tau$.

2. In $\Lambda/=_\tau$, if $\Omega \in \mathrm{Im}(C)$[12] then Ω is a fixed-point of C.

Proof. 1. It is immediate from Theorem 3.3.13 since $=_\tau$ is the induced equivalence of $\leq$.

2. As we already remarked, any $U \in \Lambda$ with $\mathrm{NF}(\mathcal{T}(U)) = \emptyset$ is minimal w.r.t $\leq$, that is, $\mathrm{NF}\mathcal{T}(C(\!|M|\!)) \supseteq \mathrm{NF}\mathcal{T}(C(\!|U|\!))$ for all $M \in \Lambda$. So $\mathrm{NF}\mathcal{T}(C(\!|U|\!)) \neq \emptyset$ entails $\mathrm{NF}\mathcal{T}(C(\!|M|\!)) \neq \emptyset$. Now reading the contrapositive of what we just obtained, we obtain point (2).

□

Remark 3.3.15. *One can easily check that for all $M \in \Lambda$,*

(1) M is normal iff $\mathrm{NF}(\mathcal{T}(M)) = \mathcal{T}(M)$ iff $\mathcal{T}(M) \subseteq \mathrm{nf}_{\mathrm{r}}(\Lambda^{\mathrm{r}})$.

(2) $\mathcal{T}(M) \cap \mathrm{nf}_{\mathrm{r}}(\Lambda^{\mathrm{r}}) \subseteq \mathrm{NF}(\mathcal{T}(M))$.

(3) If $M \twoheadrightarrow N$ then $\mathcal{T}(M) \cap \mathrm{nf}_{\mathrm{r}}(\Lambda^{\mathrm{r}}) \subseteq \mathcal{T}(N)$.

The following lemma says that Taylor expansion behaves well w.r.t substitutions.

Lemma 3.3.16. *Let $M, N \in \Lambda$ and x be a variable. We have:*

$$\mathcal{T}(M\{N/x\}) = \bigcup_{t \in \mathcal{T}(M)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} t\langle [\vec{u}]/x \rangle.$$

Proof. Case $M = x$:

$$\mathcal{T}(M\{N/x\}) = \mathcal{T}(N) = \{u \mid u \in \mathcal{T}(N)\} = \bigcup_{u \in \mathcal{T}(N)} x\langle [u]/x \rangle = \bigcup_{t \in \mathcal{T}(M)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} t\langle [\vec{u}]/x \rangle.$$

Case $M = y \neq x$:

$$\mathcal{T}(M\{N/x\}) = \mathcal{T}(M) = \{y\} = y\langle 1/x \rangle = \bigcup_{t \in \mathcal{T}(M)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} t\langle [\vec{u}]/x \rangle.$$

Case $M = \lambda y.P$:

$$\begin{array}{rcll} \mathcal{T}(M\{N/x\}) & = & \mathcal{T}(\lambda x.\, P\{N/x\}) & \\ & = & \{\lambda x.s \mid s \in \mathcal{T}(P\{N/x\})\} & \\ & = & \bigcup_{p \in \mathcal{T}(P)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} \lambda x.\, p\langle [\vec{u}]/x \rangle & \textit{(by inductive hypothesis)} \\ & = & \bigcup_{p \in \mathcal{T}(P)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} (\lambda x.p)\langle [\vec{u}]/x \rangle & \\ & = & \bigcup_{t \in \mathcal{T}(M)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} t\langle [\vec{u}]/x \rangle. & \end{array}$$

Case $M = PQ$:

$$\begin{array}{rcl} \mathcal{T}(M\{N/x\}) & = & \bigcup_{n \in \mathbb{N}} \bigcup_{p \in \mathcal{T}(P)} \bigcup_{[\vec{q}] \in\, !\mathcal{T}(Q)} \bigcup_{[\vec{s}^0], \ldots, [\vec{s}^n] \in\, !\mathcal{T}(N)} \left\{ v[w_1, \ldots, w_n] \mid v \in p\langle [\vec{s}^0]/x \rangle,\ w_i \in q_i\langle [\vec{s}^i]/x \rangle \right\} \\ & & \textit{(by inductive hypothesis)} \\ & = & \bigcup_{n \in \mathbb{N}} \bigcup_{p \in \mathcal{T}(P)} \bigcup_{[\vec{q}] \in\, !\mathcal{T}(Q)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} \sum_{\substack{([\vec{s}^0], \ldots, [\vec{s}^n]) \\ \textit{w.c. of } [\vec{u}]}} p\langle [\vec{s}^0]/x \rangle [q_1\langle [\vec{s}^1]/x \rangle, \ldots, q_n\langle [\vec{s}^n]/x \rangle] \\ & = & \bigcup_{p \in \mathcal{T}(P)} \bigcup_{[\vec{q}] \in\, !\mathcal{T}(Q)} \bigcup_{[\vec{u}] \in\, !\mathcal{T}(N)} (p[\vec{q}])\langle [\vec{u}]/x \rangle \qquad \textit{(by Figure 3.1)} \\ & = & \bigcup_{t \in \mathcal{T}(M)} \bigcup_{\vec{u} \in\, !\mathcal{T}(N)} t\langle [\vec{u}]/x \rangle. \end{array}$$

□

[12] As usual, $\mathrm{Im}(C)$ means here the image of the context function C.

The following property is important. Not only one needs it in many proofs, but also it is, in some sense, the prototypical property a "notion of approximation" should satisfy (see the end of Section 3.6).

Proposition 3.3.17. *Let $M, N \in \Lambda$. If $M \to N$ then the following holds:*

1. *For all $s \in \mathcal{T}(M)$, $s \twoheadrightarrow_{\mathrm{r}} \mathbb{T}$ for some sum $\mathbb{T} \subseteq \mathcal{T}(N)$.*
2. *For all $s' \in \mathcal{T}(N)$, there exist $s \in \mathcal{T}(M)$ such that $s \twoheadrightarrow_{\mathrm{r}} s' + \mathbb{T}$ for some sum $\mathbb{T} \subseteq \mathcal{T}(N)$.*

Proof. (1) and (2). We can write $M = C(\!|(\lambda x.P)Q|\!)$ and $N = C(\!|P\{Q/x\}|\!)$, for some $P, Q \in \Lambda$, x variable and $C = C(\!|\xi|\!)$ a single-hole context. Now go by induction on C. In the case $C = \xi$, both (1) and (2) are straightforward using Lemma 3.3.16. The only interesting case is $C = DC'$.

(1). Take $s \in \mathcal{T}(M)$. Then $s = s'[s_1, \dots, s_k]$ for some $s' \in \mathcal{T}(D)$ and $s_1, \dots, s_k \in \mathcal{T}(C'(\!|(\lambda x.P)Q|\!))$. By inductive hypothesis we have that for all $i = 1, \dots, k$, there is a $\mathbb{T}_i \subseteq \mathcal{T}(C'(\!|P\{Q/x\}|\!))$ such that $s_i \twoheadrightarrow_{\mathrm{r}} \mathbb{T}_i$. So $s = s'[s_1, \dots, s_k] \twoheadrightarrow_{\mathrm{r}} s'[\mathbb{T}_1, \dots, \mathbb{T}_k] = \sum_{s'_i \in \mathbb{T}_i} s'[s'_1, \dots, s'_k] =: \mathbb{T}$, and since each $s'[s'_1, \dots, s'_k] \in \mathcal{T}(D\,C'(\!|P\{Q/x\}|\!)) = \mathcal{T}(N)$, we are done.

(2). For (2), take $s' \in \mathcal{T}(N)$. Then $s' = s''[s_1, \dots, s_k]$ for some $s'' \in \mathcal{T}(D)$ and $s_1, \dots, s_k \in \mathcal{T}(C'(\!|P\{Q/x\}|\!))$. By inductive hypothesis we have that for all $i = 1, \dots, k$, there is an $\widetilde{s_i} \in \mathcal{T}(C'(\!|(\lambda x.P)Q|\!))$ such that $\widetilde{s_i} \twoheadrightarrow_{\mathrm{r}} s_i + \mathbb{T}_i$ for some sum $\mathbb{T}_i \subseteq \mathcal{T}(C'(\!|P\{Q/x\}|\!))$. So $\mathcal{T}(M) = \mathcal{T}(D\,C'(\!|(\lambda x.P)Q|\!)) \ni s := s''[\widetilde{s_1}, \dots, \widetilde{s_k}] \twoheadrightarrow_{\mathrm{r}} s''[s_1, \dots, s_k] + \mathbb{T} = s' + \mathbb{T}$ for some sum $\mathbb{T} \subseteq \mathcal{T}(N)$, and we are done. □

Corollary 3.3.18. *The equivalence $=_\tau$ is a λ-theory.*

Proof. We already know that it is contextual (Corollary 3.3.14), so it is enough to show that if $M \to N$ then $\mathrm{NF}\mathcal{T}(M) = \mathrm{NF}\mathcal{T}(N)$. Let us do the two inclusions.

(⊆). If $t \in \mathrm{NF}\mathcal{T}(M)$ then $t \in \mathrm{nf}_{\mathrm{r}}(s)$, for an $s \in \mathcal{T}(M)$. By Proposition 3.3.17 there is $\mathbb{S} \subseteq \mathcal{T}(N)$ s.t. $s \twoheadrightarrow_{\mathrm{r}} \mathbb{S}$. But by confluence and being t normal, it must be $t \in \mathrm{nf}_{\mathrm{r}}(\mathbb{S})$. In particular this means that $t \in \mathrm{nf}_{\mathrm{r}}(t')$ for some $t' \in \mathbb{S}$. Thus, $t \in \mathrm{NF}\mathcal{T}(N)$.

(⊇). If $t \in \mathrm{NF}\mathcal{T}(N)$ then $t \in \mathrm{nf}_{\mathrm{r}}(s')$, for an $s' \in \mathcal{T}(N)$. By Proposition 3.3.17 there is $s \in \mathcal{T}(M)$ s.t. $s \twoheadrightarrow_{\mathrm{r}} s' + \mathbb{T}$ for a sum $\mathbb{T}$. But by confluence this means $t \in \mathrm{nf}_{\mathrm{r}}(s)$. Thus, $t \in \mathrm{NF}\mathcal{T}(M)$.

□

Remark 3.3.19. *One can easily check that for all $M \in \Lambda$ normalizable, from remark 3.3.15(1) and Corollary 3.3.18 we obtain the commutation: $\mathcal{T}(\mathrm{nf}_\lambda(M)) = \mathrm{NF}(\mathcal{T}(M))$. In the Section 3.5 we will generalize this commutation to the case when M is not normalizable.*

Let us state here a fundamental result, due to [ER08]. We will quickly comment on it when we will prove it in the case of $\lambda\mu$-calculus (Theorem 4.4.51). In particular, it will be needed in Theorem 3.3.37.

Lemma 3.3.20 (Non-interference property)**.** *Let $M \in \Lambda$ and $s, t \in \mathcal{T}(M)$. If $\mathrm{nf}_{\mathrm{r}}(s) \cap \mathrm{nf}_{\mathrm{r}}(t) \neq \emptyset$ then $s = t$.*

Remark that this immediately extends to arbitrary intersections: if $t_i \in \mathcal{T}(M)$ for all $i \in \mathcal{I}$ and if $\bigcap_{i \in \mathcal{I}} \mathrm{nf}_{\mathrm{r}}(t_i) \neq \emptyset$, then $\mathrm{nf}_{\mathrm{r}}(t_i) \cap \mathrm{nf}_{\mathrm{r}}(t_j) \neq \emptyset$ for all $i, j \in \mathcal{I}$, and so $t_i = t_j$ for all $i, j \in \mathcal{I}$.

A first application of Lemma 3.3.20 is given by the following Lemma, which gives a case in which reducing each addend of a sum is possible.

Lemma 3.3.21. *1. Let $\sum_i t_i \subseteq \mathcal{T}(M)$ be a sum (thus, finite) of distinct terms. Suppose that for all i we have $\mathrm{nf_r}(t_i) \neq 0$ and $t_i \to_\mathrm{r} \mathbb{T}_i + \mathbb{S}_i$, with $\mathrm{nf_r}(u) \neq 0$ for all $u \in \mathbb{T}_i$ and $\mathrm{nf_r}(s) = 0$ for all $s \in \mathbb{S}_i$. Then:*

$$\sum_i t_i \twoheadrightarrow_\mathrm{r} \sum_i \mathbb{T}_i.$$

2. The same of point 1 still holds without the condition "$\mathrm{nf_r}(t_i) \neq 0$ for all i".

Proof. (1). Reducing t_0 in $\sum_i t_i$ (if the sum is empty there is nothing to prove), we have: $\sum_i t_i \to_\mathrm{r} \mathbb{T}_0 + \mathbb{S}_0 + \sum_{i\geq 1} t_i =: \mathbb{T}^1$. Now, each t_i, for $i \geq 1$, is not in $\mathbb{S}_0$ because it does not reduce to 0; and it is not in $\mathbb{T}_0$ because otherwise $0 \neq \mathrm{nf_r}(t_i) \subseteq \mathrm{nf_r}(\mathbb{T}_0) = \mathrm{nf_r}(t_0)$. But $\mathcal{T}(M) \ni t_0 \neq t_1 \in \mathcal{T}(M)$, so we would have a contradiction with Lemma 3.3.20. Therefore, we are allowed to reduce t_1 in $\mathbb{T}^1$. We obtain: $\mathbb{T}^1 = t_1 + \mathbb{T}_0 + \mathbb{S}_0 + \sum_{i\geq 2} t_i \to_\mathrm{r} \mathbb{T}_1 + \mathbb{S}_1 + \mathbb{T}_0 + \mathbb{S}_0 + \sum_{i\geq 2} t_i$. Remark here that we do not care about the presence of eventual duplicates. In fact, all we care is that now the same argument as before, involving Lemma 3.3.20, shows that we can reduce t_2 in the previous sum. In this way, we reduce all the t_i's, and obtain at the end: $\sum_i t_i \twoheadrightarrow_\mathrm{r} \sum_i \mathbb{T}_i + \mathbb{S}$, for some sum $\mathbb{S}$ s.t. all its element reduce to 0. Thus, by Remark 3.3.8 we are done[13].

(2). Let $\sum_i t_i =: \mathbb{T} + \mathbb{S}$, with $\mathrm{nf_r}(u) \neq 0$ for all $u \in \mathbb{T}$ and $\mathrm{nf_r}(s) = 0$ for all $s \in \mathbb{S}$.. Remark 3.3.8 gives $\sum_i t_i \twoheadrightarrow_\mathrm{r} \mathbb{T}$ and $\mathbb{T}$ is in the hypotheses of point 1. Whence, the result. □

We will not explicit the use of the previous Lemma.

We introduce now a reduction "⇓" that captures in one (big-)step the notion of r-normal form (in the sense of Lemma 3.3.24).

Definition 3.3.22 (Big-step reduction)**.** *The* big-step reduction $\Downarrow \subseteq \Lambda^\mathrm{r} \times \Lambda^\mathrm{r}$ *is the contextual closure of the rules given in Figure 3.2.*

$$\frac{t \text{ normal}}{t \Downarrow t} \qquad \frac{t \Downarrow t'}{\lambda x.t \Downarrow \lambda x.t'} \qquad \frac{t \Downarrow x \qquad u_1^1 \Downarrow v_1^1 \quad \cdots \quad u_{m_1}^1 \Downarrow v_{m_1}^1 \quad \cdots \quad u_1^k \Downarrow v_1^k \quad \cdots \quad u_{m_k}^k \Downarrow v_{m_k}^k}{t\,[u_1^1, \ldots, u_{m_1}^1] \cdots [u_1^k, \ldots, u_{m_k}^k] \Downarrow x\,[v_1^1, \ldots, v_{m_1}^1] \cdots [v_1^k, \ldots, v_{m_k}^k]}$$

$$\frac{t \Downarrow \lambda x.t' \qquad u_1 \Downarrow v_1 \quad \cdots \quad u_k \Downarrow v_k \qquad w[\vec{s}^1] \cdots [\vec{s}^h] \Downarrow t'' \qquad w \in t'\langle [\vec{v}]/x \rangle}{t\,[\vec{u}]\,[\vec{s}^1] \cdots [\vec{s}^h] \Downarrow t''}$$

Figure 3.2: Base cases of the big-step reduction for λ-calculus

It is clear that ⇓ is *not* a function, in that there can be multiple t' s.t. $t \Downarrow t'$.

Remark 3.3.23. *A $t' \in \Lambda^\mathrm{r}$ is* r*-normal as soon as $t \Downarrow t'$ for some $t \in \Lambda^\mathrm{r}$. This is immediate by induction on a reduction $t \Downarrow t'$.*

The next lemma says that, in fact, big-step reduction is simply an inductive characterization of $\mathrm{nf_r}(\cdot)$.

[13] A more formal proof would require to perform an induction, but we preferred to keep the argument non-inductive.

Lemma 3.3.24. *For all $t, t' \in \Lambda^{\mathrm{r}}$ one has: $t \Downarrow t'$ iff $t' \in \mathrm{nf}_{\mathrm{r}}(t)$.*

Proof. ($\Rightarrow$). By induction on a reduction $t \Downarrow t'$ we prove that $t \twoheadrightarrow_{\mathrm{r}} t' + \mathbb{T}$ for some sum $\mathbb{T}$. Since t' r-normal thanks to Remark 3.3.23, this ends the proof. All the cases being immediate, let us see only the one corresponding to $t = p[\vec{u}][\vec{s}^1]\cdots[\vec{s}^h]$, that is the one where $p \Downarrow \lambda x.p'$, $u_i \Downarrow v_i$ for $i = 1, \ldots, k$, and $w[\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t'$, with $w \in p'\langle[\vec{v}]/x\rangle$. In this case, using the inductive hypothesis on the bigstep-reductions above, we get:

$$\begin{array}{rcl} t = p[\vec{u}][\vec{s}^1]\cdots[\vec{s}^h] & \twoheadrightarrow_{\mathrm{r}} & (\lambda x.p')[\vec{v}][\vec{s}^1]\cdots[\vec{s}^h] + \mathbb{T} \\ & \to_{\mathrm{r}} & p'\langle[\vec{v}]/x\rangle[\vec{s}^1]\cdots[\vec{s}^h] + \mathbb{T} \\ & \ni & w[\vec{s}^1]\cdots[\vec{s}^h] \\ & \twoheadrightarrow_{\mathrm{r}} & t' + \mathbb{T}'. \end{array}$$

($\Leftarrow$). By induction on the size $\mathbf{sz}(t) \in \mathbb{N}_{\geq 1}$ of t. If $\mathbf{sz}(t) = 1$ then $t = x = t'$ is normal and we are done by the first rule defining $\Downarrow$. If $\mathbf{sz}(t) \geq 2$ then we start recalling that, as every resource term, t has the shape $t = \lambda y_1 \cdots y_m.\, u[v_1, \ldots, v_n]$ with u either a variable or a redex $(\lambda x.s_0)[s_1, \ldots, s_k]$. Now if $m \neq 0$ then t' must be of shape $t' = \lambda \vec{y}.\widetilde{t}$ for some $\widetilde{t} \in \mathrm{nf}_{\mathrm{r}}(u[\vec{v}])$. The inductive hypothesis on $u[\vec{v}]$ (of strictly smaller size than t because of the presence of at least "λy_1") gives $u[\vec{v}] \Downarrow \widetilde{t}$, whence $t \Downarrow t'$ by the second rule defining $\Downarrow$. So Wlog $m = 0$. Analogously one treats the case where $u = x$. In this case t' must have shape $t' = x[t_1, \ldots, t_n]$ for some $t_i \in \mathrm{nf}_{\mathrm{r}}(v_i)$. If $n = 0$ the result is immediate, and in the case $n \neq 0$ the inductive hypothesis on each v_i (of strictly smaller size than t because of the presence of the bag) gives $v_i \Downarrow t_i$, whence $t \Downarrow t'$ by the third rule defining $\Downarrow$. We are thus left with the case $t = (\lambda x.s_0)[\vec{v}][\vec{s}^1]\cdots[\vec{s}^h]$. But $t \to_{\mathrm{r}} s_0\langle[\vec{v}]/x\rangle[\vec{s}^1]\cdots[\vec{s}^h] \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(s_0)\langle[\mathrm{nf}_{\mathrm{r}}(v_1), \ldots, \mathrm{nf}_{\mathrm{r}}(v_n)]/x\rangle[\vec{s}^1]\cdots[\vec{s}^h]$. By confluence $t' \in \mathrm{nf}_{\mathrm{r}}(\widetilde{s}[\vec{s}^1]\cdots[\vec{s}^h])$ for some $\widetilde{s} \in s_0'\langle[v_1', \ldots, v_n']/x\rangle$, for some $s_0' \in \mathrm{nf}_{\mathrm{r}}(s_0), v_i' \in \mathrm{nf}_{\mathrm{r}}(v_i)$. Also, $\mathbf{sz}(\widetilde{s}[\vec{s}^1]\cdots[\vec{s}^h]) \leq \mathbf{sz}(t)$ and thus the inductive hypothesis on it gives $\widetilde{s}[\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t'$. But we can apply the inductive hypothesis also on s_0 and on each v_i since their size is clearly strictly smaller than that of t, and so we get $v_i \Downarrow v_i'$ as well as $s_0 \Downarrow s_0'$, whence $\lambda x.s_0 \Downarrow \lambda x.s_0$. Hence, we can apply the last rule of the definition of $\Downarrow$ and obtain $t = (\lambda x.s_0)[\vec{v}][\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t'$. □

Lemma 3.3.25. *(1) For all $t \in \mathcal{T}(M)$ and $t' \in \mathrm{nf}_{\mathrm{r}}(t)$, there is $N \in \Lambda$ such that $M \twoheadrightarrow N$ and $t' \in \mathcal{T}(N)$.*

(2) For all $t \in \mathcal{T}(M)$, there exist $N \in \Lambda$ such that $M \twoheadrightarrow N$ and $\mathrm{nf}_{\mathrm{r}}(t) \subseteq \mathcal{T}(N)$.

Proof. (1). Thanks to the previous lemma we can prove the result by induction on $t \Downarrow t'$. All the cases are similar, so let us only show the case for $t = p[\vec{u}][\vec{s}^1]\cdots[\vec{s}^h]$, that is the one where $p \Downarrow \lambda x.p'$, $u_i \Downarrow v_i$ for $i = 1, \ldots, k$, and $w[\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t'$, with $w \in p'\langle[\vec{v}]/x\rangle$. In this case it must be $M = PQ\vec{R}$, with $p \in \mathcal{T}(P)$, $[\vec{u}] \in !\mathcal{T}(Q)$ and $[\vec{s}^i] \in !\mathcal{T}(R_i)$. By inductive hypothesis on each u_i, for all $i = 1, \ldots, k$ there is $Q_i \in \Lambda$ s.t. $Q \twoheadrightarrow Q_i$ and $v_i \in \mathcal{T}(Q_i)$. By confluence there exist $Q' \in \Lambda$ such that every $Q_i \twoheadrightarrow Q'$, and since all the v_i are normal, by Remark 3.3.15(3) we have $v_1, \ldots, v_k \in \mathcal{T}(Q')$ (if $k = 0$ just take $Q' := Q$). Also, the inductive hypothesis on $p \in \mathcal{T}(P)$ (which is such that $p \Downarrow \lambda x.p'$) gives a $P' \in \Lambda$ such that $P \twoheadrightarrow \lambda x.P'$ and $p' \in \mathcal{T}(P')$. But then by Lemma 3.3.16 we have $w[\vec{s}^1]\cdots[\vec{s}^h] \in \mathcal{T}(P'\{Q'/x\}\vec{R})$, and thus the inductive hypothesis on the fact that it $\Downarrow$-reduces to t' gives an $N \in \Lambda$ such that $P'\{Q'/x\}\vec{R} \twoheadrightarrow N$ and $t' \in \mathcal{T}(N)$. Now we have $M \twoheadrightarrow (\lambda x.P')Q'\vec{R} \to P'\{Q'/x\}\vec{R} \twoheadrightarrow N$ and we are done.

(2). By (1) we have that for all $s \in \mathrm{nf}_{\mathrm{r}}(t)$, there exist $N_s \in \Lambda$ such that $M \twoheadrightarrow N_s$ and $s \in \mathcal{T}(N_s)$. Now, if $\mathrm{nf}_{\mathrm{r}}(t) = 0$ then one can choose $N := M$; otherwise, by confluence there is an $N \in \Lambda$ such that $N_s \twoheadrightarrow N$ for all $s \in \mathrm{nf}_{\mathrm{r}}(t)$, and since every such s is r-normal, by Remark 3.3.15(3) one has that $s \in \mathcal{T}(N)$. □

One can immediately lift the previous result to sums. This is done in the next Corollary 3.3.26, which will also be proved again, with a different method, at the end of the next Section 3.3.1.

Corollary 3.3.26. *For all* $\mathbb{T} \subseteq \mathcal{T}(M)$*, there exist* $N \in \Lambda$ *such that* $M \twoheadrightarrow N$ *and* $\mathrm{nf_r}(\mathbb{T}) \subseteq \mathcal{T}(N)$.

Proof. One can reason in the exact same way as for Lemma 3.3.25(2), remembering that by definition $\mathrm{nf_r}(\mathbb{T}) = \bigcup_{s\in\mathbb{T}} \mathrm{nf_r}(s)$. □

In the following of the thesis the most of the times that we will use Corollary 3.3.26 we could in fact already use the version for $\mathbb{T} = t$, that is, Lemma 3.3.25(2).

Corollary 3.3.27. *If* $N =_\tau \lambda\vec{x}.xM_1\cdots M_k$ *then* $N \twoheadrightarrow \lambda\vec{x}.xN_1\cdots N_k$*, for some* $N_i \in \Lambda$ *s.t.* $N_i =_\tau M_i$.

Proof. We know that $\mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(\lambda\vec{x}.xM_1\cdots M_k) = \lambda\vec{x}.x\,!\mathrm{NF}\mathcal{T}(M_1)\cdots!\mathrm{NF}\mathcal{T}(M_k) \neq \emptyset$. Therefore there is an $h \in \mathcal{T}(N)$ and a $\lambda\vec{x}.x[\vec{v}^1]\cdots[\vec{v}^k] \in \mathrm{nf_r}(h)$, with $v^i_j \in \mathrm{NF}\mathcal{T}(M_i)$. By Corollary 3.3.26, $N \twoheadrightarrow N'$ for some $N' \in \Lambda$ s.t. $\lambda\vec{x}.x[\vec{v}^1]\cdots[\vec{v}^k] \in \mathrm{nf_r}(h) \subseteq \mathcal{T}(N')$. So it must be $N' = \lambda\vec{x}.xN_1\cdots N_k$ for some $N_i \in \Lambda$. Now, by Corollary 3.3.18 we get $\lambda\vec{x}.xM_1\cdots M_k =_\tau \lambda\vec{x}.xN_1\cdots N_k$, and applying Remark 3.3.12, $N_i =_\tau M_i$ for all $i = 1,\dots,k$. □

3.3.1 Rigids of a resource term

We consider in this subsection the notion of rigid term. We are mainly interested in the fact that rigid terms are canonically associated with a resource term (Definition 3.3.29). Rigid-calculi have been considered, in various forma, e.g. in [TAO17], [MPV17] and [OA20], in relation to, respectively, generalized species of structures, intersection types and the combinatorial role of the factorial coefficients in the full (i.e. quantitative) Taylor expansion of a term.

The reason why we are interested in rigid terms here is double: first, as already mentioned, it allows us to give an alternative proof of Corollary 3.3.26; we will adapt the same method in Section 4.3, and it will work also for $\lambda\mu$-calculus. The second reason is that the results we will prove in this subsection about the rigids associated with a resource term (Lemma 3.3.32 and Proposition 3.3.33) are needed in the remainder of the chapter (in particular, in the proof of the Stability property – Theorem 3.3.37).

In the following, "$\langle\dots\rangle$" denotes a list (and $\langle\rangle$ the empty list).

Definition 3.3.28 (Rigids)**.** *The set of* rigid resource λ-terms *is defined by:*

$$t ::= x \mid \lambda x.t \mid t\langle t,\dots,t\rangle.$$

The set of rigid k*-context is defined as expected adding the clause "*$\xi_1 \mid \cdots \mid \xi_k$*" for the holes.*

Definition 3.3.29. *Let* c *be a resource-*k*-context. We define a set* $\mathrm{Rigid}(c)$ *of* rigid k-contexts*, whose elements are called the* rigids of c*, by induction on* c *as follows:*

1. $\mathrm{Rigid}(\xi_i) = \{\xi_i\}$
2. $\mathrm{Rigid}(x) = \{x\}$
3. $\mathrm{Rigid}(\lambda x.c_0) = \{\lambda x.c_0^\bullet \mid c_0^\bullet \in \mathrm{Rigid}(c_0)\}$
4. $\mathrm{Rigid}(c_0[c_1,\dots,c_k]) = \{c_0^\bullet\langle c^\bullet_{\sigma(1)},\dots,c^\bullet_{\sigma(k)}\rangle \mid c_i^\bullet \in \mathrm{Rigid}(c_i)$ *and* σ *permutation*$\}$.

Our set Rigid(c) is the preimage $F^{-1}(c)$ of c under the surjection F [14] from rigid contexts to resource context that forgets the order of the elements in the lists. Its graph relation is what in [OA20] is called the "representation relation". In the following definition we precisely operate such a forgetful operation, but in addition we consider terms filling the holes.

Definition 3.3.30. *Let $c^\bullet$ be a rigid of a resource-k-context c and, for $i = 1, \dots, k$, let $\vec{v}^i := \langle v^i_1, \dots, v^i_{\deg_{\xi_i}(c)}\rangle$ be a list[15] of resource terms. We define, by induction on c, a resource term $c^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!)$ as follows:*

1. *If $c = \xi_i$ then $c^\bullet = \xi_i$; we set $c^\bullet(\!|\langle\rangle, \dots, \langle\rangle, \langle v^i_1\rangle, \langle\rangle, \dots, \langle\rangle|\!) := v^i_1$.*
2. *If $c = x$ then $c^\bullet = x$; we set $c^\bullet(\!|\langle\rangle, \dots, \langle\rangle|\!) := x$.*
3. *If $c = \lambda x.c_0$ then $c^\bullet = \lambda x.c^\bullet_0$ where $c^\bullet_0$ is a rigid of c_0; we set $c^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) = \lambda x.c^\bullet_0(\!|\vec{v}^1, \dots, \vec{v}^k|\!)$.*
4. *If $c = c_0[c_1, \dots, c_n]$, then $c^\bullet = c^\bullet_0\langle c^\bullet_{\sigma(1)}, \dots, c^\bullet_{\sigma(n)}\rangle$ where $c^\bullet_i$ is a rigid of c_i. So each list $\vec{v}^i$ factorizes as a concatenation $\vec{w}^{i0}\vec{w}^{i1}\cdots\vec{w}^{in}$ of lists where $\vec{w}^{i0}$ has exactly $\deg_{\xi_i}(c_0)$ elements and $\vec{w}^{ij}$ has exactly $\deg_{\xi_i}(c_{\sigma(j)})$ elements[16]; we set:*

$$c^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) := c^\bullet_0(\!|\vec{w}^{10}, \dots, \vec{w}^{k0}|\!)[c^\bullet_{\sigma(1)}(\!|\vec{w}^{11}, \dots, \vec{w}^{k1}|\!), \dots, c^\bullet_{\sigma(n)}(\!|\vec{w}^{1n}, \dots, \vec{w}^{kn}|\!)].$$

Remark 3.3.31. *One has that if $v \twoheadrightarrow_{\mathrm{r}} \mathbb{V}$ then:*

$$c^\bullet(\!|\cdots, \langle\cdots, v, \cdots\rangle, \cdots|\!) \twoheadrightarrow_{\mathrm{r}} \sum_{w\in\mathbb{V}} c^\bullet(\!|\cdots, \langle\cdots, w, \cdots\rangle, \cdots|\!) =: c^\bullet(\!|\cdots, \langle\cdots, \mathbb{V}, \cdots\rangle, \cdots|\!).$$

Let us extend the definition of Taylor expansion to resource k-contexts by adding, in its definition, the clause:

$$\mathcal{T}(\xi_i) := \{\xi_i\}.$$

It is clear that is C is a k-context then all elements of $\mathcal{T}(C)$ are resource k-contexts.

In the following, if $\vec{v}$ is a list, we denote with $[\vec{v}]$ the multiset associated with $\vec{v}$ (same elements but disordered).

Lemma 3.3.32. *Let C be a k-context and $c_1, c_2 \in \mathcal{T}(C)$. Let $c^\bullet_1$ and $c^\bullet_2$ rigids respectively of c_1 and c_2. For $i = 1 \dots, k$, let $\vec{v}^i = \langle v^i_1, \dots, v^i_{\deg_{\xi_i}(c_1)}\rangle$ and $\vec{u}^i = \langle u^i_1, \dots, u^i_{\deg_{\xi_i}(c_2)}\rangle$ be lists of resource terms. If $c^\bullet_1(\!|\vec{v}^1, \dots, \vec{v}^k|\!) = c^\bullet_2(\!|\vec{u}^1, \dots, \vec{u}^k|\!)$ then $c_1 = c_2$ and $[\vec{v}^i] = [\vec{u}^i]$ for all i.*

Proof. Induction on C.

Case $C = \xi_i$. Then $c_1 = \xi_i = c_2$, and $\vec{v}^i = \langle v^i_1\rangle$, $\vec{u}^i = \langle u^i_1\rangle$ and $\vec{v}^j = \langle\rangle = \vec{u}^j$ for $j \neq i$. So $v^i_1 = c^\bullet_1(\!|\vec{v}^1, \dots, \vec{v}^k|\!) = c^\bullet_2(\!|\vec{u}^1, \dots, \vec{u}^k|\!) = u^i_1$.

Case $C = x$. Trivial.

Case $C = \lambda x.C_0$. Straightforward by inductive hypothesis.

[14] In [OA20] F is called $\| \cdot \|$, but here wrote F, for "forgetful".

[15] If $\deg_{\xi_i}(c) = 0$ we mean the empty list.

[16] This is just a concise way of saying that we take $\vec{w}^{i1} := \langle v^i_1, \dots, v^i_{\deg_{\xi_i}(c_{\sigma(1)})}\rangle$, $\vec{w}^{i2} := \langle v^i_{1+\deg_{\xi_i}(c_{\sigma(1)})}, \dots, v^i_{\deg_{\xi_i}(c_{\sigma(2)})+\deg_{\xi_i}(c_{\sigma(1)})}\rangle$ etc, and similarly for $\vec{w}^{i0}$.

Case $C = C'C''$. Then, for $i = 1, 2$, one has $c_i = c_{i0}[c_{i1}, \ldots, c_{in_i}]$ with $c_{i0} \in \mathcal{T}(C')$ and $c_{ij} \in \mathcal{T}(C'')$ for $j \geq 1$. So $c_i^\bullet = c_{i0}^\bullet \langle c_{i\sigma_i(1)}^\bullet, \ldots, c_{i\sigma_i(n_i)}^\bullet \rangle$ where σ_i is a permutation on n_i elements. So:

$$c_1^\bullet(\!|\vec{v}^1, \ldots, \vec{v}^k|\!) = c_{10}^\bullet(\!|\vec{w}^{110}, \ldots, \vec{w}^{1k0}|\!)[\, c_{1\sigma_1(1)}^\bullet(\!|\vec{w}^{111}, \ldots, \vec{w}^{1k1}|\!), \ldots, c_{1\sigma_1(n_1)}^\bullet(\!|\vec{w}^{11n_1}, \ldots, \vec{w}^{1kn_1}|\!)]$$

$$c_2^\bullet(\!|\vec{v}^1, \ldots, \vec{v}^k|\!) = c_{20}^\bullet(\!|\vec{w}^{210}, \ldots, \vec{w}^{2k0}|\!)[\, c_{2\sigma_2(1)}^\bullet(\!|\vec{w}^{211}, \ldots, \vec{w}^{2k1}|\!), \ldots, c_{2\sigma_2(n_2)}^\bullet(\!|\vec{w}^{21n_2}, \ldots, \vec{w}^{2kn_2}|\!)]$$

where the concatenation $\vec{w}^{1j1} \cdots \vec{w}^{1jn_1}$ equals $\vec{v}^j$ and the concatenation $\vec{w}^{2j1} \cdots \vec{w}^{2jn_2}$ equals $\vec{u}^j$. From $c_1^\bullet(\!|\vec{v}^1, \ldots, \vec{v}^k|\!) = c_2^\bullet(\!|\vec{u}^1, \ldots, \vec{u}^k|\!)$ we get that $n_1 = n_2 =: n$, that:

$$c_{10}^\bullet(\!|\vec{w}^{110}, \ldots, \vec{w}^{1k0}|\!) = c_{20}^\bullet(\!|\vec{w}^{210}, \ldots, \vec{w}^{2k0}|\!) \tag{3.6}$$

and that there exist a permutation ρ on n elements which identifies each term of the written bag of $c_1^\bullet(\!|\vec{v}^1, \ldots, \vec{v}^k|\!)$ with the respective one of the written bag of $c_2^\bullet(\!|\vec{u}^1, \ldots, \vec{u}^k|\!)$. That is, for all $h = 1, \ldots, n$, one has:

$$c_{1\sigma_1(h)}^\bullet(\!|\vec{w}^{\,1\,1\,h}, \ldots, \vec{w}^{\,1\,k\,h}|\!) = c_{2\sigma_2(\rho(h))}^\bullet(\!|\vec{w}^{\,2\,1\,\rho(h)}, \ldots, \vec{w}^{\,2\,k\,\rho(h)}|\!). \tag{3.7}$$

Now the inductive hypothesis on (3.6) gives $c_{10}^\bullet = c_{20}^\bullet$ as well as $[\vec{w}^{\,1\,i\,0}] = [\vec{w}^{\,2\,i\,0}]$, and the inductive hypothesis on (3.7) gives, after some reindexing, $[c_{11}^\bullet, \ldots, c_{1n}^\bullet] = [c_{21}^\bullet, \ldots, c_{2n}^\bullet]$ as well as $[\vec{w}^{1j1} \cdots \vec{w}^{1jn}] = [\vec{w}^{2j1} \cdots \vec{w}^{2jn}]$ for $j \geq 1$. Putting these things together, we have the desired result. □

Lemma 3.3.33. *1. Let C be a k-context. Then:*

$$\mathcal{T}(C(\!|M_1, \ldots, M_k|\!)) = \{c^\bullet(\!|\vec{s}^1, \ldots, \vec{s}^k|\!) \mid c \in \mathcal{T}(C),\ c^\bullet \text{ rigid of } c \text{ and } \vec{s}^i \text{ list of elements of } \mathcal{T}(M_i)\}.$$

2. Let $c = c(\!|\xi|\!)$ be a single-hole resource context, $M \in \Lambda$ and $s_0 \in \Lambda^{\mathrm{r}}$. If $c(\!|s_0|\!) \in \mathcal{T}(M)$, then there is a context $C = C(\!|\xi|\!)$, an $N \in \Lambda$, a resource context $\widetilde{c} \in \mathcal{T}(C)$, a rigid $\widetilde{c}^\bullet$ of $\widetilde{c}$ and $s_1 \ldots, s_{\deg_\xi \widetilde{c}-1} \in \mathcal{T}(N)$, s.t.:

(a) $M = C(\!|N|\!)$

(b) $s_0 \in \mathcal{T}(N)$

(c) $c(\!|t|\!) = \widetilde{c}^\bullet(\!|\langle t, s_1 \ldots, s_{\deg_\xi(\widetilde{c})-1}\rangle|\!)$ for all $t \in \Lambda^{\mathrm{r}}$.

Proof. 1. Straightforward induction on C.

2. Induction on the single-hole resource context c.

Case $c = \xi$. Take $C := \xi$, $N := M$, $\widetilde{c} := \xi$, $\widetilde{c}^\bullet := \xi$ (there are no s_i's since $\deg_\xi(\widetilde{c}) - 1 = 0$).

Case $c = \lambda x.c_1$. Since $c(\!|s_0|\!) \in \mathcal{T}(M)$ then $M = \lambda x.M_1$ with $c_1(\!|s_0|\!) \in \mathcal{T}(M_1)$. We can thus take $C := \lambda x.C_1$, $\widetilde{c} := \lambda x.\widetilde{c_1}$, $\widetilde{c}^\bullet := \lambda x.\widetilde{c_1}^\bullet$, where $C_1, N, \widetilde{c_1}, \widetilde{c_1}^\bullet$ and $s_1, \ldots, s_{\deg_\xi(\widetilde{c_1})}$ are given by the inductive hypothesis.

Case $c = c'[\vec{u}]$. Analogous as above.

Case $c = u[c', u_1, \ldots, u_n]$. Since $u[c'(\!|s_0|\!), \vec{u}] = c(\!|s_0|\!) \in \mathcal{T}(M)$ then M must have shape $M = PQ$ with $u \in \mathcal{T}(P)$ and $c'(\!|s_0|\!), u_i \in \mathcal{T}(Q)$. By induction hypothesis, we can write $Q = C_0(\!|N|\!)$ for an appropriate context C_0 and $N \in \mathfrak{L}$ s.t. $s_0 \in \mathcal{T}(N)$, and also there is a resource context $c_0 \in \mathcal{T}(C_0)$ and $c_0^\bullet$ rigid of c_0, together with a list $\vec{s}^0 := \langle s_1^0, \ldots, s_{\deg_\xi(c_0)-1}^0\rangle$ of elements of $\mathcal{T}(N)$, such that $c'(\!|t|\!) = c_0^\bullet(\!|\langle t, \vec{s}^0\rangle|\!)$ for all $t \in \Lambda^{\mathrm{r}}$. But $u_i \in \mathcal{T}(Q) = \mathcal{T}(C_0(\!|N|\!))$, so by the point (1), each u_i have shape $u_i = c_i^\bullet(\!|\vec{s}^i|\!)$ for an appropriate $c_i^\bullet$ rigid of some $c_i \in \mathcal{T}(C_0)$ and some $\vec{s}^i$ list of elements of $\mathcal{T}(N)$. Now, we

have: $M = C\langle\!|N|\!\rangle$ for $C := PC_0$. Moreover, putting $\widetilde{c} := u[c_0, c_1, \ldots, c_n] \in \mathcal{T}(C)$ and choosing its rigid $\widetilde{c}^\bullet := u^\bullet\langle c_0^\bullet, c_1^\bullet, \ldots, c_k^\bullet\rangle$ (which rigid $u^\bullet$ of u one chooses does not matter), we have: $c\langle\!|t|\!\rangle = u[c'\langle\!|t|\!\rangle, u_1 \ldots, u_n] = u[c_0^\bullet\langle\!|\langle t, \vec{s}\rangle|\!\rangle, c_1^\bullet\langle\!|\vec{s}^{\,1}|\!\rangle, \ldots, c_k^\bullet\langle\!|\vec{s}^{\,k}|\!\rangle] = \widetilde{c}^\bullet\langle\!|\langle t, \vec{s}\rangle|\!\rangle$ for all $t \in \Lambda^{\mathrm{r}}$, where the list $\vec{s}$ is the concatenation $\vec{s}^{\,0}\vec{s}^{\,1}\cdots\vec{s}^{\,n}$ and is made of exactly $\deg_\xi(\widetilde{c}) - 1$ elements of $\mathcal{T}(N)$. □

Let us notice how, as previously mentioned, using resource contexts one can give an alternative proof of Corollary 3.3.26 not using the bigstep argument.

- First prove the following:

Proposition 3.3.34. *If $\mathcal{T}(M) \supseteq \mathbb{T} \to_{\mathrm{r}} \mathbb{T}'$ then there is $N \in \mathfrak{L}$ and a sum $\widetilde{\mathbb{T}} \subseteq \mathcal{T}(N)$ s.t. $M \to N$ and $\mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \widetilde{\mathbb{T}}$.*

Proof. Saying that $\mathbb{T} \to_{\mathrm{r}} \mathbb{T}'$ means that $\mathbb{T}$ has shape $\mathbb{T} = \sum_i t_i + c\langle\!|(\lambda x.s_0)[\vec{u}]|\!\rangle$ and $\mathbb{T}'$ has shape $\mathbb{T}' = \sum_i t_i + c\langle\!|s_0\langle[\vec{u}]/x\rangle|\!\rangle$, for some single-hole resource context c. So Lemma 3.3.33(2) gives us a context C_0, a term $N' \in \Lambda$, a resource context $c_0 \in \mathcal{T}(C_0)$, a rigid $c_0^\bullet$ of c_0 and resource terms $\vec{s} \in \mathcal{T}(N')$ s.t. $M = C_0\langle\!|N'|\!\rangle$, $(\lambda x.s_0)[\vec{u}] \in \mathcal{T}(N')$ and $c\langle\!|u|\!\rangle = c_0^\bullet\langle\!|u, \vec{s}|\!\rangle$ for all $u \in \Lambda^{\mathrm{r}}$. Thus $N' = (\lambda x.P)Q$, with $s_0 \in \mathcal{T}(P)$ and $[\vec{u}] \in !\mathcal{T}(Q)$, and so $M \to C\langle\!|N''|\!\rangle =: N$, with $N'' := P\{Q/x\}$. Also, $s_0\langle[\vec{u}]/x\rangle \in \mathcal{T}(N'')$ thanks to Lemma 3.3.16. Now: every $t_i \in \mathcal{T}(M) = \mathcal{T}(C_0\langle\!|N'|\!\rangle)$, so by Lemma 3.3.33(1) it must have shape $t_i = c_i^\bullet\langle\!|\vec{v}^{\,i}|\!\rangle$ for some resource terms $v_j^i \in \mathcal{T}(N')$, a context $c_i \in \mathcal{T}(C_0)$ and a rigid $c_i^\bullet$ of c_i. But since $N' \to N''$ we can apply Proposition 3.3.17 on v_j^i and obtain that $v_j^i \twoheadrightarrow_{\mathrm{r}} \mathbb{V}_j^i$ for some sum $\mathbb{V}_j^i \subseteq \mathcal{T}(N'')$. So $t_i \twoheadrightarrow_{\mathrm{r}} c_i^\bullet\langle\!|\vec{\mathbb{V}}^{\,i}|\!\rangle =: \mathbb{T}_i$. Using again Lemma 3.3.33(1) one has that $\mathbb{T}_i \subseteq \mathcal{T}(N)$. Now, let's use again Proposition 3.3.17, this time on each $s_i \in \mathcal{T}(N')$. Since $N' \to N''$ we obtain sums $\mathbb{S}_i \subseteq \mathcal{T}(N'')$ s.t. $s_i \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_i$. So we have: $c\langle\!|s_0\langle[\vec{u}]/x\rangle|\!\rangle = c_0^\bullet\langle\!|s_0\langle[\vec{u}]/x\rangle, \vec{s}|\!\rangle \twoheadrightarrow_{\mathrm{r}} c_0^\bullet\langle\!|s_0\langle[\vec{u}]/x\rangle, \vec{\mathbb{S}}|\!\rangle =: \mathbb{U}$. But since $s_0\langle[\vec{u}]/x\rangle \subseteq \mathcal{T}(N'')$ and every $\mathbb{S}_i \subseteq \mathcal{T}(N'')$, again thanks to Lemma 3.3.33(1) one has $\mathbb{U} \subseteq \mathcal{T}(C_0\langle\!|N''|\!\rangle) = \mathcal{T}(N)$. This ends the proof, since letting $\widetilde{\mathbb{T}} := \sum_i \mathbb{T}_i + \mathbb{U} \subseteq \mathcal{T}(N)$ one has $\mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \widetilde{\mathbb{T}}$. □

- and then prove Corollary 3.3.26:

Alternative proof of Corollary 3.3.26. We prove, by induction on $\ell \in \mathbb{N}$, that if ℓ is the length of a maximal reduction (let's call it ϕ) from a sum $\mathbb{T}$ to its normal form $\mathrm{nf}_{\mathrm{r}}(\mathbb{T})$, then the statement of Corollary 3.3.26 holds. If $\ell = 0$ then just take $N := M$. If $\ell \geq 1$, then ϕ factorizes as $\mathbb{T} \to_{\mathrm{r}} \mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(\mathbb{T})$ for some sum $\mathbb{T}'$. Since $\mathbb{T} \in \mathcal{T}(M)$ by hypothesis, by Proposition 3.3.34 there exist $N' \in \mathfrak{L}$ and $\mathbb{T}'' \subseteq \mathcal{T}(N')$ such that $M \to N'$ and $\mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \mathbb{T}''$. Let $k \geq 0$ be the length of this last reduction. By confluence, we have also $\mathbb{T}'' \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(\mathbb{T})$. Now take the maximal reduction from $\mathbb{T}''$ to $\mathrm{nf}_{\mathrm{r}}(\mathbb{T}'') = \mathrm{nf}_{\mathrm{r}}(\mathbb{T})$ and let ℓ' be its length. Due to the maximality of ϕ, it must be $\ell' + k + 1 \leq \ell$, so $\ell' < \ell$, and now we can apply the inductive hypothesis to ℓ' (because it is the maximal length of a reduction between a sum $\mathbb{T}''$ and its normal form). Since we already found that $\mathbb{T}'' \subseteq \mathcal{T}(N')$, we get an $N \in \mathfrak{L}$ such that $M \to N' \twoheadrightarrow N$ and $\mathrm{nf}_{\mathrm{r}}(\mathbb{T}) = \mathrm{nf}_{\mathrm{r}}(\mathbb{T}'') \subseteq \mathcal{T}(N)$. □

3.3.2 Stability and Perpendicular Lines Property

Let us now apply the developed tools in order to obtain, first, the Stability property and, then, the Perpendicular Lines Lemma (PLP for short). Both of them are, for now, stated with respect to the λ-theory $=_\tau$.

Stability Let us set up some notations used in the statement of the Stability theorem.

Definition 3.3.35. *Given a non-empty subset $\mathcal{X} \subseteq \Lambda$, define its $\mathcal{T}$-*infimum *$\bigcap \mathcal{X} \subseteq \Lambda^{\mathrm{r}}$ (also written $\bigcap_{M\in\mathcal{X}} M$) as:*

$$\bigcap \mathcal{X} := \bigcap_{M\in\mathcal{X}} \mathrm{NF}\mathcal{T}(M).$$

*We say that $\mathcal{X}$ is $\mathcal{T}$-*bounded *iff there exists an $L \in \Lambda$ such that $M \leq L$ for all $M \in \mathcal{X}$.*

We write $M = \bigcap \mathcal{X}$ instead of $\mathrm{NF}\mathcal{T}(M) = \bigcap \mathcal{X}$.

Example 3.3.36. *1. For all $M \in \Lambda$, we have $\bigcap M = M$.*

2. $\mathsf{I} \cap \Omega = \emptyset = \Omega$ and $\lambda x.x\Omega \cap \Delta = \lambda x.x1$.

3. $\bigcap_{n\geq 2} \lambda f.f^n\Omega = \{\lambda f.f[\underbrace{f1,\ldots,f1}_{k}] \mid k \in \mathbb{N}\}$.

Observe that the M s.t. $M = \bigcap \mathcal{X}$, in case it exists, need *not* to be unique; so writing only $\bigcap \mathcal{X}$ does not identify a unique λ-term. The following Stability theorem gives sufficient conditions for when a context commutes with intersection, in the sense that one informally has (with abuse of notation):

$$C(\!|\bigcap_{N\in\mathcal{X}_1} N\ ,\ldots,\ \bigcap_{N\in\mathcal{X}_n} N|\!) = \bigcap_{N_1\in\mathcal{X}_1} \cdots \bigcap_{N_n\in\mathcal{X}_n} C(\!|N_1,\ldots,N_n|\!).$$

Theorem 3.3.37 (Stability property)**.** *Let C be an n-context and fix non-empty $\mathcal{T}$-bounded $\mathcal{X}_1,\ldots,\mathcal{X}_n \subseteq \Lambda$. For all $M_1,\ldots,M_n \in \Lambda$ s.t.*

$$M_i = \bigcap \mathcal{X}_i \quad \textit{(for } i = 1,\ldots,n\textit{)}$$

we have:

$$C(\!|M_1,\ldots,M_n|\!) = \bigcap_{\substack{N_1\in\mathcal{X}_1\\ \vdots\\ N_n\in\mathcal{X}_n}} C(\!|N_1,\ldots,N_n|\!).$$

Proof. Since every $\mathcal{X}_i$ is $\mathcal{T}$-bounded, for $i = 1,\ldots,n$ there exists $L_i \in \Lambda$ s.t. $\bigcup_{N\in\mathcal{X}_i} \mathrm{NF}\mathcal{T}(N) \subseteq \mathrm{NF}\mathcal{T}(L_i)$. Fix now $M_1,\ldots,M_n \in \Lambda$ s.t. $\mathrm{NF}\mathcal{T}(M_i) = \bigcap_{N\in\mathcal{X}_i} \mathrm{NF}\mathcal{T}(N)$. We have to show that:

$$\mathrm{NF}\mathcal{T}(C(\!|M_1,\ldots,M_n|\!)) = \bigcap_{N_1\in\mathcal{X}_1} \cdots \bigcap_{N_n\in\mathcal{X}_n} \mathrm{NF}\mathcal{T}(C(\!|N_1,\ldots,N_n|\!)).$$

($\subseteq$). Clearly, for all $i = 1,\ldots,n$ and $N_i \in \mathcal{X}_i$, we have $\mathrm{NF}\mathcal{T}(M_i) \subseteq \mathrm{NF}\mathcal{T}(N_i)$, therefore we conclude $\mathrm{NF}\mathcal{T}(C(\!|M_1,\ldots,M_n|\!)) \subseteq \mathrm{NF}\mathcal{T}(C(\!|N_1,\ldots,N_n|\!))$ by Monotonicity (Proposition 3.3.13).

($\supseteq$). Let $t \in \bigcap_{\vec{N}\in\vec{\mathcal{X}}} \mathrm{NF}\mathcal{T}(C(\!|N_1,\ldots,N_n|\!))$ (where $\vec{N} := (N_1,\ldots,N_n)$ and $\vec{\mathcal{X}} := (\mathcal{X}_1,\ldots,\mathcal{X}_n)$). For every $\vec{N} \in \vec{\mathcal{X}}$, by Lemma 3.3.33 there exist an n-resource-context $c_{\vec{N}} \in \mathcal{T}(C)$ and, for every $i = 1,\ldots,n$, a list $\vec{v}^{\,i}_{\vec{N}} = \langle v^{i1}_{\vec{N}},\ldots,v^{id_i}_{\vec{N}}\rangle$ (where $d_i := \deg_{\xi_i}(c_{\vec{N}})$) of elements of $\mathcal{T}(N_i)$ and such that $t \in \mathrm{nf}_{\mathrm{r}}(c^{\bullet}_{\vec{N}}(\!|\vec{v}^{\,1}_{\vec{N}},\ldots,\vec{v}^{\,n}_{\vec{N}}|\!))$, for $c^{\bullet}_{\vec{N}}$ a rigid of $c_{\vec{N}}$. Fix any reduction from $c^{\bullet}_{\vec{N}}(\!|\vec{v}^{\,1}_{\vec{N}},\ldots,\vec{v}^{\,n}_{\vec{N}}|\!)$ to its normal form, and confluence allows to factorize it as follows:

$$c^{\bullet}_{\vec{N}}(\!|\mathrm{nf}_{\mathrm{r}}(v^{11}_{\vec{N}}),\ldots,\mathrm{nf}_{\mathrm{r}}(v^{1d_1}_{\vec{N}}),\ldots,\mathrm{nf}_{\mathrm{r}}(v^{n1}_{\vec{N}}),\ldots,\mathrm{nf}_{\mathrm{r}}(v^{nd_n}_{\vec{N}})|\!) \quad \twoheadrightarrow_{\mathrm{r}} \quad \mathrm{nf}_{\mathrm{r}}(c^{\bullet}_{\vec{N}}(\!|\vec{v}^{\,1}_{\vec{N}},\ldots,\vec{v}^{\,n}_{\vec{N}}|\!)) \ni t.$$

So for all $i = 1, \dots, n$ and $j = 1, \dots, d_i$, there exist $w^{ij}_{\vec{N}} \in \mathrm{nf_r}(v^{ij}_{\vec{N}})$ such that:

$$\mathrm{nf_r}(c^\bullet_{\vec{N}} \langle\!| \vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}} |\!\rangle) \ni t \tag{3.8}$$

and being $N_i \in \mathcal{X}_i$ which is bounded by L_i, we have $w^{ij}_{\vec{N}} \in \mathrm{nf_r}(v^{ij}_{\vec{N}}) \subseteq \mathrm{NF}\mathcal{T}(N_i) \subseteq \mathrm{NF}\mathcal{T}(L_i)$. From each inclusion $w^{ij}_{\vec{N}} \in \mathrm{NF}\mathcal{T}(L_i)$ we obtain a resource term $u^{ij}_{\vec{N}} \in \mathcal{T}(L_i)$ such that:

$$w^{ij}_{\vec{N}} \in \mathrm{nf_r}(u^{ij}_{\vec{N}}) \tag{3.9}$$

By composing thus a reduction from $u^{ij}_{\vec{N}}$ to $w^{ij}_{\vec{N}}$ with a reduction from $c^\bullet_{\vec{N}} \langle\!| \vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}} |\!\rangle$ to t, we find that $t \in \mathrm{nf_r}(c^\bullet_{\vec{N}} \langle\!| \vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}} |\!\rangle)$. This holds for all $\vec{N} \in \vec{\mathcal{X}}$, i.e.:

$$t \in \bigcap_{\vec{N} \in \vec{\mathcal{X}}} \mathrm{nf_r}(c_{\vec{N}} \langle\!| \vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}} |\!\rangle). \tag{3.10}$$

Now, Lemma 3.3.33 gives $c^\bullet_{\vec{N}} \langle\!| \vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}} |\!\rangle \in \mathcal{T}(C \langle\!| L_1, \dots, L_n |\!\rangle)$. But since the L_i's are independent from $N_1, \dots, N_n$, and thanks to (3.10), we can apply Lemma 3.3.20, and obtain that the set $\{c^\bullet_{\vec{N}} \langle\!| \vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}} |\!\rangle \mid \vec{N} \in \vec{\mathcal{X}}\}$ is actually a singleton. Therefore, Lemma 3.3.32 tells us that also the terms $c_{\vec{N}}$ and the bags $[\vec{u}^i_{\vec{N}}]$ are independent from $\vec{N} \in \vec{\mathcal{X}}$. The unique element of the previous sigleton has hence shape $c^\bullet \langle\!| \vec{u}^i, \dots, \vec{u}^n |\!\rangle$, with $c^\bullet$ a rigid of a $c \in \mathcal{T}(C)$, and $\vec{u}^i$ a list of elements of $\mathcal{T}(L_i)$. Recalling now that $\sum_j u^i_j \subseteq \mathcal{T}(L_i)$, we can apply Corollary 3.3.26 in order to obtain, for each $i = 1, \dots, n$, an $L'_i \in \Lambda$ s.t. $L_i \twoheadrightarrow L'_i$ and, using (3.9), $\sum_j w^{ij} \subseteq \mathcal{T}(L'_i)$. Thus Lemma 3.3.33 tells us that, for every $\vec{N} \in \vec{\mathcal{X}}$, we have:

$$c^\bullet \langle\!| \vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}} |\!\rangle \in \mathcal{T}(C \langle\!| L'_1, \dots, L'_n |\!\rangle). \tag{3.11}$$

But now thanks to (3.11) and (3.8) (which holds for all $\vec{N} \in \vec{\mathcal{X}}$), we can apply again Lemma 3.3.20 in order to find that the set $\{c^\bullet \langle\!| \vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}} |\!\rangle \mid \vec{N} \in \vec{\mathcal{X}}\}$ is a singleton. Again by Lemma 3.3.32, we have that all the bags $[\vec{w}^1_{\vec{N}}], \dots, [\vec{w}^n_{\vec{N}}]$ for $\vec{N} \in \vec{\mathcal{X}}$, coincide respectively to some bags $[\vec{w}^1], \dots, [\vec{w}^n]$ which are independent from $\vec{N} \in \vec{\mathcal{X}}$. So the only element of the previous singleton has shape $c^\bullet \langle\!| \vec{w}^1, \dots, \vec{w}^n |\!\rangle$, and (3.8) becomes:

$$t \in \mathrm{nf_r}(c^\bullet \langle\!| \vec{w}^1, \dots, \vec{w}^n |\!\rangle). \tag{3.12}$$

Now for all $i = 1, \dots, n$, we already know that $[\vec{w}^i] = [\vec{w}^i_{\vec{N}}]$ which is a bag of elements of $\mathrm{NF}\mathcal{T}(N)$, and this holds for all $N \in \mathcal{X}_i$. That is, we have:

$$\sum_j \vec{w}^{ij} \subseteq \bigcap_{N \in \mathcal{X}_i} \mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(M_i) \tag{3.13}$$

where we finally used the hypothesis. From (3.12), (3.13) and Lemma 3.3.33 we finally conclude that $t \in \mathrm{nf_r}(c^\bullet \langle\!| \vec{w}^1, \dots, \vec{w}^n |\!\rangle) \subseteq \mathrm{NF}\mathcal{T}(C \langle\!| M_1, \dots, M_n |\!\rangle)$. □

We immediately obtain the non implementability of the following parallel-or, where we use the usual encoding of couples as $(M, N) := \lambda z.zMN \in \Lambda$.

Corollary 3.3.38. *There is* no $\mathtt{Por} \in \Lambda$ *s.t. for all* $M, N \in \Lambda$,

$$\begin{cases} \mathtt{Por}(M, N) =_\tau \mathtt{True} & \text{if } M \neq_\tau \Omega \text{ or } N \neq_\tau \Omega \\ \mathtt{Por}(M, N) =_\tau \Omega & \text{if } M =_\tau N =_\tau \Omega. \end{cases}$$

Proof. Otherwise, for the context $C := \mathtt{Por}\,\xi$, the set $\mathcal{X} := \{(\mathtt{True}, \Omega), (\Omega, \mathtt{True})\} \subseteq \Lambda$ which is bounded by $(\mathtt{True}, \mathtt{True}) \in \Lambda$, Stability (Theorem 3.3.37) would give the contradiction:

$$\mathtt{True} =_\tau C \langle\!| (\mathtt{True}, \Omega) |\!\rangle \cap C \langle\!| (\Omega, \mathtt{True}) |\!\rangle =_\tau C \langle\!| (\Omega, \Omega) |\!\rangle =_\tau \Omega. \qquad \square$$

The Perpendicular Lines Property The perpendicular lines property (PLP, for short) states that if a term $\lambda z_1 \dots \lambda z_n.F$ is constant modulo $=_\tau$ on n perpendicular lines, then it must be constant modulo $=_\tau$ everywhere. Let us explain what does this terminology mean:

- When we say a "line" we mean a notion miming, in λ-calculus, the one of line parallel to the axes in Cartesian geometry: we fix all the coordinates but one. That is, a "λ-calculus line" l is a subset $l = \{(M_1, \dots, M_{i-1}, Z, M_{i+1}, \dots, M_n) \in \Lambda^n \mid Z \in \Lambda\}$ of Λ^n (which corresponds to a line parallel to the i-th axe in geometry). We will denote it is simply by:

$$l : (M_1, \dots, M_{i-1}, Z, M_{i+1}, \dots, M_n)_{Z \in \Lambda}.$$

- Miming again geometry, when we say that two lines l_1, l_2 are perpendicular, we just mean that the axis they are parallel to are different, i.e. the free coordinate of l_1 is different from the free coordinate of l_2. For example, the line $(\mathtt{I}, Z)_{Z \in \Lambda}$ is perpendicular to the line $(Z, \mathtt{II})_{Z \in \Lambda}$.

- Finally, when we say that $\lambda \vec{z}.F$ is constant modulo $=_\tau$ on a line l as before, we mean that the context function $(\lambda \vec{z}.F)\xi : (\Lambda/_{=_\tau})^n \to \Lambda/_{=_\tau}$ is constant on $l \subseteq (\Lambda/_{=_\tau})^n$. That is, there exist $N \in \Lambda$ s.t. $(\lambda \vec{z}.F)\ M_1\ \cdots\ M_{i-1}\ Z\ M_{i+1}\ \cdots\ M_n =_\tau N$ for all $Z \in \Lambda$.

With this terminology in place, the desired property is expressed exactly as in the statement of Theorem 3.3.40. It is important, however, to keep in mind that the geometric intuition mentioned above has a very short life and stops here: in λ-calculus there is no such notion as scalar product, and the best one can do is to reproduce the notion of perpendicularity but only in the trivial case of lines parallel to the axes.

As for Stability, we will use in the proof of PLP all the power of the technique of resource approximation. Indeed, resource terms can never behave too bad, the worst can happen being reducing to 0, and they must respect the strong constraints of linearity, strong normalization and confluence. For instance, let us consider how can a function $b \in !\,\Lambda^{\mathrm{r}} \mapsto \mathrm{nf}_{\mathrm{r}}((\lambda z.t)b) \in \mathscr{P}(\Lambda^{\mathrm{r}})$ be constant. The intuition is that there are few reasons:

1. either because $t \twoheadrightarrow_{\mathrm{r}} 0$ by itself
2. either because z's did not occur in t at all.

That is, we are left with only the trivial cases. These are the intuitions behind the Lemma 3.3.39 below, which is the crucial ingredient for the proof of PLP.

In the next section we will compare the situation just described with the much more complicated one present when asking the same question in $\Lambda/_{=_{\mathcal{B}}}$. Thanks to the *commutation formula*, the respective PLP are equivalent. The following proof could also be written in a coinductive fashion for $\Lambda/_{=_{\mathcal{B}}}$.

Lemma 3.3.39. *Let $C := (\lambda \vec{z}.F)\xi_1 \cdots \xi_n$ be an n-context and let $t \in \Lambda^{\mathrm{r}}$. Suppose that:*

i. $\mathrm{nf}_{\mathrm{r}}(t) \neq 0$

ii. $t \in \mathcal{T}(F)$

iii. there are $\{M_{ij}\}_{1 \leq i \neq j \leq n} \subseteq \Lambda$ s.t. C is constant, mod $=_\tau$, on the following perpendicular lines:

$$\begin{array}{l} l_1 = \{(Z,\ M_{12},\ \dots\dots,\ M_{1n}) \mid Z \in \Lambda\} \\ l_2 = \{(M_{21},\ Z,\ \dots\dots,\ M_{2n}) \mid Z \in \Lambda\} \\ \qquad\qquad \ddots \\ l_n = \{(M_{n1},\ \dots,\ M_{n(n-1)},\ Z) \mid Z \in \Lambda\}. \end{array} \tag{3.14}$$

Then:

$$\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0.$$

Proof. Induction on the size $\mathbf{sz}(t)$ of $t \in \Lambda^{\mathrm{r}}$.

Case $\mathbf{sz}(t) = 1$. Then t is a variable. Now if $t = z_i$ for some i, then the i-th line of (3.14) gives the contradiction:

$$N_i =_\tau (\lambda\vec{z}.z_i)M_{i1}\cdots M_{i(i-1)}ZM_{i(i+1)}\cdots M_{in} =_\tau Z$$

for all $Z \in \Lambda$. Hence, it must be $t \neq z_i$ for all i, i.e. $\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0$.

Case $\mathbf{sz}(t) > 1$. By (i) there is $u \in \mathrm{nf}_{\mathrm{r}}(t)$. Being u normal, it has shape $u = \lambda\vec{x}.y[\vec{u}^1]\cdots[\vec{u}^m]$ for some $m \geq 0$, some variable y, some normal bags $[\vec{u}^j]$. By (ii) $t \in \mathcal{T}(F)$, so that by Corollary 3.3.26 there is $Q \in \Lambda$ s.t. $F \twoheadrightarrow Q$ and $u \in \mathcal{T}(Q)$. So Q must have shape: $Q = \lambda\vec{x}.yQ_1\cdots Q_m$ for some Q_j's in Λ s.t. $[\vec{u}^j] \in !\mathcal{T}(Q_j)$ for all $j = 1,\ldots,m$. Now there are two possibilities: either $y = z_i$ for some $i = 1,\ldots,n$, either $y \neq z_i$ for all i. Suppose $y = z_i$. Then, for $\vec{q} := q_1,\ldots,q_m$ fresh variables, we can chose $Z := \lambda\vec{q}.\mathtt{True} \in \Lambda$ (or $Z := \mathtt{True}$ if $m = 0$) in the i-th line l_i of (3.14), and since by (iii) $\lambda\vec{z}.F$ is constant (mod $=_\tau$) on l_i, we can compute its value as:

$$\begin{array}{rcl}(\lambda\vec{z}.F)M_{i1}\cdots M_{i(i-1)}\,(\lambda\vec{q}.\mathtt{True})\,M_{i(i+1)}\cdots M_{in} & =_\tau & Q\{M_{i1}/z_1,\ldots,(\lambda\vec{q}.\mathtt{True})/z_i,\ldots,M_{in}/z_n\}\\ & =_\tau & \lambda\vec{x}.(\lambda\vec{q}.\mathtt{True})\widetilde{Q_{i1}}\cdots\widetilde{Q_{im}}\\ & =_\tau & \lambda\vec{x}.\mathtt{True}\end{array}$$

where we set:

$$\widetilde{Q_{ij}} := Q_j\{M_{i1}/z_1,\ldots,(\lambda\vec{q}.\mathtt{True})/z_i,\ldots,M_{in}/z_n\}.$$

The first equality holds because $F \twoheadrightarrow Q$ and $=_\tau$ is finer than $=_{\boldsymbol{\lambda}}$, and the second equality holds because $y = z_i$. In the same way, choosing $Z := \lambda\vec{q}.\mathtt{False} \in \Lambda$ in l_i we find that the value (mod $=_\tau$) of $\lambda\vec{z}.F$ on l_i is $\lambda\vec{x}.\mathtt{False}$. But this is impossible because $\mathtt{True} \neq_\tau \mathtt{False}$.

Therefore, it must be $y \neq z_i$ for all i. Note that wlog $m \geq 1$ (indeed if $m = 0$, from the fact that $y \neq z_i$ for all i we already get $\deg_{z_i}(u) = 0$ and, as $u \in \mathrm{nf}_{\mathrm{r}}(t)$ and in Λ^{r} one cannot erase non-empty bags, we are done). Now fix $i \in \{1,\ldots,n\}$ and $Z', Z'' \in \Lambda$. Similarly as before, choosing $Z := Z'$ in l_i and using what we found so far, putting

$$Q'_{ij} := Q_j\{M_{i1}/z_1,\ldots,Z'/z_i,\ldots,M_{in}/z_n\}$$

since $\lambda\vec{z}.F$ is constant (mod $=_\tau$) on l_i, we can compute its value as:

$$\begin{array}{rcl}(\lambda\vec{z}.F)M_{i1}\cdots M_{i(i-1)}Z'M_{i(i+1)}\cdots M_{in} & =_\tau & Q\{M_{i1}/z_1,\ldots,Z'/z_i,\ldots,M_{in}/z_n\}\\ & =_\tau & \lambda\vec{x}.yQ'_{i1}\cdots Q'_{im}\end{array}$$

where the last equality holds since y is *not* one of the z_i's. Choosing Z'' instead of Z' and putting Q''_{ij} the same as Q'_{ij} but with Z'' instead of Z', one has that the value (mod $=_\tau$) of $\lambda\vec{z}.F$ on l_i is $\lambda\vec{x}.yQ''_{i1}\cdots Q''_{im}$. Therefore we have $\lambda\vec{x}.yQ'_{i1}\cdots Q'_{im} = \lambda\vec{x}.yQ''_{i1}\cdots Q''_{im}$, and Lemma 3.3.12 entails:

$$\begin{array}{c}Q'_{i1} =_\tau Q''_{i1}\\ \vdots\\ Q'_{im} =_\tau Q''_{im}\end{array}$$

But by construction it is:

$$\begin{array}{l}Q'_{ij} =_\tau (\lambda\vec{z}.Q_j)M_{i1}\cdots M_{i(i-1)}Z'M_{i(i+1)}\cdots M_{in}\\ Q''_{ij} =_\tau (\lambda\vec{z}.Q_j)M_{i1}\cdots M_{i(i-1)}Z''M_{i(i+1)}\cdots M_{in}.\end{array}$$

So if we remember that Z', Z'' were generic in Λ, the previous equalities $Q'_{ij} = Q''_{ij}$ precisely say that $\lambda\vec{z}.Q_j$ is constant on the line l_i. And since this holds for all $i = 1, \dots, n$, we have just found that $\lambda\vec{z}.Q_j$ satisfies (iii). And since we have equalities $Q'_{ij} = Q''_{ij}$ for all $j = 1, \dots, m$, we have that each $\lambda\vec{z}.Q_1, \dots, \lambda\vec{z}.Q_k$ satisfies (iii). We can now comfortably apply the induction hypothesis on any $s \in [\vec{u}^j]$. In fact, as $[\vec{u}^j]$ is normal, $\mathrm{nf_r}(s) \neq 0$, i.e. (i); as $[\vec{u}^j] \in !\mathcal{T}(Q_j)$, we have $s \in \mathcal{T}(Q_j)$, i.e. (ii); and we just found that $\lambda\vec{z}.Q_j$ satisfies (iii); finally, s is a strict subterm of $u \in \mathrm{nf_r}(t)$, thus $\mathrm{ms}(s) < \mathrm{ms}(u) \leq \mathrm{ms}(t)$. Therefore, the inductive hypothesis gives $\deg_{z_1}(s) = \cdots = \deg_{z_n}(s) = 0$. Since this is true for all s in all $[\vec{u}^j]$, $j = 1, \dots, m$, we get $\deg_{z_1}(u) = \cdots = \deg_{z_n}(u) = 0$. And now $u \in \mathrm{nf_r}(t)$ entails $\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0$. □

Now we can safely prove that the term algebra $\Lambda/_{=_\tau}$ enjoys the PLP.

Theorem 3.3.40 (Perpendicular lines property)**.** *Suppose that for some fixed* $\{M_{ij}\}_{1\leq i\neq j\leq n}$, $\{N_i\}_{1\leq i\leq n} \subseteq \Lambda$, *the system of equations:*

$$\begin{cases} (\lambda z_1 \cdots z_n.F)\ \ Z\ M_{12}\ \cdots\cdots\ M_{1n} &=_\tau\ N_1 \\ (\lambda z_1 \cdots z_n.F)\ M_{21}\ Z\ \ \cdots\cdots\ M_{2n} &=_\tau\ N_2 \\ \qquad\qquad\qquad\qquad \ddots \quad\ \vdots & \\ (\lambda z_1 \cdots z_n.F)\ M_{n1}\ \cdots\ M_{n(n-1)}\ Z &=_\tau\ N_n \end{cases}$$

holds for all $Z \in \Lambda$. *Then:*

$$(\lambda z_1 \cdots z_n.F)Z_1 \cdots Z_n =_\tau N_1$$

for all $Z_1, \dots, Z_n \in \Lambda$.

Proof. Fix $i \in \{1, \dots, n\}$ and $\vec{Z} \in \Lambda \times \cdots \times \Lambda$, and let us prove that $\mathrm{NF}\mathcal{T}((\lambda\vec{z}.F)\vec{Z}) = \mathrm{NF}\mathcal{T}(N_i)$.

(⊆). If $s \in \mathrm{NF}\mathcal{T}((\lambda\vec{z}.F)\vec{Z})$ then $s \in \mathrm{nf_r}(t')$ for some $t' = (\lambda\vec{z}.t)[\vec{u}^1]\cdots[\vec{u}^n]$, for some $t \in \mathcal{T}(F)$ and bags $[\vec{u}^j] \in !\mathcal{T}(Z_j)$. Since $t' \not\twoheadrightarrow_{\mathrm{r}} 0$ then $\mathrm{nf_r}(t) \neq 0$. Moreover, $\lambda\vec{z}.F$ is constant (mod $=_\tau$) on n perpendicular lines by hypothesis. Therefore t satisfies all the hypothesis of Lemma 3.3.39, which gives $z_1 \dots, z_n \notin t$. But the only possibility for that situation to cohabit with $t' \not\twoheadrightarrow_{\mathrm{r}} 0$ is that all $[\vec{u}^j]$'s trivialize to the empty bag: $[\vec{u}^1] = \cdots = [\vec{u}^n] = 1$. But then, $t' = (\lambda\vec{z}.t)1\cdots 1 \in \mathcal{T}((\lambda\vec{z}.F)M_{i1}\cdots M_{i(i-1)}Z_iM_{i(i+1)}\cdots M_{in})$ and thus $s \in \mathrm{NF}\mathcal{T}((\lambda\vec{z}.F)M_{i1}\cdots M_{i(i-1)}Z_iM_{i(i+1)}\cdots M_{in}) \subseteq \mathrm{NF}\mathcal{T}(N_i)$ by the i-th line of the system of equations of the hypothesis.

(⊇). If $s \in \mathrm{NF}\mathcal{T}(N_i) \subseteq \mathrm{NF}\mathcal{T}((\lambda\vec{z}.F)M_{i1}\cdots M_{i(i-1)}Z_iM_{i(i+1)}\cdots M_{in})$ (the inclusion is by the i-th line of the system of equations of the hypothesis), then $s \in \mathrm{nf_r}((\lambda\vec{z}.t)[\vec{u}^1]\cdots[\vec{u}^n])$ for some $t \in \mathcal{T}(F)$, bags $[\vec{u}^j] \in !\mathcal{T}(M_{ij})$ for $j \neq i$ and $[\vec{u}^i] \in !\mathcal{T}(Z_i)$. In the exact same way as before Lemma 3.3.39 gives $z_1 \dots, z_n \notin t$, and since $(\lambda\vec{z}.t)[\vec{u}^1]\cdots[\vec{u}^n] \not\twoheadrightarrow_{\mathrm{r}} 0$, the only possibility is that $[\vec{u}^1] = \cdots = [\vec{u}^n] = 1$. So $t' = (\lambda\vec{z}.t1\cdots 1) \in \mathcal{T}((\lambda\vec{z}.F)\vec{Z})$, from which $s \in \mathrm{NF}\mathcal{T}((\lambda\vec{z}.F)\vec{Z})$. □

We immediately obtain the non implementability of the following parallel-or:

Corollary 3.3.41. *There is* no $\mathtt{Por}' \in \Lambda$ *s.t. for all* $M, N \in \Lambda$,

$$\begin{cases} \mathtt{Por}'MN &=_\tau\ \mathtt{True} & \text{if } M \neq_\tau \Omega \text{ or } N \neq_\tau \Omega \\ \mathtt{Por}'MN &=_\tau\ \Omega & \text{if } M =_\tau N =_\tau \Omega. \end{cases}$$

Proof. Such a $\mathtt{Por}'$ would be such that $(\lambda xy.\mathtt{Por}'xy)\,\mathtt{I}\,Z =_\tau (\lambda xy.\mathtt{Por}'xy)\,Z\,\mathtt{I} =_\tau \mathtt{I}$ for all $Z \in \Lambda$ while $(\lambda xy.\mathtt{Por}'xy)\,\Omega\,\Omega =_\tau \Omega$, thus contradicting PLP (Theorem 3.3.40). □

Actually, the existence of this $\texttt{Por}'$ is equivalent to that of the $\texttt{Por}$ of Corollary 3.3.38. Indeed, one can take $\texttt{Por} := \lambda c.\texttt{Por}'(\texttt{True}\, c)(\texttt{False}\, c)$ and $\texttt{Por}' := \lambda x'y'.\texttt{Por}(x', y')$.

PLP also immediately entails the non implementability of the following other version of the parallel-or:

Corollary 3.3.42. *There is* no $\mathtt{Por}'' \in \Lambda$ *s.t. for all* $Z \in \Lambda$ *one has:*

$$\begin{cases} \mathtt{Por}''\ \mathtt{True}\, Z =_\tau \mathtt{True} \\ \mathtt{Por}''\ Z\ \mathtt{True} =_\tau \mathtt{True} \\ \mathtt{Por}''\ \mathtt{False}\,\mathtt{False} =_\tau \mathtt{False}. \end{cases}$$

3.4 Interlude: Call-by-value λ-calculus

In this "interlude" we consider the *call-by-value* λ-calculus (cbv-λ-calculus, for short), introduced long time ago in [Plo75]. In this section we will sometimes refer, as usual, to the ordinary λ-calculus as the "cbn" one.

We study the cbv framework via the tool of resource approximation. In fact, it turns out we can adapt the constructions already presented for the (cbn-)λ-calculus in order to prove the *Stability Property* in the cbv-λ-calculus. We refer [KMP20] for the basic definitions and the construction of the cbv-resource calculus. Their constructions follow the same schema of the ones we presented in Section 3.3, with some little technical modifications due to the fact that in the cbv setting one linearises terms in a slightly different way.

We first recall the main properties of cbv-resource-λ-calculus. Then, we show how to adapt the constructions of rigid terms associated with resource terms and prove the Stability property (Theorem 3.4.20) in the cbv framework.

Definition 3.4.1 (Cbv-λ-calculus)**.** *The set* Λ_{cbv} *of the cbv-λ-terms is the same as the set of (ordinary call-by-name) λ-terms. The set* Val *of* values *is the set containing the variables and abstractions. Contexts (with multiple holes, one hole and single-hole) are defined as in cbn-λ-calculus. The reduction* $\to_{\mathrm{v}} \subseteq \Lambda_{\mathrm{cbv}} \times \Lambda_{\mathrm{cbv}}$ *of cbv-λ-calculus is defined in Definition 1.5 of [KMP20]. It is known that* $\to_{\mathrm{v}}$ *is confluent.*

Resource cbV-λ-calculus in a nutshell

Definition 3.4.2 (Resource cbv-λ-calculus)**.** *The set* $\Lambda^{\mathrm{r}}_{\mathrm{cbv}}$ *of the resource-cbv-λ-terms is defined in Definition 3.1 of [KMP20]. We report it here:*

$$\Lambda^{\mathrm{r}}_{\mathrm{cbv}} := \mathrm{Val}^{\mathrm{r}} \cup \mathrm{Simp}^{\mathrm{r}}$$

where the set $\mathrm{Val}^{\mathrm{r}}$ *or* resource values *and the set* $\mathrm{Simp}^{\mathrm{r}}$ *of* resource simple terms *are defined by mutual induction by:*

$$\begin{array}{lcll} \mathrm{Val}^{\mathrm{r}} & ::= & x \mid \lambda x.s & \text{for} \quad s \in \mathrm{Simp}^{\mathrm{r}} \\ \mathrm{Simp}^{\mathrm{r}} & ::= & s_1 s_2 \mid [v_1, \dots, v_n] & \text{for} \quad s_i \in \mathrm{Simp}^{\mathrm{r}} \text{ and } v_i \in \mathrm{Val}^{\mathrm{r}}. \end{array}$$

The reduction $\to_{\mathrm{r}}$ *of resource-cbv-calculus is defined in Definitions 3.3 and 3.4 of [KMP20].*

We do not report the definition of the reduction because we will not need its actual definition but only the properties it verifies. For instance, it is known that:

Proposition 3.4.3. *The reduction* $\to_{\mathrm{r}}$ *is confluent and strongly normalising.*

Definition 3.4.4 ((Qualitative) cbv-Taylor expansion)**.** *The* (qualitative) cbv-Taylor expansion *is the map* $\mathcal{T} : \Lambda_{\mathrm{cbv}} \to \mathcal{P}(\mathrm{Simp}^{\mathrm{r}})$ *defined in Definition 3.9 of [KMP20]. We report it here:*

$$\begin{array}{rcl} \mathcal{T}(x) & := & \{\, [x, \overset{(n)}{\ldots}, x] \mid n \in \mathbb{N}\} \\ \mathcal{T}(\lambda x.M) & := & \{\, [\lambda x.s_1, \ldots, \lambda x.s_n] \mid n \in \mathbb{N} \text{ and } s_i \in \mathcal{T}(M)\} \\ \mathcal{T}(M_1 M_2) & := & \{s_1 s_2 \mid s_i \in \mathcal{T}(M_i)\}. \end{array}$$

One defines as usual the set $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{nf}_{\mathrm{r}}(\mathrm{Simp}^{\mathrm{r}})$*. The inclusion* $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N)$ *defines as usual a partial preorder* $M \leq N$*, as well as the equivalence* $=_\tau$*, which is its symmetric closure.*

Remark 3.4.5. *One has* $\mathrm{nf}_{\mathrm{v}}(MN) = \mathrm{nf}_{\mathrm{v}}(\mathrm{nf}_{\mathrm{v}}(M)\mathrm{nf}_{\mathrm{v}}(N))$ *and* $\mathrm{nf}_{\mathrm{v}}(\lambda x.M) = \lambda x.\mathrm{nf}_{\mathrm{v}}(M)$*, whenever the written normal forms do exist. Also:*

1. $\mathrm{NF}\mathcal{T}(\lambda x.M) := \{\, [\lambda x.s_1, \ldots, \lambda x.s_n] \mid n \in \mathbb{N} \text{ and } s_i \in \mathrm{NF}\mathcal{T}(M)\}$
2. $\mathrm{NF}\mathcal{T}(MN) := \bigcup_{s_i \in \mathrm{NF}\mathcal{T}(M)} \mathrm{nf}_{\mathrm{r}}(s_1 s_2)$.

We can now prove the Monotonicity property for cbv-λ-calculus. It is already proven in [KMP20] (where it is called "Context Lemma") but we provide below a more concise proof, very similar to the one already given in the case of λ-calculus.

Theorem 3.4.6 (Monotonicity property)**.** *Any context* $C : \Lambda_{\mathrm{cbv}} \to \Lambda_{\mathrm{cbv}}$ *is monotone w.r.t.* $\leq$*.*

Proof. Induction on C.
If $C = \xi$ then $\mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) = \mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle)$.
If $C = x$ then $\mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) = \{x\} = \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle)$.
If $C = \lambda x.C_1$ then:

$$\begin{array}{rcl} \mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) & = & \{\, [\lambda x.s_1, \ldots, \lambda x.s_n] \mid n \in \mathbb{N} \text{ and } s_i \in \mathrm{NF}\mathcal{T}(C_1\langle\!|M|\!\rangle)\} \\ & \subseteq & \{\, [\lambda x.s_1, \ldots, \lambda x.s_n] \mid n \in \mathbb{N} \text{ and } s_i \in \mathrm{NF}\mathcal{T}(C_1\langle\!|N|\!\rangle)\} \\ & = & \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle). \end{array}$$

If $C = C_1 C_2$ then:

$$\begin{array}{rcl} \mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) & = & \bigcup_{s_i \in \mathrm{NF}\mathcal{T}(C_i\langle\!|M|\!\rangle)} \mathrm{nf}_{\mathrm{r}}(s_1 s_2) \\ & \subseteq & \bigcup_{s_i \in \mathrm{NF}\mathcal{T}(C_i\langle\!|N|\!\rangle)} \mathrm{nf}_{\mathrm{r}}(s_1 s_2) \\ & = & \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle). \end{array}$$

□

As always, we can simulate $\to_{\mathrm{v}}$ via $\twoheadrightarrow_{\mathrm{r}}$.

Proposition 3.4.7 (Resource simulation property)**.** *If* $M \to_{\mathrm{v}} N$ *then:*

1. *for all* $s \in \mathcal{T}(M)$ *there exist* $\mathbb{T} \subseteq \mathcal{T}(N)$ *s.t.* $s \twoheadrightarrow_{\mathrm{r}} \mathbb{T}$
2. *for all* $s' \in \mathcal{T}(N)$ *s.t.*[17] $s' \not\to_0 0$*, there exist* $s \in \mathcal{T}(M)$ *s.t.* $s \twoheadrightarrow_{\mathrm{r}} s' + \mathbb{T}$ *for some sum* $\mathbb{T}$*.*

Proof. See Lemma 4.4 of [KMP20]. □

[17] The following condition $s' \not\to_0 0$ refers to a particular reduction $\to_0$, which we did not specify, that one considers in cbv-λ-calculus.

As in cbn-λ-calculus, we can define a cbv-λ-theory as a congruence which contains the reflexive symmetric and transitive closure $=_{\mathrm{v}}$ of $\to_{\mathrm{v}}$.

Corollary 3.4.8 (Taylor normal form cbv-λ-theory)**.** *The equivalence $=_\tau$ is a cbv-λ-theory.*

Proof. It is a congruence thanks to the Monotonicity property, and the Simulation property allows to prove as usual that it contains $=_{\mathrm{v}}$ (this last fact is already present as Corollary 4.5 of [KMP20]). □

We state now some useful properties for the proof of Theorem 3.4.20 .

Remark 3.4.9. *If $V \in \mathrm{Val}$ then $\mathcal{T}(V) \subseteq !\,\mathrm{Val}^{\mathrm{r}}$.*

Remark 3.4.10. *Let $t \in \Lambda^{\mathrm{r}}_{\mathrm{cbv}}$ normal and belonging to $\mathcal{T}(M)$. If $M \to_{\mathrm{v}} N$ then $t \in \mathcal{T}(N)$.*

Proposition 3.4.11. *If $t \in \mathrm{NF}\mathcal{T}(M)$, there exist $N \in \Lambda_{\mathrm{cbv}}$ s.t. $M \twoheadrightarrow_{\mathrm{v}} N$ and $t \in \mathcal{T}(N)$.*

Proof. See Lemma 4.9(2) of [KMP20]. □

Call-by-value λ-calculus enjoys the same "non-interference" property we already encoutered for λ-calculus:

Proposition 3.4.12. *For all $t, s \in \mathcal{T}(M)$ s.t. $t \neq s$, you have $\mathrm{nf}_{\mathrm{r}}(t) \cap \mathrm{nf}_{\mathrm{r}}(s) = \emptyset$.*

Proof. See Lemma 4.9(3) of [KMP20]. □

Rigids

We adapt now the notion of rigid terms associated with a resource term (or, in general, with a multi-hole context) in the current cbv-setting.

Definition 3.4.13 (Cbv-resource-contexts)**.** *The set $\mathrm{Cbv}^{\mathrm{r}}_k$ of* cbv-resource-k-contexts *is defined as follows:*

$$\mathrm{Cbv}^{\mathrm{r}}_k := \mathrm{Val}^{\mathrm{r}}_k \cup \mathrm{Simp}^{\mathrm{r}}_k$$

where $\mathrm{Val}^{\mathrm{r}}_k$ and $\mathrm{Simp}^{\mathrm{r}}_k$ are defined by mutual induction by:

$$\begin{array}{lcll} \mathrm{Val}^{\mathrm{r}}_k & ::= & \xi_1 \mid \cdots \mid \xi_k \mid x \mid \lambda x.c^s & \text{for } \ c^s \in \mathrm{Simp}^{\mathrm{r}}_k \\ \mathrm{Simp}^{\mathrm{r}}_k & ::= & c^s_1 c^s_2 \mid [c^v_1, \ldots, c^v_n] & \text{for } \ c^s_i \in \mathrm{Simp}^{\mathrm{r}}_k \text{ and } c^v_j \in \mathrm{Val}^{\mathrm{r}}_k. \end{array}$$

We extend the definition of Taylor expansion on each Cbv_k by setting:

$$\mathcal{T}(\xi_i) := \{\, [\xi_i, \overset{(n)}{\ldots}, \xi_i] \mid n \in \mathbb{N} \} \subseteq \mathrm{Simp}^{\mathrm{r}}_k.$$

Definition 3.4.14 (Rigid cbv-λ-terms)**.** *1. A* rigid-k-context *is built as a resource-k-context but taking lists of rigid terms instead of bags of resource terms. In particular, a rigid term is a rigid context with no occurrences of the holes. As for cbv-terms, rigid contexts are divided into rigid-value-contexts and rigid-simple-contexts (and of course this distincion coincides with that of terms when a context has no holes).*

2. Let now c be a resource-k-context. We define a set $\mathrm{Rigid}(c)$ *of* rigid k-contexts *associated with c, whose elements are called the* rigids of c*, by mutual induction on $\mathrm{Val}^{\mathrm{r}}_k$ and $\mathrm{Simp}^{\mathrm{r}}_k$ as follows:*

$$\mathrm{Rigid}(\xi_i) = \{\xi_i\}$$
$$\mathrm{Rigid}(x) = \{x\}$$

$$\mathrm{Rigid}(\lambda x.c_0) = \{\lambda x.c_0^\bullet \mid c_0^\bullet \in \mathrm{Rigid}(c_0)\}$$
$$\mathrm{Rigid}(c_0 c_1) = \{c_0^\bullet c_1^\bullet \mid c_i^\bullet \in \mathrm{Rigid}(c_i)\}$$
$$\mathrm{Rigid}([c_1,\dots,c_k]) = \{\langle c^\bullet_{\sigma(1)} \dots, c^\bullet_{\sigma(k)}\rangle \mid \sigma \textit{ permutation and } c_i^\bullet \in \mathrm{Rigid}(c_i)\}.$$

The above definition makes sense since one immediately sees that if c is a resource-k-value-context (resp. resource-k-simple-context) then any of its rigids $c^\bullet$ is a rigid-k-value-context (resp. rigid-k-simple-context).

Definition 3.4.15. *Let $c^\bullet$ be a rigid of a cbv-resource-k-context c and, for $i = 1,\dots,k$, let $\vec{v}^{\,i} := \langle v_1^i,\dots,v^i_{\deg_{\xi_i}(c)}\rangle$ be a list*[18] *of resource* values *(that is, elements of* $\mathrm{Val}^{\mathrm{r}}$*). We define, by mutual induction on* $\mathrm{Val}^{\mathrm{r}}_k$ *and* $\mathrm{Simp}^{\mathrm{r}}_k$*, a resource term $c^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) \in \Lambda^{\mathrm{r}}_{\mathrm{cbv}}$ s.t. if $c \in \mathrm{Val}^{\mathrm{r}}_k$ (resp. $\in \mathrm{Simp}^{\mathrm{r}}_k$) then $c^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) \in \mathrm{Val}^{\mathrm{r}}$ (resp. $\in \mathrm{Simp}^{\mathrm{r}}$). The definition goes as follows:*

1. *If $c = \xi_i$ then $c^\bullet = \xi_i$; we set $c^\bullet(\!|\langle\rangle,\dots,\langle\rangle,\langle v_1^i\rangle,\langle\rangle,\dots,\langle\rangle|\!) := v_1^i$*
2. *If $c = x$ then $c^\bullet = x$; we set $c^\bullet(\!|\langle\rangle,\dots,\langle\rangle|\!) := x$*
3. *If $c = \lambda x.c_0$ then $c^\bullet = \lambda x.c_0^\bullet$ where $c_0^\bullet$ is a rigid of c_0; we set $c^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) = \lambda x.c_0^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!)$*
4. *If $c = c_1c_2$, then $c^\bullet = c_1^\bullet c_2^\bullet$ where $c_i^\bullet$ is a rigid of c_i. So each list $\vec{v}^{\,i}$ factorizes as a concatenation $\vec{w}^{\,i1}\vec{w}^{\,i2}$ of lists where $\vec{w}^{\,ij}$ has exactly $\deg_{\xi_i}(c_j)$ elements; we set:*
$$c^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) := c_1^\bullet(\!|\vec{w}^{11},\dots,\vec{w}^{k1}|\!)c_2^\bullet(\!|\vec{w}^{12},\dots,\vec{w}^{k2}|\!).$$
5. *If $c = [c_1,\dots,c_n]$, then $c^\bullet = \langle c^\bullet_{\sigma(1)},\dots,c^\bullet_{\sigma(n)}\rangle$ where σ is a permutation and $c_i^\bullet$ is a rigid of c_i. So each list $\vec{v}^{\,i}$ factorizes as a concatenation $\vec{w}^{\,i1}\cdots\vec{w}^{\,in}$ of lists where $\vec{w}^{\,ij}$ has exactly $\deg_{\xi_i}(c_\sigma(j))$ elements; we set:*
$$c^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) := [c^\bullet_{\sigma(1)}(\!|\vec{w}^{\,11},\dots,\vec{w}^{\,k1}|\!),\dots,c^\bullet_{\sigma(n)}(\!|\vec{w}^{\,1n},\dots,\vec{w}^{\,kn}|\!)].$$

Remark 3.4.16. *One has that if $v \twoheadrightarrow_{\mathrm{r}} \mathbb{V}$ then:*

$$c^\bullet(\!|\cdots,\langle\cdots,v,\cdots\rangle,\cdots|\!) \in \Lambda^{\mathrm{r}}_{\mathrm{cbv}} \twoheadrightarrow_{\mathrm{r}} \sum_{w\in\mathbb{V}} c^\bullet(\!|\cdots,\langle\cdots,w,\cdots\rangle,\cdots|\!).$$

The following lemma will be used in the proof of Theorem 3.4.20. As always, if $\vec{v}$ is a list we denote with $[\vec{v}]$ the multiset associated with $\vec{v}$ (same elements but disordered).

Lemma 3.4.17. *Let C be a k-context and $c_1, c_2 \in \mathcal{T}(C)$ (hence, in particular, c_1, c_2 are resource-k-contexts). Let $c_1^\bullet$ and $c_2^\bullet$ rigids respectivly of c_1 and c_2. For $i = 1\dots,k$, let $\vec{v}^{\,i} = \langle v_1^i,\dots,v^i_{\deg_{\xi_i}(c_1)}\rangle$ and $\vec{u}^{\,i} = \langle u_1^i,\dots,u^i_{\deg_{\xi_i}(c_2)}\rangle$ be lists of resource values. If $c_1^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) = c_2^\bullet(\!|\vec{u}^1,\dots,\vec{u}^k|\!)$ then $c_1 = c_2$ and $[\vec{v}^{\,i}] = [\vec{u}^{\,i}]$ for all i.*

Proof. Simple induction on C.

Case $C = \xi_i$. Then $c_1 = [\xi_i,\overset{(n)}{\dots},\xi_i]$, $c_2 = [\xi_i,\overset{(m)}{\dots},\xi_i]$, $\vec{v}^{\,i} = \langle v^{i1},\dots,v^{in}\rangle$, $\vec{u}^{\,i} = \langle u^{i1},\dots,u^{im}\rangle$ and $\vec{v^j} = \langle\rangle = \vec{u^j}$ for $j \neq i$. So $[v^{i1},\dots,v^{in}] = c_1^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) = c_2^\bullet(\!|\vec{u}^1,\dots,\vec{u}^k|\!) = [u^{i1},\dots,u^{im}]$, thus $n = m$, i.e. $c_1 = c_2$.

Case $C = x$. Then $c_1 = [x,\overset{(n)}{\dots},x]$, $c_2 = [x,\overset{(m)}{\dots},x]$ and $\vec{v}^{\,i} = \langle\rangle = \vec{u}^{\,i}$. So $[x,\overset{(n)}{\dots},x] = c_1^\bullet(\!|\vec{v}^1,\dots,\vec{v}^k|\!) = c_2^\bullet(\!|\vec{u}^1,\dots,\vec{u}^k|\!) = [x,\overset{(m)}{\dots},x]$, thus $n = m$, i.e. $c_1 = c_2$.

[18] If $\deg_{\xi_i}(c) = 0$ we mean the empty list.

Case $C = \lambda x.C_0$. Then, for $i = 1, 2$, one has $c_i = [\lambda x.c_{i1}, \dots, \lambda x.c_{in_i}]$ with $c_{ij} \in \mathcal{T}(C_0)$ for all i, j. So $c_i^\bullet = [\lambda x.c^\bullet_{i\sigma_i(1)}, \dots, \lambda x.c^\bullet_{i\sigma_i(n_i)}]$ where σ_i is a permutation on n_i elements. By Definition 3.4.15 we have:

$$c_i^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) = [\,\lambda x.c^\bullet_{i\widetilde{\sigma}_i(1)}(\!|\vec{w}^{i11}, \dots, \vec{w}^{ik1}|\!), \dots, \lambda x.c^\bullet_{i\widetilde{\sigma}_i(n_i)}(\!|\vec{w}^{i1n_i}, \dots, \vec{w}^{ikn_i}|\!)]$$

where $\widetilde{\sigma}_i$ is some permutation on n_i elements and the concatenation $\vec{w}^{ij1} \cdots \vec{w}^{ijn_i}$ gives $\vec{v}^j$ if $i = 1$ and gives $\vec{u}^j$ if $i = 2$. From $c_1^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) = c_2^\bullet(\!|\vec{u}^1, \dots, \vec{u}^k|\!)$ we get then $n_1 = n_2 =: n$ and that there exist a permutation ρ on n elements which identifies each term of the bag $c_1^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!)$ with the respective one of the bag $c_2^\bullet(\!|\vec{u}^1, \dots, \vec{u}^k|\!)$. That is, for all $j = 1, \dots, n$, one has:

$$c^\bullet_{1j}(\!|\vec{w}^{1\,1\,\widetilde{\sigma_1}^{-1}(j)}, \dots, \vec{w}^{1\,k\,\widetilde{\sigma_1}^{-1}(j)}|\!) = c^\bullet_{2\rho(j)}(\!|\vec{w}^{2\,1\,\widetilde{\sigma_2}^{-1}(\rho(j))}, \dots, \vec{w}^{2\,k\,\widetilde{\sigma_2}^{-1}(\rho(j))}|\!).$$

The inductive hypothesis gives $c^\bullet_{1j} = c^\bullet_{2\rho(j)}$ for all j, and, reeumerating the indeces, $\vec{w}^{1\,i\,j} = \vec{w}^{2\,i\,\widetilde{\sigma_2}^{-1}(\rho(\widetilde{\sigma_1}(j)))}$ for all $i = 1, \dots, k$. Now, the former equality gives $c_1^\bullet = c_2^\bullet$, while the latter gives $[\vec{v}^j] = [\vec{w}^{1j1} \cdots \vec{w}^{1jn}] = [\vec{w}^{2j1} \cdots \vec{w}^{2jn}] = [\vec{u}^j]$.

Case $C = C'C''$. Analogous and easier than the above case. □

Lemma 3.4.18. *Let C be a k-context and $V_1, \dots, V_k \in \mathrm{Val}$. Then:*

$$\mathcal{T}(C(\!|V_1, \dots, V_k|\!)) = \{c^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) \mid c \in \mathcal{T}(C),\ c^\bullet \text{ rigid of } c,\ [\vec{v}^i] \in \mathcal{T}(V_i)\}.$$

Proof. Induction on C. The base case $C = x$ and the step cases of the induction are straightforward. We only show the remaining base case, that is the one for $C = \xi_i$. We have:

$$\begin{array}{ll} & \{c^\bullet(\!|\vec{v}^1, \dots, \vec{v}^k|\!) \mid c \in \mathcal{T}(C),\ c^\bullet \text{ rigid of } c,\ [v_1^i, \dots, v^i_{\deg_{\xi_i}(c)}] \in \mathcal{T}(V_i)\} \\ = & \{\langle \xi_i, \overset{(n)}{\dots}, \xi_i\rangle(\!|\langle\rangle, \overset{(i-1)}{\dots}, \langle\rangle, \langle v_1^i, \dots, v_n^i\rangle, \langle\rangle, \overset{(k-i)}{\dots}, \langle\rangle|\!) \mid n \in \mathbb{N},\ [v_1^i, \dots, v_n^i] \in \mathcal{T}(V_i)\} \\ = & \{\,[v_1^i, \dots, v_n^i] \mid [v_1^i, \dots, v_n^i] \in \mathcal{T}(V_i)\} \\ = & \mathcal{T}(V_i) \\ = & \mathcal{T}(C(\!|V_1, \dots, V_k|\!)). \end{array}$$

□

Stability

Definition 3.4.19. *Given a non-empty subset $\mathcal{X} \subseteq \Lambda_{\mathrm{cbv}}$, define its $\mathcal{T}$-*infimum $\bigwedge \mathcal{X} \subseteq \Lambda^{\mathrm{r}}$ *as:*

$$\bigwedge \mathcal{X} := \bigcap_{M \in \mathcal{X}} \mathrm{NF}\mathcal{T}(M).$$

We say that $\mathcal{X}$ is bounded *iff there exists an $L \in \Lambda$ such that $M \leq L$ for all $M \in \mathcal{X}$.*

We write $M =_\tau \bigwedge \mathcal{X}$ instead of $\mathrm{NF}\mathcal{T}(M) = \bigwedge \mathcal{X}$.

In the cbv setting, the arguments passed to the functions are *values*. Since in the Stability theorem we are intuitively looking at contexts as functions, it makes sense to restrict their application on values only. Remark that this restriction was already present in the previous sections (see Definition 3.4.14, Definition 3.4.15, Lemma 3.4.17 and Lemma 3.4.18), and it is this restriction that makes the previous contructions work.

The proof of the following result is a modification of the proof in the cbn framework. The fact that one can adapt it in a cbv framework can be seen as an extra strenght of the orginal proof.

Theorem 3.4.20 (Stability property)**.** *Let C be an n-context and fix non-empty $\mathcal{X}_1, \dots, \mathcal{X}_n \subseteq \mathrm{Val}$ bouded by a value. For all $V_1, \dots, V_n \in \mathrm{Val}$ such that for $i = 1, \dots, n$ one has:*

$$V_i = \bigcap \mathcal{X}_i$$

we have:

$$C(\!|V_1, \dots, V_n|\!) = \bigcap_{\substack{N_1 \in \mathcal{X}_1 \\ \cdots \\ N_n \in \mathcal{X}_n}} C(\!|N_1, \dots, N_n|\!).$$

Proof. Since every $\mathcal{X}_i$ is $\mathcal{T}$-bounded, for $i = 1, \dots, n$ there exists $L_i \in \mathrm{Val}$ s.t.

$$\bigcup_{N \in \mathcal{X}_i} \mathrm{NF}\mathcal{T}(N) \subseteq \mathrm{NF}\mathcal{T}(L_i).$$

Fix now $V_1, \dots, V_n \in \mathrm{Val}$ s.t. $\mathrm{NF}\mathcal{T}(V_i) = \bigcap_{N \in \mathcal{X}_i} \mathrm{NF}\mathcal{T}(N)$. We have to show that:

$$\mathrm{NF}\mathcal{T}(C(\!|V_1, \dots, V_n|\!)) = \bigcap_{N_1 \in \mathcal{X}_1} \cdots \bigcap_{N_n \in \mathcal{X}_n} \mathrm{NF}\mathcal{T}(C(\!|N_1, \dots, N_n|\!)).$$

($\subseteq$). Clearly, for all $i = 1, \dots, n$ and $N_i \in \mathcal{X}_i$, we have $\mathrm{NF}\mathcal{T}(V_i) \subseteq \mathrm{NF}\mathcal{T}(N_i)$, therefore we conclude $\mathrm{NF}\mathcal{T}(C(\!|V_1, \dots, V_n|\!)) \subseteq \mathrm{NF}\mathcal{T}(C(\!|N_1, \dots, N_n|\!))$ by Monotonicity.
($\supseteq$). Let $t \in \bigcap_{\vec{N} \in \vec{\mathcal{X}}} \mathrm{NF}\mathcal{T}(C(\!|N_1, \dots, N_n|\!))$ (where $\vec{N} := (N_1, \dots, N_n)$ and $\vec{\mathcal{X}} := (\mathcal{X}_1, \dots, \mathcal{X}_n)$). For every $\vec{N} \in \vec{\mathcal{X}}$, by Lemma 3.4.18 there exist a cbv-n-resource-context $c_{\vec{N}} \in \mathcal{T}(C)$ and, for every $i = 1, \dots, n$, a list $\vec{v}^i_{\vec{N}} = \langle v^{i1}_{\vec{N}}, \dots, v^{id_i}_{\vec{N}} \rangle$ (where $d_i := \deg_{\xi_i}(c_{\vec{N}})$) with $[\vec{v}^i_{\vec{N}}] \in \mathcal{T}(N_i)$ and such that $t \in \mathrm{nf}(c^\bullet_{\vec{N}}(\!|\vec{v}^1_{\vec{N}}, \dots, \vec{v}^n_{\vec{N}}|\!))$, for $c^\bullet_{\vec{N}}$ a rigid of $c_{\vec{N}}$. Confluence allows to factorize the reduction from $c^\bullet_{\vec{N}}(\!|\vec{v}^1_{\vec{N}}, \dots, \vec{v}^n_{\vec{N}}|\!)$ to t as follows:

$$c^\bullet_{\vec{N}}(\!|\mathrm{nf}(v^{11}_{\vec{N}}), \dots, \mathrm{nf}(v^{1d_1}_{\vec{N}}), \dots, \mathrm{nf}(v^{n1}_{\vec{N}}), \dots, \mathrm{nf}(v^{nd_n}_{\vec{N}})|\!) \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}(c^\bullet_{\vec{N}}(\!|\vec{v}^1_{\vec{N}}, \dots, \vec{v}^n_{\vec{N}}|\!)) \ni t.$$

So for all $i = 1, \dots, n$ and $j = 1, \dots, d_i$, there exist $w^{ij}_{\vec{N}} \in \mathrm{nf}(v^{ij}_{\vec{N}})$ such that:

$$\mathrm{nf}(c^\bullet_{\vec{N}}(\!|\vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}}|\!)) \ni t \tag{3.15}$$

and being $N_i \in \mathcal{X}_i$ which is bounded by L_i, we have $[\vec{w}^i_{\vec{N}}] \in \mathrm{nf}([\vec{v}^i_{\vec{N}}]) \subseteq \mathrm{NF}\mathcal{T}(N_i) \subseteq \mathrm{NF}\mathcal{T}(L_i)$. From the inclusion $[\vec{w}^i_{\vec{N}}] \in \mathrm{NF}\mathcal{T}(L_i)$ we obtain, thanks to Remark 3.4.9 because L_i is a value, a simple term $[\vec{u}^i_{\vec{N}}] \in \mathcal{T}(L_i)$ such that:

$$[\vec{w}^i_{\vec{N}}] \in \mathrm{nf}([\vec{u}^i_{\vec{N}}]) \tag{3.16}$$

i.e. they have the same number of elements and $\mathrm{nf}(u^{ij}_{\vec{N}}) \ni w^{ij}_{\vec{N}}$ for all $i, j, \vec{N}$. By composing thus a reduction from $c^\bullet_{\vec{N}}(\!|\vec{w}^1_{\vec{N}}, \dots, \vec{w}^n_{\vec{N}}|\!)$ to t with a reduction from $u^{ij}_{\vec{N}}$ to $w^{ij}_{\vec{N}}$, we find that $t \in \mathrm{nf}(c^\bullet_{\vec{N}}(\!|\vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}}|\!))$. This holds for all $\vec{N} \in \vec{\mathcal{X}}$, i.e.:

$$t \in \bigcap_{\vec{N} \in \vec{\mathcal{X}}} \mathrm{nf}(c_{\vec{N}}(\!|\vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}}|\!)). \tag{3.17}$$

Now, Lemma 3.4.18 gives $c^\bullet_{\vec{N}}(\!|\vec{u}^1_{\vec{N}}, \dots, \vec{u}^n_{\vec{N}}|\!) \in \mathcal{T}(C(\!|L_1, \dots, L_n|\!))$. But since the L_i's are independent from $N_1, \dots, N_n$, and thanks to (3.17), we can apply Proposition 3.4.12, and obtain that

the set $\{c^\bullet_{\vec{N}}(\!|\vec{u}^1_{\vec{N}},\dots,\vec{u}^n_{\vec{N}}|\!) \mid \vec{N}\in\vec{\mathcal{X}}\}$ is actually a singleton. Therefore, Lemma 3.4.17 tells us that also the terms $c^\bullet_{\vec{N}}$ and the bags $[\vec{u}^i_{\vec{N}}]$ are independent from $\vec{N}\in\vec{\mathcal{X}}$. The unique element of the previous sigleton has hence shape $c^\bullet(\!|\vec{u}^i,\dots,\vec{u}^n|\!)$, with $c^\bullet$ a rigid of a $c\in\mathcal{T}(C)$, and $[\vec{u}^i]\in\mathcal{T}(L_i)$. Recalling now that $[\vec{w}^i_{\vec{N}}]\in \mathrm{NF}\mathcal{T}(L_i)$, we can apply Proposition 3.4.11 in order to obtain, for each $i=1,\dots,n$, an $L'_{[\vec{w}^i_{\vec{N}}]}\in\Lambda_{\mathrm{cbv}}$ such that $L_i\twoheadrightarrow_{\mathrm{v}} L'_{[\vec{w}^i_{\vec{N}}]}$ and $[\vec{w}^i_{\vec{N}}]\in\mathcal{T}(L'_{[\vec{w}^i_{\vec{N}}]})$. Remark that these $L_{[\vec{w}^i_{\vec{N}}]}$'s must in fact be values, since they are reducts of the values L_i. Consider now the set $\{[\vec{w}^i_{\vec{N}}]\mid\vec{N}\in\vec{\mathcal{X}}\}$, which can be *a priori* infinite. Since for i fixed, the set $\{[\vec{u}^i_{\vec{N}}]\mid\vec{N}\in\vec{\mathcal{X}}\}$ is a singleton $\{[\vec{u}^i]\}$, (3.16) entails that $[\vec{w}^i_{\vec{N}}]\in\mathrm{nf}([\vec{u}^i])$, and our set $\{[\vec{w}^i_{\vec{N}}]\mid\vec{N}\in\vec{\mathcal{X}}\}$ must thus in fact be finite. Therefore we can invoke confluence in order to say that the *finitely many* $L'_{[\vec{w}^i_{\vec{N}}]}$'s share a commond reduct, call it L'_i, which as the notation shows is now independent from $[\vec{w}^i_{\vec{N}}]$ (but still depends on i). Of course L'_i is also a reduct of L_i, and it is still a value. Also, since each $[\vec{w}^i_{\vec{N}}]$ belongs to $\mathcal{T}(L'_{[\vec{w}^i_{\vec{N}}]})$ and is normal, by Remark 3.4.10 it is $\{[\vec{w}^i_{\vec{N}}]\mid\vec{N}\in\vec{\mathcal{X}}\}\subseteq\mathcal{T}(L'_i)$. Thus we can apply Lemma 3.4.18 and find that, for every $\vec{N}\in\vec{\mathcal{X}}$, we have:

$$c^\bullet(\!|\vec{w}^1_{\vec{N}},\dots,\vec{w}^n_{\vec{N}}|\!)\in\mathcal{T}(C(\!|L'_1,\dots,L'_n|\!)). \tag{3.18}$$

But now thanks to (3.15) (which holds for all $\vec{N}\in\vec{\mathcal{X}}$) and (3.18), we can apply again Proposition 3.4.12 in order to find that the set $\{c^\bullet(\!|\vec{w}^1_{\vec{N}},\dots,\vec{w}^n_{\vec{N}}|\!)\mid\vec{N}\in\vec{\mathcal{X}}\}$ is a singleton. Again by Lemma 3.4.17, we have that all the bags $[\vec{w}^1_{\vec{N}}],\dots,[\vec{w}^n_{\vec{N}}]$ for $\vec{N}\in\vec{\mathcal{X}}$, coincide respectively to some bags $[\vec{w}^1],\dots,[\vec{w}^n]$ which are independent from $\vec{N}\in\vec{\mathcal{X}}$. So the only element of the previous singleton has shape $c^\bullet(\!|\vec{w}^1,\dots,\vec{w}^n|\!)$, and by (3.15) we get:

$$t\in\mathrm{nf}(c^\bullet(\!|\vec{w}^1,\dots,\vec{w}^n|\!)). \tag{3.19}$$

Now, for all i, remembering what we found already, we have $[\vec{w}^i]=[\vec{w}^i_{\vec{N}}]\in\mathrm{NF}\mathcal{T}(N)$ for all $N\in\mathcal{X}_i$. That is,

$$[\vec{w}^i]\in\bigcap_{N\in\mathcal{X}_i}\mathrm{NF}\mathcal{T}(N)=\mathrm{NF}\mathcal{T}(V_i) \tag{3.20}$$

where we finally used the hypothesis. From (3.20) and Lemma 3.4.18 one can now easily conclude that $t\in\mathrm{nf}(c^\bullet(\!|\vec{w}^1,\dots,\vec{w}^n|\!))\subseteq\mathrm{NF}\mathcal{T}(C(\!|V_1,\dots,V_n|\!))$. □

As usual, one obtains as a Corollary the non-existence of the following *parallel-or*. We use the usual encoding of pairs: $(M,N):=\lambda z.zMN$. Remark that thus a pair is a value.

Corollary 3.4.21 (No parallel-or)**.** *There is* no $\mathtt{Por}\in\Lambda_{\mathrm{cbv}}$ *s.t. for all* $M,N\in\Lambda_{\mathrm{cbv}}$*,*

$$\begin{cases}\mathtt{Por}(M,N) &=_\tau \mathtt{True} & \text{if } M\neq_\tau\Omega \text{ or } N\neq_\tau\Omega\\ \mathtt{Por}(M,N) &=_\tau \Omega & \text{if } M=_\tau N=_\tau\Omega.\end{cases}$$

Proof. Otherwise, applying Theorem 3.4.20 for $C:=\mathtt{Por}\,\xi$, $\mathcal{X}=\{(\mathtt{True},\Omega),(\Omega,\mathtt{True})\}$ which is bounded by the value $(\mathtt{True},\mathtt{True})$, and the value $V:=(\Omega,\Omega)=(\mathtt{True},\Omega)\cap(\mathtt{True},\Omega)$, we would have the contradiction:

$$\mathtt{True}=C(\!|(\mathtt{True},\Omega)|\!)\cap C(\!|(\Omega,\mathtt{True})|\!)=C(\!|(\Omega,\Omega)|\!)=_\tau\Omega.$$ □

3.5 Taylor subsumes Böhm

In the present section we show how one can apply the resource approximation in order to obtain important results of λ-calculus. Those results constitute the core of Chapter 14 in [Bar84] and

are based on what we have called Böhm's approximation. Not only our techniques can serve as "substitute" of the traditional techniques, but they are also "subsume" them in some way, as better explained in the final comments (see Section 3.6).

The present section is organized as follows:

- First, we consider the relation between Böhm's approximation (recall Section 2.2.2) and the resource approximation. As it is well known, the relation between them culminates in the fact that the relations $=_\tau$ and $=_\mathcal{B}$ coincide (Corollary 3.5.13). This is obtained via some results on some particular classes of resource terms, which we call *linearized* and *affined* (see 3.5.8), and via the *commutation formula* of [ER06a]. In that paper, it is proved in the general quantitative case; we will only consider it in the qualitative case, since this is all we need. We give an "easier" proof of this formula, in the sense that the results of Section 3.3 are basically all one needs in order to prove it.

- Then we transfer the PLP from the world of Taylor normal form to the world of Böhm trees (Theorem 3.5.14). This is immediate thanks to the commutation formula. Interestingly, our proof automatically allows to positively answer to the question: does the PLP hold in the term algebra of closed terms? This was an open question. We also present some equivalent ways of stating the PLP.

- We can also transfer the Stability property in the same way (Theorem 3.5.28). This is not immediate due to the presence of intersections; we thus prove the needed properties about the intersection of Taylor normal forms and of approximants of a term (Lemma 3.5.27).

- We then turn to the Genericity property (Theorem 3.5.29), for which we provide a proof based on the property of linearity of resource terms. Compare it to the original proof, which follows a topological argument.

- Finally, we consider the Continuity Lemma (Lemma 3.5.33). One remarks again how our proof is different from the original one, and contains the same crucial argument involving the linearity of the resource terms involved.

The commutation formula

Definition 3.5.1. *One extends the Taylor expansion to $\Lambda_\perp$ (and thus, in particular, to Böhm approximants) setting:*

$$\mathcal{T}(\perp) := \emptyset.$$

One extends it also on subsets $\mathcal{X} \subseteq \mathcal{A}pp$ simply by setting $\mathcal{T}(\mathcal{X}) := \bigcup_{M\in\mathcal{X}} \mathcal{T}(M)$. In particular, the Taylor expansion $\mathcal{T}(\mathcal{A}(M)) \subseteq \Lambda^{\mathrm{r}}$ of the Böhm approximants of a term is defined.

Remark 3.5.2. *Actually, the previous definition lifts to Böhm trees, in the sense that one can define:*

$$\mathcal{T}(\mathrm{BT}(M)) := \bigcup_{P\in\mathcal{A}(M)} \mathcal{T}(P).$$

This is well defined as a function $\mathfrak{B}_\Lambda \longrightarrow \mathscr{P}(\Lambda^{\mathrm{r}})$, because one knows that $\mathrm{BT}(M) = \mathrm{BT}(N)$ entails $\mathcal{A}(M) = \mathcal{A}(N)$, so that $\mathcal{T}(\mathrm{BT}(M)) = \mathcal{T}(\mathrm{BT}(N))$.

Remark also that, by definition and since $\mathrm{BT}(M) \sqsubseteq \mathrm{BT}(N)$ iff $\mathcal{A}(M) \subseteq \mathcal{A}(N)$, the Taylor expansion is monotone on $\mathfrak{B}_\Lambda$ w.r.t. $\sqsubseteq$.

The next Lemma says that the Taylor expansion is monotone also on $\Lambda_\perp$ w.r.t. $\sqsubseteq$.

Lemma 3.5.3. *For $M, N \in \Lambda_\perp$, if $M \sqsubseteq N$ then $\mathcal{T}(M) \subseteq \mathcal{T}(N)$.*

Proof. Straightforward induction on M. □

Remark 3.5.4. *If $P \in \mathcal{A}pp$ then all $t \in \mathcal{T}(P)$ are* r*-normal. This is immediate to see, since the elements of the Taylor expansion of a term must follow the structure of the syntax tree of it.*

Lemma 3.5.5. *Let $M \in \Lambda$ and $t \in \mathcal{T}(M)$. If t is* r*-normal, then there exists $P \in \mathcal{A}pp$ such that $t \in \mathcal{T}(P)$ and $P \sqsubseteq M$. In particular, one has therefore $\mathcal{T}(M) \subseteq \mathcal{T}(\mathrm{BT}(M))$.*

Proof. Being r-normal, $t = \lambda\vec{x}.y[\vec{u}^1]\cdots[\vec{u}^k] \in \mathcal{T}(M)$ for u^i_j r-normal. So $M = \lambda\vec{x}.yM_1\cdots M_k$ with $[\vec{u}^i] \in !\,\mathcal{T}(M_i)$.

We go now by induction on the r-normal structure[19] of t.

If $k = 0$, then $P := \lambda\vec{x}.y$ does the job.

If $k \geq 1$, then fix $i = \{1, \ldots, k\}$ and apply the inductive hypothesis on each $u^i_j \in [\vec{u}^i]$, obtaining a $P^i_j \in \mathcal{A}pp$ s.t. $P^i_j \sqsubseteq M_i$ and $u^i_j \in \mathcal{T}(P^i_j)$. Let $P_i := \bigsqcup_j P^i_j$. Being it a *sup*, it must be $P_i \sqsubseteq M_i$ and $P^i_j \sqsubseteq P_i$ for every i, j. From the former inequality we have $\lambda\vec{x}.yP_1\cdots P_k \sqsubseteq M$, and from the latter (thanks to Lemma 3.5.3) we have $u^i_j \in \mathcal{T}(P_i)$. Whence, the thesis for $P := \lambda\vec{x}.yP_1\cdots P_k$. □

Everything is already in place to prove the commutation formula:

Theorem 3.5.6 (Qualitative commutation formula)**.** *Let $M \in \Lambda$. Then:*

$$\mathrm{NF}\mathcal{T}(M) = \mathcal{T}(\mathrm{BT}(M)).$$

Proof. ($\subseteq$). If $t \in \mathrm{NF}\mathcal{T}(M)$ then $t \in \mathrm{nf}(t')$ for some $t' \in \mathcal{T}(M)$, so by Corollary 3.3.26 there is $N \in \Lambda$ such that $M \twoheadrightarrow N$ and $t \subseteq \mathcal{T}(N)$, and Lemma 3.5.5 gives $t \in \mathcal{T}(\mathrm{BT}(N)) = \mathcal{T}(\mathrm{BT}(M))$.

($\supseteq$). If $t \in \mathcal{T}(\mathrm{BT}(M))$, there is $P \in \mathcal{A}pp$ and $N \in \Lambda$ such that $M \twoheadrightarrow N \sqsupseteq P$ and $t \in \mathcal{T}(P)$. By Lemma 3.5.3 we get $t \in \mathcal{T}(N)$, and t is r-normal by Remark 3.5.4. So $t \in \mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(M)$. □

One immediately gets the following consequence:

Corollary 3.5.7 (Characterization of solvability)**.** *M is solvable iff* $\mathrm{NF}\mathcal{T}(M) \neq \emptyset$.

Proof. M solvable iff $\mathrm{BT}(M) \neq \bot$ iff there exist $P \in \mathcal{A}(M) - \{\bot\}$ iff there exist $P \in \mathcal{A}(M)$ such that $\mathcal{T}(P) \neq \emptyset$ iff $\mathrm{NF}\mathcal{T}(M) = \mathcal{T}(\mathrm{BT}(M)) = \bigcup_{P \in \mathcal{A}(M)} \mathcal{T}(P) \neq \emptyset$. □

As previously mentioned, let us introduce a class of resource terms that are interesting as they contain just enough information to easily reconstruct a term in Λ and $\Lambda_\bot$, respectively.

Definition 3.5.8 (Linearized and affined resource terms)**.** *A resource term t is* linearised *(resp.* affined*) if every bag in t has cardinality exactly* 1 *(resp. at most* 1*). Let's call* Lin, Aff *the set of linearised and affined resource terms.*

Lemma 3.5.9. *There is an injection $\underline{(\cdot)} : \mathrm{Aff} \longrightarrow \Lambda_\bot$ defined as:*

$$\underline{x} = x \qquad \underline{\lambda x.t} = \lambda x.\underline{t} \qquad \underline{s[t]} = \underline{s}\,\underline{t} \qquad \underline{s1} = \underline{s}\,\bot.$$

There is an injection $\overline{(\cdot)} : \Lambda_\bot - \{\bot\} \longrightarrow \mathrm{Aff}$ defined as[20]*:*

$$\overline{x} = x \qquad \overline{(\lambda x.M)} = \lambda x.\overline{M} \qquad \overline{(MN)} = \overline{M}[\,\overline{N}\,] \qquad \overline{(M\bot)} = \overline{M}1.$$

The maps above are bijections between:

[19] Recall Remark3.3.6.

[20] Here we are using the fact that there is a canonical bijection between the set $\Lambda_\bot - \{\bot\}$ and the inductive grammar $M ::= x \mid \lambda x.M \mid M\bot \mid MM$.

1. Λ *and* Lin*, and between* $\mathrm{nf}_\lambda(\Lambda)$ *and* $\mathrm{nf}_\mathrm{r}(\mathrm{Lin})$*.*
2. $\mathcal{A}pp - \{\bot\}$ *and the set* $\mathrm{nf}_\mathrm{r}(\mathrm{Aff})$ *of* r*-normal affined resource terms.*

Proof. Simple inductions. □

The following is an easily checked remark.

Remark 3.5.10. *Let* $M \in \Lambda$*. Then:*

(1) $\mathcal{T}(M) \cap \mathrm{Lin} = \{\overline{M}\}$

(2) For all $P \in \mathcal{A}pp$ *s.t.* $\bot \neq P \sqsubseteq M$*, one has* $\overline{P} \in \mathcal{T}(M)$*.*

The following lemma contains the main properties of interest of the linearized and affined terms.

Lemma 3.5.11. *Let* $M \in \Lambda$ *and* $P, Q \in \mathcal{A}pp$ *with* $P \neq \bot$*. Then:*

1. $\overline{P} \in \mathcal{T}(Q)$ *entails* $P \sqsubseteq Q$*.*
2. $\overline{P} \in \mathrm{NF}\mathcal{T}(M)$ *iff* $P \in \mathcal{A}(M)$*.*
3. M *normalisable iff* $\mathrm{NF}\mathcal{T}(M) \cap \mathrm{Lin} \neq \emptyset$ *iff* $\mathrm{NF}\mathcal{T}(M) \cap \mathrm{Lin} = \{\overline{\mathrm{nf}(M)}\}$*.*

Proof. (1). Simple induction on P.

(2). ($\Rightarrow$). If $\overline{P} \in \mathrm{NF}\mathcal{T}(M)$ then $\overline{P} \in \mathrm{nf}_\mathrm{r}(t)$, for some $t \in \mathcal{T}(M)$. But by Lemma 3.5.5, $t \in \mathcal{T}(Q)$ for some $Q \in \mathcal{A}(M)$, and being thus Q normal, we have $\mathcal{T}(Q) = \mathrm{NF}\mathcal{T}(Q) \ni \overline{P}$. We conclude by Point (1) and the fact that $\mathcal{A}(M)$ is downward closed w.r.t. $\sqsubseteq$.

($\Leftarrow$). If $P \in \mathcal{A}(M)$, then $M \twoheadrightarrow M'$ for some M' such that $P \sqsubseteq M'$, thus $\overline{P} \in \mathcal{T}(M')$ by Remark 3.5.10. By Proposition 3.3.17(2), there is $t \in \mathcal{T}(M)$ such that $t \twoheadrightarrow_\mathrm{r} \overline{P} + \mathbb{T}$ for some sum $\mathbb{T}$. But being it r-normal, one has $\overline{P} \in \mathrm{nf}(t)$, whence $\overline{P} \in \mathrm{NF}\mathcal{T}(M)$.

(3). If M has a normal form $\mathrm{nf}_\lambda(M)$, then $\mathrm{nf}_\lambda(M) \in \mathcal{A}(M) - \{\bot\}$, so by (2) $\overline{\mathrm{nf}(M)} \in \mathrm{NF}(\mathcal{T}(M))$ and by definition it is linearized.

If $t \in \mathrm{NF}\mathcal{T}(M) \cap \mathrm{Lin}$ then $\underline{t} \in \mathrm{nf}_\lambda(\Lambda) \subseteq \mathcal{A}pp - \{\bot\}$. Moreover $t = \overline{(\underline{t})}$, so that by Point (2) there is a reduction $M \twoheadrightarrow M'$ s.t. $\underline{t} \sqsubseteq M'$. Since $\underline{t}$ is even $\bot$-free the only possibility is that $\underline{t} = M'$, which, being normal, is therefore the normal form $\mathrm{nf}_\lambda(M)$ of M.

This proves the first equivalence, as well as the second one. □

Remark 3.5.12. *From point 2 of the previous Lemma one gets a ways of computing the Böhm approximants from the resource ones:*

$$\mathcal{A}(M) = \underline{\mathrm{NF}\mathcal{T}(M) \cap \mathrm{Aff}} \cup \{\bot\}.$$

The commutation formula and Lemma 3.5.11(2) immediately allow to prove that in fact $\sqsubseteq$ is nothing more than $\leq$.

Corollary 3.5.13 (Equivalence between Taylor normal form and Böhm preorders)**.** *Let* $M, N \in \Lambda$*. Then:*

$$M \sqsubseteq N \text{ iff } M \leq N$$

Note that this immediately implies that:

$$M =_\mathcal{B} N \text{ iff } M =_\tau N.$$

Proof. ($\Rightarrow$). $\mathrm{NF}\mathcal{T}(M) = \mathcal{T}(\mathrm{BT}(M)) \subseteq \mathcal{T}(\mathrm{BT}(N)) = \mathrm{NF}\mathcal{T}(N)$, where we have used the Commutation formula and Remark 3.5.2.

($\Leftarrow$). If $P \in \mathcal{A}(M)$, then by Lemma 3.5.11(2) $\overline{P} \in \mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N)$, so $P \in \mathcal{A}(N)$ by the same Lemma.

□

Being $=_\tau$ contextual (this is the Monotonicity Theorem) – and thus a λ-theory – also $\mathcal{B}$ is:

$$\text{If } \mathrm{BT}(M) = \mathrm{BT}(N) \text{ then } \mathrm{BT}(C(\!|M|\!)) = \mathrm{BT}(C(\!|N|\!)).$$

Of course we know this facts from the 60's, but this is another way to prove it, by passing only via resource approximation.

Perpendicular Lines Property We already encountered the Perpendicular Lines Property (PLP) as Theorem 3.3.40. It is there formulated in $\Lambda/_{=_\tau}$ instead of $\Lambda/_{=_\mathcal{B}}$ as it is in the literature, but because the λ-theories of Böhm and Taylor normal form coincide, we automatically obtain it for the former:

Theorem 3.5.14 (Perpendicular lines property for Böhm trees)**.** *If $(\lambda z_1 \dots z_n.F)\xi_1 \dots \xi_n : \Lambda/_{=_\mathcal{B}} \times \cdots \times \Lambda/_{=_\mathcal{B}} \longrightarrow \Lambda/_{=_\mathcal{B}}$ is constant on n perpendicular lines, then it is constant on all $\Lambda/_{=_\mathcal{B}} \times \cdots \times \Lambda/_{=_\mathcal{B}}$.*

Actually, the number n of abstracted variables and the number of arguments need not be equal. In fact, one can state PLP in a (*a priori*) more general form considering directly a term F instead of the abstraction $\lambda\vec{z}.F$, while keeping the same n, and these two variants of PLP are equivalent: using the one for $\lambda z_1 \dots z_n.F$ we can prove the one for F (and the vice-versa is obvious), as the following proposition shows.

Proposition 3.5.15. *Suppose there are $\{M_{ij}\}_{1\le i\ne j\le n}, \{N_i\}_{1\le i\le n} \subseteq \Lambda$ s.t. the system of equations:*

$$\begin{cases} F\ \ Z\ M_{12}\ \dots\dots\ M_{1n} &=_\mathcal{B}\ N_1 \\ F\ M_{21}\ Z\ \dots\dots\ M_{2n} &=_\mathcal{B}\ N_2 \\ \qquad\qquad \ddots \qquad\quad \vdots & \\ F\ M_{n1}\ \dots\ M_{n(n-1)}\ Z &=_\mathcal{B}\ N_n. \end{cases} \tag{3.21}$$

holds for all $Z \in \Lambda$. Then for all $Z_1, \dots, Z_n \in \Lambda$ one has:

$$F\ Z_1 \dots Z_n =_\mathcal{B} N_1.$$

Proof. Wlog F is solvable, otherwise, the result trivially holds: if $F =_\tau \Omega$ then by Monotonicity we have: $F\vec{Z} =_\tau \Omega\vec{Z} =_\tau \emptyset$ for all $\vec{Z}$. Now let us remark that if F is solvable and satisfies equations (3.21) then, once called $m \in \mathbb{N}$ the (necessary unique) integer s.t. $F \twoheadrightarrow_\mathrm{h} \lambda z_1 \dots \lambda z_m.yP_1 \dots P_k$, it must be: $m \ge n$. In fact if $m < n$ then we have two cases: either $y = z_i$ for some $i \in \{1, \dots, m\}$ in which case computing the i-th equation of (3.21) we find the following contradiction (for fresh variables $\vec{q}$ and the appropriate $\widetilde{P_i}$):

$$\begin{array}{rcl} N_i &=_\mathcal{B}& F\, M_{i1} \dots M_{i(i-1)}\, (\lambda q_1 \dots q_{k+n-m}.Z)\, M_{i(i+1)} \cdots M_{in} \\ &=_\mathcal{B}& (\lambda z_1 \dots z_m.\, z_i P_1 \cdots P_k) M_{i1} \dots M_{i(i-1)}\, (\lambda q_1 \dots q_{k+n-m}.Z)\, M_{i(i+1)} \cdots M_{in} \\ &=_\mathcal{B}& (\lambda q_1 \dots q_{k+n-m}.Z) \widetilde{P}_1 \cdots \widetilde{P}_k\, M_{i(m+1)} \cdots M_{in} \\ &=_\mathcal{B}& Z \end{array}$$

for all Z; either $y \neq z_i$ for all $i \in \{1,\dots,m\}$ and then computing the n-th equation of (3.21) we find the following contradiction (for fresh variables $\vec{q}$ and the appropriate $\widetilde{P_i}$):

$$\begin{array}{rcl} N_n & =_{\mathcal{B}} & F\, M_{n1} \cdots M_{n(n-1)}\, Z \\ & =_{\mathcal{B}} & (\lambda z_1 \dots z_m.\, z_i P_1 \cdots P_k)\, M_{n1} \cdots M_{n(n-1)}\, Z \\ & =_{\mathcal{B}} & y\, \widetilde{P_1} \cdots \widetilde{P_k}\, M_{n(m+1)} \cdots M_{n(n-1)}\, Z \end{array}$$

for all Z. But now since $m \geq n$, for all $Z_i \in \Lambda$ we have:

$$\begin{array}{rcl} F Z_1 \cdots Z_n & =_{\mathcal{B}} & (\lambda z_1 \dots z_m.F') Z_1 \cdots Z_n \\ & =_{\mathcal{B}} & \lambda z_{n+1} \dots z_m.\, F'\{Z_1/z_1, \dots, Z_n/z_n\} \\ & =_{\mathcal{B}} & \lambda z_{n+1} \dots z_m.\, (\lambda z_1 \dots z_n.F') Z_1 \cdots Z_n. \end{array}$$

So instantiating the Z_i's by the i perpendicular lines of (3.21) we obtain that each N_i must be s.t. $N_i =_{\mathcal{B}} \lambda z_{n+1} \dots z_m.N_i'$ for some N_i' satisfying:

$$(\lambda z_1 \dots z_n.F') M_{i1} \cdots M_{i(i-1)}\, Z\, M_{i(i+1)} \cdots M_{in} =_{\mathcal{B}} N_i'.$$

That is, the term $\lambda z_1 \dots z_n.F'$ is constant (mod $=_{\mathcal{B}}$) on n perpendicular lines, and thus thanks to the PLP in the already proved version (Theorem 3.5.14) it is constant (mod $=_{\mathcal{B}}$) on all $\Lambda \times \cdots \times \Lambda$ (n times). Hence, $\lambda z_{n+1} \dots z_m.\, (\lambda z_1 \dots z_n.F') Z_1 \cdots Z_n$ also is constant (mod $=_{\mathcal{B}}$) in $Z_1 \dots Z_n$, and the same for $F Z_1 \dots Z_n$. This is exactly what we wanted to prove. □

The PLP can also be equivalently stated for n-contexts instead of terms. We mean that the statement of PLP using a term F (or, equivalently, a term $\lambda z_1 \cdots z_n.F$) is equivalent to the following statement:

Let C be an n-context and suppose there are $\{M_{ij}\}_{1 \leq i \neq j \leq n}, \{N_i\}_{1 \leq i \leq n} \subseteq \Lambda$ s.t. the system of equations:

$$\left\{ \begin{array}{lcl} C(\!|\, Z,\ M_{12}, \dots\dots, M_{1n}\, |\!) & =_{\mathcal{B}} & N_1 \\ C(\!|\, M_{21}, Z, \dots\dots\dots, M_{2n} |\!) & =_{\mathcal{B}} & N_2 \\ \qquad\qquad \ddots & \vdots & \vdots \\ C(\!|\, M_{n1}, \dots, M_{n(n-1)}, Z\ |\!) & =_{\mathcal{B}} & N_n \end{array} \right.$$

holds for all $Z \in \Lambda$. Then for all $Z_1, \dots, Z_n \in \Lambda$ one has:

$$C(\!|\, Z_1 \dots Z_n |\!) =_{\mathcal{B}} N_1.$$

We have:

Proposition 3.5.16. *Both for the open term algebra and the closed term algebra of any λ-theory T, the PLP stated for contexts holds iff PLP stated for terms does.*

Proof. One can check that the following argument holds both in the open and in the closed term algebra.
($\Rightarrow$). Trivial taking $C := F\xi_1 \dots \xi_n$.
($\Leftarrow$). We suppose that for all $F \in \Lambda$, if F is constant (mod T) on n perpendicular lines then F is constant (mod T) on all $\Lambda \times \cdots \times \Lambda$ (n times). We fix a generic n-context C which is constant (mod T) on the following n perpendicular lines:

$$l_i \ :\ (M_{i1}, \dots, M_{i(i-1)}, Z, M_{i(i+1)}, \dots, M_{in})_{Z \in \Lambda}$$

for $i = 1, \dots, n$. We want to prove that C is constant (mod T) everywhere. To that end[21], for each $i = 1, \dots, k$, let $\vec{v}_i$ be a list (the order of the elements will not matter) of all the variables

[21] Intuitively, one would be tempted to immediately conclude by considering $F = \lambda x_1 \dots \lambda x_n.C(\!|\, x_1, \dots, x_n\, |\!)$. However since substitution in a context is may capture variables, in this way we do not necessary have that $F\vec{Z} =_{\mathcal{T}} C(\!|\, \vec{Z}\, |\!)$. So we have to be careful in "simulating" the capturing variables substitution in contexts with the capture free substitution in terms.

bound[22] in C that one encounters on all the paths from the root of C to the occurrences of ξ_i, that is, the bound variables of C in whose scope there is an occurrence of ξ_i. To say it differently, $\vec{v}_i$ are the only variables that, when replacing ξ_i for a term, C can capture. Take now the term:

$$F := \lambda z_1 \dots z_n . C\langle\!| z_1\vec{v}_1, \dots, z_n\vec{v}_n |\!\rangle \in \Lambda$$

where $z_1, \dots, z_n$ are distinct fresh variables. First of all, let us show that F is constant on the following n perpendicular lines:

$$l_i \ : \ (\lambda\vec{v}_1.M_{i1}, \dots, \lambda\vec{v}_{i-1}.M_{i(i-1)}, Z, \lambda\vec{v}_{i+1}.M_{i(i+1)}, \dots, \lambda\vec{v}_n.M_{in})_{Z\in\Lambda}$$

for $i = 1, \dots, n$. In fact, remembering that in general $(\lambda\vec{v}.P)\vec{v} =_{\boldsymbol{\lambda}} P$, we have:

$$\begin{array}{ll} & F\ (\lambda\vec{v}_1.M_{i1}) \cdots (\lambda\vec{v}_{i-1}.M_{i(i-1)})\ Z\ (\lambda\vec{v}_{i+1}.M_{i(i+1)}) \cdots (\lambda\vec{v}_n.M_{in}) \\ =_{\boldsymbol{\lambda}} & C\langle\!|(\lambda\vec{v}_1.M_{i1})\vec{v}_1, \dots, (\lambda\vec{v}_{i-1}.M_{i(i-1)})\vec{v}_{i-1},\ Z\vec{v}_i,\ (\lambda\vec{v}_{i+1}.M_{i(i+1)})\vec{v}_{i+1}, \dots, (\lambda\vec{v}_n.M_{in})\vec{v}_n|\!\rangle \\ =_{\boldsymbol{\lambda}} & C\langle\!| M_{i1}, \dots, M_{i(i-1)},\ Z\vec{v}_i,\ M_{i(i+1)}, \dots, M_{in} |\!\rangle \end{array}$$

which is constant (mod T) in Z thanks to the hypothesis on C. So thanks to the hypothesis on PLP for terms, F is constant (mod T) on all $\Lambda \times \cdots \times \Lambda$ (n times). Now, reasoning analogously as before, we have:

$$\begin{array}{rll} C\langle\!| Z_1, \dots, Z_n |\!\rangle & =_{\boldsymbol{\lambda}} & C\langle\!|(\lambda\vec{v}_1.Z_1)\vec{v}_1, \dots, (\lambda\vec{v}_n.Z_n)\vec{v}_n|\!\rangle \\ & =_{\boldsymbol{\lambda}} & F\,(\lambda\vec{v}_1.Z_1) \dots (\lambda\vec{v}_n.Z_n) \end{array}$$

and so being F constant, C is is constant (mod T) as well. □

It is important to understand the great advantage of using Taylor expansion in order to prove the PLP: in fact, if we considered it with respect to $=_{\mathcal{B}}$, a context $C\langle\!|\xi|\!\rangle$ can display a constant behaviour for several reasons:

1. $C\langle\!|\xi|\!\rangle$ does not contain the hole ξ at all (the trivial case);
2. ξ is "erased" during its reduction as in $C\langle\!|\xi|\!\rangle = (\lambda xy.y)\xi$;
3. ξ is "hidden" behind an unsolvable as in $C\langle\!|\xi|\!\rangle = \Omega\,\xi$;
4. ξ is pushed into infinity as in $C\langle\!|\xi|\!\rangle = \mathsf{P}\xi$, where $\mathsf{P} := \mathsf{Y}(\lambda yzx.x(yz))$ is s.t. $\mathsf{P}z =_{\boldsymbol{\lambda}} \lambda x.x(\mathsf{P}z)$.

As we already remarked in Section 3.3.2, in Λ^{r} we are only left with the trivial case. But in the case of $=_{\mathcal{B}}$, all the cases are possible. We recognise here the "coinductive" character of Böhm trees. Speaking of coinduction, our proof could also be written in a "coinductive fashion", thus allowing to prove the PLP, using coinduction, while remaining in $\Lambda/_{=_{\mathcal{B}}}$.

We talked about the PLP in the term algebra $\Lambda/_{=_\tau} = \Lambda/_{=_{\mathcal{B}}}$. One can ask what happens for other λ-theories, starting with $=_{\boldsymbol{\lambda}}$. The fact that $\Lambda/_{=_\lambda}$ satisfies PLP was only suggested in [Bar84], and proved later on by Endrullis and de Vrijer [EdV08] that applied van Daalen's *Reduction under Substitution* property [vD80], which is a strengthening of the famous "Barendregt Lemma". As we are now considering $=_{\boldsymbol{\lambda}}$, a constant context $C\langle\!|\xi|\!\rangle$ can only display two possible behaviours, namely (1) or (2) in the list above. Moreover, the hypotheses of PLP must hold when taking $Z = x$ and $Z = y$ for $x \neq y$. This is a strong assumption because x and y are completely defined different values thus enforcing a "maximal" the distinction among the terms. These are the main ingredients used in [EdV08] to derive that $\Lambda/_{=_\lambda}$ satisfies PLP. In

[22] Remember that contexts are *not* considered up to renaming of bound variables, contrary to terms, and so the notion of bound variables occurring in a context is well defined.

fact, they are crucial: as shown in [SB99], the property *fails* in the closed term model $\Lambda^o/_{=_\lambda}$ (in which we cannot, thus, take $Z = x$ and $Z = y$ as mentioned above). The last result relies on the existence of so-called Plotkin terms [Plo74], namely λ-distinct terms $M, N \in \Lambda^o$ satisfying $ML =_{\boldsymbol{\lambda}} NL$ for all $L \in \Lambda^o$.

So one sees that in the *closed* term algebra things may be different. It is thus interesting to ask whether for our equivalent λ-theories $=_\tau$ and $=_{\mathcal{B}}$, PLP holds or not when restricting to the closed term algebra. To the best of our knowledge, that is an open question. Actually, we can immediate answer it:

Theorem 3.5.17. *PLP holds in the* closed *term algebra* $\Lambda^o/_{=_{\mathcal{B}}} = \Lambda^o/_{=_\tau}$.

Proof. One can check that the proof of Theorem 3.3.40 work also in the case of closed terms. □

Let us resume in Figure 3.5 what we now know about the validity of PLP, the top right case being new, and the top left case being now proved solely via Taylor expansion and resource approximation.

PLL	Λ	Λ^o
$=_{\mathcal{B}} = =_\tau$	$\surd$	$\surd$
$=_{\boldsymbol{\lambda}}$	$\surd$	$\times$

Figure 3.3: The new PLP validity table

As already remarked, one has that the following "parallel or" is undefinable, and since being solvable is a recursively enumerable property, it means that in λ-calculus one cannot implement parallel computations.

Corollary 3.5.18 (No parallel-or)**.** *There is* no $\mathtt{por} \in \Lambda$ *which semi-decides the solvability of any two terms, that is, such that for all* $M, N \in \Lambda$ *one has:*

$$\begin{cases} \mathtt{por}\, MN =_{\boldsymbol{\lambda}} \mathtt{True} & \text{if } M \text{ or } N \text{ is solvable} \\ \mathtt{por}\, MN \text{ unsolvable} & \text{otherwise.} \end{cases}$$

Another consequence of PLP is that there is *no* λ-term which decides whether its input is $=_{\mathcal{B}}$ to a Church numeral or not. That is, if we denote – only in this Corollary (and its proof) – by $\underline{n} \in \Lambda$ the Church numeral of $n \in \mathbb{N}$, we have:

Corollary 3.5.19. *There is* no $\mathtt{isNum?} \in \Lambda$ *s.t. for all* $M \in \Lambda$ *one has:*

$$\begin{cases} \mathtt{isNum?}\, M =_{\mathcal{B}} \mathtt{True}, & \text{if } M =_{\mathcal{B}} \underline{n} \text{ for some } n \in \mathbb{N}, \\ \mathtt{isNum?}\, M =_{\mathcal{B}} \mathtt{False}, & \text{otherwise.} \end{cases}$$

Proof. Suppose there is such $\mathtt{isNum?} \in \Lambda$. Now take $f : \mathbb{N}^k \longrightarrow \mathbb{N}$ a *computable* function which is constantly equal to a certain $n_0 \in \mathbb{N}$ on all the k axes while $f(\vec{m}_0) \neq \vec{n}_0$ for some $\vec{m}_0 \in \mathbb{N}^k$. It is clear that such functions *do* exist. By Church thesis, there is $F \in \Lambda$ representing f. Now let:

$$\widetilde{F} := \lambda z_1 \dots z_k.\, \mathtt{isNum?}\, z_1(\mathtt{isNum?}\, z_2(\dots \mathtt{isNum?}\, z_n(F z_1 \cdots z_k)\, \underline{n}_0)\, \underline{n}_0)\, \underline{n}_0 \in \Lambda.$$

That is:

$$\widetilde{F} M_1 \cdots M_k =_{\mathcal{B}} \begin{cases} \underline{f(n_1, \dots, n_k)}, & \text{if } M_1 =_{\mathcal{B}} \underline{n_1}, \dots, M_k =_{\mathcal{B}} \underline{n_k} \text{ for some } \vec{n} \in \mathbb{N}^k, \\ \underline{n_0}, & \text{otherwise.} \end{cases}$$

Thus, $\widetilde{F}$ is constantly equal (mod $=_{\mathcal{B}}$) to $\underline{n_0}$ on the k perpendicular lines $(Z, \underline{0}, \dots, \underline{0})_{Z \in \Lambda}$, $(\underline{0}, Z, \underline{0}, \dots, \underline{0})_{Z \in \Lambda}$, $\dots$, $(\underline{0}, \dots, \underline{0}, Z)_{Z \in \Lambda}$. So by PLP it is constant (mod $=_{\mathcal{B}}$) everywhere, but this is a contradiction with the fact that $\widetilde{F}\underline{\vec{m}_0} =_{\mathcal{B}} \underline{f(\vec{m}_0)} \neq_{\mathcal{B}} \underline{\vec{n}_0}$ (two *different* Church numerals are always $=_{\mathcal{B}}$-distinct, as one immediately sees). □

Stability The theory of stability was developed by Berry, while studying sequential computations and full abstraction for PCF [Ber78]. We exhibit a new proof of the Stability Theorem as formulated in [Bar84, Thm. 14.4.10]. The original proof exploits a *causality relation* capturing the fact that suitable subtrees of $\mathrm{BT}(C(\!|M_1,\dots,M_n|\!))$ "are caused by" some argument M_i.

We first, we will need some results:

Recall that $P_1, P_2 \in \mathcal{A}pp$ are said to be comparable iff $P_1 \sqsubseteq Q \sqsupseteq P_2$ for some $Q \in \mathcal{A}pp$. This is the same to say that they have the same structure: either at least one of them is $\bot$, either $P_1 = \lambda\vec{x}.yP_1^1\cdots P_1^k$, $P_2 = \lambda\vec{x}.yP_2^1\cdots P_2^k$, and P_1^j, P_2^j are comparable. Finitely many comparable Böhm approximants always admit a *sup* in $\mathcal{A}pp$, and this operation can be lifted to countably infinite many pairwise comparable ones by yielding a *sup* in $\mathfrak{B}$. We show below that the same can be done for the *inf*, but remaining inside $\mathcal{A}pp$.

Definition 3.5.20. *For comparable $P_1, P_2 \in \mathcal{A}pp$, we inductively define a term $P_1 \sqcap P_2 \in \Lambda_\bot$ as follows:*

$$P_1 \sqcap P_2 = \begin{cases} \lambda\vec{x}.y(P_{11}\sqcap P_{21})\cdots(P_{1k}\sqcap P_{2k}) & \text{if } P_1 = \lambda\vec{x}.yP_{11}\cdots P_{1k} \text{ and } P_2 = \lambda\vec{x}.yP_{21}\cdots P_{2k}, \\ \bot & \text{if at least one between } P_1 \text{ and } P_2 \text{ is } \bot. \end{cases}$$

It is trivial to check by induction that thanks to the compatibility, the definition makes sense and gives the inf *of P_1, P_2 in $\mathcal{A}pp$. It is straightforward to check by induction (on P_1, for example) that if countably many P_i's are compatible with each other, then $\prod_{i\in\mathcal{I}} P_i = \inf_{i\in\mathcal{I}} P_i$, where the* inf *is taken in $\mathcal{A}pp$ (and not in $\mathfrak{B}$ as for the* sup*). In particular, in $\mathcal{A}pp$ there exists $\inf_{P\in\mathcal{A}(M)} P = \prod_{P\in\mathcal{A}(M)} P$.*

Remark 3.5.21. *Let $P_i \in \mathcal{A}pp$ for $i \in \mathcal{I}$. If $\bigcap_i \mathcal{T}(P_i) \neq \emptyset$, then the P_i's are pairwise comparable. Indeed, fix $t \in \bigcap_i \mathcal{T}(P_i)$. Since the P_i's belong to $\mathcal{A}pp$, such a t must be* r*-normal. We go now by induction on its* r*-normal structure*[23]*: If $t = \lambda\vec{x}.y$ then it is trivial. If $t = \lambda\vec{x}.y[\vec{u}^1]\cdots[\vec{u}^k]$ (with $k \geq 1$) then each P_i must has shape $P_i = \lambda\vec{x}.yP'_{i1}\cdots P'_{ik}$, with $u_j^m \in \bigcap_i \mathcal{T}(P_{im})$ for all m, j. Now one can conclude by the inductive hypothesis.*

Lemma 3.5.22. *Let $\mathcal{X} \subseteq \Lambda$ and take a family $\{P_N \in \mathcal{A}(N) \mid N \in \mathcal{X}\}$. If the P_N's are all pairwise compatible, one has:*

$$\mathcal{T}(\prod_{N\in\mathcal{X}} P_N) \subseteq \mathcal{T}(\bigcap_{N\in\mathcal{X}} \mathcal{A}(N)).$$

Proof. Fix $N \in \mathcal{X}$. Since $\prod_{N\in\mathcal{X}} P_N \sqsubseteq P_N \in \mathcal{A}(N)$, one has $\prod_{N\in\mathcal{X}} P_N \in \mathcal{A}(N)$ because $\mathcal{A}(M)$ is downward closed. We have proved that $\prod_{N\in\mathcal{X}} P_N \in \bigcap_{N\in\mathcal{X}} \mathcal{A}(N)$. And this entails[24] that $\mathcal{T}(\prod_{N\in\mathcal{X}} P_N) \subseteq \mathcal{T}(\bigcap_{N\in\mathcal{X}} \mathcal{A}(N))$. □

Lemma 3.5.23. *Let $\mathcal{X} \subseteq \mathcal{A}pp$. If the elements of $\mathcal{X}$ are pairwise compatible, one has:*

$$\bigcap_{P\in\mathcal{X}} \mathcal{T}(P) \subseteq \mathcal{T}(\prod_{P\in\mathcal{X}} P).$$

[23] Recall Remark 3.3.6.

[24] We are using the trivial fact that if $Q \in \mathcal{X}$ for a term Q and a set $\mathcal{X}$ of terms, then $\mathcal{T}(\mathcal{X}) = \bigcup_{P\in\mathcal{X}} \mathcal{T}(P) \supseteq \mathcal{T}(Q)$.

Proof. First remark that if $t \in \bigcap_{P\in\mathcal{X}} \mathcal{T}(P)$ then for all $P \in \mathcal{X}$ it must be $\mathcal{T}(P) \neq \emptyset$, i.e. $P \neq \bot$, i.e. $P = \lambda\vec{x}_P.y_P Q_P^1 \cdots Q_P^{k_P}$ for some variables $\vec{x}_P, y_P$ and some $Q_P^j \in \mathcal{A}pp$. But since t is in the Taylor expansion of all of them, it must be that all $\vec{x}_P$, all y_P and all k_P coincide, i.e. for all $P \in \mathcal{X}$, $P =: \lambda\vec{x}.yQ_P^1 \cdots Q_P^k$. Also, let us remark that since all the P's are in $\mathcal{A}pp$, such a t must be r-normal. So let us show by induction on the r-normal structure of t, that if $t \in \bigcap_{P\in\mathcal{X}} \mathcal{T}(P)$ then $t \in \mathcal{T}(\prod_{P\in\mathcal{X}} P)$. If $t = \lambda\vec{x}.y$ then all P must coincide with $\lambda\vec{x}.y$ and the result is trivial. If $t = \lambda\vec{x}.y[\vec{u}^1]\cdots[\vec{u}^k]$ (with $k \geq 1$) then all P must follow the same shape, and thus in particular we have: $\prod_{P\in\mathcal{X}} P = \lambda\vec{x}.y(\prod_{P\in\mathcal{X}} Q_P^1)\cdots(\prod_{P\in\mathcal{X}} Q_P^k)$. Now if we show that $u_j^i \in \mathcal{T}(\prod_{P\in\mathcal{X}} Q_P^i)$, then we are done. But $u_j^i \in \bigcap_{P\in\mathcal{X}} \mathcal{T}(Q_P^i)$ by hypothesis, and thus the inductive hypothesis precisely gives $u_j^i \in \mathcal{T}(\prod_{P\in\mathcal{X}} Q_P^i)$. □

Lemma 3.5.24. *Let $\mathcal{Y}$ a collection of subsets of Λ. One has:*

$$\mathcal{T}(\bigcap_{\mathcal{N}\in\mathcal{Y}} \mathcal{N}) \subseteq \bigcap_{\mathcal{N}\in\mathcal{Y}} \mathcal{T}(\mathcal{N}).$$

Proof. Take $t \in \mathcal{T}(\bigcap_{\mathcal{N}\in\mathcal{Y}} \mathcal{N}) = \bigcup_{F\in\bigcap_{\mathcal{N}\in\mathcal{Y}}\mathcal{N}} \mathcal{T}(F)$, that is, $t \in \mathcal{T}(F)$ for some F s.t. $F \in \mathcal{N}$ for all $\mathcal{N} \in \mathcal{Y}$. But then for all fixed $\mathcal{N} \in \mathcal{Y}$ one has $t \in \bigcup_{F\in\mathcal{N}} \mathcal{T}(F) = \mathcal{T}(\mathcal{N})$. So $t \in \bigcap_{\mathcal{N}\in\mathcal{Y}} \mathcal{T}(\mathcal{N})$. □

The previous inclusion can be strict. For example, take $\mathcal{Y} := \{\mathcal{N}_1, \mathcal{N}_2\}$ with $\mathcal{N}_1 := \{xy\}$ and $\mathcal{N}_2 := \{xz\}$. Then $\mathcal{T}(\mathcal{N}_1 \cap \mathcal{N}_2) = \mathcal{T}(\emptyset) = \emptyset$ but $x1 \in \mathcal{T}(xy) \cap \mathcal{T}(xz) = \mathcal{T}(\mathcal{N}_1) \cap \mathcal{T}(\mathcal{N}_1)$.

However, if the elements of $\mathcal{Y}$ are sets of Böhm approximants of terms, then the equality holds, as the next lemma shows. This particular case is precisely what we will need.

Lemma 3.5.25. *Let $\mathcal{X} \subseteq \Lambda$. One has:*

$$\mathcal{T}(\bigcap_{N\in\mathcal{X}} \mathcal{A}(N)) = \bigcap_{N\in\mathcal{X}} \mathcal{T}(\mathcal{A}(N)).$$

Proof. ($\subseteq$). It is Lemma 3.5.24 with $\mathcal{Y} := \{\mathcal{A}(N) \mid N \in \mathcal{X}\}$.
($\supseteq$). If $t \in \bigcap_{N\in\mathcal{X}} \mathcal{T}(\mathcal{A}(N)) = \bigcap_{N\in\mathcal{X}} \bigcup_{P\in\mathcal{A}(N)} \mathcal{T}(P)$ then for all $N \in \mathcal{X}$ there exist a $P_N \in \mathcal{A}(N)$ s.t. $t \in \mathcal{T}(P_N)$. So $t \in \bigcap_{N\in\mathcal{X}} \mathcal{T}(P_N) \neq \emptyset$, and by Remark 3.5.21 this implies that the P_N's (for $N \in \mathcal{X}$) are pairwise compatible. Hence, Lemma 3.5.22 and Lemma 3.5.23 give: $t \in \mathcal{T}(\prod_{N\in\mathcal{X}} P_N) \subseteq \mathcal{T}(\bigcap_{N\in\mathcal{X}} \mathcal{A}(N))$. □

We can now finally set up the notations and prove the Stability property for Böhm trees, in the same way we already did in the framework of Taylor expansion.

Definition 3.5.26. *Given a non-empty subset $\mathcal{X} \subseteq \Lambda$, define its $\mathcal{B}$-*infimum $\bigsqcap \mathcal{X} \subseteq \mathcal{A}pp$ *as:*

$$\bigsqcap \mathcal{X} := \bigcap_{M\in\mathcal{X}} \mathcal{A}(M).$$

We say that $\mathcal{X}$ is bounded *iff there exists an $L \in \Lambda$ such that $M \sqsubseteq L$ for all $M \in \mathcal{X}$.*

By commutation formula we instantly get that $\mathcal{A}(M) \subseteq \mathcal{A}(L)$ iff $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(L)$, so $\mathcal{X}$ is bounded iff $\mathcal{X}$ is $\mathcal{T}$-bounded.

We write $M = \bigsqcap \mathcal{X}$ instead of $\mathcal{A}(M) = \bigsqcap \mathcal{X}$.
As in the case of Taylor normal form, the Stability property gives sufficient conditions for when a context is "stable" under intersection of Böhm trees.

In order to translate Theorem 3.3.37 to the framework of Böhm trees, we need a last lemma saying that "$\bigsqcap \mathcal{X} = \bigcap \mathcal{X}$", in the sense of the statement.

Lemma 3.5.27. *Let $M \in \Lambda$ and $\mathcal{X} \subseteq \Lambda$. Then:*

$$M = \bigsqcap \mathcal{X} \text{ iff } M = \bigcap \mathcal{X}.$$

Proof. ($\Rightarrow$). Applying $\mathcal{T}(\cdot)$ on both sides of the hypothesis $\mathcal{A}(M) = \bigcap\limits_{N\in\mathcal{X}} \mathcal{A}(N)$ one gets:

$$\mathrm{NF}\mathcal{T}(M) = \mathcal{T}(\mathcal{A}(M)) = \mathcal{T}(\bigcap_{N\in\mathcal{X}} \mathcal{A}(N)) = \bigcap_{N\in\mathcal{X}} \mathcal{T}(\mathcal{A}(N)) = \bigcap_{N\in\mathcal{X}} \mathrm{NF}\mathcal{T}(N)$$

where we have used the Commutation Formula as well as Lemma 3.5.25.

($\Leftarrow$). ($\subseteq$). Let $P \in \mathcal{A}(M)$. Then:

$$\mathrm{NF}\mathcal{T}(P) = \mathcal{T}(P) \subseteq \bigcup_{Q\in\mathcal{A}(M)} \mathcal{T}(Q) = \mathcal{T}(\mathcal{A}(M)) = \mathrm{NF}\mathcal{T}(M) = \bigcap_{\vec{N}\in\vec{\mathcal{X}}} \mathrm{NF}\mathcal{T}(N)$$

where we have used the Commutation Formula and the hypothesis (and $\mathrm{NF}\mathcal{T}(P) = \mathcal{T}(P)$ since P is $\lambda\bot$-normal). So, for every $\vec{N} \in \vec{\mathcal{X}}$, one has $P \sqsubseteq N$, and this means that $P \in \mathcal{A}(P) \subseteq \bigcap\limits_{\vec{N}\in\vec{\mathcal{X}}} \mathcal{A}(N)$ (where $P \in \mathcal{A}(P)$ holds again because P is $\lambda\bot$-normal).

($\supseteq$). Let $P \in \bigcap\limits_{\vec{N}\in\vec{\mathcal{X}}} \mathcal{A}(N)$. Then, for all fixed $\vec{N} \in \vec{\mathcal{X}}$, via a similar argument as before one has:

$$\mathrm{NF}\mathcal{T}(P) = \mathcal{T}(P) \subseteq \bigcup_{Q\in\mathcal{A}(N)} \mathcal{T}(Q) = \mathcal{T}(\mathcal{A}(N)) = \mathrm{NF}\mathcal{T}(N).$$

Thus, $\mathrm{NF}\mathcal{T}(P) \subseteq \bigcap\limits_{\vec{N}\in\vec{\mathcal{X}}} \mathrm{NF}\mathcal{T}(N) = \mathrm{NF}\mathcal{T}(M)$, and this means that $P \in \mathcal{A}(P) \subseteq \mathcal{A}(M)$. □

Theorem 3.5.28 (Stability for Böhm trees)**.** *Let C be an n-context, $M_1, \ldots, M_n \in \Lambda$ and fix non-empty bounded $\mathcal{X}_1, \ldots, \mathcal{X}_n \subseteq \Lambda$. If for all $i \in \{1, \ldots, n\}$ one has:*

$$M_i = \bigsqcap \mathcal{X}_i$$

then one has:

$$C(\!|M_1, \ldots, M_n|\!) = \bigsqcap_{\substack{N_1\in\mathcal{X}_1 \\ \cdots \\ N_n\in\mathcal{X}_n}} C(\!|N_1, \ldots, N_n|\!).$$

Proof. Thanks to the fact that being bounded means being $\mathcal{T}$-bounded, and thanks to the previous Lemma 3.5.27, the hypotheses and the conclusions are exactly those of the Stability for $=_\tau$ (Theorem 3.3.37), so there is nothing more to prove. □

Genericity property of unsolvables Let us start by remarking that Corollary 3.3.14 can be equivalently formulated saying that if $C : \Lambda \longrightarrow \Lambda$ is a context, then:

$$\mathrm{Im}(C \restriction_{\mathrm{Unsol}}) \subseteq \mathrm{SOL} \textit{ entails } \mathrm{Im}(C) \subseteq \mathrm{SOL}.$$

Here we mean of course that "Im" is the image of a function, that "$\restriction$" is the restriction of a function on a subset of its domain, and that "Unsol" is the set of unsolvable λ-terms.

The Genericity property reinforces the conclusion of the previous statement, under a stronger hypothesis. Its informal meaning is that if a λ-term M reducing to a completely defined output (a normal form) contains a subterm U which is unsolvable, then U has *no* influence on the computation of this value, in the sense that it may be replaced by any other λ-term and the computation would yield the exact same result. This is an important result from a conceptual point of view, because it motivates the equivalence between "meaningless" and unsolvable λ-terms. Moreover it gives the idea that unsolvable terms should be identified all each other: this identification forms indeed the λ-theory $\mathcal{H}$, which plays a crucial role in the study of λ-theories.

In the following statement we call "Normalisables" the set of normalisable λ-terms.

Theorem 3.5.29 (Genericity Property)**.** *Let $C : \Lambda \longrightarrow \Lambda$ be a context. If:*

$$\mathrm{Im}(C \restriction_{\mathrm{Unsol}}) \cap \mathrm{Normalisables} \neq \emptyset$$

then:

$$C \textit{ is constant mod } =_{\boldsymbol{\lambda}} \textit{ on all } \Lambda.$$

Proof. Take U unsolvable with $C(\!|U|\!)$ normalisable, and let us show that $C(\!|U|\!) =_{\boldsymbol{\lambda}} C(\!|M|\!)$ for all $M \in \Lambda$. By Lemma 3.5.11(3), there is $t \in \mathrm{NF}\mathcal{T}(C(\!|U|\!)) \cap \mathrm{Lin}$ s.t. $\underline{t} = \mathrm{nf}_\lambda(C(\!|U|\!))$. As $t \in \mathrm{NF}\mathcal{T}(C(\!|U|\!))$, by Lemma 3.3.33(1), there exist $c = c(\!|\xi|\!) \in \mathcal{T}(C)$ and $s_1, \dots, s_k \in \mathcal{T}(U)$ s.t. $t \in \mathrm{nf}_{\mathrm{r}}(c^\bullet(\!|s_1, \dots, s_k|\!))$, where $c^\bullet$ is a rigid of c. Since $\mathrm{NF}\mathcal{T}(U) = \emptyset$ (U is unsolvable) we have $\mathrm{nf}_{\mathrm{r}}(s_i) = 0$ and, by confluence, any reduction from $c^\bullet(\!|s_1, \dots, s_k|\!)$ to its normal form $\mathrm{nf}_{\mathrm{r}}(c^\bullet(\!|s_1, \dots, s_k|\!)) = t + \mathbb{T} \neq 0$ factorizes as:

$$c^\bullet(\!|s_1, \dots, s_k|\!) \twoheadrightarrow_{\mathrm{r}} c^\bullet(\!|0, \dots, 0|\!) \twoheadrightarrow_{\mathrm{r}} t + \mathbb{T}.$$

Now, if ξ actually occurs in c then $c^\bullet(\!|0, \dots, 0|\!) = 0$, but since the reduction $0 \twoheadrightarrow_{\mathrm{r}} t + \mathbb{T}$ is impossible, it must be that ξ does *not* occur in c. But this means that $c^\bullet(\!|s_1, \dots, s_k|\!) \in \mathcal{T}(C(\!|M|\!))$ for all $M \in \Lambda$, hence $t \in \mathrm{NF}\mathcal{T}(C(\!|M|\!))$. Since t is linearised, again by Lemma 3.5.11(3) we get that $C(\!|M|\!)$ is normalisable with $\mathrm{nf}_\lambda(C(\!|M|\!)) = \underline{t}$. Recalling the second line of this proof, we have: $\mathrm{nf}_\lambda(C(\!|U|\!)) = \underline{t} = \mathrm{nf}_\lambda(C(\!|M|\!))$. □

An interesting fact about this proof is that it does *not* use Monotonicity (the reader can check that all the properties we used in the proof, and their proofs, do not use this result). However, it is also possible to prove it using Monotonicity, and it actually makes it easier, contracting the second to the penultimate line of the above proof to a trivial invocation of it:

Proof with Monotonicity. By Lemma 3.5.11(3), there is a linearized $t \in \mathrm{NF}\mathcal{T}(C(\!|U|\!))$ such that $\underline{t} = \mathrm{nf}_\lambda(C(\!|U|\!))$. As $\mathrm{NF}\mathcal{T}(U) = \emptyset$, Monotonicity gives $t \in \mathrm{NF}\mathcal{T}(C(\!|M|\!))$, so that by Lemma 3.5.11(3) $C(\!|M|\!)$ is normalisable and since t is linearised always Lemma 3.5.11(3) gives $\mathrm{nf}_\lambda(C(\!|M|\!)) = \underline{t}$. □

Moreover, using Monotonicity of Böhm trees, one immediately remarks that the hypothesis "$\mathrm{Im}(C \restriction_{\mathrm{Unsol}}) \cap$ normalisables $\neq \emptyset$" is equivalent to "$\mathrm{Im}(C \restriction_{\mathrm{Unsol}}) \subseteq$ normalisables". In fact if there is an U unsolvable with $C(\!|U|\!)$ normalisable, then by the contextuality of $=_{\mathcal{B}}$, any $C(\!|U'|\!)$ for U' unsolvable has the same Böhm tree as $C(\!|U|\!)$, and hence it is normalisable.

We can give an even shorter proof by passing through Böhm trees (and always using monotonicity):

Proof with Böhm trees' Monotonicity. Since $\mathrm{BT}(U) = \bot$ then $U \sqsubseteq M$ so monotonicity yields $C(\!|U|\!) \sqsubseteq C(\!|M|\!)$, and since $\mathrm{BT}(C(\!|U|\!)) = \mathrm{BT}(\mathrm{nf}_\lambda(C(\!|U|\!)))$ is $\bot$-free it must be $C(\!|M|\!) =_{\boldsymbol{\lambda}} C(\!|U|\!)$. □

However this last proof is just a "wrap" for the previous one, in the sense that behind Monotonicity of Böhm trees there is all the theory of resource approximantion and Taylor expansion (and in particular Lemma 3.5.11), which already contains the crucial result used in the previous proof of Lemma 3.5.11; so one might as well directly use them.

Continuity We conclude this section by demonstrating Scott's syntactic continuity lemma, as formulated in [Bar84, Prop. 14.3.19]. The meaning of this statement is that for any context $C : \Lambda/_{=_\mathcal{B}} \longrightarrow \Lambda/_{=_\mathcal{B}}$, any "finite portion" of its output can always be already generated by an appropriate "finite portion" of its input. In this setting, a "finite portion" of the input (resp. output) M (resp. $C(\!|M|\!)$) of C corresponds of course to a Böhm approximant $Q \in \mathcal{A}(M)$ (resp. $P \in \mathcal{A}(C(\!|M|\!))$).

In order to apply our technique based on resource approximation, we are going to associate a resource term $t \in \mathcal{T}(\mathrm{BT}(M))$ with an approximant $P_t \in \mathcal{A}(M)$ by mapping the empty bag 1 to $\bot$ and taking the supremum $\sqcup$ w.r.t. $\sqsubseteq$.

Definition 3.5.30. *We define a* partial *map* $P_{(\cdot)} : \mathrm{nf_r}(\Lambda^{\mathrm{r}}) \longrightarrow \mathcal{A}pp - \{\bot\}$*, by induction on the* r*-normal structure*[25] *of* $\mathrm{nf_r}(\Lambda^{\mathrm{r}})$*, as follows:*

$$\begin{array}{lcll} P_{\lambda\vec{x}.y} & := & \lambda\vec{x}.y & \\ P_{\lambda\vec{x}.y\,[\vec{u}^{\,1}]\cdots[\vec{u}^{\,k}]} & := & \lambda\vec{x}.y\,(\bigsqcup\limits_j P_{u_j^1})\cdots(\bigsqcup\limits_j P_{u_j^k}) & \textit{if the } \text{sups} \textit{ do exist} \\ P_t & \textit{is} & \textit{undefined} & \textit{otherwise.} \end{array}$$

where we took $\bigsqcup\limits_{P\in\emptyset} P := \bot$*. It is clear that the definition makes sense.*

Lemma 3.5.31. *Let* $M \in \Lambda$ *solvable. For all* $t \in \mathrm{NF}\mathcal{T}(M)$*, one has that* P_t *is defined and* $P_t \in \mathcal{A}(M)$*.*

Proof. By induction on the r-normal structure of t, remembering that $\mathrm{NF}\mathcal{T}(M) = \mathrm{NF}\mathcal{T}(\mathrm{nf_h}(M))$ as well as $\mathcal{A}(M) = \mathcal{A}(\mathrm{nf_h}(M))$. If $t = \lambda\vec{x}.y$ then $\mathrm{nf_h}(M) = \lambda\vec{x}.y$ and the result is immediate. If $t = \lambda\vec{x}.y[\vec{u}^{\,1}]\cdots[\vec{u}^{\,k}]$ (with $k \geq 1$) then $\mathrm{nf_h}(M) = \lambda\vec{x}.yM_1\cdots M_k$, with $u_j^i \in \mathrm{NF}\mathcal{T}(M_i)$. Thus M_i is solvable (Corollary 3.5.7) and by inductive hypothesis $P_{u_j^i}$ is defined and $P_{u_j^i} \in \mathcal{A}(M_i)$. Since $\mathcal{A}(M_i)$ is directed, there exist the sup $\bigsqcup\limits_j P_{u_j^i} \in \mathcal{A}(M_i)$, and thus $P_t = \lambda\vec{x}.y\,(\bigsqcup\limits_j P_{u_j^1})\cdots(\bigsqcup\limits_j P_{u_j^k})$ is defined. But it is immediately seen that $\mathcal{A}(\lambda\vec{x}.yM_1,\cdots,M_k) = \lambda\vec{x}.y\mathcal{A}(M_1)\cdots\mathcal{A}(M_k)$, and so $P_t \in \mathcal{A}(\mathrm{nf_h}(M)) = \mathcal{A}(M)$. □

If $t \in \mathrm{NF}\mathcal{T}(M)$, by the commutation formula this means that there is some $P \in \mathcal{A}(M)$ s.t. $t \in \mathcal{T}(P)$. The next Lemma says that P_t is precisely one of such approximants.

Lemma 3.5.32. *Let* $M \in \Lambda$ *solvable. For all* $t \in \mathrm{NF}\mathcal{T}(M)$ *one has* $t \in \mathcal{T}(P_t)$*.*

Proof. By induction on the r-normal structure of t. If $t = \lambda\vec{x}.y$ the result is immediate. If $t = \lambda\vec{x}.y[\vec{u}^{\,1}]\cdots[\vec{u}^{\,k}]$ (with $k \geq 1$) then $\mathrm{nf_h}(M) = \lambda\vec{x}.yM_1\cdots M_k$, with $u_j^i \in \mathrm{NF}\mathcal{T}(M_i)$. Now the inductive hypothesis on each u_j^i says that it belongs to $\mathcal{T}(P_{u_j^i})$. But by Lemma 3.5.3, we have $u_j^i \in \mathcal{T}(P_{u_j^i}) \subseteq \mathcal{T}(\bigsqcup\limits_j P_{u_j^i})$, so we conclude $t \in \mathcal{T}(\lambda\vec{x}.y\,(\bigsqcup\limits_{s_1\in b_1} P_{s_1})\cdots(\bigsqcup\limits_{s_k\in b_k} P_{s_k}))$. □

[25] Recall Remark 3.3.6.

Example 1.

1. *For all affined* $t \in \mathcal{T}(\mathrm{BT}(M))$ *we have* $P_t = \underline{t}$.
2. $P_{(\lambda x.x[x,x])} = \Delta$ *and* $P_{(\lambda x.x1)} = \lambda x.x\bot$.
3. $P_{(\lambda f.f[f1,f[f1]])} = \lambda f.f(f(f\bot))$.

Everything is now in place to prove Scott's continuity lemma. The scrupulous reader will notice that we apply to terms in $\Lambda_\bot$ some definitions and results originally stated for Λ, e.g., $\mathcal{A}(\cdot)$, $\sqsubseteq_{\mathcal{B}}$, Lemma 3.5.11. This should not be troubling, because the constant $\bot$ is observationally indistinguishable from the λ-term Ω, i.e., $\mathcal{T}(\bot) = \mathrm{NF}\mathcal{T}(\Omega) = \emptyset$.

Lemma 3.5.33 (Continuity lemma)**.** *Let* $M \in \Lambda$ *and* C *be a context. For all* $P \in \mathcal{A}(C\langle\!|M|\!\rangle)$*, there exists* $Q \in \mathcal{A}(M)$ *such that* $P \sqsubseteq C\langle\!|Q|\!\rangle$.

Proof. Let $P \in \mathcal{A}(C\langle\!|M|\!\rangle)$. Since P is $\lambda\bot$-normal, we have $\mathcal{A}(P) = \{P' \in \mathcal{A}pp \mid P' \sqsubseteq P\}$, whence it is a finite set $\mathcal{A}(P) = \{P_1, \ldots, P_k\} \cup \{\bot\}$ where $k \geq 0$ and $P_i \neq \bot$. Since it is clear that $\mathcal{A}(P) \subseteq \mathcal{A}(C\langle\!|M|\!\rangle)$, by Lemma 3.5.11(2), for all $i = 1, \ldots, k$ we have $\overline{P_i} \in \mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle)$ so that there exist $t_i \in \mathcal{T}(C\langle\!|M|\!\rangle)$ s.t. $\overline{P}_i \in \mathrm{nf}(t_i)$. By Lemma 3.3.33(1), there are $c_i = c\langle\!|\xi|\!\rangle \in \mathcal{T}(C)$ and $s^i_1, \ldots, s^i_{n_i} \in \mathcal{T}(M)$ s.t. $t_i = c^\bullet_i\langle\!|\vec{s}^i|\!\rangle$, where $c^\bullet_i$ is a rigid of c_i. Hence, any reduction $t_i \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(t_i)$ factorizes as:

$$t_i = c^\bullet_i\langle\!|\vec{s}^i|\!\rangle \twoheadrightarrow_{\mathrm{r}} c^\bullet_i\langle\!|\mathrm{nf}_{\mathrm{r}}(s^i_1), \ldots, \mathrm{nf}_{\mathrm{r}}(s^i_{n_i})|\!\rangle \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(t_i) \ni \overline{P}_i$$

where $\mathrm{nf}_{\mathrm{r}}(s^i_j) \subseteq \mathrm{NF}(\mathcal{T}(M))$ for all $i = 1, \ldots, k$ and $1 \leq j \leq n_i$. Now, if $\mathrm{nf}_{\mathrm{r}}(s^i_j) = 0$ for all i, j, then the only possibility is that ξ does *not* occur in c_i, and thus $t_i = c^\bullet_i \in \mathcal{T}(C\langle\!|\Omega|\!\rangle)$, so that $\overline{P_i} \in \mathrm{NF}\mathcal{T}(C\langle\!|\Omega|\!\rangle)$ and by Lemma 3.5.11, $P_i \in \mathcal{A}(C\langle\!|\Omega|\!\rangle) = \mathcal{A}(C\langle\!|\bot|\!\rangle)$, which is the thesis. So Wlog let us assume that $\mathrm{nf}_{\mathrm{r}}(s^i_j) \neq 0$ for all i, j. As there are at most kn_i such s^i_j's, and each $\mathrm{nf}(s^i_j)$ is a finite sum, the following set:

$$\mathcal{Q} := \bigcup_{i=1}^{k} \bigcup_{j=1}^{n_i} \mathrm{nf}(s^i_j) \subseteq \mathrm{NF}\mathcal{T}(M)$$

is *finite*. Since by Lemma 3.5.31 $P_u \in \mathcal{A}(M)$ for all $u \in \mathcal{Q}$, and since being $\mathcal{A}(M)$ directed it contains the sup of finitely many of its elements, we can take:

$$Q := \bigsqcup_{u \in \mathcal{Q}} P_u \in \mathcal{A}(M).$$

Now, recalling that $\overline{P_i} \in \mathrm{nf}_{\mathrm{r}}(t_i) = \mathrm{nf}_{\mathrm{r}}(c^\bullet_i\langle\!|\mathrm{nf}_{\mathrm{r}}(s^i_1), \ldots, \mathrm{nf}_{\mathrm{r}}(s^i_{n_i})|\!\rangle)$, we get $\overline{P_i} \in \mathrm{nf}_{\mathrm{r}}(c^\bullet_i\langle\!|u^i_1, \ldots, u^i_{n_i}|\!\rangle)$ for some $u^i_j \in \mathcal{Q}$ $(1 \leq j \leq n_i)$. But since by Lemmas 3.5.32 and 3.5.3 every $u^i_j \in \mathcal{T}(P_{u^i_j}) \subseteq \mathcal{T}(Q)$, using Lemma 3.3.33 we derive that $\overline{P}_i \in \mathrm{NF}\mathcal{T}(C\langle\!|Q|\!\rangle)$. By Lemma 3.5.11(2), we finally conclude that $P_i \in \mathcal{A}(C\langle\!|Q|\!\rangle)$. If we remember that the P_i's are exactly the elements of $\mathcal{A}(M) - \{\bot\}$, we just proved that $\mathcal{A}(P) \subseteq \mathcal{A}(C\langle\!|Q|\!\rangle)$, i.e. $P \sqsubseteq C\langle\!|Q|\!\rangle$. □

The reason why the above result is called "continuity Lemma" is because it is the main ingredients, together with Monotonicity, of the following theorem, of which we omit the proof since one finds it in [Bar84, Corollary 14.3.21].

Theorem 3.5.34. *All contexts* $C : \Lambda \longrightarrow \Lambda$ *are continuous w.r.t. Scott-topology.*

3.6 Conclusive comments

We have seen that approximating the behaviour of a λ-term by calculating its Taylor expansion allows to abandon proof-techniques based on coinduction in favour of the inductive principle. Already in [Wad76], Wadsworth defined a *labelled λ-calculus* to calculate an approximant $P \in \mathcal{A}(M)$ in an effective way, and this is the historical way the results of Section 3.5 were found. Let us quickly mention the idea and compare the two approaches.

The idea is to annotate every subterm of M with a certain *amount of energy*, represented by a natural number n, which is then decremented along the reduction:

$$\begin{array}{lll} (\lambda x.M)^{n+1}N & \to_{\beta_\ell} & (M\{N^n/x\})^n \\ (\lambda x.M)^0 N & \to_{\beta_\perp} & (M\{\perp/x\})^0 \\ (M^n)^m & \to_\ell & M^{\min\{m,n\}} \end{array}$$

The labelled λ-calculus Λ^ℓ so obtained clearly enjoys strongly normalization, and can be proved to be confluent as well. One retrieves an approximant $P \in \mathcal{A}(M)$ starting from a fully annotated version M^ℓ of M by first computing $\mathrm{nf}_{\beta_\ell\beta_\perp\ell\perp}(M^\ell)$ and then removing the labels. By exploiting the properties above, it is possible to give proofs of Church-Rosser and standardization for λ-calculus. Moreover, in [Bar84, Ch. 14] the labelled calculus constitutes the base ground on which are built the results that we, instead, proved in Section 3.5 via resource approximation. From a semantic perspective, the labelled λ-calculus can be interpreted in well-stratified reflexive objects $\mathcal{D}$ living in cpo-enriched categories [Man09]. It was first used by Hyland [Hyl76] to prove that Scott's $\mathcal{D}_\infty$ and Plotkin's $\mathcal{P}_\omega$ induce a λ-theory including $\mathcal{B}$. This means that also labelling is an interesting technique.

Comparing the labelled λ-calculus and the resource calculus we can find a common principle: in both cases the idea of harness infinite reductions by bounding the availability of subprograms. However, this idea is applied in different (somewhat dual) ways: in Λ^ℓ the bound is on the contraction of redexes, so it is the λ-abstraction that exhibits a restricted behaviour, while in Λ^{r} the restriction is on the amount of resources a program has available. Another difference is that the labels of Λ^ℓ give *an upper bound* on the energy that can be consumed by a λ-term M, while a resource term $t \in \mathcal{T}(M)$ that does not reduce to 0 must contain *the exact number* of resources needed by M in order to compute its outcome. Moreover, while both Λ^ℓ and Λ^{r} are confluent and strongly normalising, only the latter enjoys *linearity* that prevents a resource term from erasing or duplicating its subterms during its execution — a property that is crucial in our proofs. A last important difference is that, while the "old" techniques are essentially based on non-trivial analysis of reductions in the labelled λ-calculus, as well as on the notion of coinduction, the techniques based on Taylor expansion are completely inductive[26].

In conclusion, we believe that the labelled λ-calculus is a valid instrument, but the resource calculus is arguably more natural because it arises from Girard's translation $(\cdot)^\bullet$ of λ-calculus into Linear Logic proof-nets sending $(MN)^\bullet$ to $M^\bullet(N^\bullet)^!$. In fact, the resource calculus stands on the solid ground provided by Differential Linear Logic [ER06b], and this is the reason why it has been generalized so easily to non-deterministic, probabilistic, algebraic call-by-name and call-by-value calculi. Also from the semantic perspective, Λ^{r} can be naturally interpreted in every (linear) reflexive object living a (Cartesian closed) differential category [BCS09], and the Taylor expansion can be used to obtain combinatorial proofs of (denotational) Approximation Theorems [MR14].

It is interesting to remark that, while following [Bar84] the results are often dependent one on another, our proofs and our formulation of the matter allows to discover that actually they

[26] A side effect is, from this perspective, the proliferation of indices in the proofs due to the presence of bags.

are all independent: in Figure 3.4, the continuous lines indicated the "old" theory (as one finds it in [Bar84]), and the dashed ones indicate the situation that we find using the resource approximation/Taylor expansion as we described in the present chapter.

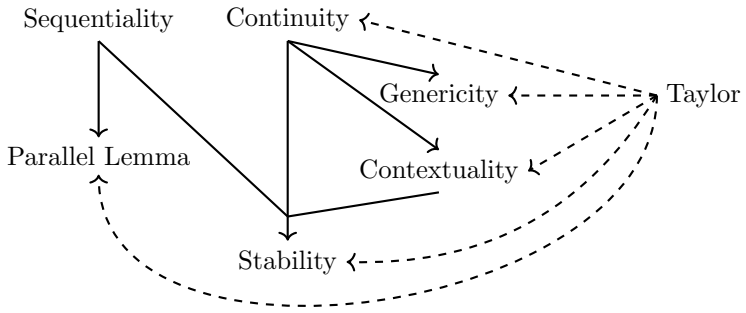


Figure 3.4: The dependencies between the results of [Bar84, Chapter 14]

Let us end this chapter by what we believe to be a very interesting question for the future research: we now dispose of two theories of the approximation, and we know (thanks to the commutation formula) that in a certain sense they give equivalent notions of approximation. Some questions immediately arise:

- What does it mean, in general, *to approximate* a programming language (or at least the λ-calculus)? That is, can we axiomatically *define* the general notion of approximation?
- Would Böhm's and Taylor's be the only two possible instances of such a definition, or are there other ways of approximating programs?
- Can we find *new* results about λ-calculus via approximation techniques?

We have no answer to any of these questions at this day, but we believe an investigation of those would be of a primary interest. To the best of our knowledge, the only current tentative of attacking this questions has been recently proposed by Mazza in the unpublished note [Maz21]. His suggestion is to consider an approximation theory[27] as a certain span of functors enjoying some properties which are, basically, the translation in a categorical setting of what we called the Simulation property (Proposition 3.3.17). Of course both the known approximation theories falls in Mazza's "definition", but the framework is still too premature to be any close to satisfaction. However, we believe that it is a promising starting point.

[27] Or an approximation system, as he calls it.

Chapter 4

The resource approximation for $\lambda\mu$-calculus

> Models of the simply typed λ-calculus, of the untyped λ-calculus and of the simply typed $\lambda\mu$-calculus are well understood, but what about models of the untyped $\lambda\mu$-calculus? As far as we know, this question has been almost ignored.
>
> *Olivier Laurent – [Lau04]*

4.1 Plan of the Chapter

Despite the quote above explicitly mentions the models of $\lambda\mu$-calculus, we will *not* talk about any denotational semantics for it; the reason of the presence of that quote is because the task which we attack in the present chapter is (a small step towards) the development of a *mathematical theory* of $\lambda\mu$-calculus; if we can say that the long time studied λ-calculus admits a true mathematical theory, rich and with its own questions, the one of $\lambda\mu$-calculus remains much less known. Now, when we think of a mathematical theory of a language, we basically mean a theory of its denotational semantics (and "λ-theories"), but also a *theory of its approximation*. Laurent's quote mentions an "ignorance" with respect to the models; we will attack the question of the theory of approximation. From a certain point of view, it is quite natural that the study of $\lambda\mu$-calculus is less developed than the one of λ-calculus: the former is a "canonical" object of study, since it is at the bases of all the other functional languages, therefore all the other ones are less canonical, $\lambda\mu$-calculus included. But from a Curry-Howard perspective, one could say that $\lambda\mu$-calculus does present a sort of canonicity: it is "the simpler" extension of λ-calculus able to include classical logic in the correspondence. Such a statement is of course arguable, because there are many other ways of extending the correspondence to classical logic, such as Krivine's classical realisability [Kri09] (which however does not relate to the same kind of Curry-Howard correspondence as the other calculi, as we will briefly mention later), and the reader can find an interesting discussion of other extensions in [Her95].

The Taylor expansion has been adapted to many languages ([Cho19, Vau19, LL19b, CT20, KMP20], for example), but to the best of our knowledge not to $\lambda\mu$-calculus. There is a notable

exception which has to be mentioned: in [Vau07b], Vaux defines a full differential $\lambda\mu$-calculus following the steps of [ER03]. However, he stops right before defining, for instance, the Taylor expansion. Not because there are difficulties in its definition (one could still follow [ER03]), but because it seems that the author did not have in mind to directly define an approximation theory for the calculus, ready to be used. His version takes coefficients into account; our version is actually qualitative (no coefficients), and we show that this (*a priori*) simpler case is enough for our goals. Our goals are to follow [BM20] in order to adapt the qualitative approximation theory of λ-calculus to $\lambda\mu$-calculus, and apply it. The fact that we see the approximation theory as a tool is also a difference with the other mentioned works, which usually aim more at showing that one can in fact define it, and then concentrate on the relations with normalisation.

We start the chapter with a quick introduction on $\lambda\mu$-calculus and its relevance (Section 4.2).

Before moving to the resource approximation of it, we provide a useful "syntactic interlude" (Section 4.3). The reader can jump to the beginning of that section for an explanation of it content and its meaning.

In Section 4.4 we finally attack the question of the resource approximation: first of all, we define the resource $\lambda\mu$-calculus (in particular we define its resource-reduction 4.4.6) and prove that it is strongly normalizing and confluent (Corollaries 4.4.15 and 4.4.47). In order to achieve the strong normalisation, we define a multiset measure which is strictly decreasing along reductions, and which is ordered via a well-founded order. For the confluence, we basically show that the diagrams of all the critical pairs can be closed. This is not trivial and will take us many pages.

In Section 4.4.1 we define the qualitative Taylor expansion and prove its main properties.

In Section 4.4.2 we prove that the $=_\tau$ on (the equality $=_\tau$ of Taylor normal forms) forms a sensible "$\lambda\mu$-theory" (Corollary 4.4.61), just as in λ-calculus.

In Section 4.4.3 we apply these approximation tools in order to prove the Stability property (Theorem 4.4.62) and the Perpendicular lines property (PLP for short, Theorem 4.4.66). We obtain these results by adapting the new proofs we presented in Chapter 3 for λ-calculus, and we consider the fact that this adaptation is possible as an added value of our proofs. As a consequence, we obtain the sequentiality of $\lambda\mu$-calculus (Corollaries 4.4.63 and 4.4.67).

We end the chapter with some comments and some possible future directions of research. In particular, we briefly discuss the role of Saurin's $\Lambda\mu$-calculus, a variant of Parigot's one, which we are going to quickly mention in the following Section 4.2.

4.2 Introduction to the $\lambda\mu$-calculus

As we already discussed, the celebrated *Curry-Howard correspondence* states that a class of programs, written in a suitable programming language, and intuitionistic logic proofs, written in an suitable formal system, are the same mathematical objects. The typical suitable programming language is λ-calculus, and the typical suitable formal system is intuitionistic natural deduction NJ; under this correspondence, simply typed λ-calculus is identified with NJ. A natural question is what happens for *classical* logic proofs, and whether it is possible to find such a correspondence at all. As we already discussed, in the 90's several ways for generalising such a correspondence to this framework appeared, starting from Griffin's suggestion (in [Gri90]) to type control operators with Peirce law. In this chapter we are going to consider the $\lambda\mu$-calculus, introduced by Parigot in [Par92], which has the advantage of allowing the correspondence to take the exact same form as in the intuitionistic case: just like λ-calculus is the Turing-complete programming language in which intuitionistic logic expresses its computational content, $\lambda\mu$-calculus is the one expressing the computational content of classical logic.

Definition 4.2.1 ($\lambda\mu$-calculus)**.** *Fix a countable set whose elements are called* variables *and a*

(disjoint) countable set whose elements are called names. *The set* $\lambda\mu$ *of* $\lambda\mu$-terms *is defined by the following grammar:*

$$M ::= x \mid \lambda x.M \mid MM \mid \mu\alpha.{}_\beta|M| \quad \textit{(for } x \textit{ a variable and } \alpha, \beta \textit{ names)}$$

in which, as usual, λ *binds* x *in* M *as well as* μ *binds* α *in* ${}_\beta|M|$.

The previous definition means that one proceeds as in λ-calculus: first, define pre-$\lambda\mu$-terms as the words in the above grammar, then define as expected the sets of free variables and free names in a pre-$\lambda\mu$-term, and finally define a $\lambda\mu$-term as pre-$\lambda\mu$-term up to renaming of bound variables and bound names[1].

Despite not being $\lambda\mu$-terms, words of shape ${}_\alpha|M|$ are traditionally called *named terms*, and M is said to be *named under* α. Historically named terms are written as $[\alpha]M$ (as in [Par92]). But this notation is not possible for us, since the use of square brackets is already imperatively taken by the finite multisets, which we will encounter constantly in the following. Another notation, used in [Sau12] for $\Lambda\mu$-calculus, is to write $M\alpha$ just like in an application. However, in our framework, we find this notation not completely clear. Our notation ${}_\alpha|M|$ should, hopefully, clearly show what is "inside a naming" and what is not.

Definition 4.2.2. *Fix a new countable set* $\{\xi_1, \xi_2, \dots\}$ *whose elements are called* holes. *Define a* k-context $C = C\{\xi_1, \dots, \xi_k\}$ *by:*

$$C ::= x \mid \xi_1 \mid \cdots \mid \xi_k \mid \lambda x.C \mid CC \mid \mu\alpha.{}_\beta|C|$$

1-*contexts are simply called* contexts. *A context* $C = C\{\xi\}$ *with exactly one occurrence of the hole is called* single-hole, *and it can be given the following inductive characterisation:*

$$C ::= \xi \mid \lambda x.C \mid CM \mid MC \mid \mu\alpha.{}_\beta|C| \quad \textit{(where } M \textit{ is a } \lambda\mu\textit{-term).}$$

In order to understand where the syntax of the calculus comes from, and what is its relevance, let us quickly consider its programming and logical interpretation.

From the point of view of programming, a $\lambda\mu$-term is meant to represent a process which yields more than one result; more precisely, it is able to produce side-effects by redirecting auxiliary outputs on "channels" different from the standard output, or taking its input from auxiliary channels different from the standard one. One can think of Unix-like Shells' ">" or, better, ">>", and of "<". Under this analogy, a named term ${}_\alpha|M|$ corresponds to Shell's "$M >>$ *channel* α". The coexistence of multiple outputs is then given by the fact that, in a $\lambda\mu$-term M, the standard output is the term M itself, while an auxiliary output N is a term appearing inside a naming ${}_\beta|N|$, for the channel name β free in M. The, in a certain sense, dual constructor of the naming ${}_\alpha|\cdot|$, is the μ-abstraction constructor, and a term of shape $C(\!|\mu\alpha.\dots|\!)$ would intuitively correspond to Shell's "$C <$ *channel* α". The way one achieves this behaviour is through the operational semantics which we will describe in a moment.

But first, let us see what is the logical counterpart of this calculus. As we mentioned in the introduction, the crucial difference between intuitionistic and classical formalisms for proofs, is in the fact that in classical derivations one has *multiple* formulas at the right of "$\vdash$", that is multiple conclusions. This is exactly what the multiple outputs of a $\lambda\mu$-term are about. In addition, as we mentioned in the introduction, in order to have a well-behaving cut-elimination, one has to distinguish between an active output and passive ones, which will correspond to standard and auxiliary outputs in $\lambda\mu$-terms. Let us give the typing rules, in a more convenient variant of Parigot's original Classical Natural Deduction, which can be found, e.g., in [Sel03].

[1] We will avoid calling this an "α-equivalence", since α is now usually a name, and the two terminologies could enter in conflict.

Definition 4.2.3. *The simply typed λμ-calculus is given by the following typing rules*[2]*:*

$$\frac{}{\Gamma, x : A \vdash x : A \mid \Delta} \qquad \frac{\Gamma, x : A \vdash M : B \mid \Delta}{\Gamma \vdash \lambda x.M : A \to B \mid \Delta} \qquad \frac{\Gamma \vdash M : A \to B \mid \Delta \quad \Gamma \vdash N : B \mid \Delta}{\Gamma \vdash MN : B \mid \Delta}$$

$$\frac{\Gamma \vdash M : A \mid \alpha : A, \Delta, \beta : B}{\Gamma \vdash \mu\beta.{}_\alpha|M| : B \mid \alpha : A, \Delta} \text{ if } \alpha \neq \beta \qquad \frac{\Gamma \vdash M : A \mid \alpha : A, \Delta}{\Gamma \vdash \mu\alpha.{}_\alpha|M| : A \mid \Delta}$$

The interesting rules are the last two: if we forget about the terms and about the distinction active/passive formulas, they are both operating a contraction on the right of the "$\vdash$" (together with some exchanges for the rule at the left). Weakenings (on both the sides of "$\vdash$") are handled through the axiom rule. That is, the underlying logical system is the classical system NK. If now we forget about terms and the distinction active/passive formula, we obtain Parigot's Classical Natural Deduction [Par92] (actually, a different version of it, because of the slightly different syntax we are considering). Furthermore, and most importantly, cut-elimination corresponds to the operational semantics of λμ-calculus which we describe below:

Definition 4.2.4. *The reduction relation $\to$ of λμ-calculus*[3] *is the contextual closure of the union $\to_{\text{base}}$ of:*

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

$$(\mu\alpha.{}_\beta|M|)N \to_\mu \mu\alpha.({}_\beta|M|)_\alpha N$$

$$\mu\gamma.{}_\alpha|\mu\beta.{}_\eta|M|| \to_\rho \mu\gamma.\,({}_\eta|M|\{\alpha/\beta\})$$

where $M\{N/x\}$ is the usual capture-free substitution of N for all free occurrences of x in M, and $(M)_\alpha N$ is the substitution $(M)_\alpha N := M\{{}_\alpha|(\cdot)N|/{}_{\alpha|\cdot|}\}$.

The notation $(M)_\alpha N$ comes from the fact that every subterm (indicated with $\cdot$) of M which is named under α, receives a copy of N. This operation is defined as a substitution: every named subterm ${}_\alpha|\cdot|$ of M gets substituted with the named term ${}_\alpha|(\cdot)N|$. Nevertheless, intuitively speaking, it is an application: one applies all subterms of M which are named under α to N. Hence, the notation "$(M)_\alpha N$" which is reminiscent of the application of M to N, and the term $(M)_\alpha N$ is called the *named application of M to N through α*. This notation is due to Vaux [Vau07b].

Remark 4.2.5. *The named application $(M)_\alpha N$ can be inductively characterized as:*

$$(x)_\alpha N := x \qquad (\lambda x.M)_\alpha N := \lambda x.(M)_\alpha N \qquad (MP)_\alpha N := ((M)_\alpha N)((P)_\alpha N)$$

$$(\mu\beta.{}_\alpha|M|)_\alpha N := \mu\beta.{}_\alpha|((M)_\alpha N)N| \qquad (\mu\beta.{}_\gamma|M|)_\alpha N := \mu\beta.{}_\gamma|(M)_\alpha N| \quad \textit{(if } \gamma \neq \alpha\textit{)}.$$

The reduction λ is the usual reduction of λ-calculus[4]. Remark that the renaming $\{\alpha/\beta\}$ in the ρ-reduction cannot be, in general, "taken out" of the parenthesis, because if $\alpha = \gamma$ then α is bound. The reduction ρ is just a renaming of names; one could be tempted to consider

[2] With all the usual conventions: Γ is a variable declaration context, so different variables with possibly equal types, Δ is a name declaration context, so different names with possibly equal types, and writing a sequent "$\Gamma \vdash M : A \mid \Delta$" is just a comfortable way of writing the quadruple (Γ, M, A, Δ). Also, remark that when we write "$\alpha : A, \Delta$", or "$\alpha : A, \Delta, \beta : B$", we mean that respectively α and α, β are not declared in Δ.

[3] So we use for the reduction in λμ-calculus the same symbol for the reduction in λ-calculus. This should not be confusing, since in this chapter we will only talk of λμ-calculus. If we want to specify a base case of a reduction, we will explicitly write the subscript.

[4] This is why we denoted it "λ" and not "β" in all the thesis: we want to avoid confusion with the names in this chapter.

terms up to that renaming, but this is not possible because μ-reduction is not well defined on the ρ-equivalence classes. For example (for $\gamma \neq \eta \neq \alpha$):

$$\begin{array}{lll} (\mu\alpha.{}_\alpha|\mu\gamma.{}_\eta|x||)y & \to_\rho & (\mu\alpha.{}_\eta|x|)y \\ \downarrow_\mu & & \downarrow_\mu \\ \mu\alpha.{}_\alpha|(\mu\gamma.{}_\eta|x|)y| & \neq_\rho & \mu\alpha.{}_\eta|x|. \end{array}$$

This is an important critical pair because it also shows that the ρ-reduction and the μ-reduction do not commute (Definition 2.1.2): in the bottom-left reduct, the ρ-redex is "blocked" by an application; it is only via the μ-reduction $\mu\alpha.{}_\alpha|(\mu\gamma.{}_\eta|x|)y| \to_\mu \mu\alpha.{}_\alpha|\mu\gamma.{}_\eta|x||$ that we can "unblock it" and finally close the diagram by performing the ρ-reduction: $\mu\alpha.{}_\alpha|\mu\gamma.{}_\eta|x|| \to_\rho \mu\alpha.{}_\eta|x|$. We will find the analogue of this example when studying the confluence of the associated resource calculus.

Witnessing the fact that $\lambda\mu$-calculus is a well-behaved version of classical logic, one has that the typed-calculus is strongly normalising and confluent. Actually, as it happens for λ-calculus, the untyped version is also confluent (but of course not strongly normalising).

Theorem 4.2.6. *The $\lambda\mu$-calculus $(\lambda\mu, \to)$ is confluent.*

Proof. See proof of Theorem 4.1 of [Py98]. □

As we already said, from a programming viewpoint the characteristic feature of $\lambda\mu$-calculus is to allow the encoding of control operators, and one says for that reason that it is a functional "impure" functional language. The encoding of `callcc` in $\lambda\mu$-calculus can be found by adding terms to the usual derivation of Peirce's law:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y : (A \to B) \to A \vdash y : (A \to B) \to A \mid \alpha : A} \quad \dfrac{\dfrac{\dfrac{}{y : (A \to B) \to A, x : A \vdash x : A \mid \alpha : A, \delta : B}}{y : (A \to B) \to A, x : A \vdash \mu\delta.{}_\alpha|x| : B \mid \alpha : A}}{y : (A \to B) \to A \vdash \lambda x.\mu\delta.{}_\alpha|x| : A \to B \mid \alpha : A}}{y : (A \to B) \to A \vdash y(\lambda x.\mu\delta.{}_\alpha|x|) : A \mid \alpha : A}}{y : (A \to B) \to A \vdash \mu\alpha.{}_\alpha|y(\lambda x.\mu\delta.{}_\alpha|x|)| : A \mid}}{\vdash \lambda y.\mu\alpha.{}_\alpha|y(\lambda x.\mu\delta.{}_\alpha|x|)| : ((A \to B) \to A) \to A \mid}$$

So one sets:

$$\texttt{callcc} := \lambda y.\mu\alpha.{}_\alpha|y(\lambda x.\mu\delta.{}_\alpha|x|)|.$$

Let's verify that it has the expected backtracking behaviour described in the introduction, by applying it to a non-empty stack "$M, N_1, \ldots, N_k$". One has:

$$\begin{array}{lll} \texttt{callcc}\, M\, N_1 \cdots N_k & \to_\lambda & (\mu\alpha.{}_\alpha|M\, (\lambda x.\mu\delta.{}_\alpha|x|)|)\, N_1 \ldots N_k \\ & \to_\mu & (\mu\alpha.{}_\alpha|M\, (\lambda x.\mu\delta.{}_\alpha|x\, N_1|)\, N_1|)\, N_2 \ldots N_k \\ & \to_\mu \cdots \to_\mu & \mu\alpha.{}_\alpha|M\, (\lambda x.\mu\delta.{}_\alpha|x\, \vec{N}|)\, \vec{N}|. \end{array}$$

The term that we just found represents a process sending the output of $M(\lambda x.\mu\delta.{}_\alpha|x\, \vec{N}|)\vec{N}$ on the channel α, but immediately after it is sending the content of channel α on the standard output. That is, it is just sending the output of $M(\lambda x.\mu\delta.{}_\alpha|x\, \vec{N}|)\vec{N}$ on the standard output. Now let us see what this latter term does: it represents the term M which is launched in front of the stack $\vec{N}$, but in addition to that, it is also provided with the additional term $\lambda x.\mu\delta.{}_\alpha|x\, \vec{N}|$ to potentially make use of. This last term allows to take a term and start its execution in front of the stack $\vec{N}$, that is, in the exact same "environment" of the initial call of M. Since the term in last line of the reductions above is a μ-abstraction on α (so the content of the channel α is the standard output), the two bound occurrences of α in that term can be read as follows: the process $\mu\alpha.{}_\alpha|M\, (\lambda x.\mu\delta.{}_\alpha|x\, \vec{N}|)\, \vec{N}|$ can keep the execution of M in front of the original stack $\vec{N}$

as long as it wants (and this will thus modify the execution stack), by ignoring the additional tool "$\lambda x.\mu\delta.{}_\alpha|x\,\vec{N}|$"; but, at any moment, it can use this tool in order to pop a new term M', restore the original execution stack $\vec{N}$, and launch M' in front of the restored original stack, forgetting about the modified current one. That is, it can backtrack and modify its behaviour after having, potentially, computed some more information, mimicking a sort of "try/catch" piece of code. This is in fact the same backtracking behaviour of the real `callcc` command in the Scheme programming language, as well as Java's handling of exceptions. One could consider Felleisen's C operator in a similar way.

The two new typing rules of simply typed $\lambda\mu$-calculus handle the passage of a formula from active to passive and vice-versa, but this aspect is not very clear with the particular rules that we have written. In fact one could perform both those rules via a two step procedure, first producing a named term and then μ-abstracting it. This is done via the following rules:

$$\frac{\Gamma \vdash M : A \mid \alpha : A, \Delta}{\Gamma \vdash {}_\alpha|M| : \bot \mid \alpha : A, \Delta} \qquad \frac{\Gamma \vdash {}_\beta|M| : \bot \mid \alpha : A, \Delta}{\Gamma \vdash \mu\alpha.{}_\beta|M| : A \mid \Delta}$$

where we added the constant $\bot$ to the formulas. One observes that the left rule is the introduction rule for $\bot$, and performs a contraction, while the right one is the elimination rule for $\bot$. As we already mentioned, these rules show clearly how a formula passes from active to passive and vice-versa. In fact, one could embed simply-typed $\lambda\mu$-calculus inside Linear Logic. Laurent showed how to do this by making use of his *Polarized proof-nets*[5] [Lau03], in which the duality active/passive formula – or standard/auxiliary output – becomes that of positive/negative formulas, an important notion in Linear Logic, particularly in the study of focalization. In the framework of polarized proof-nets the above two new rules become constructions of proof-nets, and show even better how one handles the duality positive/negative.

Let us make a last comment. As we just saw, the more natural rules are the ones which operate on a named term, constructing it or μ-abstracting it. The same could be said for the very syntax or $\lambda\mu$-calculus: it is natural to say that the construction of a term $M = \mu\alpha.{}_\beta|N|$ from N is *not* done in a single step, but via two different constructors: first, one creates ${}_\beta|N|$ from N, and then M from ${}_\beta|N|$. It is natural to consider thus the following syntax:

$$M ::= x \;\mid\; \lambda x.M \;\mid\; MM \;\mid\; \mu\alpha.M \mid {}_\beta|M|.$$

Remark that now a named term is really a term and not just a syntactic word. One can ask whether the two calculi are "the same or not". The question has been negatively answered by Saurin in [Sau12]: while the first syntax (Parigot $\lambda\mu$-calculus) does *not* satisfy Böhm's separation theorem – something which was shown by David and Py in [DP01] – this new syntax *does* satisfy it. The new extended syntax in now known as Saurin's $\Lambda\mu$-calculus (with a big "λ"). We do not consider it in this thesis, but we will come back on it and make some comments at the end of this chapter.

4.3 Interlude: A syntactic shortcut

In reproducing the same arguments of the previous chapter for the case of $\lambda\mu$-calculus, we will often encounter the same kind of proofs. Moreover, many of the properties do not really depend on the fact that one has a λ-abstraction or a μ-abstraction, but only on the fact that the resource calculus linearises application while treating the other constructors as linear and that the Taylor expansion is defined accordingly. Thus, in this interlude, we will forget that a constructor is called λ-abstraction and takes a variable, and the other is called μ-abstraction and takes two

[5] Actually, he also gave polarized proof-nets for Girard's LC, see [Lau99].

names, and we just consider them instances of a same kind of constructor, called $g(v,\cdot)$, taking any possible kind of "variable" – usual variables in the case of λ and pairs of names for μ. We show what is the skeleton of the constructions we carried on in the last chapter, a skeleton which is, thus, valid as soon as one can instantiate this constructor and considers a resource calculus together with a Taylor expansion defined accordingly. Of course the previous case of λ-calculus falls in this case, and in the next section we will see that $\lambda\mu$-calculus does too. The value of this approach is to clearly distinguish what is proper to each particular language, and what can be carried on independently because relying only on the way one treats linearisation.

But a disclaimer must be made: the content of this section must be seen as a "syntactic shortcut" in order to handle, at the same time for λ and for $\lambda\mu$-calculus, the proofs that do not depend on their particular syntax, and to show what is the shared path which we are following in the cases of $\lambda/\lambda\mu$-calculus. But it is not a real generalization; in fact, as it will be clear from the first definitions, we are really simply mimicking the syntax and the situation of $\lambda/\lambda\mu$-calculus. In particular, the "language" we consider here, might not be general enough to include other interesting programming languages other than the two we are interested in. Also, and maybe most importantly, we *already know* how to linearise λ-calculus and $\lambda\mu$-calculus (the latter will be discussed at the beginning of Section 4.4). So we already know how to define, for this "general language", the crucial notions of resource approximants and Taylor expansion. The way one gets to know this would be by relating the language to linear logic, so a real generalization would first of all need to consider that.

We fix disjoint non-empty countably infinite sets $\mathrm{Var}_1,\dots,\mathrm{Var}_h$. Fix $h_1 \leq h$ and $h_2 \in \mathbb{N}$. We consider a programming language $\mathfrak{L}$ with syntax:

$$M ::= v \mid g_j(v_{k_j}, M_0) \mid @(M_1, M_2)$$

where $v \in \mathrm{Var}_1 \cup \dots \cup \mathrm{Var}_{h_1}$, $j = 1,\dots,h_2$ and $v_{k_j} \in \mathrm{Var}_{k_j}$ for some fixed $k_1,\dots,k_{h_2} \leq h$, and $@ : \mathfrak{L}\times\mathfrak{L} \to \mathfrak{L}$ and $g_i : \mathrm{Var}_{k_i} \times \mathfrak{L} \to \mathfrak{L}$ some constructors. We ask this constructors to be injective w.r.t. the equality on $\mathfrak{L}$, i.e. $g_j(v_{k_j}, M) = g_j(v_{k_j}, M')$ iff $M = M'$ and $@(M_1, M_2) = @(M_1', M_2')$ iff $M_1 = M_1'$ and $M_2 = M_2'$. This is the case, for instance, for the equality on λ-calculus or on $\lambda\mu$-calculus, which is syntactic equality modulo renaming of, respectively, bound variables and bound variables and names. In the following we will reason *by induction* on a such terms; we ask, thus, that the equality allows such reasoning. For example equality being the syntactic equality, or a quotient which only allows to change the name of the variables, thus maintaining the same inductive structure.

Example 2.
1. *The set $\mathfrak{L} = \Lambda$ of λ-terms falls in the syntax above: take $h = h_1 = h_2 = 1$ and $k_1 = 1$; take $@(M,N) :=$ the word MN; for $x \in \mathrm{Var}_1$, take $g_1(x,M) :=$ the word $\lambda x.M$ modulo renaming of the variable x.*
2. *The set $\mathfrak{L} = \Lambda\mu$ of $\Lambda\mu$-terms* à la *Saurin falls in the syntax above: take $h = 2$, $h_1 = 1$, $h_2 = 3$ and $k_1 = 1$, $k_2 = k_3 = 2$; take $@(M,N) :=$ the word MN; for $x \in \mathrm{Var}_1$, take $g_1(x,M) :=$ the word $\lambda x.M$ modulo renaming of the variable x; for $\alpha \in \mathrm{Var}_2$ take $g_2(\alpha,M) :=$ the word $\mu\alpha.M$ modulo renaming of the name α and $g_3(\alpha,M)$ the word ${}_\alpha|M|$.*
3. *The set $\mathfrak{L} = \lambda\mu$ of $\lambda\mu$-terms* à la *Parigot falls in the syntax above: take $h = 2$, $h_1 = 1$, $h_2 = 2$ and $k_1 = 1$, $k_2 = 2$; take $\mathrm{Var}_1 := \{x_0, x_1, \dots\}$ and $\mathrm{Var}_2 := \{(\alpha_{i_0}, \alpha_{j_0}), (\alpha_{i_1}, \alpha_{j_1}), \dots\}$ (with $i_0, j_0, i_1, j_1, \dots \in \mathbb{N}$) where $\{\alpha_0, \alpha_1, \dots\}$ is a countably infinite set; take $@(M,N) :=$ the word MN; for $x \in \mathrm{Var}_1$, take $g_1(x,M)$ as before; for $(\alpha,\beta) \in \mathrm{Var}_2$ take $g_2((\alpha,\beta),M) :=$ the word $\mu\alpha.{}_\beta|M|$ modulo renaming of the name α.*

Multihole contexts and single-hole contexts are defined as expected, as well as the contextual closure of a binary relation $\to_{\mathrm{base}}$ on $\mathfrak{L}$.

Definition 4.3.1. *The* resource sensitive version $\mathfrak{L}^{\mathrm{r}}$ *of* $\mathfrak{L}$ *is given by the syntax*[6]*:*

$$M ::= v \mid g^{\mathrm{r}}_j(v_{k_j}, t) \mid @^{\mathrm{r}}(t, [\overbrace{t, \ldots, t}^{n}])$$

where $n \in \mathbb{N}$ *(and as usual* $v \in \mathrm{Var}_1 \cup \cdots \cup \mathrm{Var}_{h_1}$*,* $j = 1, \ldots, h_2$ *and* $v_{k_j} \in \mathrm{Var}_{k_j}$*). In the exact same way as before one defines resource multihole-contexts* $c = c(\!|\xi_1, \ldots, \xi_k|\!)$ *and single-hole resource contexts.*

We also consider the set $2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *of sums, defined by the free* $\mathbb{Z}_2$*-module construction described in Section 2.1. By linearity,* $2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *inherits from* $\mathfrak{L}^{\mathrm{r}}$ *the same constructors* $g^{\mathrm{r}}_j : \mathrm{Var}_{\mathrm{k_j}} \times 2\langle\mathfrak{L}^{\mathrm{r}}\rangle \to 2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *and* $@^{\mathrm{r}} : 2\langle\mathfrak{L}^{\mathrm{r}}\rangle \times\, !\, 2\langle\mathfrak{L}^{\mathrm{r}}\rangle \to 2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *by setting as expected:*

$$g^{\mathrm{r}}_j\left(v, \sum_i t_i\right) := \sum_i g^{\mathrm{r}}_j(v, t_i)$$

$$@^{\mathrm{r}}\left(\sum_{i\in I_0} t_i, \left[\sum_{i\in I_1} t_i, \ldots, \sum_{i\in I_n} t_i\right]\right) := \sum_{(i_0,\ldots,i_n)\in I_0\times\cdots\times I_n} @^{\mathrm{r}}(t_{i_0}, [t_{i_1}, \ldots, t_{i_n}]).$$

Given $\to_{\mathrm{r}} \subseteq \mathfrak{L}^{\mathrm{r}} \times 2\langle\mathfrak{L}^{\mathrm{r}}\rangle$*, one extends it to all* $2\langle\mathfrak{L}^{\mathrm{r}}\rangle \times 2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *as described in Section 2.1.*

Remember that if $\to_{\mathrm{r}}$ is confluent and strongly normalising, all resource terms t have a unique r-normal form $\mathrm{nf}_r(t) \in 2\langle\mathfrak{L}^2\rangle$, which can be 0.

Remark that Lemma 3.3.9, that still holds in this framework.

Definition 4.3.2. *The qualitative Taylor expansion is the map* $\mathcal{T} : \mathfrak{L} \to \mathcal{P}(\mathfrak{L}^{\mathrm{r}})$ *defined by:*

1. $\mathcal{T}(v) := \{v\}$
2. $\mathcal{T}(g_i(v, M)) := \{g^{\mathrm{r}}_i(v, t) \mid t \in \mathcal{T}(M)\}$
3. $\mathcal{T}(@(M, N)) := \{@^{\mathrm{r}}(t, [\vec{u}]) \mid t \in \mathcal{T}(M) \text{ and } [\vec{u}] \in !\,\mathcal{T}(N)\}$.

From now on, we suppose that:

Assumption 1. *It is fixed a binary relation* $\to_{\mathrm{base}}$ *on* $\mathfrak{L}$*. Moreover, its contextual closure* $\to \subseteq \mathfrak{L} \times \mathfrak{L}$ *is s.t. every term has at most one normal form* $\mathrm{nf}(M)$*, and* $\mathrm{nf}(@(M, N)) = \mathrm{nf}(@(\mathrm{nf}(M), \mathrm{nf}(N)))$ *and* $\mathrm{nf}(g_i(v, M)) = g_i(v, \mathrm{nf}(M))$*, whenever the written normal forms do exist.*

Assumption 2. *We have fixed a reduction* $\to_{\mathrm{r}} \subseteq 2\langle\mathfrak{L}^{\mathrm{r}}\rangle \times 2\langle\mathfrak{L}^{\mathrm{r}}\rangle$ *which is confluent and strongly normalising. We also ask that* $\to_{\mathrm{r}}$ *satisfies the clauses in the last line of Assumption 1.*

The previous two assumptions hold in λ-calculus.

By assumption 2, for all $M \in \mathfrak{L}$ there always exist $\mathrm{NF}\mathcal{T}(M) := \bigcup_{t\in\mathcal{T}(M)} \mathrm{nf}_r(t) \subseteq \mathfrak{L}^{\mathrm{r}}$. This allows to endow $\mathfrak{L}$ with the partial order: $M \leq N$ iff $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N)$. It also endows it with the induced equivalence $M =_\tau N$ iff $\mathrm{NF}\mathcal{T}(M) = \mathrm{NF}\mathcal{T}(N)$.

Remark 4.3.3. 1. $\mathrm{NF}\mathcal{T}(g_i(v, M)) = \{g^{\mathrm{r}}_i(v, t) \mid t \in \mathrm{NF}\mathcal{T}(M)\}$

2. $\mathrm{NF}\mathcal{T}(@(M, N)) = \bigcup_{\substack{t\in\mathrm{NF}\mathcal{T}(M)\\ [\vec{u}]\in !\,\mathrm{NF}\mathcal{T}(N)}} \mathrm{nf}_r(@^{\mathrm{r}}(t, [\vec{u}]))$.

We can now prove the monotonicity in the exact same way as already done in λ-calculus:

[6] We take on $\mathfrak{L}^{\mathrm{r}}$ "the same" equality as in $\mathfrak{L}$.

Theorem 4.3.4 (Monotonicity)**.** *Any context $C : \mathfrak{L} \to \mathfrak{L}$ is monotone w.r.t. $\leq$.*

Proof. The proof shows, by induction on C, that for all $M, N \in \mathfrak{L}$ s.t. $\mathrm{NF}\mathcal{T}(M) \subseteq \mathrm{NF}\mathcal{T}(N)$, one has $\mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) \subseteq \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle)$. The proof we gave in λ-calculus (Theorem 3.3.13) can be immediately adapted to the present framework, simply by treating the variables v as the variables x, and the constructor g as a "λ". Let us write this last case for the seek of clarity: If $C = g_i(v, C')$ then:

$$\begin{aligned}\mathrm{NF}\mathcal{T}(C\langle\!|M|\!\rangle) &= \{g_i^{\mathrm{r}}(v,t) \mid t \in \mathrm{NF}\mathcal{T}(C'\langle\!|M|\!\rangle)\}\\ &\subseteq \{g_i^{\mathrm{r}}(v,t) \mid t \in \mathrm{NF}\mathcal{T}(C'\langle\!|N|\!\rangle)\}\\ &= \mathrm{NF}\mathcal{T}(C\langle\!|N|\!\rangle).\end{aligned}$$

□

As we already remarked in Chapter 3, monotonicity of 1-contexts entails monotonicity of k-contexts, so the previous theorem can be extended to multihole-contexts.

The following "simulation assumption" is the analogous of Proposition 3.3.17.

Assumption 3. *If $M \to_{\mathrm{base}} N$ then:*

1. *for all $s \in \mathcal{T}(M)$ there exist $\mathbb{T} \subseteq \mathcal{T}(N)$ s.t. $s \twoheadrightarrow_{\mathrm{r}} \mathbb{T}$*
2. *for all $s' \in \mathcal{T}(N)$ there exist $s \in \mathcal{T}(M)$ s.t. $s \twoheadrightarrow_{\mathrm{r}} s' + \mathbb{T}$ for some sum $\mathbb{T} \subseteq \mathcal{T}(N)$.*

We can lift the previous assumption from $\to_{\mathrm{base}}$ to $\to$:

Proposition 4.3.5. *If $M \to N$ then point (1) and (2) of the previous assumption both hold.*

Proof. By induction on the single-hole context C s.t. $M = C\langle\!|M'|\!\rangle$, $N = C\langle\!|N'|\!\rangle$ and $M' \to_{\mathrm{base}} N'$.

Case $C = \xi$. Then $M = M'$ and $N = N'$, so the result is exactly Assumption 3.

Case $C = g^{\mathrm{Cxt}}(v, C')$.
(1). If $s \in \mathcal{T}(M)$ then $s = g^{\mathrm{r}}(v, s_0)$, with $s_0 \in \mathcal{T}(C'\langle\!|M'|\!\rangle)$. So by induction hypothesis there is a sum $\mathbb{T}_0 \subseteq \mathcal{T}(C'\langle\!|N'|\!\rangle)$ s.t. $s \to_{\mathrm{r}} g^{\mathrm{r}}(v, \mathbb{T}_0) \subseteq \mathcal{T}(g(v, C'\langle\!|N'|\!\rangle)) = \mathcal{T}(N)$.
(2). If $s' \in \mathcal{T}(N)$ then $s' = g^{\mathrm{r}}(v, s'_0)$, with $s'_0 \in \mathcal{T}(C'\langle\!|N'|\!\rangle)$. So by induction hypothesis there is $t \in \mathcal{T}(C'\langle\!|M'|\!\rangle)$ and a sum $\mathbb{T}_0 \subseteq \mathcal{T}(C'\langle\!|N'|\!\rangle)$ s.t. $t \to_{\mathrm{r}} s'_0 + \mathbb{T}_0$. Hence, $\mathcal{T}(M) = \mathcal{T}(g(v, C'\langle\!|M'|\!\rangle)) \ni g(v,t) \to_{\mathrm{r}} g(v, s'_0) + g(v, \mathbb{T}_0) = s' + g(v, \mathbb{T}_0) \subseteq \mathcal{T}(C\langle\!|N'|\!\rangle) = \mathcal{T}(N)$.

Case $C = @^{\mathrm{Cxt}}(C', P)$. Analogous as above.

Case $C = @^{\mathrm{Cxt}}(P, C')$.
(1). If $s \in \mathcal{T}(M)$ then $s = @^{\mathrm{r}}(p, [s_1, \ldots, s_k])$, with $p \in \mathcal{T}(P)$ and $s_j \in \mathcal{T}(C'\langle\!|M'|\!\rangle)$. So, for all $j = 1, \ldots, k$, by inductive hypothesis there is $\mathbb{T}_j \subseteq \mathcal{T}(C'\langle\!|N'|\!\rangle)$ s.t. $s \twoheadrightarrow_{\mathrm{r}} @^{\mathrm{r}}(p, [\mathbb{T}_1, \ldots, \mathbb{T}_k]) = \sum_{t_j \in \mathbb{T}_j} @^{\mathrm{r}}(p, [t_1, \ldots, t_k]) \subseteq \mathcal{T}(@^{\mathrm{r}}(p, C'\langle\!|N'|\!\rangle)) = \mathcal{T}(N)$.
(2). If $s' \in \mathcal{T}(N)$ then $s' = @^{\mathrm{r}}(p, [s'_1, \ldots, s'_k])$, with $p \in \mathcal{T}(P)$ and $s'_j \in \mathcal{T}(C'\langle\!|N'|\!\rangle)$. So, for all $j = 1, \ldots, k$, by induction hypothesis there is $t_j \in \mathcal{T}(C'\langle\!|M'|\!\rangle)$ and a sum $\mathbb{T}_j \subseteq \mathcal{T}(C'\langle\!|N'|\!\rangle)$ s.t. $t_j \to_{\mathrm{r}} s'_j + \mathbb{T}_j$. Hence, $\mathcal{T}(M) = \mathcal{T}(@^{\mathrm{r}}(P, C'\langle\!|M'|\!\rangle)) \ni @^{\mathrm{r}}(p, [t_1, \ldots, t_k]) \twoheadrightarrow_{\mathrm{r}} @^{\mathrm{r}}(p, [s'_1 + \mathbb{T}_1, \ldots, s'_k + \mathbb{T}_k]) = @^{\mathrm{r}}(p, [s'_1, \ldots, s'_k]) + \mathbb{T} = s' + \mathbb{T}$, for some sum $\mathbb{T} \subseteq \mathcal{T}(C\langle\!|N'|\!\rangle) = \mathcal{T}(N)$. □

Mimicking the definitions for λ-calculus one can give the definition of *congruence* on $\mathfrak{L}$, of $(\mathfrak{L}, \to)$-theory and of *term algebra*.

It is clear that $=_\tau$ is an equivalence. But also:

Corollary 4.3.6. *The equivalence $=_\tau$ is an $(\mathfrak{L}, \to)$-theory.*

Proof. The proof is exactly the same of Corollary 3.3.18, using Proposition 4.3.5 instead of Proposition 3.3.17. □

We now reproduce the same arguments of Section 3.3.1.

Definition 4.3.7. *The set of* rigid resource terms *is defined by:*

$$t ::= x \mid g^{\mathrm{r}}_j(v_{k_j}, t) \mid @^{\mathrm{r}}(t, \langle t \dots, t \rangle).$$

The set of rigid k-context is defined as expected adding the clause "$\xi_1 \mid \cdots \mid \xi_k$" for the holes.

Definition 4.3.8. *Let c be a resource-k-context. We define a set* $\mathrm{Rigid}(c)$ *of* rigid k-contexts*, whose elements are called the* rigids of c*, by induction on c as follows:*

1. $\mathrm{Rigid}(\xi_i) = \{\xi_i\}$
2. $\mathrm{Rigid}(x) = \{x\}$
3. $\mathrm{Rigid}(g^{\mathrm{r}}(v, c_0)) = \{g^{\mathrm{r}}(v, c_0^\bullet) \mid c_0^\bullet \in \mathrm{Rigid}(c_0)\}$
4. $\mathrm{Rigid}(@^{\mathrm{r}}(c_0, [c_1, \dots, c_k])) = \{\, @^{\mathrm{r}}(c_0^\bullet, \langle c^\bullet_{\sigma(1)}, \dots, c^\bullet_{\sigma(k)}\rangle) \mid c_i^\bullet \in \mathrm{Rigid}(c_i)$ *and σ permutation*$\}$.

Definition 4.3.9. *Let $c^\bullet$ be a rigid of a resource-k-context c and, for $i = 1, \dots, k$, let $\vec{v}^{\,i} := \langle v^i_1, \dots, v^i_{\deg_{\xi_i}(c)}\rangle$ be a list[7] of resource terms. We define as expected, by induction on c, a resource term $c^\bullet(\!|\vec{v}^{\,1}, \dots, \vec{v}^{\,k}|\!)$ following Definition 3.3.30.*

Remark 4.3.10. *One has that if $v \twoheadrightarrow_{\mathrm{r}} \mathbb{V}$ then:*

$$c^\bullet(\!|\dots, \langle \dots, v, \dots\rangle, \dots|\!) \twoheadrightarrow_{\mathrm{r}} \sum_{w \in \mathbb{V}} c^\bullet(\!|\dots, \langle \dots, w, \dots\rangle, \dots|\!) =: c^\bullet(\!|\dots, \langle \dots, \mathbb{V}, \dots\rangle, \dots|\!).$$

Let us extend the definition of Taylor expansion to resource k-contexts by adding, in its definition, the clause:

$$\mathcal{T}(\xi_i) := \{\xi_i\}.$$

It is clear that if C is a k-context then all elements of $\mathcal{T}(C)$ are resource k-contexts.

In the following, if $\vec{v}$ is a list, we denote with $[\vec{v}]$ the multiset associated with $\vec{v}$ (same elements but unordered).

Lemma 4.3.11. *Let C be a k-context and $c_1, c_2 \in \mathcal{T}(C)$. Let $c_1^\bullet$ and $c_2^\bullet$ rigids respectively of c_1 and c_2. For $i = 1 \dots, k$, let $\vec{v}^{\,i} = \langle v^i_1, \dots, v^i_{\deg_{\xi_i}(c_1)}\rangle$ and $\vec{u}^{\,i} = \langle u^i_1, \dots, u^i_{\deg_{\xi_i}(c_2)}\rangle$ be lists of resource terms. If $c_1^\bullet(\!|\vec{v}^{\,1}, \dots, \vec{v}^{\,k}|\!) = c_2^\bullet(\!|\vec{u}^{\,1}, \dots, \vec{u}^{\,k}|\!)$ then $c_1 = c_2$ and $[\vec{v}^{\,i}] = [\vec{u}^{\,i}]$ for all i.*

The proof of the previous lemma is a trivial adaptation of the one of Lemma 3.3.32, where the case of the constructor g corresponds to the case of the λ-abstraction.

Lemma 4.3.12. *1. Let C be a k-context. Then:*

$$\mathcal{T}(C(\!|M_1, \dots, M_k|\!)) = \{c^\bullet(\!|\vec{s}^{\,1}, \dots, \vec{s}^{\,k}|\!) \mid c \in \mathcal{T}(C),\ c^\bullet \text{ rigid of } c \text{ and } \vec{s}^{\,i} \text{ list of elements of } \mathcal{T}(M_i)\}.$$

2. *Let $c = c(\!|\xi|\!)$ be a single-hole resource context, $M \in \mathfrak{L}$ and $s_0 \in \mathfrak{L}^{\mathrm{r}}$. If $c(\!|s_0|\!) \in \mathcal{T}(M)$, then there is a context $C = C(\!|\xi|\!)$, an $N \in \mathfrak{L}$, a resource context $\widetilde{c} \in \mathcal{T}(C)$, a rigid $\widetilde{c}^\bullet$ of c and $s_1 \dots, s_{\deg_\xi \widetilde{c}-1} \in \mathcal{T}(N)$, s.t.:*

[7] If $\deg_{\xi_i}(c) = 0$ we mean the empty list.

(a) $M = C(\!|N|\!)$

(b) $s_0 \in \mathcal{T}(N)$

(c) $c(\!|t|\!) = \widetilde{c}^{\bullet}(\!|\langle t, s_1 \ldots, s_{\deg_\xi(\widetilde{c})-1}\rangle|\!)$ *for all* $t \in \lambda\mu^{\mathrm{r}}$.

Proof. The proof is a trivial adaptation of the one of Lemma 3.3.33, where the case of the constructor g corresponds to the case of the λ-abstraction. Let us write it for the seek of clarity: Let $c = g^{\mathrm{r}}(v, c_1)$. Since $c(\!|s_0|\!) \in \mathcal{T}(M)$ then $M = g(v, M_1)$ with $c_1(\!|s_0|\!) \in \mathcal{T}(M_1)$. We can thus take $C := g(v, C_1)$, $\widetilde{c} := g^{\mathrm{r}}(v, \widetilde{c_1})$, $\widetilde{c}^{\bullet} := g^{\mathrm{r}}(v, \widetilde{c_1}^{\bullet})$, where $C_1, N, \widetilde{c_1}, \widetilde{c_1}^{\bullet}$ and $s_1, \ldots, s_{\deg_\xi(\widetilde{c_1})}$ are given by the inductive hypothesis. □

From now on, we suppose the analogous of Ehrhard-Regnier's "non-interference" property (Lemma 3.3.20), that is:

Assumption 4. *For all* $t, s \in \mathcal{T}(M)$ *s.t.* $t \neq s$, *you have* $\mathrm{nf}_r(t) \cap \mathrm{nf}_r(s) = \emptyset$.

Remark that thanks to the previous Assumption 4 and thanks to the strong normalisation one (Assumption 2), the analogue of Lemma 3.3.21 holds. As usual, we will not explicitly mention its use.

From now on, we suppose that:

Assumption 5. *For all* $t \in \mathcal{T}(M)$, *if* $t \to_{\mathrm{base}^{\mathrm{r}}} \mathbb{T}'$ *then there is* $N \in \mathfrak{L}$ *s.t.* $M \to N$ *and* $\mathbb{T}' \subseteq \mathcal{T}(N)$.

This assumption holds in λ-calculus (it is a particular case of Proposition 3.3.34).

As in λ-calculus, we can prove the following proposition. The proof is almost the same of the one of Proposition 3.3.34, but we write it for the seek of clarity and because we use here Assumption 5 where in Proposition 3.3.34 we use Lemma 3.3.16. When we will prove that this assumption also holds in $\lambda\mu$-calculus (Proposition 4.4.52), we will use the analogous of Lemma 3.3.16, that is Lemma 4.4.48.

Proposition 4.3.13. *If* $\mathcal{T}(M) \supseteq \mathbb{T} \to_{\mathrm{r}} \mathbb{T}'$ *then there is* $N \in \mathfrak{L}$ *and a sum* $\widetilde{\mathbb{T}} \subseteq \mathcal{T}(N)$ *s.t.* $M \to N$ *and* $\mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \widetilde{\mathbb{T}}$.

Proof. Saying that $\mathbb{T} \to_{\mathrm{r}} \mathbb{T}'$ means that $\mathbb{T}$ has shape $\mathbb{T} = \sum_i t_i + c(\!|h|\!)$ and $\mathbb{T}'$ has shape $\mathbb{T}' = \sum_i t_i + c(\!|\mathcal{H}|\!)$, for some single-hole resource context c, a resource term h and a sum $\mathcal{H}$ s.t. $h \to_{\mathrm{base}} \mathcal{H}$. But since $c(\!|h|\!) \in \mathbb{T} \subseteq \mathcal{T}(M)$, by Lemma 4.3.12(2) we get a context C_0, a term $N' \in \mathfrak{L}$, a resource context $c_0 \in \mathcal{T}(C_0)$, a rigid $c_0^{\bullet}$ of c_0 and resource terms $\vec{s} \in \mathcal{T}(N')$ s.t. $M = C_0(\!|N'|\!)$, $h \in \mathcal{T}(N')$ and $c(\!|u|\!) = c_0^{\bullet}(\!|u, \vec{s}|\!)$ for all $u \in \mathfrak{L}^{\mathrm{r}}$. Now we can apply Assumption 5 to $h \in \mathcal{T}(N')$ obtaining an $N'' \in \mathfrak{L}$ s.t. $N' \to N''$ and $\mathcal{H} \subseteq \mathcal{T}(N'')$. Set $N := C_0(\!|N''|\!)$, so that $M = C_0(\!|N'|\!) \to N$. Now: every $t_i \in \mathcal{T}(M) = \mathcal{T}(C_0(\!|N'|\!))$, so by Lemma 4.3.12(1) it must have shape $t_i = c_i^{\bullet}(\!|\vec{v}^{\,i}|\!)$ for some resource terms $v_j^i \in \mathcal{T}(N')$, a context $c_i \in \mathcal{T}(C_0)$ and a rigid $c_i^{\bullet}$ of c_i. But since $N' \to N''$ we can apply Proposition 4.3.5(1) on v_j^i and obtain that $v_j^i \twoheadrightarrow_{\mathrm{r}} \mathbb{V}_j^i$ for some sum $\mathbb{V}_j^i \subseteq \mathcal{T}(N'')$. So $t_i \twoheadrightarrow_{\mathrm{r}} c_i^{\bullet}(\!|\vec{\mathbb{V}}^{\,i}|\!) =: \mathbb{T}_i$. Using again Lemma 4.3.12(1) one has that $\mathbb{T}_i \subseteq \mathcal{T}(N)$. Now, let's use again Proposition 4.3.5(1), this time on $s \in \mathcal{T}(N')$. Since $N' \to N''$ we obtain sums $\mathbb{S}_i \subseteq \mathcal{T}(N'')$ s.t. $s_i \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_i$. So we have: $c(\!|\mathcal{H}|\!) = c_0^{\bullet}(\!|\mathcal{H}, \vec{s}|\!) \twoheadrightarrow_{\mathrm{r}} c_0^{\bullet}(\!|\mathcal{H}, \vec{\mathbb{S}}|\!) =: \mathbb{U}$. But since $\mathcal{H} \subseteq \mathcal{T}(N'')$ and every $\mathbb{S}_i \subseteq \mathcal{T}(N'')$, again thanks to Lemma 4.3.12(1) one has $\mathbb{U} \subseteq \mathcal{T}(C_0(\!|N''|\!)) = \mathcal{T}(N)$. Now letting $\widetilde{\mathbb{T}} := \sum_i \mathbb{T}_i + \mathbb{U} \subseteq \mathcal{T}(N)$ one has $\mathbb{T}' \twoheadrightarrow_{\mathrm{r}} \widetilde{\mathbb{T}}$ and we are done. □

As a corollary, we obtain the following analogous of Corollary 3.3.26, whose proof can be immediately adapted from the alternative proof given at page 54.

Corollary 4.3.14. *For all $\mathbb{T} \subseteq \mathcal{T}(M)$, there exist $N \in \mathfrak{L}$ s.t. $M \twoheadrightarrow N$ and $\mathrm{nf_r}(\mathbb{T}) \subseteq \mathcal{T}(N)$.*

Exactly as in λ-calculus, we can set the same notions needed for the Stability Property.

Definition 4.3.15. *Given a non-empty subset $\mathcal{X} \subseteq \Lambda$, define its $\mathcal{T}$-*infimum *$\bigcap \mathcal{X} \subseteq \mathfrak{L}^{\mathrm{r}}$ as:*

$$\bigcap \mathcal{X} := \bigcap_{M \in \mathcal{X}} \mathrm{NF}\mathcal{T}(M).$$

We say that $\mathcal{X}$ is bounded *iff there exists an $L \in \mathfrak{L}$ such that $M \leq_{\mathcal{T}} L$ for all $M \in \mathcal{X}$.*

Now we have:

Theorem 4.3.16 (Stability)**.** *Let C be an n-context and fix non-empty bounded $\mathcal{X}_1, \dots, \mathcal{X}_n \subseteq \mathfrak{L}$. For all $M_1, \dots, M_n \in \mathfrak{L}$ s.t. (for $i = 1, \dots, n$)*

$$M_i = \bigcap \mathcal{X}_i$$

we have:

$$C(\!|M_1, \dots, M_n|\!) = \bigcap_{\substack{N_1 \in \mathcal{X}_1 \\ \cdots \\ N_n \in \mathcal{X}_n}} C(\!|N_1, \dots, N_n|\!).$$

Proof. We do not write it again, because it is exactly the same proof as we did for λ-calculus in Theorem 3.3.37. Remark, in particular, that is uses the "non-interference assumption" (Assumption 4) as well as the notion of rigids. □

Once all the Assumptions will be proven for λμ-calculus, we will get "for free" the Stability property (Theorem 4.4.62).

4.4 The resource λμ-calculus

In Example 2 we already remarked that the syntax of λμ-calculus is of the shape of those considered in the previous Section 4.3. Same for contexts and reduction, which clearly satisfies Assumption 1. We turn to consider now the resource approximation of this calculus. It is natural to treat the μ-abstraction constructor as linear, so we only have to linearize applications, just like one does in resource λ-calculus.

Definition 4.4.1. *The set $\lambda\mu^{\mathrm{r}}$ of* resource λμ-terms *is defined by:*

$$t ::= x \;\mid\; \lambda x.t \;\mid\; t[t, \dots, t] \;\mid\; \mu\alpha.{}_\beta|t|$$

where $[t, \dots, t] \in !\lambda\mu^{\mathrm{r}}$. Resource terms are considered up to renaming of variables and names. Resource contexts are defined as expected. For ν a variable or a name, the degree $\deg_\nu(t) \in \mathbb{N}$ *of ν in t, is defined as the number of (free) occurrences of ν in t.*

This definition is of the shape of Definition 4.3.1. Same for resource contexts.

The following is easily proven.

Lemma 4.4.2. *Every $\lambda\mu^{\mathrm{r}}$-term t has the following shape:*

$$t = \lambda\vec{x}_1.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x}_k.\mu\alpha_k.{}_{\beta_k}|p[\vec{u}^{\,1}]\cdots[\vec{u}^{\,n}]||$$

where p is either a variable, or a λ-redex or a μ-redex. p is called the head redex *of t if it is a λμ-redex, and it is called the* head variable *of t otherwise. The sequence $\lambda\vec{x}_1.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x}_k.\mu\alpha_k.{}_{\beta_k}|*||$ is called the* head *of t. The same holds for λμ-terms as well.*

We use Definition 4.3.1 in order to get a notion of sums of $\lambda\mu^{\mathrm{r}}$-terms, and we are going to define now the reduction in $\lambda\mu^{\mathrm{r}}$, which is extended to all $2\langle\lambda\mu^{\mathrm{r}}\rangle$ following the same Definition. We want it to satisfy Assumption 2. In order to define such a reduction, we need to be able to divide a multiset into a certain number of "blocks". The notion which we want to define for bags is similar to the combinatorial notion of *partition of an integer*, and more specifically to that of *weak composition*. The extension of these notions from integers to multisets is well known and studied in the literature of combinatorics (see for example [Ben74]).

Definition 4.4.3 (Weak composition of a multiset)**.** *A* partition *(resp.* weak partition*) of a multiset* $[\vec{u}]$ *is a multiset* $[[\vec{v}^1],\dots,[\vec{v}^k]]$ *of* non-empty *(resp.* possibly empty*) multisets, called* blocks*, s.t.* $[\vec{u}]=[\vec{v}^1]*\dots*[\vec{v}^k]$.
A composition *(resp.* weak composition – w.c. *for short) of a multiset* $[\vec{u}]$ *is a tuple* $([\vec{v}^1],\dots,[\vec{v}^k])$ *of multisets s.t.* $[[\vec{v}^1],\dots,[\vec{v}^k]]$ *is a partition (resp. weak partition) of* $[\vec{u}]$.

Observe that the empty bag 1 admits no partitions but admits infinite weak partitions: they are the multisets of shape $[1,\dots,1]$ ($h\geq 0$ times 1). Let us give some other examples: the set of all the weak partitions of the bag $[x]$ is:

$$\{[[x]],[[x],1],[[x],1,1],\dots\}.$$

The set of all weak partitions of $[x,x]$ is:

$$\{[[x,x]],[[x],[x]],[[x,x],1],[[x],[x],1],[[x,x],1,1],[[x],[x],1,1],\dots\}.$$

The w.c.'s are then simply obtained by choosing an order on the weak partitions.

Definition 4.4.4. *Let* $t\in\lambda\mu^{\mathrm{r}}$ *and* $[\vec{u}]=[u_1,\dots,u_k]\in !\,\lambda\mu^{\mathrm{r}}$.

1. *We define as usual the* linear substitution $t\langle[u_1,\dots,u_k]/x\rangle\in 2\langle\lambda\mu^{\mathrm{r}}\rangle$ *as* $t\langle[u_1,\dots,u_k]/x\rangle:=0$ *if* $\deg_x(t)\neq k$*, and* $t\langle[u_1,\dots,u_k]/x\rangle:=\sum_{\sigma\in\mathfrak{S}_k} t\left\{u_{\sigma(1)}/x^{(1)},\dots,u_{\sigma(k)}/x^{(k)}\right\}$ *if* $\deg_x(t)=k$.
   Here $x^{(1)},\dots,x^{(k)}$ *is any fixed enumeration of the occurrences*[8] *of* x *in* t *and* $\mathfrak{S}_k$ *is the set of bijections on* k*-elements.*

2. *In order to linearize the* μ*-reduction we introduce the* linear named application $\langle t\rangle_\alpha[\vec{u}]\in 2\langle\lambda\mu^{\mathrm{r}}\rangle$*, defined as:*

   if $\deg_\alpha(t)=0$ *then* $\langle t\rangle_\alpha 1:=t$ *and* $\langle t\rangle_\alpha[v,\vec{u}]:=0$
   if $\deg_\alpha(t)=:d\neq 0$ *then:*

$$\langle t\rangle_\alpha[\vec{u}]:=\sum t\left\{{}_\alpha|(\cdot)[\vec{s}^1]|/_{{}_\alpha|\cdot|^{(1)}},\dots,{}_\alpha|(\cdot)[\vec{s}^d]|/_{{}_\alpha|\cdot|^{(d)}}\right\}$$

   where the sum is taken over all $([\vec{s}^1],\dots,[\vec{s}^d])$ *w.c. of* $[\vec{u}]$ *of length* d *(which are necessarily in finite number), and* ${}_\alpha|\cdot|^{(1)},\dots,{}_\alpha|\cdot|^{(d)}$ *is any fixed enumeration of the occurrences of* α *in* t*. The same exact definition is given also in the case* t *is a* named term.

Remark 4.4.5. *The linear substitution and the linear named application are well defined since they do not depend on the particular enumeration of the chosen occurrences. Another way of defining them is by induction, as presented in Figure 4.1 and Figure 4.2.*

In Definition 4.4.4 we write that "the same exact definition is given also in the case t is a *named term*". This is reflected in the inductive definition of Figure 4.2, where in fact we

[8] We are of course employing in the usual way the *informal* notion of "occurrence" in a resource term (due to the presence of "unordered" bags). The same for the next part of the definition.

The linear substitution $t\langle[\vec{u}]/x\rangle$ is inductively defined as in Figure 3.1 with the additional clause:

$$(\mu\alpha._\beta|s|)\langle[\vec{u}]/x\rangle = \mu\alpha._\beta|s\langle[\vec{u}]/x\rangle|$$

Figure 4.1: Inductive characterisation of linear substitution in λμ-calculus

$$\begin{array}{ccc}
\langle x\rangle_\alpha[v,\vec{u}] = 0 & \langle x\rangle_\alpha 1 = x & \langle_\eta|t|\rangle_\alpha[\vec{u}] = {}_\eta|\langle t\rangle_\alpha[\vec{u}]| \quad (\text{if } \eta \neq \alpha) \\
\langle\mu\gamma._\eta|t|\rangle_\alpha[\vec{u}] = \mu\gamma.\langle_\eta|t|\rangle_\alpha[\vec{u}] & \langle\lambda y.t\rangle_\alpha[\vec{u}] = \lambda y.\langle t\rangle_\alpha[\vec{u}] & \langle_\alpha|t|\rangle_\alpha[\vec{u}] = \sum\limits_{\substack{([\vec{w}^1],[\vec{w}^2]) \\ w.c.\ of\ [\vec{u}]}} {}_\alpha|\big(\langle t\rangle_\alpha[\vec{w}^1]\big)\,[\vec{w}^2]| \\
\langle t[v_1,\dots,v_n]\rangle_\alpha[\vec{u}] = & \sum\limits_{([\vec{w}^0],\dots,[\vec{w}^n])\ w.c.\ of\ [\vec{u}]} & \big(\langle t\rangle_\alpha[\vec{w}^0]\big)\,\big[\,\langle v_1\rangle_\alpha[\vec{w}^1],\dots,\langle v_n\rangle_\alpha[\vec{w}^n]\,\big]\,.
\end{array}$$

Figure 4.2: Inductive characterisation of linear named application

consider also the case $\langle_\eta|t|\rangle_\alpha[\vec{u}]$ (for η equal or not to α) of the linear application of a named term to a bag through a name. That is, we are actually defining the operation $\langle\cdot\rangle_\alpha[\vec{u}]$ on Saurin's $\Lambda\mu$-calculus[9] (but $[\vec{u}]$ is still a bag of $\lambda\mu$-terms, because that is enough for us). This raises no problems, since Parigot's $\lambda\mu$-calculus is an inductive subsystem of it. But it is important to notice that many of our definitions do already make sense in Saurin's calculus and, in order to get things right in our framework ($\lambda\mu$-calculus), we need sometimes to apply definitions on $\Lambda\mu$-calculus. An example of this phenomenon is Lemma 4.4.50. In general, as we will mention in the conclusions, this raises the question of whether it is possible to lift the present work from $\lambda\mu$-calculus to $\Lambda\mu$-calculus.

Let us now define our reduction for the resource $\lambda\mu$-calculus.

Definition 4.4.6. *Define a reduction*[10] $\to_{\mathrm{r}} \subseteq \lambda\mu^{\mathrm{r}} \times 2\langle\lambda\mu^{\mathrm{r}}\rangle$ *as the resource-context closure of the union* $\to_{\mathrm{base}^{\mathrm{r}}}$ *of:*

$$(\lambda x.t)[\vec{u}] \to_{\lambda^{\mathrm{r}}} t\langle[\vec{u}]/x\rangle$$
$$(\mu\alpha._\beta|t|)[\vec{u}] \to_{\mu^{\mathrm{r}}} \mu\alpha.\langle_\beta|t|\rangle_\alpha[\vec{u}]$$
$$\mu\gamma._\alpha|\mu\beta._\eta|t|| \to_{\rho^{\mathrm{r}}} \mu\gamma.\,(_\eta|t|\{\alpha/\beta\})\,.$$

We extend it to all $2\langle\lambda\mu^{\mathrm{r}}\rangle \times 2\langle\lambda\mu^{\mathrm{r}}\rangle$ *as explained in the "formal sums" paragraph of Section 2.1.*

Observe that, just like $\lambda\mu$-terms, every r-normal $\lambda\mu^{\mathrm{r}}$-term t has a head variable, has no ρ-redexes in its head and its bags contain only r-normal $\lambda\mu^{\mathrm{r}}$-terms.

The reduction $\to_{\lambda^{\mathrm{r}}}$ is the usual linearisation of $\to_\lambda$ and $\to_{\rho^{\mathrm{r}}}$ simply simulates $\to_\rho$. For $\to_{\mu^{\mathrm{r}}}$ we believe it is interesting the following consideration:

Remark 4.4.7. *It is known (for instance this is considered in [Sau12]), that one can decompose* $\to_\mu$ *introducing a new reduction* $\to_{\mathrm{fst}}$*. The decomposition transforms a* μ *into a* λ*, and then reduces all the complexity of* $\to_\mu$ *to an appropriate* $\to_\lambda$*. The reduction* $\to_{\mathrm{fst}}$ *is:*

$$\mu\alpha._\beta|M| \to_{\mathrm{fst}} \lambda z.\mu\alpha.(_\beta|M|)_\alpha z$$

[9] Whose syntax is $M ::= x \mid \lambda x.M \mid MM \mid \mu\alpha.M \mid {}_\alpha|M|$.

[10] Here we use again the same symbol as the resource reduction for λ-calculus. See footnote 3.

(for z fresh), and the decomposition of $\to_\mu$ *is as follows:*

$$(\mu\alpha.{}_\beta|M|)N \to_{\mathrm{fst}} (\lambda z.\mu\alpha.({}_\beta|M|)_\alpha z)N \to_\lambda \mu\alpha.({}_\beta|M|)_\alpha N.$$

One may wonder if our resource reduction $\to_{\mu^{\mathrm{r}}}$ *is good enough to be compatible with this fine decomposition. For that, we need to linearise Saurin's decomposition, and since we already know what the linearised* λ*-reduction is, we only need to define a correct linearisation* $\to_{\mathrm{fst}^{\mathrm{r}}}$ *of* fst*. Then, we have to verify whether we can still decompose* $\to_{\mu^{\mathrm{r}}}$ *by first applying* $\to_{\mathrm{fst}^{\mathrm{r}}}$ *and then* $\to_{\lambda^{\mathrm{r}}}$*. The natural choice for* $\to_{\mathrm{fst}^{\mathrm{r}}}$ *is:*

$$\mu\alpha.{}_\beta|t| \to_{\mathrm{fst}^{\mathrm{r}}} \lambda z.\mu\alpha.\sum_{m\geq 0}\langle {}_\beta|t|\rangle_\alpha[\overbrace{z,\dots,z}^{m}]$$

(for z fresh). Note that this reduction $\to_{\mathrm{fst}^{\mathrm{r}}}$ *gives an infinite sum. Of course to manipulate such an object one should pay attention; but we will not, because we want to keep this computation just as an heuristic. Let us now verify that the decomposition "*$\to_{\mu^{\mathrm{r}}} = \twoheadrightarrow_{\lambda^{\mathrm{r}}} \circ \to_{\mathrm{fst}^{\mathrm{r}}}$*" holds. We have:*

$$(\mu\alpha.{}_\beta|t|)[\vec{u}] \to_{\mathrm{fst}^{\mathrm{r}}} \left(\lambda z.\mu\alpha.\sum_{m\geq 0}\langle {}_\beta|t|\rangle_\alpha[\overbrace{z,\dots,z}^{m}]\right)[\vec{u}] \twoheadrightarrow_{\lambda^{\mathrm{r}}} \mu\alpha.\sum_{m\geq 0}(\langle {}_\beta|t|\rangle_\alpha[\overbrace{z,\dots,z}^{m}])\langle[\vec{u}]/z\rangle.$$

If $\deg_\alpha({}_\beta|t|) = 0$ *then, in the last sum, all the addends with* $m \neq 0$ *are* 0*; so we are left with:* $\mu\alpha.(\langle {}_\beta|t|\rangle_\alpha 1)\langle[\vec{u}]/z\rangle = \mu\alpha.\langle {}_\beta|t|\rangle_\alpha[\vec{u}]$ *and we are done. And if* $\deg_\alpha({}_\beta|t|) =: d \geq 1$ *then by definition of linear substitution one easily sees that, in the previous sum, all the addends with* $m \neq n$ *are* 0 *(*n *being the length of* $[\vec{u}]$*); since the w.c.'s of* $[\overbrace{z,\dots,z}^{n}]$ *of length* d *are the* d*-uples* $([\overbrace{z,\dots,z}^{k_1}],\dots,[\overbrace{z,\dots,z}^{k_d}])$ *s.t.* $k_1+\dots+k_d = n$*, the last sum becomes:*

$$\mu\alpha.\sum_{k_1+\dots+k_d=n} {}_\beta|t|\{{}_\alpha|(\cdot)[\overbrace{z,\dots,z}^{k_1}]|/_{{}_\alpha|\cdot|^{(1)}},\dots,{}_\alpha|(\cdot)[\overbrace{z,\dots,z}^{k_d}]|/_{{}_\alpha|\cdot|^{(d)}}\}\,\langle[\vec{u}]/z\rangle.$$

It is now easy from it to reconstruct the term $\mu\alpha.\langle {}_\beta|t|\rangle_\alpha[\vec{u}]$ *and we are done again.*

This remark can be seen as an a posteriori justification of the correctness of our linearisation $\to_{\mu^{\mathrm{r}}}$ *of* $\to_\mu$*, or as an heuristic to find it.*

A resource calculus is useful because, in addition to linearity, resource sensitiveness entails strong properties such as strong normalization and confluence. This is what we are going to prove now. Compared with resource λ-calculus in our setting those properties require some more thinking.

Strong normalization With $\to_{\lambda^{\mathrm{r}}}$ we erase exactly one λ, and with $\to_{\rho^{\mathrm{r}}}$ we erase exactly one μ. With $\to_{\mu^{\mathrm{r}}}$ however, the situation is more subtle: we are not creating nor erasing λ's or μ's (which remain thus in constant number), but we are eventually making the the reduct grow by creating an arbitrarily large number of new applications, possibly even creating new (λ or μ-)redexes. However, these new applications are created "deep inside" the reduct. In fact, in order to pass from the μ-redex $(\mu\alpha.{}_\beta|t|)[\vec{u}]$ to a reduct $t' \in \mu\alpha.\langle {}_\beta|t|\rangle_\alpha[\vec{u}]$, we:

- first, decompose $[\vec{u}]$ in several blocks
- then, erase $[\vec{u}]$
- finally, put each block *inside* a certain *named* subterm of ${}_\beta|t|$.

The idea is thus that we replaced a bag with many new bags which are at a "deeper depth". Our aim is to find a decreasing measure, while this intuition of "depth" is increasing. But as we will see in Remark 4.4.9, it will be immediate to recognize that actually this number is necessary bounded by the number of μ-occurrences in the term, which is invariant under $\to_{\mu^{\mathrm{r}}}$, so the former subtracted from the latter should decrease. Remark that there is one case in which we do not create new bags and we simply erase one already existing one: it is the case where $[\vec{u}] = 1$ and $\deg_\mu(_\beta|t|) = 0$, so we have to make sure our measure decreases in this case as well.

Definition 4.4.8. *Let $t \in \lambda\mu^{\mathrm{r}}$ and let b be an* occurrence *of a bag or of a subterm of t. Define the* depth $d_t(b) \in \mathbb{N}$ *of b in t as the number of* named *subterms of t containing b.*

Remark 4.4.9. *By definition of the grammar of the $\lambda\mu^{\mathrm{r}}$-calculus there are as many named subterms of t as μ-abstractions*[11] *in t, which are $\deg_\mu(t)$ by definition. So it must always be:*

$$d_t(b) \leq \deg_\mu(t).$$

Definition 4.4.10. *Recalling the notation of multisets in Section 2.1, we define the* multiset measure $\mathrm{m}(t) \in !\mathbb{N}$ *of a $\lambda\mu^{\mathrm{r}}$-term t as:*

$$\mathrm{m}(t) := \deg_\mu(t) - [\, d_t(b) \mid b \textit{ occurrence of bag in } t\,].$$

Remark 4.4.9 assures that $\mathrm{m}(t) \in !\mathbb{N}$ (and not in $!\mathbb{Z}$), which is well-founded w.r.t. the multiset order. We order thus multiset measures w.r.t to this order.

In this chapter we will have to talk about many occurrences of bags in terms. This is why we will sometimes use the notation "b" for such occurrences, just like we did in the previous definition, instead of explicitly writing the bag as "$[\vec{u}]$".

The measure $\mathrm{m}(\cdot)$ is "almost" the good one for strong normalization:

Proposition 4.4.11. *If $t \to_{\mu^{\mathrm{r}}} t' + \mathbb{T}$ then $\mathrm{m}(t) > \mathrm{m}(t')$.*

Proof. If $t \to_{\mu^{\mathrm{r}}} t' + \mathbb{T}$ then $t = c(\!|(\mu\alpha.{}_\beta|s|)b_0|\!)$ and $t' = c(\!|h|\!)$ with $h \in \mu\alpha.\langle {}_\beta|s|\rangle_\alpha b_0$ and c a single-hole resource context. Call $k := \deg_\mu(t) = \deg_\mu(t')$ and consider $\deg_\alpha(_\beta|s|) \in \mathbb{N}$. The are two cases:

> Case $\deg_\alpha(_\beta|s|) = 0$. By definition of $\to_{\mu^{\mathrm{r}}}$ this is possible only if $b_0 = 1$ (otherwise $t \to_{\mu^{\mathrm{r}}} 0$) and $h = \mu\alpha.{}_\beta|s|$. So in t there are the exact same occurrences of bags as in t' and they are at the same depth, except for b_0 which is in t but not in t'. This means that $\mathrm{m}(t) = \mathrm{m}(t') * [\, k - d_t(b_0)\,] > \mathrm{m}(t')$.
>
> Case $\deg_\alpha(_\beta|s|) =: n \geq 1$. Then: $h = \mu\alpha.{}_\beta|s|\left\{ {}_\alpha|(\cdot)b_1|/_{{}_\alpha|\cdot|^{(1)}}, \ldots, {}_\alpha|(\cdot)b_n|/_{{}_\alpha|\cdot|^{(n)}} \right\}$ for a w.c. $(b_1, \ldots, b_n)$ of b_0. So $\mathrm{m}(t') = k - A'$ and $\mathrm{m}(t) = k - A$, with A' and A the multisets:
>
> $$\begin{array}{rl} A' = & [\, d_{t'}(b_1), \ldots, d_{t'}(b_n)\,] * [\, d_{t'}(b) \mid b \textit{ in } c\,] * [\, d_{t'}(b) \mid b \textit{ in } s\,] * [\, d_{t'}(b) \mid b \textit{ in a } v \in b_i \textit{ for an } i\,]) \\ A = & \quad [\, d_t(b_0)\,] \quad * [\, d_t(b) \mid b \textit{ in } c\,] * [\, d_t(b) \mid b \textit{ in } s\,] * \quad [\, d_t(b) \mid b \textit{ in a } v \in b_0]\,]). \end{array}$$
>
> Now for $i = 1, \ldots, n$ we have: $d_{t'}(b_i) = d_{t'}(h) + d_h(b_i) > d_t(b_0)$ since as one sees from the expression of h, we have $d_{t'}(h) = d_t(b_0)$ and $d_h(b_i) > 0$. Also, it is easily understood that for all b occurring in c, or occurring in s, we have: $d_{t'}(b) = d_t(b)$. Finally, observe that since $(b_1, \ldots, b_n)$ is a w.c. of b_0, then: b occurs in some $v \in b_0$ iff b occurs in some $v \in b_i$ for some i. And for all such b we have: $d_{t'}(b) = d_{t'}(v) + d_v(b) > d_t(v) + d_v(b) = d_t(b)$ since $d_{t'}(v) = d_{t'}(b_i) > d_t(b_0) = d_t(v)$. All these considerations precisely mean $\mathrm{m}(t) > \mathrm{m}(t')$. □

[11] Here there is a difference with Saurin's $\Lambda\mu$-calculus, in which one can have as many named subterms as one wants, independently from μ-abstractions.

Proposition 4.4.12. *If* $t \to_{\lambda^{\mathrm{r}}} t' + \mathbb{T}$ *then* $\mathrm{m}(t) > \mathrm{m}(t')$.

Proof. If $t \to_{\lambda^{\mathrm{r}}} t'+\mathbb{T}$ then $t = c(\!|(\lambda x.s)b_0|\!)$ and $t' = c(\!|h|\!)$ with $h = s\{u_{\sigma(1)}/x^{(1)}, \ldots, u_{\sigma(n)}/x^{(n)}\} \in s\langle b_0/x\rangle$, for c a single-hole resource context, σ a permutation and $b_0 = [u_1, \ldots, u_n]$. Call $k := \deg_\mu(t) = \deg_\mu(t')$. We have:

$$\mathrm{m}(t') \;=\; k \;-\; [\, d_{t'}(b) \mid b \textit{ in } c \,] * [\, d_{t'}(b) \mid b \textit{ in } s \,] * [\, d_{t'}(b) \mid b \textit{ in some } u_i \textit{ in } b_0 \,]$$

$$\mathrm{m}(t) \;=\; k \;-\; [\, d_t(b) \mid b \textit{ in } c] * [\, d_t(b) \mid b \textit{ in } s] * [\, d_t(b) \mid b \textit{ in some } u_i \textit{ in } b_0 \,] * [\, d_t(b_0)].$$

Now, it is easily understood that if b occurs in c, or b occurs in s, then $d_{t'}(b) = d_t(b)$. Furthermore, for all b occurrence of bag in some $u_{\sigma(i)}$ belonging to b_0, one has:

$$d_{t'}(b) = d_c(\xi) + d_s(x^{(i)}) + d_{u_{\sigma(i)}}(b) \geq d_c(\xi) + d_{u_{\sigma(i)}}(b) = d_t(b).$$

Thus:

$$\mathrm{m}(t) \geq \mathrm{m}(t') * [\, k - d_t(b_0) \,] > \mathrm{m}(t').$$

□

However, only $\mathrm{m}(t)$ is not enough to prove strong normalization. In fact:

Proposition 4.4.13. *If* $t \to_{\rho^{\mathrm{r}}} t'+\mathbb{T}$ *then* $\mathrm{m}(t) \geq \mathrm{m}(t')$*, and there are cases in which the equality holds.*

Proof. If $t \to_{\rho^{\mathrm{r}}} t'+\mathbb{T}$ then $t = c(\!|h|\!)$ and $t' = c(\!|h'|\!)$, with $h = \mu\gamma.{}_\alpha|\mu\beta.{}_\eta|s||$ and $h' = \mu\gamma.{}_\eta|s|\{\alpha/\beta\}$ and c a single-hole resource context. Therefore:

$$\mathrm{m}(t') = [\, \deg_\mu(t') - d_{t'}(b) \mid b \textit{ in } c \,] * [\, \deg_\mu(t') - d_{t'}(b) \mid b \textit{ in } s \,].$$

$$\mathrm{m}(t) = [\, \deg_\mu(t) - d_t(b) \mid b \textit{ in } c] * [\, \deg_\mu(t) - d_t(b) \mid b \textit{ in } s].$$

First, remark that $\deg_\mu(t') = 1 + \deg_\mu(c) + \deg_\mu(s)$ and $\deg_\mu(t) = 2 + \deg_\mu(c) + \deg_\mu(s)$.

Also, notice as usual that if b occurs in c then $d_{t'}(b) = d_c(b) = d_t(b)$. Putting these things together we have that, if c *contains at least one bag*:

$$[\, \deg_\mu(t') - d_{t'}(b) \mid b \textit{ in } c \,] = (1+\deg_\mu(s))+\mathrm{m}(c) < (2+\deg_\mu(s))+\mathrm{m}(c) = [\, \deg_\mu(t) - d_t(b) \mid b \textit{ in } c \,].$$

On the other hand, if c does *not* contain any bag, the previous multisets are both empty, thus equal.

Now let's see what happens if b occurs in s. We have: $d_{t'}(b) = 1 + d_c(\xi) + d_s(b)$ and $d_t(b) = 2 + d_c(\xi) + d_s(b)$. Therefore:

$$[\, \deg_\mu(t') - d_{t'}(b) \mid b \textit{ in } s \,] = (\deg_\mu(c) - d_c(\xi)) + \mathrm{m}(s) = [\, \deg_\mu(t) - d_t(b) \mid b \textit{ in } s \,].$$

These facts immediately imply that $\mathrm{m}(t) \geq \mathrm{m}(t')$ and, to give an example, one has (for $\beta \neq \eta$) $\mathrm{m}(\mu\gamma.{}_\alpha|\mu\beta.{}_\eta|x||) = 1 = \mathrm{m}(\mu\gamma.{}_\eta|x|)$ while $\mu\gamma.{}_\alpha|\mu\beta.{}_\eta|x|| \to_{\rho^{\mathrm{r}}} \mu\gamma.{}_\eta|x|$. □

That is why, in order to get a strongly normalising measure, we add another component:

Definition 4.4.14. *We define the measure:*

$$\widetilde{\mathrm{m}}(t) := (\mathrm{m}(t), \deg_\mu(t)) \in !\mathbb{N} \times \mathbb{N}$$

where the couple is ordered by the lexicographic order, which is well-founded.

Corollary 4.4.15 (Strong normalisation)**.** *If* $t \to_{\mathrm{r}} t' + \mathbb{T}$ *then* $\widetilde{\mathrm{m}}(t) > \widetilde{\mathrm{m}}(t')$*. Therefore, the resource reduction* $\to_{\mathrm{r}}$ *on sums is strongly normalising.*

Proof. The only case in which $\mathrm{m}(\cdot)$ may remain constant is along a ρ^{r}-reduction, but in this case $\deg_\mu(t)$ strictly decreases, so $t \to_{\mathrm{r}} t' + \mathbb{T}$ entails $\widetilde{\mathrm{m}}(t) > \widetilde{\mathrm{m}}(t')$. Now one concludes as usual: if we associate the multiset $[\widetilde{\mathrm{m}}(t) \mid t \in \mathbb{T}]$ with any sum $\mathbb{T}$ (in particular, a single element sum), it is immediate to see that if $\mathbb{T} \to_{\mathrm{r}} \mathbb{S}$, then the multiset associated with $\mathbb{T}$ is strictly smaller, in the multiset order, than the one associated with $\mathbb{S}$ (the empty multiset 1 being associated with the empty sum 0). Therefore the well-foundeness of the multiset order (since the order on $\widetilde{\mathrm{m}}(\cdot)$ is well-founded) gives the non-existence of infinite (non-trivial) reductions. □

Let us see some properties of the measure $\mathrm{m}(\cdot)$.

Lemma 4.4.16. *Let* $c = c\langle\!|\xi|\!\rangle$ *be a single-hole context and* t *a* $\lambda\mu$*-term. Then:* $\mathrm{m}(c\langle\!|t|\!\rangle) \geq \mathrm{m}(t)$.

Proof. One has:

$$\mathrm{m}(c\langle\!|t|\!\rangle) = A * [\,\deg_\mu(c\langle\!|t|\!\rangle) - d_{c\langle\!|t|\!\rangle}(b) \mid b \text{ in } t\,]$$

where $A := [\,\deg_\mu(c\langle\!|t|\!\rangle) - d_{c\langle\!|t|\!\rangle}(b) \mid b \text{ in } c\,]$. But $\deg_\mu(c\langle\!|t|\!\rangle) = \deg_\mu(c) + \deg_\mu(t)$ and, for all occurrence b in t, we have: $d_{c\langle\!|t|\!\rangle}(b) = d_t(b) + d_c(\xi) \leq d_t(b) + \deg_\mu(c)$. Thus, for all occurrence b of bag in t, we have: $\deg_\mu(c\langle\!|t|\!\rangle) - d_{c\langle\!|t|\!\rangle}(b) \geq \deg_\mu(t) - d_t(b)$ and this last integer is exactly a generic element of $\mathrm{m}(t)$ (if it is non-empty). Hence $\mathrm{m}(c\langle\!|t|\!\rangle) \geq A * \mathrm{m}(t) \geq \mathrm{m}(t)$. □

However, there are cases in which $\mathrm{m}(c\langle\!|t|\!\rangle) = \mathrm{m}(t)$ even if $c \neq \xi$. For example, taking $c = \lambda x.\xi$ one has $\mathrm{m}(c\langle\!|t|\!\rangle) = 1 = \mathrm{m}(t)$ for all $t \in \lambda\mu^{\mathrm{r}}$ *not* containing any bags. This is exactly why, in the following, we will consider a slightly different size, called **ms** (defined in Corollary 4.4.19).

Lemma 4.4.17. *Let* $c = c\langle\!|\xi|\!\rangle$ *be a single-hole resource context and* $t \in \lambda\mu$*. Then:*

$$\mathrm{m}(c\langle\!|t|\!\rangle) = (\deg_\mu(t) + \mathrm{m}(c)) * ((\deg_\mu(c) - d_c(\xi) + \mathrm{m}(t)).$$

Proof. Easily checked, thanks to the clear fact that if b is the occurrence of a bag in c, then $d_{c\langle\!|t|\!\rangle}(b) = d_c(b)$. □

An immediate consequence of the previous lemma is the following:

Lemma 4.4.18. *Let* c *be a single-hole context and* t, s $\lambda\mu$*-terms s.t.* $\deg_\mu(s) \leq \deg_\mu(t)$*. If* $\mathrm{m}(s) < \mathrm{m}(t)$ *then* $\mathrm{m}(c\langle\!|s|\!\rangle) < \mathrm{m}(c\langle\!|t|\!\rangle)$.

In the following, we will need a strong normalising measure which, in addition, satisfies the properties of the following Corollary 4.4.19. However, the previous Lemma 4.4.16 shows that $\widetilde{\mathrm{m}}(\cdot)$ is not adapted for that. This is why we operate a last slight modification.

First, let us consider the size $\mathbf{sz}(t) \in \mathbb{N}_{\geq 1}$ of resource $\lambda\mu$-terms. We mean the analogous of the size of λ-terms, that is, the same of Remark 3.3.3 with the additional clause: $\mathbf{sz}(\mu\alpha._\beta|s|) := 1 + \mathbf{sz}(s)$. Of course $\mathbf{sz}(t) = 1$ iff t is a variable, and for all c single-hole context, $\mathbf{sz}(c\langle\!|t|\!\rangle) \geq \mathbf{sz}(t)$ where the equality holds iff $c = \xi$. Now we have:

Corollary 4.4.19. *Define a measure* $\boldsymbol{ms}(\cdot)$ *of* $\lambda\mu$*-terms as:*

$$\boldsymbol{ms}(t) := (\widetilde{\mathrm{m}}(t), \mathbf{sz}(t)) = (\mathrm{m}(t), \deg_\mu(t), \mathbf{sz}(t)) \in !\,\mathbb{N} \times \mathbb{N} \times \mathbb{N}$$

ordered lexicographically (and thus well-founded). Then:

1. *t is a variable iff $\boldsymbol{ms}(t)$ takes its minimal value (which is $(1, 0, 1)$).*
2. *For all single-hole context $c = c\langle\!|\xi|\!\rangle$, we have $\boldsymbol{ms}(c\langle\!|t|\!\rangle) \geq \boldsymbol{ms}(t)$, and the equality holds iff $c = \xi$.*

3. If $t \to_{\mathrm{r}} t' + \mathbb{T}$ *then* $\boldsymbol{ms}(t) > \boldsymbol{ms}(t')$.

Proof. Since $\widetilde{\mathrm{m}}(t) = (\mathrm{m}(t), \deg_\mu(t))$ is ordered lexicographically, Lemma 4.4.16 and Lemma 4.4.18 immediately hold for $\widetilde{\mathrm{m}}(t)$. The three points are now immediate to check. □

This measure satisfies thus the wanted properties, and we will use it when needed.

Confluence Due to the presence of three different reductions, the confluence or our resource λ-calculus is not easy. Another difficulty is raised from the fact that we placed ourselves in a qualitative setting, that is, with idempotent sums, so that we cannot always reduce a sum component-wise. This is why we split the problem of the confluence in two steps: first, we show that the *quantitative* resource λμ-calculus (that is, where sum is *not* idempotent, and thus coefficients matter) is confluent; second, we show that its confluence implies the confluence of the calculus with no coefficients. Actually, since our calculus is strongly normalising, we will be able to just consider the local confluence.

Before attacking the first step, let us give the necessary definitions and make some remarks.

Definition 4.4.20. *The quantitative resource λμ-calculus is (with the notations of Definition 3.2.2)* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$. *That is, the only difference with usual sums (*$2\langle\lambda\mu^{\mathrm{r}}\rangle$*) is that now "+" is non-idempotent*[12].

We define the three base-case reductions $\to^+_{\lambda^{\mathrm{r}}}, \to^+_{\mu^{\mathrm{r}}}, \to^+_{\rho^{\mathrm{r}}}$ *in* $\lambda\mu^{\mathrm{r}} \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$. *The reduction* $\to^+_{\rho^{\mathrm{r}}}$ *is defined exactly as in Definition 4.4.6, while* $\to^+_{\lambda^{\mathrm{r}}}$ *and* $\to^+_{\mu^{\mathrm{r}}}$ *are defined as in Definition 4.4.6, except for the fact that the linear substitution and linear named application are replaced with a modified version of them, which we denote respectively* $t\langle[\vec{u}]/x\rangle^+$ *and* $\langle t\rangle^+_\alpha[\vec{u}]$. *They are defined in the next Definition 4.4.23.*

The contextual union of the base-reductions $\to^+_{\lambda^{\mathrm{r}}}, \to^+_{\mu^{\mathrm{r}}}, \to^+_{\rho^{\mathrm{r}}}$ *forms a reduction* $\to^+_{\mathrm{r}}$ *on* $\lambda\mu^{\mathrm{r}} \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ *which is extended to all* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ *simply by setting* $t + \mathbb{S} \to^+_{\mathrm{r}} \mathbb{T} + \mathbb{S}$ *whenever* $t \to^+_{\mathrm{r}} \mathbb{T}$ *(that is, we dropped the condition* $t \notin \mathbb{S}$ *since now coefficients matter).*

For the following Definition 4.4.23, we need the following terminology:

Terminology 4.4.21. *Fix a function*[13] $W : \{1, \dots, k\} \longrightarrow \{0, \dots, n\}$ *and a bag* $[u_1, \dots, u_k]$. *Remember that the writing* $[u_1, \dots, u_k]$ *means that we have fixed an enumeration* $u_1, \dots, u_k$ *of the* k *elements (counted with their multiplicity) of the bag. Now, with respect to this enumeration, we say that the w.c.*

$$([u_j \mid j \in W^{-1}(0)], \dots, [u_j \mid j \in W^{-1}(n)])$$

of $[u_1, \dots, u_k]$ *is* the w.c. of $[u_1, \dots, u_k]$ generated by W. *Remark that the cardinality of the bag* $[u_j \mid j \in W^{-1}(i)]$ *is* $W^{-1}(i)$.

In order to lighten the notations we will often employ the following notation: when we use $W : \{1, \dots, k\} \longrightarrow \{0, \dots, n\}$ *in order to generate a w.c. of* $[u_1, \dots, u_k]$ *(w.r.t. the written enumeration), then we denote* W *by* $W : (u_1, \dots, u_k) \longrightarrow \{0, \dots, n\}$, *by* $W : (\vec{u}) \longrightarrow \{0, \dots, n\}$. *when we do not precise the elements of* $[\vec{u}]$. *In this case, the above defined w.c. of* $[u_1, \dots, u_k]$ *generated by* W, *is denoted by* $([\vec{w}^{\,0}], \dots, [\vec{w}^{\,n}])$. *That is,* $[\vec{w}^{\,i}]$ *stands for the bag* $[u_j \mid j \in W^{-1}(i)]$. *We will also write, as usual,* $[\vec{w}^{\,i}] =: [w^i_1, \dots, w^i_{h_i}]$ *(for some* $h_i \in \mathbb{N}$*).*

In the case $[\vec{u}] = 1$, *we write* $W : () \longrightarrow \{0, \dots, n\}$ *and we say that there is exactly one w.c. generated by* W, *which is* $(1, \dots, 1)$.

[12] Analogously, one can see a sum as a finite multiset of resource terms, where the multiplicity of a term is given by its coefficient in the sum.

[13] Attention: in the rest of the thesis, W usually denotes a w.c. of some bag. Since in this confluence section the sums over the w.c.'s are replaced with sums over such functions, then we use the same letter, but we will always precise that it is a function and not a w.c.

Example 4.4.22. *Using the notations and terminologies above, we have:*

1. *The function* $W : (x, y) \longrightarrow \{0, 1\}$ *defined by* $W(1) = 1, W(2) = 0$*, generates the w.c.* $([\vec{w}^0], [\vec{w}^1])$ *of* $[x, y]$ *(with the enumeration from left to right) given by:* $[\vec{w}^0] = [y]$, $[\vec{w}^1] = [x]$.

2. *The function* $D : (y, y) \longrightarrow \{0, 1\}$ *defined by* $D(1) = 0, D(2) = 1$*, generates the w.c.* $([\vec{d}^0], [\vec{d}^1])$ *of* $[y, y]$ *(with the enumeration from left to right) given by:* $[\vec{d}^0] = [\vec{d}^1] = [y]$.

3. *The same* identical *w.c. of* $[y, y]$ *(with the enumeration from left to right) as the one above (that is,* $([y], [y])$*) is also the one generated by the* different *function* $P : \{1, 2\} \longrightarrow \{0, 1\}$ *defined by* $P(1) = 1, P(2) = 0$.

Definition 4.4.23 (Quantitative linear substitution and quantitative linear named application)**.** *The quantitative version* $t\langle[\vec{u}]/x\rangle^+$ *of the linear substitution is defined exactly as in Figure 4.1 but by replacing, in Figure 3.1, the sum on all the* $([\vec{w}^0], \ldots, [\vec{w}^n])$ *w.c. of* $[\vec{u}]$ *with the sum on all* $W : (\vec{u}) \longrightarrow \{0, \ldots, n\}$*, and by taking the above w.c.'s as the one given by* W *in the previously explained sense.*

The quantitative version $\langle t\rangle^+_\alpha[\vec{u}]$ *of the linear named application is defined exactly as in Figure 4.2 but by replacing, in the case of an application, the sum on all the* $([\vec{w}^0], \ldots, [\vec{w}^n])$ *w.c. of* $[\vec{u}]$ *with the sum on all* $W : (\vec{u}) \longrightarrow \{0, \ldots, n\}$*, and by taking the above w.c.'s as the one given by* W *in the previously explained sense. Analogously, in the case of a* μ*-abstraction, we replace the sum on all the* $([\vec{w}^1], [\vec{w}^2])$ *w.c. of* $[\vec{u}]$ *with the sum on all* $W : (\vec{u}) \longrightarrow \{0, 1\}$*, and by taking the above w.c.'s as the one given by* W *in the previously explained sense.*

Example 4.4.24. *For instance, one has:*

$$(\mu\alpha_{\cdot\alpha}|\mu\eta_{\cdot\alpha}|x||)[y, y] \to^+_{\mathrm{r}} \mu\alpha_{\cdot\alpha}|(\mu\eta_{\cdot\alpha}|x1|)[y, y]| + 2\,\mu\alpha_{\cdot\alpha}|(\mu\eta_{\cdot\alpha}|x[y]|)[y]| + \mu\alpha_{\cdot\alpha}|(\mu\eta_{\cdot\alpha}|x[y, y]|)1|.$$

The first term of the sum comes from the w.c. $(1, [yy])$ *of* $[y, y]$ *given by the function* $W : (y, y) \longrightarrow \{0, 1\}$ *defined by* $W(1) = W(2) = 1$*; the last term of the sum comes from the w.c.* $([yy], 1)$ *of* $[y, y]$ *given by the function* $W : (y, y) \longrightarrow \{0, 1\}$ *defined by* $W(1) = W(2) = 0$*; the second term of the sum, with coefficient* 2*, comes from the two* identical *w.c.'s* $([y], [y])$ *of* $[y, y]$ *given by the two* different *functions* $W : (y, y) \longrightarrow \{0, 1\}$ *defined by* $W(1) = 0, W(2) = 1$ *and* $W : (y, y) \longrightarrow \{0, 1\}$ *defined by* $W(1) = 1, W(2) = 0$.

Remark 4.4.25. *It is clear by the definitions that if, for* $t \in \lambda\mu^{\mathrm{r}}$*, one has* $t \to_{\lambda\mu^{\mathrm{r}}} \mathbb{T}$ *(in* $2\langle\lambda\mu^{\mathrm{r}}\rangle$*) and* $t \to^+_{\mathrm{r}} \mathbb{S}$ *(in* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$*)* by reducing the same redex*, then* $\mathrm{supp}(\mathbb{S}) = \mathbb{T}$*. That is, the two reductions only differ for the coefficients.*

Remark 4.4.26. *Since we already know that the reduction* $\to_{\mathrm{r}}$ *is strongly normaling in* $\lambda\mu^{\mathrm{r}}$ *(Corollary 4.4.15), then the reduction* $\to^+_{\mathrm{r}}$ *is strongly normaling in* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$*. Indeed, it sufficies to extend the strongly normalising measure* $\widetilde{\mathrm{m}}(\cdot)$ *of* $\lambda\mu^{\mathrm{r}}$ *(Definition 4.4.14) to* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ *by setting* $\widetilde{\mathrm{m}}(\mathbb{T}) := [\widetilde{\mathrm{m}}(t) \mid t \in \mathbb{T}] \in !(!\mathbb{N} \times \mathbb{N})$*, and use the multiset order.*

Remark 4.4.27 (Embedding inside the differential $\lambda\mu$-calculus)**.** *One could think of proving the confluence of* $\to^+_{\mathrm{r}}$ *by using the fact that in [Vau07a], Vaux proves the confluence of his differential* $\lambda\mu$*-calculus. Let us call it* $(\lambda\mu^\partial, \to_\partial)$ *in this remark. In fact, our resource* $\lambda\mu$*-calculus* $2\langle\lambda\mu^{\mathrm{r}}\rangle$*, as well as the quantitative version* $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$*, translates into* $\lambda\mu^\partial$ *via the translation* $(\cdot)^\partial : \lambda\mu^{\mathrm{r}} \longrightarrow \lambda\mu^\partial$ *defined as*[14]*:*

$$x^\partial := x \qquad \lambda x.t^\partial := \lambda x.t^\partial \qquad \mu\alpha_{\cdot\beta}|t|^\partial := \mu\alpha_{\cdot\beta}|t^\partial|$$

[14]Remark that the case $(t[u_1, \ldots, u_k])^\partial$ is well-defined since a $\lambda\mu^\partial$-term of shape $\mathtt{D}^k\,\Theta' \bullet (\Theta_1, \ldots, \Theta_k)$ does not depend on the enumeration of the Θ_i's (Corollary 6.17 of [Vau07a]).

$$(t[u_1,\dots,u_k])^\partial := \Big(\not{D}^k\, t^\partial \bullet (u_1^\partial,\dots,u_k^\partial)\Big)\, 0.$$

We can extend it to sums, both in $2\langle\lambda\mu^{\mathrm{r}}\rangle$ and in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$, by linearity. In the qualitative case (that is, if we consider $(\cdot)^\partial : 2\langle\lambda\mu^{\mathrm{r}}\rangle \longrightarrow \lambda\mu^\partial$), it is not a well-behaved embedding, because it does not preserve reductions. On the contrary, it does in the quantitative case (that is, if we consider $(\cdot)^\partial : \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle \longrightarrow \lambda\mu^\partial$), as one can prove that: if $t \to^+_{\lambda\mu^{\mathrm{r}}} \mathbb{T}$ in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$, then $t^\partial \twoheadrightarrow_\partial \mathbb{T}^\partial$ in $\lambda\mu^\partial$. One can now entail the local confluence of $\to^+_{\lambda\mu^{\mathrm{r}}}$ from the confluence of $\to_\partial$. And in turn, as we are going to see (using the reduction in Definition 4.4.44), this entails that $\to_{\lambda\mu^{\mathrm{r}}}$ is locally confluent as well.

However, as the reader has probably noticed, we only talked about $\to_{\lambda\mu^{\mathrm{r}}}$, and not about the whole $\to_{\mathrm{r}} = \to_{\lambda\mu^{\mathrm{r}}} \cup \to_{\rho^{\mathrm{r}}}$. This is simply because in [Vau07a] the ρ-reduction is not considered, thus the reduction $\to_\partial$ for which the confluence is there proven only contains the cases for $\to_\lambda$ and $\to_\mu$, not for $\to_\rho$.

Remark that, even if it is possible to prove the confluence of $\to_{\rho^{\mathrm{r}}}$ by itself, we cannot use this result in order to entail the confluence of $\to_{\mathrm{r}} = \to_{\lambda\mu^{\mathrm{r}}} \cup \to_{\rho^{\mathrm{r}}}$ by invoking the well-known Hindley-Rosen lemma[15]*. This is because $\to^+_{\rho^{\mathrm{r}}}$ and $\to^+_{\lambda\mu^{\mathrm{r}}}$ do* not *commute, as the following example shows*[16] *(where $\gamma \neq \eta \neq \alpha$):*

$$\begin{array}{ccc} (\mu\alpha.{}_\alpha|\mu\gamma.{}_\eta|x||)1 & \to_{\rho^{\mathrm{r}}} & (\mu\alpha.{}_\eta|x|)1 \\ \downarrow_{\mu^{\mathrm{r}}} & & \downarrow_{\mu^{\mathrm{r}}} \\ \mu\alpha.{}_\alpha|(\mu\gamma.{}_\eta|x|)1| & \not\to_{\rho^{\mathrm{r}}} & \mu\alpha.{}_\eta|x|. \end{array}$$

We can attack the first of the two steps mentioned some pages above: we will prove the local confluence of $\to^+_{\mathrm{r}}$. We present here a proof which essentially consists in considering all the possible critical pairs and showing that one can always close the respective diagram[17].

Remark 4.4.28. *We can extend the definition of linear substitution and linear named application to sums by linearity:*

$$\mu\alpha.\,\langle\mathbb{T}\rangle^+_\alpha\,[\mathbb{U}_1,\dots,\mathbb{U}_k] := \sum_{t\in\mathbb{T}}\sum_{u_1\in\mathbb{U}_1}\cdots\sum_{u_k\in\mathbb{U}_k} \mu\alpha.\,\langle t\rangle^+_\alpha\,[u_1,\dots,u_k]$$

and the same for the linear substitution. Analogously, the renaming of a sum $\mathbb{T}\{\alpha/\beta\}$ is defined component-wise. With these definitions in place one checks that base-step-reduction lifts to sums, i.e.:

$$\begin{array}{c} (\mu\alpha.{}_\beta|\mathbb{T}|)\,[\vec{\mathbb{U}}] \twoheadrightarrow^+_{\mu^{\mathrm{r}}} \mu\alpha.\,\langle\mathbb{T}\rangle^+_\alpha\,[\vec{\mathbb{U}}], \\ (\lambda x.\mathbb{T})\,[\vec{\mathbb{U}}] \twoheadrightarrow^+_{\lambda^{\mathrm{r}}} \mathbb{T}\langle[\vec{\mathbb{U}}]\,/\,x\rangle^+, \qquad \mu\alpha.{}_\beta|\mu\gamma.{}_\eta|\mathbb{T}|| \twoheadrightarrow^+_{\rho^{\mathrm{r}}} \mu\alpha.{}_\eta|\mathbb{T}|\{\beta/\gamma\}. \end{array}$$

One can check that $\to^+_{\mathrm{r}}$ on $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ is contextual.

We need a number of technical lemmas handling the interaction of two successive substitutions. The proofs of these lemmas are tedious and long inductions.

[15] Whose statement is that if $\to_1$, $\to_2$ are two confluent binary relations on X s.t. $\twoheadrightarrow_1$, $\twoheadrightarrow_2$ commute, then $\to_1 \cup \to_2$ is confluent.

[16] The same of course holds also in the quantitative case.

[17] Another possibility would be to consider a "parallel reduction", in the style of the well known Tait&Martin-Löf technique. We are currently working on such a proof as well.

Notation 4.4.29. *In this section we will sometimes use the following notation: for α, β, η names, we set $\delta^{\alpha}_{\eta}(\beta)$ to be*[18] *α if $\beta = \eta$, or η otherwise. Remark that with this notation the ρ-reduction takes the form $\mu\alpha.{}_{\beta}|\mu\gamma.{}_{\eta}|t|| \to^{+}_{\rho^{\mathrm{r}}} \mu\alpha.{}_{\delta^{\beta}_{\eta}(\gamma)}|t\{\beta/\gamma\}|$.*

Lemma 4.4.30. *Let $t \in \lambda\mu^{\mathrm{r}}$ and $\alpha, \beta, \gamma, \eta$ names.*
If $\alpha \neq \eta$, $\beta \neq \eta$ and $\beta \neq \gamma$, then $t\{\alpha/\beta\}\{\gamma/\eta\} = t\{\gamma/\eta\}\{\alpha/\beta\}$.

The following lemma says how a renaming behaves with respect to the linear substitution and the linear named application.

Lemma 4.4.31. *Let $t \in \lambda\mu^{\mathrm{r}}$ and $[\vec{u}]$ a bag. Then:*

1. $t\langle[\vec{u}]/x\rangle^{+}\{\alpha/\beta\} = t\{\alpha/\beta\}\langle[\vec{u}\{\alpha/\beta\}]/x\rangle^{+}$.
2. *If $\alpha \neq \gamma \neq \beta$ then:* $(\,\langle t\rangle^{+}_{\gamma}[\vec{u}]\,)\{\alpha/\beta\} = \langle t\{\alpha/\beta\}\rangle^{+}_{\gamma}[\vec{u}\{\alpha/\beta\}]$.
3. *If $\alpha \neq \gamma \neq \beta$ then:* $(\,\langle {}_{\eta}|t|\rangle^{+}_{\gamma}[\vec{u}]\,)\{\alpha/\beta\} = \langle {}_{\eta}|t|\{\alpha/\beta\}\rangle^{+}_{\gamma}[\vec{u}\{\alpha/\beta\}]$.

The next lemma says how two linear substitutions operate when applied consecutively.

Lemma 4.4.32. . *Let $t \in \lambda\mu^{\mathrm{r}}$, $[\vec{v}] =: [v_1, \ldots, v_n]$ and $[\vec{u}]$ bags and $y \neq x$ variables with y* not *occurring in $[\vec{u}]$. Then:*

$$t\langle[\vec{v}]/y\rangle\langle[\vec{u}]/x\rangle = \sum_{W} t\langle[\vec{w}^{\,0}]/x\rangle\langle[v_1\langle[\vec{w}^{\,1}]/x\rangle, \ldots, v_n\langle[\vec{w}^{\,n}]/x\rangle]/y\rangle$$

with $W : (\vec{u}) \longrightarrow \{0, \ldots, n\}$.

The condition $x \neq y$ in the previous lemma cannot be suppressed: for instance for $x = y$, $t = x$, $[\vec{v}] = 1$ and $[\vec{u}] = [z]$, with $z \neq y$, the left hand-side of the equality becomes $x\langle 1/x\rangle^{+}\langle[z]/x\rangle^{+} = 0$ while the right hand-side becomes $x\langle[z]/x\rangle^{+}\langle 1/x\rangle^{+} = z$.
The next lemma says how a linear substitution operates on linear named application.

Lemma 4.4.33. . *Let $t \in \lambda\mu^{\mathrm{r}}$, $[\vec{v}] =: [v_1, \ldots, v_n]$ and $[\vec{u}]$ bags and α a name with*[19] $\deg_{\alpha}([\vec{u}]) = 0$. *Then:*

$$(\,\langle t\rangle^{+}_{\alpha}[\vec{v}]\,)\,\langle[\vec{u}]/x\rangle = \sum_{W} \langle t\langle[\vec{w}^{\,0}]/x\rangle\rangle^{+}_{\alpha}[v_1\langle[\vec{w}^{\,1}]/x\rangle, \ldots, v_n\langle[\vec{w}^{\,n}]/x\rangle]$$

with $W : (\vec{u}) \longrightarrow \{0, \ldots, n\}$.

The following two remarks are easy:

Remark 4.4.34. *Let α be a name, $t \in \lambda\mu^{\mathrm{r}}$ and $[\vec{u}], [\vec{v}]$ bags. If $\deg_{\alpha}([\vec{v}]) = 0$ then one has:*

$$\langle t[\vec{v}]\rangle^{+}_{\alpha}[\vec{u}] = (\langle t\rangle^{+}_{\alpha}[\vec{u}]\,)\,[\vec{v}].$$

Remark 4.4.35. *Let $\alpha \neq \beta$ be names, $t \in \lambda\mu^{\mathrm{r}}$ and $[\vec{v}] =: [v_1, \ldots, v_n], [\vec{u}]$ bags. If $\deg_{\beta}(t) = 0$ then one has:*

$$\langle\langle t\rangle^{+}_{\alpha}[\vec{v}]\rangle^{+}_{\beta}[\vec{u}] = \sum_{W} \langle t\rangle^{+}_{\alpha}[\langle v_1\rangle^{+}_{\beta}[\vec{w}^{\,1}], \ldots, \langle v_n\rangle^{+}_{\beta}[\vec{w}^{\,n}]].$$

with $W : (\vec{u}) \longrightarrow \{1, \ldots, n\}$.

The next lemma says in which sense and when, in some cases, one can swap the order of two linear applications.

[18] Note that we may use the letter "δ" for a name as well: in this notation – which is however used only in some proofs – it is of course not a name, and there should be no ambiguity.
[19] By "$\deg_{\beta}([\vec{u}]) = 0$" we mean that $\deg_{\beta}(u) = 0$ for all $u \in [\vec{u}]$.

Lemma 4.4.36. *Let $t \in \lambda\mu^{\mathrm{r}}$, $[\vec{u}], [\vec{v}]$ bags and $\alpha \neq \beta$ names.*

1. *If $\deg_\alpha([\vec{v}]) = 0$ and $\deg_\beta([\vec{u}]) = 0$, then:*
$$\langle\langle t\rangle^+_\alpha[\vec{u}]\rangle^+_\beta[\vec{v}] = \langle\langle t\rangle^+_\beta[\vec{v}]\rangle^+_\alpha[\vec{u}].$$
2. *If $\deg_\alpha([\vec{v}]) = 0$ then, taking δ a fresh name, one has:*
$$\langle\langle t\rangle^+_\alpha[\vec{u}]\rangle^+_\beta[\vec{v}] = \sum_W \langle\langle\langle t\rangle^+_\beta[\vec{w}^{\,1}]\rangle^+_\alpha[\vec{u}\{\delta/\beta\}]\rangle^+_\delta[\vec{w}^{\,2}]\ \{\beta/\delta\}$$
with $W : (\vec{v}) \longrightarrow \{1, 2\}$.
3. *If $\deg_\beta([\vec{u}]) = 0$ then, taking δ a fresh name, one has:*
$$\langle\langle t\rangle^+_\alpha[\vec{u}]\rangle^+_\beta[\vec{v}] = \langle\langle t\rangle^+_\beta[\vec{v}\{\delta/\alpha\}]\rangle^+_\alpha[\vec{u}]\ \{\alpha/\delta\}.$$

The next lemma says how a linear named applications on a name operates on a renaming involving the same name.

Lemma 4.4.37. *Let $t \in \lambda\mu^{\mathrm{r}}$, $[\vec{u}]$ a bag and $\alpha \neq \beta$ names with $\deg_\beta([\vec{u}]) = 0$. Then:*
$$\langle t\{\alpha/\beta\}\rangle^+_\alpha[\vec{u}] = \sum_W \langle\langle t\rangle^+_\alpha[\vec{w}^{\,1}]\rangle^+_\beta[\vec{w}^{\,2}]\,\{\alpha/\beta\}$$
with $W : (\vec{u}) \longrightarrow \{1, 2\}$.

The condition $\alpha \neq \beta$ in the previous lemma cannot be suppressed: for instance, for $t = \mu\gamma.{}_\alpha|x|$ and $[\vec{u}] = 1$, the left hand-side of the equality becomes $\mu\gamma.{}_\alpha|x1|$ while the right hand-side becomes:
$$\sum_{W:()\longrightarrow\{1,2\}} \mu\gamma.{}_\alpha|x\,[\vec{w}^{\,1}]\,[\vec{w}^{\,2}]| = \mu\gamma.{}_\alpha|x\,1\,1|.$$
Also the condition $\deg_\beta([\vec{u}]) = 0$ cannot be suppressed: for instance, for $t = \mu\gamma.{}_\alpha|x|$ and $[\vec{u}] = [\mu\gamma'.{}_\beta|y|]$, the left hand-side becomes $\mu\gamma.{}_\alpha|x[\mu\gamma'.{}_\beta|y|]|$ while the right hand-side becomes $\mu\gamma.{}_\alpha|x[\mu\gamma'.{}_\beta|y1|]|$.

The following lemma says how a linear named application operates on a linear substitution.

Lemma 4.4.38. *Let $t \in \lambda\mu^{\mathrm{r}}$, $[\vec{v}] =: [v_1, \ldots, v_n], [\vec{u}]$ bags, x a variable and α a name s.t. $\deg_x([\vec{u}]) = 0$.*
$$\langle t\langle[\vec{v}]/x\rangle\rangle^+_\alpha[\vec{u}] = \sum_W (\langle t\rangle^+_\alpha[\vec{w}^{\,0}])\langle[\langle v_1\rangle^+_\alpha[\vec{w}^{\,1}], \ldots, \langle v_n\rangle^+_\alpha[\vec{w}^{\,n}]]/x\rangle$$
with $W : (\vec{u}) \longrightarrow \{0, \ldots, n\}$.

The condition $\deg_x([\vec{u}]) = 0$ in the previous lemma cannot be suppressed: for instance, for $t = \mu\gamma.{}_\alpha|y|$ (with $y \neq x$), $[\vec{v}] = 1$ and $[\vec{u}] = [x]$, the left hand-side becomes $\mu\gamma.{}_\alpha|y[x]|$ while the right hand-side becomes 0.

The next lemma says how two linear named application operate consecutively.

Lemma 4.4.39. *Let $t \in \lambda\mu^{\mathrm{r}}$, $\alpha \neq \gamma$ names and $[\vec{v}] =: [v_1, \ldots, v_n], [\vec{u}]$ bags s.t. $\deg_\gamma([\vec{u}]) = 0$. We have:*
$$\langle\langle t\rangle^+_\gamma[\vec{v}]\rangle^+_\alpha[\vec{u}] = \sum_W \langle\langle t\rangle^+_\alpha[\vec{w}^{\,0}]\rangle^+_\gamma[\langle v_1\rangle^+_\alpha[\vec{w}^{\,1}], \ldots, \langle v_n\rangle^+_\alpha[\vec{w}^{\,n}]]$$

with $W : (\vec{u}) \longrightarrow \{0, \dots, n\}$.

Furthermore, the same holds also for named terms. That is, under the same hypothesis and for η *a name, we have*[20]*:*

$$\langle\langle_\eta|t|\rangle^+_\gamma[\vec{v}]\rangle^+_\alpha[\vec{u}] = \sum_W \langle\langle_\eta|t|\rangle^+_\alpha[\vec{w}^{\,0}]\rangle^+_\gamma[\langle v_1\rangle^+_\alpha[\vec{w}^{\,1}], \dots, \langle v_n\rangle^+_\alpha[\vec{w}^{\,n}]]$$

with $W : (\vec{u}) \longrightarrow \{0, \dots, n\}$.

The condition $\alpha \neq \gamma$ in the previous lemma cannot be suppressed: for instance, for $t = \mu\delta._\alpha|x|$, $[\vec{u}] = [\mu\delta._\delta|x|]$ and $[\vec{v}] = [\mu\delta._\delta|y|]$, the left hand-side becomes $\mu\delta._\alpha|x[\mu\delta._\delta|y|][\mu\delta._\delta|x|]|$ while the right hand-side becomes $\mu\delta._\alpha|x[\mu\delta._\delta|x|][\mu\delta._\delta|y|]|$.

Also the condition $\deg_\gamma([\vec{u}]) = 0$ in the previous lemma cannot be suppressed: for instance, for $t = \mu\delta._\alpha|x|$, $[\vec{u}] = [\mu\delta._\gamma|x|]$ and $[\vec{v}] = [\mu\delta._\delta|x|]$, the left hand-side becomes 0 while the right hand-side becomes $\mu\delta._\alpha|x[\mu\delta._\gamma|x[\mu\delta._\delta|x|]|]|$.

With these technical lemmas in place, we can now can prove the crucial Lemma 4.4.41 for the confluence of the quantitative resource calculus. To say it in other words, it states that reduction on sums (see Remark 4.4.28) remains contextual also if we include $(\cdot)\langle[\cdot, \dots, \cdot]/x\rangle$ and $\langle\cdot\rangle^+_\alpha[\cdot, \dots, \cdot]$ as (multi-hole-)contexts.

In the proof of Lemma 4.4.41 we also use the following remark.

Remark 4.4.40. *Let* $[p, \vec{q}]$ *be a bag. Then, every function* $W' : \{p, u_1, \dots, u_k\} \longrightarrow \{0, \dots, n\}$ *is uniquely determined by the choice of* $W'(p) \in \{0, \dots, n\}$ *plus the choice of a function* $W : \{u_1, \dots, u_k\} \longrightarrow \{0, \dots, n\}$*. This is reflected in the equality* $(n+1)^{k+1} = (n+1)^k \cdot (n+1)$*. Now, for* $0 \leq i, j \leq n$*, let us set* $[p]^j_i$ *the singleton multiset* $[p]$ *if* $i = j$*, and the empty mulitset* 1 *if* $i \neq j$*. Therefore, the w.c. of* $[p, \vec{q}]$ *generated by a* $W' : \{p, u_1, \dots, u_k\} \longrightarrow \{0, \dots, n\}$ *is of shape* $([\vec{w}^{\,0}] * [p]^{W'(p)}_0, \dots, [\vec{w}^{\,n}] * [p]^{W'(p)}_n)$*, for* $([\vec{w}^{\,0}], \dots, [\vec{w}^{\,n}])$ *a w.c. of* $[\vec{q}]$ *generated by a* $W : \{u_1, \dots, u_k\} \longrightarrow \{0, \dots, n\}$*.*

Lemma 4.4.41. *Let* $t, s \in \lambda\mu^{\mathrm{r}}$*,* x *a variable,* α, β *names and* $[\vec{u}]$ *a bag. If* $s \to^+_{\mathrm{r}} \mathbb{S}$ *then:*

1. $s\{\alpha/\beta\} \twoheadrightarrow^+_{\mathrm{r}} \mathbb{S}\{\alpha/\beta\}$
2. $t\langle[s, \vec{u}]/x\rangle^+ \twoheadrightarrow^+_{\mathrm{r}} t\langle[\mathbb{S}, \vec{u}]/x\rangle^+$
3. $s\langle[\vec{u}]/x\rangle^+ \twoheadrightarrow^+_{\mathrm{r}} \mathbb{S}\langle[\vec{u}]/x\rangle^+$
4. $\langle t\rangle^+_\alpha[s, \vec{u}] \twoheadrightarrow^+_{\mathrm{r}} \langle t\rangle^+_\alpha[\mathbb{S}, \vec{u}]$
5. $\mu\alpha.\langle_\beta|t|\rangle^+_\alpha[s, \vec{u}] \twoheadrightarrow^+_{\mathrm{r}} \mu\alpha.\langle_\beta|t|\rangle^+_\alpha[\mathbb{S}, \vec{u}]$
6. $\langle s\rangle^+_\alpha[\vec{u}] \twoheadrightarrow^+_{\mathrm{r}} \langle\mathbb{S}\rangle^+_\alpha[\vec{u}]$.
7. $\mu\alpha.\langle_\beta|s|\rangle^+_\alpha[\vec{u}] \twoheadrightarrow^+_{\mathrm{r}} \mu\alpha.\langle_\beta|\mathbb{S}|\rangle^+_\alpha[\vec{u}]$.

Proof. 1. Induction on s:

Case s variable: impossible.

Case $s = \lambda y.s'$: straightforward by inductive hypothesis.

Case $s = \mu\gamma._\eta|s'|$: we have two subcases[21]:

[20] The following is just an equality between sets of words – the named terms.

[21] In the following we do not explicitly say it, but of course we will take bound names different from α and β, as well as from other bound names.

Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing s':
then $\mathbb{S} = \mu\gamma.\eta|\mathbb{S}'|$ with $s' \to_{\mathrm{r}}^{+} \mathbb{S}'$, and so (regardless of whether η equals γ or not):

$$s\{\alpha/\beta\} = \mu\gamma.{}_{\delta_\eta^\alpha(\beta)}|s'\{\alpha/\beta\}| \to_{\mathrm{r}}^{+} \mu\gamma.{}_{\delta_\eta^\alpha(\beta)}|\mathbb{S}'\{\alpha/\beta\}| = \mu\gamma.\eta|\mathbb{S}'|\{\alpha/\beta\} = \mathbb{S}\{\alpha/\beta\}$$

where we used the inductive hypothesis.

Subcase $s' = \mu\gamma'.{}_{\eta'}|s''|$ and $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing its leftmost ρ-redex:
Then $s = \mu\gamma.\eta|\mu\gamma'.{}_{\eta'}|s''||$, $\mathbb{S} = \mu\gamma.\eta|s''|\{\eta/\gamma'\}$ and we have four sub-subcases:
Sub-subcase $\eta = \beta$ and $\eta' = \beta$: then $\mathbb{S}\{\alpha/\beta\} = \mu\gamma.{}_\beta|s''|\{\alpha/\beta, \alpha/\gamma'\} = \mu\gamma.{}_\alpha|s''\{\alpha/\beta, \alpha/\gamma'\}|$ and:

$$\begin{array}{rcl} s\{\alpha/\beta\} = & \mu\gamma.{}_\alpha|\mu\gamma'.{}_\alpha|s''\{\alpha/\beta\}|| \to_\rho^{+} \mu\gamma.{}_\alpha|s''\{\alpha/\beta\}|\{\alpha/\gamma'\} \\ = & \mu\gamma.{}_\alpha|s''\{\alpha/\beta, \alpha/\gamma'\}| = \mathbb{S}\{\alpha/\beta\}. \end{array}$$

Sub-subcase $\eta = \beta$ and $\eta' \neq \beta$: then

$$\mathbb{S}\{\alpha/\beta\} = \mu\gamma.{}_{\eta'}|s''|\{\alpha/\beta, \alpha/\gamma'\} = \mu\gamma.{}_{\delta_{\eta'}^\alpha(\gamma')}|s''\{\alpha/\beta, \alpha/\gamma'\}|$$

and:

$$\begin{array}{rcl} s\{\alpha/\beta\} = & \mu\gamma.{}_\alpha|\mu\gamma'.{}_{\eta'}|s''\{\alpha/\beta\}|| \to_\rho^{+} \mu\gamma.{}_{\eta'}|s''\{\alpha/\beta\}|\{\alpha/\gamma'\} \\ = & \mu\gamma.{}_{\delta_{\eta'}^\alpha(\gamma')}|s''\{\alpha/\beta, \alpha/\gamma'\}| = \mathbb{S}\{\alpha/\beta\}. \end{array}$$

Sub-subcase $\eta \neq \beta$ and $\eta' = \beta$: then

$$\mathbb{S}\{\alpha/\beta\} = \mu\gamma.{}_\beta|s''|\{\eta/\gamma'\}\{\alpha/\beta\} = \mu\gamma.{}_\alpha|s''\{\eta/\gamma'\}\{\alpha/\beta\}|$$

and:

$$s\{\alpha/\beta\} = \mu\gamma.{}_\eta|\mu\gamma'.{}_\alpha|s''\{\alpha/\beta\}|| \to_\rho^{+} \mu\gamma.{}_\alpha|s''\{\alpha/\beta\}|\{\eta/\gamma'\} = \mu\gamma.{}_\alpha|s''\{\alpha/\beta\}\{\eta/\gamma'\}|$$

and the result follows by Lemma 4.4.30.

Sub-subcase $\eta \neq \beta$ and $\eta' \neq \beta$: then

$$\mathbb{S}\{\alpha/\beta\} = \mu\gamma.{}_{\delta_{\eta'}^\eta(\gamma')}|s''\{\eta/\gamma'\}|\{\alpha/\beta\} = \mu\gamma.{}_{\delta_{\eta'}^\eta(\gamma')}|s''\{\eta/\gamma'\}\{\alpha/\beta\}|$$

and:

$$s\{\alpha/\beta\} = \mu\gamma.{}_\eta|\mu\gamma'.{}_{\eta'}|s''\{\alpha/\beta\}|| \to_\rho^{+} \mu\gamma.{}_{\eta'}|s''\{\alpha/\beta\}|\{\eta/\gamma'\} = \mu\gamma.{}_{\delta_{\eta'}^\eta(\gamma')}|s''\{\alpha/\beta\}\{\eta/\gamma'\}|$$

and the result follows by Lemma 4.4.30.

Case $s = s'[\vec{v}]$: we have four subcases:

Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing the s': straightforward by inductive hypothesis.

Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing one $v_i \in [\vec{v}]$: straightforward by inductive hypothesis.

Subcase $s' = \lambda x.s''$ and $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing the λ-redex s: then $\mathbb{S} = s''\langle[\vec{v}]/x\rangle^{+}$ and:

$$s\{\alpha/\beta\} = (\lambda x.s''\{\alpha/\beta\})[\vec{v}\{\alpha/\beta\}] \to_{\lambda^{\mathrm{r}}} s''\{\alpha/\beta\}\langle[\vec{v}\{\alpha/\beta\}]/x\rangle^{+} = s''\langle[\vec{v}]/x\rangle^{+}\{\alpha/\beta\} = \mathbb{S}\{\alpha/\beta\}$$

where the second-last equality holds thanks to Lemma 4.4.31.

Subcase $s' = \mu\gamma.{}_\eta|s''|$ and $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing the μ-redex s: then $\mathbb{S} = \mu\gamma.\langle{}_\eta|s''|\rangle_\gamma^{+}[\vec{v}]$ and:

$$\begin{array}{rcl} s\{\alpha/\beta\} & = & (\mu\gamma.{}_{\delta_\eta^\alpha(\beta)}|s''\{\alpha/\beta\}|)[\vec{v}\{\alpha/\beta\}] \\ & \to_{\mu^{\mathrm{r}}} & \mu\gamma.\langle{}_{\delta_\eta^\alpha(\beta)}|s''\{\alpha/\beta\}|\rangle_\gamma^{+}[\vec{v}\{\alpha/\beta\}] \\ & = & \mu\gamma.\langle{}_\eta|s''|\{\alpha/\beta\}\rangle_\gamma^{+}[\vec{v}\{\alpha/\beta\}] \end{array}$$

and the equality with $\mathbb{S}\{\alpha/\beta\}$ follows from Lemma 4.4.31.

2. Induction on t, using Figure 4.1. The only non-trivial case is[22] $t = v_0[v_1, \dots, v_n]$. In this case, by Figure 4.1 plus Remark 4.4.40, we have:

$$(\, v_0[v_1, \dots, v_n]\,)\,\langle [s, \vec{u}]/x\rangle^+ = \sum_W \sum_{j=0}^{n} (\, v_0\langle [\vec{w}^{\,0}] * [s]_0^j/x\rangle\,)\,[\dots, v_i\langle [\vec{w}^{\,i}] * [s]_i^j/x\rangle, \dots].$$

where $W : (\vec{u}) \longrightarrow \{1, \dots, n\}$. Fix now a $W : (\vec{u}) \longrightarrow \{1, \dots, n\}$ (together with its generated w.c.) and consider each of the $n+1$ elements of the sum on j. We write the case for $j = 0$, but the other cases are exactly the same. Since $j = 0$, the element is $(\, v_0\langle [\vec{w}^{\,0}] * [s]/x\rangle\,)\,[\dots, v_i\langle [\vec{w}^{\,i}]/x\rangle, \dots]$ and by inductive hypothesis we have:

$$(\, v_0\langle [\vec{w}^{\,0}] * [s]/x\rangle\,)\,[\dots, v_i\langle [\vec{w}^{\,i}] * [s]/x\rangle, \dots] \twoheadrightarrow_{\mathrm{r}}^+ (\, v_0\langle [\vec{w}^{\,0}] * [\mathbb{S}]/x\rangle\,)\,[\dots, v_i\langle [\vec{w}^{\,i}]/x\rangle, \dots].$$

Now summing up all the elements for $j = 0, \dots, n$ and $W : (\vec{u}) \longrightarrow \{1, \dots, n\}$ we obtain the following sum:

$$\sum_W \sum_{j=0}^{n} (\, v_0\langle [\vec{w}^{\,0}] * [\mathbb{S}]_0^j/x\rangle\,)[\dots, v_i\langle [\vec{w}^{\,i}] * [\mathbb{S}]_i^j/x\rangle, \dots].$$

Using again Remark 4.4.40 plus Figure 4.1, the above sum becomes:

$$(\, v_0[v_1, \dots, v_n]\,)\,\langle [\mathbb{S}, \vec{u}]/x\rangle^+$$

which is the desired result.

3. Induction on s.

   Case s variable: impossible.

   Case $s = \lambda$-abstraction: straightforward by Figure 4.1 and inductive hypothesis.

   Case $s = \mu\alpha._\beta|s'|$: we have two subcases:

   Subcase $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing s': immediate by Figure 4.1.

   Subcase $s' = \mu\gamma._\eta|s''|$ and $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing its leftmost ρ-redex: then $\mathbb{S} = \mu\alpha._\eta|s''|\{\beta/\gamma\}$ and (call $[\vec{u}] =: [u_1, \dots, u_k]$)

$$\begin{array}{lll}
s\langle [\vec{u}]/x\rangle^+ & = & \mu\alpha._\beta|\mu\gamma._\eta|s''\langle [\vec{u}]/x\rangle^+|| \qquad \textit{(by Figure 4.1)} \\
& \to_{\rho^{\mathrm{r}}} & \mu\alpha._\eta|s''\langle [\vec{u}]/x\rangle^+|\{\beta/\gamma\} \\
& = & \mu\alpha._\eta|s''|\langle [\vec{u}]/x\rangle^+\{\beta/\gamma\} \qquad \textit{(by Figure 4.1)} \\
& = & \mu\alpha._\eta|s''|\{\beta/\gamma\}\langle [\vec{u}]/x\rangle^+ \quad \textit{(because, being bound, } \gamma \textit{ does not occur in } \vec{u}\textit{)} \\
& = & \mathbb{S}\langle [\vec{u}]/x\rangle^+.
\end{array}$$

   Case $s = s'[\vec{v}]$: we have four subcases:

   Subcase $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing s': straightforward by Figure 4.1 and inductive hypothesis.

   Subcase $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing the v_i in $[\vec{v}]$: straightforward by Figure 4.1 and inductive hypothesis.

   Subcase $s' = \lambda y.s''$ and $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing the λ-redex s: follows by Figure 4.1 and Lemma 4.4.32 (where $x \neq y$ because y is bound and x is fixed).

   Subcase $s' = \mu\alpha._\beta|s''|$ and $s \to_{\mathrm{r}}^+ \mathbb{S}$ is performed by reducing the μ-redex s: follows by Figure 4.2 and Lemma 4.4.33 (where $\deg_\alpha([\vec{u}]) = 0$ because α is bound and $[\vec{u}]$ is fixed).

[22] In the case $t = y \neq x$, and if $\mathbb{S} = 0$ and $\vec{u}$ is empty, note that $t\langle [\mathbb{S}, \vec{u}]/x\rangle^+ = y\langle [0]/x\rangle^+ = \sum_{s' \in 0} y\langle [s']/x\rangle^+ = 0$ (and *not* "$y\langle [0]/x\rangle^+ = y\langle 1/x\rangle^+ = y$"), so the result still holds in this case.

4. Induction on t, using Figure 4.1. The non-trivial cases are $t = v_0[v_1, \ldots, v_n]$ and $t = \mu\gamma.{}_\eta|t'|$. Both are done following the same argument as we did in point (2); first, apply Figure 4.2, then Remark 4.4.40, then the inductive hypothesis and thus close the argument by applying Remark 4.4.40 again and finally Figure 4.2 again.

5. It is immediate discriminating the cases $\alpha = \beta$ and $\alpha \neq \beta$, using Figure 4.2, and concluding by point (4).

6. Induction on $s \in \lambda\mu$.

   Case $s =$ variable. Impossible.

   Case $s = \lambda$-abstraction. Straightforward by Figure 4.2 and inductive hypothesis.

   Case $s = \mu\beta.{}_\gamma|s'|$: we have two subcases:

   Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing s': then $\mathbb{S} = \mu\beta.{}_\gamma|\mathbb{S}'|$ with $s' \to_{\mathrm{r}} \mathbb{S}'$. By Figure 4.2 we have $\langle s\rangle_\alpha^+[\vec{u}] = \mu\beta.\langle {}_\gamma|s'|\rangle_\alpha^+[\vec{u}]$. Remark that we *cannot* immediately apply the inductive hypothesis on ${}_\gamma|s'|$, simply because ${}_\gamma|s'| \notin \lambda\mu^{\mathrm{r}}$ (it is a named term). However we can split in the two subcases whether $\gamma = \alpha$ or $\gamma \neq \alpha$ and now in both subcases we can conclude straightforwardly by Figure 4.2 and the inductive hypothesis.

   Subcase $s' = \mu\gamma'.{}_\eta|s''|$ (with $\gamma \neq \gamma'$) and $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing its leftmost ρ-redex: then $\mathbb{S} = \mu\beta.{}_\eta|s''|\{\gamma/\gamma'\}$. We split in two sub-subcases:

   Sub-subcase $\alpha \neq \gamma$:

$$\begin{array}{rcll}
\langle s\rangle_\alpha^+[\vec{u}] & = & \mu\beta.{}_\gamma|\mu\gamma'.\langle {}_\eta|s''|\rangle_\alpha^+[\vec{u}]| & \textit{(by Figure 4.2)} \\
& \twoheadrightarrow_{\rho^{\mathrm{r}}}^{+} & \mu\beta.\langle {}_\eta|s''|\rangle_\alpha^+[\vec{u}]\,\{\gamma/\gamma'\} & \\
& = & \langle \mu\beta.{}_\eta|s''|\{\gamma/\gamma'\}\rangle_\alpha^+[\vec{u}] & \textit{(by Lemma 4.4.31 plus } \deg_{\gamma'}([\vec{u}]) = 0\textit{)} \\
& = & \langle \mathbb{S}\rangle_\alpha^+[\vec{u}]. &
\end{array}$$

   Sub-subcase $\alpha = \gamma$. By Figure 4.2, we have:

$$\begin{array}{rcll}
\langle s\rangle_\alpha^+[\vec{u}] & = & \sum\limits_W \mu\beta.{}_\alpha|\,(\,\mu\gamma'.\langle {}_\eta|s''|\rangle_\alpha^+[\vec{w}^{\,1}]\,)\,[\vec{w}^{\,2}]| & \textit{where } W : (\vec{u}) \longrightarrow \{1,2\} \\
& \twoheadrightarrow_{\mu^{\mathrm{r}}}^{+} & \sum\limits_W \mu\beta.{}_\alpha|\mu\gamma'.\langle\langle {}_\eta|s''|\rangle_\alpha^+[\vec{w}^{\,1}]\rangle_{\gamma'}^+[\vec{w}^{\,2}]| & \\
& \twoheadrightarrow_{\rho^{\mathrm{r}}}^{+} & \sum\limits_W \mu\beta.\langle\langle {}_\eta|s''|\rangle_\alpha^+[\vec{w}^{\,1}]\rangle_{\gamma'}^+[\vec{w}^{\,2}]\,\{\alpha/\gamma'\} & \\
& = & \sum\limits_W \langle\langle \mu\beta.{}_\eta|s''|\rangle_\alpha^+[\vec{w}^{\,1}]\rangle_{\gamma'}^+[\vec{w}^{\,2}]\,\{\alpha/\gamma'\} & \\
& = & \langle \mu\beta.{}_\eta|s''|\,\{\alpha/\gamma'\}\rangle_\alpha^+[\vec{u}] & \textit{(by Lemma 4.4.37)} \\
& = & \langle \mathbb{S}\rangle_\alpha^+[\vec{u}] & \textit{(since } \alpha = \gamma\textit{).}
\end{array}$$

   Case $s = s'[v_1, \ldots, v_n]$: we have four subcases:

   Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing s': straightforward by Figure 4.2 and inductive hypothesis.

   Subcase $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing the v_i in $[\vec{v}]$: straightforward by Figure 4.2 and inductive hypothesis.

   Subcase $s' = \lambda x.s''$ and $s \to_{\mathrm{r}}^{+} \mathbb{S}$ is performed by reducing the λ-redex s. Then $\mathbb{S} =$

$s''\langle[\vec{v}]/x\rangle$ and using Figure 4.2 we have:

$$\begin{array}{rcl}\langle s\rangle^+_\alpha[\vec{u}] & = & \sum_W(\lambda x.\langle s''\rangle^+_\alpha[\vec{w}^{\,0}])\,[\dots,\langle v_i\rangle^+_\alpha[\vec{w}^{\,i}],\dots] \;\; \textit{with } W:(\vec{u})\longrightarrow\{0,\dots,n\}\\ & \twoheadrightarrow^+_{\lambda^{\mathrm{r}}} & \sum_W(\langle s''\rangle^+_\alpha[\vec{w}^{\,0}])\langle[\dots,\langle v_i\rangle^+_\alpha[\vec{w}^{\,i}],\dots]/x\rangle\\ & = & \langle s''\langle[\vec{v}]/x\rangle\rangle^+_\alpha[\vec{u}] \qquad \textit{(by Lemma 4.4.38)}\\ & = & \langle\mathbb{S}\rangle^+_\alpha[\vec{u}].\end{array}$$

Subcase $s' = \mu\gamma.{}_\eta|s''|$ (with $\gamma\neq\alpha$) and $s\to^+_{\mathrm{r}}\mathbb{S}$ is performed by reducing the μ-redex s. Then $\mathbb{S} = \mu\gamma.\langle{}_\eta|s''|\rangle^+_\gamma[\vec{v}]$ and using Figure 4.2 we have:

$$\begin{array}{rcl}\langle s\rangle^+_\alpha[\vec{u}] & = & \sum_W(\langle\mu\gamma.{}_\eta|s''|\rangle^+_\alpha[\vec{w}^{\,0}])\,[\dots,\langle v_i\rangle^+_\alpha[\vec{w}^{\,i}],\dots] \qquad \textit{with } W:(\vec{u})\longrightarrow\{0,\dots,n\}\\ & \twoheadrightarrow^+_{\mu^{\mathrm{r}}} & \mu\gamma.\sum_W\langle\langle{}_\eta|s''|\rangle^+_\alpha[\vec{w}^{\,0}]\rangle^+_\gamma[\dots,\langle v_i\rangle^+_\alpha[\vec{w}^{\,i}],\dots]\\ & = & \mu\gamma.\langle\langle{}_\eta|s''|\rangle^+_\gamma[\vec{v}]\rangle^+_\alpha[\vec{u}] \qquad \textit{(by Lemma 4.4.39)}\\ & = & \langle\mathbb{S}\rangle^+_\alpha[\vec{u}].\end{array}$$

7. It is immediate by discriminating the cases $\alpha=\beta$ and $\alpha\neq\beta$, using Figure 4.2, and then concluding by point (6). □

Now we can finally state and prove the local confluence of quantitative resource $\lambda\mu$-calculus.

Proposition 4.4.42. *The reduction $\to^+_{\mathrm{r}}$ is locally confluent in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$.*

Proof. We show, by induction on a single-hole resource context c, that if:

$$t\to^+_{\mathrm{base}^{\mathrm{r}}}\mathbb{T} \;\textit{ and }\; c(\!|t|\!)\to^+_{\mathrm{r}}\mathbb{T}_2$$

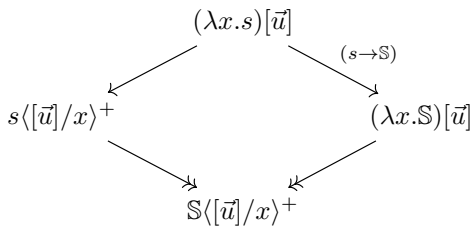
then there is $\mathbb{T}'\in\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ s.t.

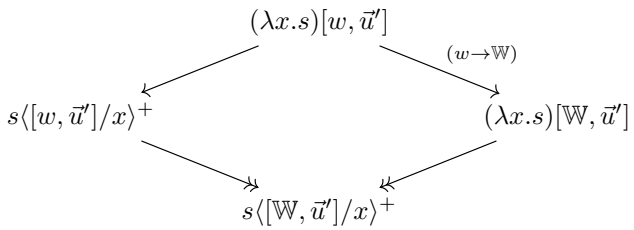
$$c(\!|\mathbb{T}|\!)\twoheadrightarrow^+_{\mathrm{r}}\mathbb{T}'\,{}^+_{\mathrm{r}}\!\!\twoheadleftarrow\mathbb{T}_2.$$

In all the following diagrams we write "$\to$" but of course we mean "$\to^+_{\mathrm{r}}$".

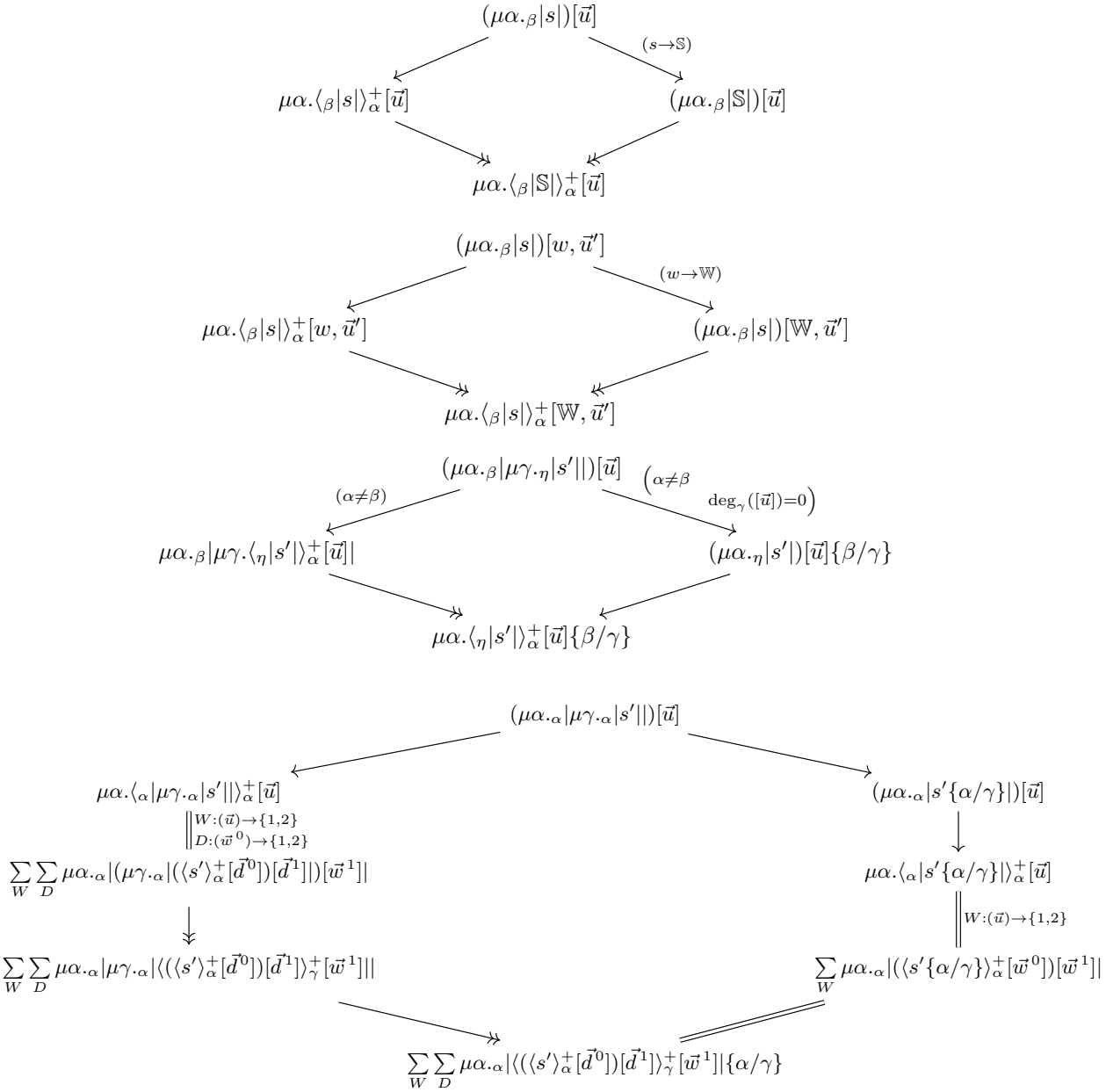
1. Case $c=\xi$. So $c(\!|t|\!) = t\to^+_{\mathrm{base}^{\mathrm{r}}}\mathbb{T}$ and we only have the three base-cases of Definition 4.4.20. In all the diagrams of this case, when not explicitly said differently or not clear by an easy reduction, the bottom-left reduction follows from Lemma 4.4.41 and the bottom-right from Remark 4.4.28.

   Subcase $t=(\lambda x.s)[\vec{u}]$ and $\mathbb{T}=s\langle[\vec{u}]/x\rangle^+$. Then $c(\!|t|\!)=t\to^+_{\mathrm{r}}\mathbb{T}_2$ (on a different redex than t) can only be performed either by reducing s, or by reducing an element w of $[\vec{u}]$. We have thus the following two diagrams:





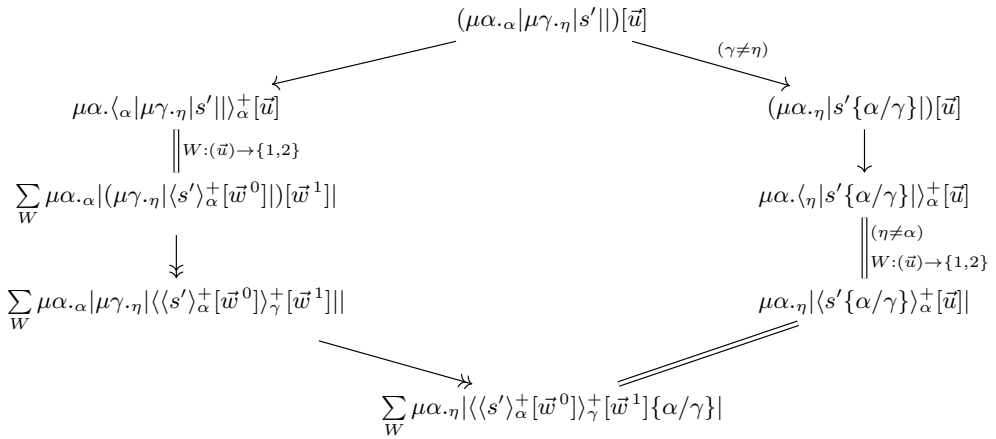
Subcase $t = (\mu\alpha.{}_\beta|s|)[\vec{u}]$ and $\mathbb{T} = \mu\alpha.\langle_\beta|s|\rangle^+_\alpha[\vec{u}]$. Then $c(\!|t|\!) = t \to^+_{\mathrm{r}} \mathbb{T}_2$ (on a different redex than t) can only be performed either by reducing s, or by reducing an element w of $[\vec{u}]$, or if $s = \mu\gamma.{}_\eta|s'|$ and we reduce the ρ-redex $...{}_\beta|\mu\gamma.\,...|$. In the latter case we split into the case $\alpha \neq \beta$, the case $\alpha = \beta, \gamma \neq \eta, \eta = \alpha$, the case $\alpha = \beta, \gamma \neq \eta, \eta \neq \alpha$, and the case $\alpha = \beta, \eta = \gamma$ (with necessary $\gamma \neq \alpha$). The above six cases respectively correspond to the following six diagrams:



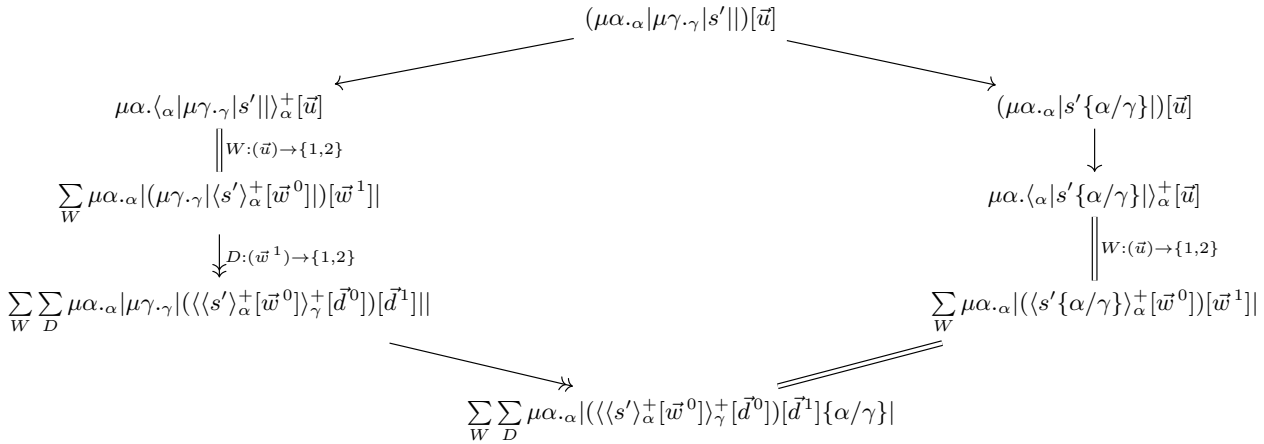
where the bottom right equality holds because, by Lemma 4.4.37, Remark 4.4.34 and since $\deg_\gamma([\vec{u}]) = 0$, each addend of the bottom right sum is:

$$\begin{aligned}\mu\alpha.{}_\alpha|(\langle s'\{\alpha/\gamma\}\rangle^+_\alpha[\vec{w}^{\,0}])[\vec{w}^{\,1}]| &= \sum_{D:(\vec{w}^{\,0})\to\{1,2\}} \mu\alpha.{}_\alpha|(\langle\langle s'\rangle^+_\alpha[\vec{d}^{\,1}]\rangle^+_\gamma[\vec{d}^{\,0}])[\vec{w}^{\,1}]|\{\alpha/\gamma\}\\ &= \sum_D \mu\alpha.{}_\alpha|\langle(\langle s'\rangle^+_\alpha[\vec{d}^{\,1}])[\vec{w}^{\,1}]\rangle^+_\gamma[\vec{d}^{\,0}]|\{\alpha/\gamma\}\end{aligned}$$

and since we are then summing up on all possible $W : (\vec{u}) \to \{1, 2\}$, the resulting sum is the same as the one at the bottom of the above diagram.

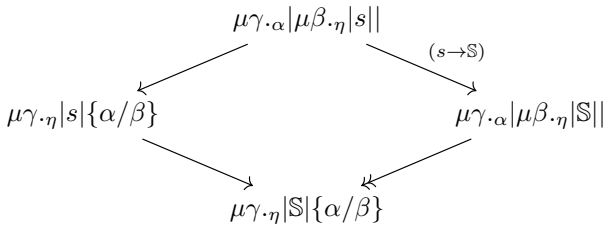


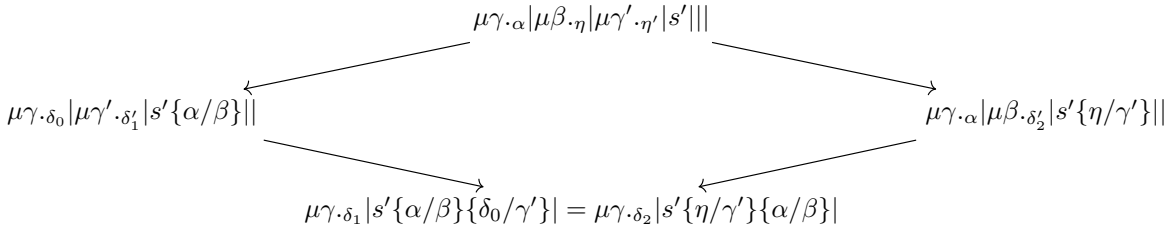
where the bottom right equality holds by Lemma 4.4.37 and because $\deg_\gamma([\vec{u}]) = 0$.



where the bottom right equality holds by Lemma 4.4.37 and because $\deg_\gamma([\vec{u}]) = 0$.

Subcase $t = \mu\gamma.{}_\alpha|\mu\beta.{}_\eta|s||$ and $\mathbb{T} = \mu\gamma.{}_\eta|s|\{\alpha/\beta\}$. Then $c(\!|t|\!) = t \to^+_{\mathrm{r}} \mathbb{T}_2$ (on a different redex than t) can be only performed either by reducing s, or if $s = \mu\gamma'.{}_{\eta'}|s'|$ and we reduce the ρ-redex $...{}_\eta|\mu\gamma'.\,...|$. We have thus the following two diagrams:





where $\delta_0 := \delta^{\alpha}_{\eta}(\beta)$, $\delta'_1 := \delta^{\alpha}_{\eta'}(\beta)$, $\delta_1 := \delta^{\delta_0}_{\delta'_1}(\gamma')$, $\delta'_2 := \delta^{\eta}_{\eta'}(\gamma')$ and $\delta_2 := \delta^{\alpha}_{\delta'_2}(\beta)$. Let us prove the equality in the last diagram: we first show that $s'\{\alpha/\beta\}\{\delta_0/\gamma'\} = s'\{\eta/\gamma'\}\{\alpha/\beta\}$ and then that $\delta_1 = \delta_2$. For the former equality, we have:

if $\beta \neq \eta$, then $\delta_0 = \eta$ and the equality follows from Lemma 4.4.30;

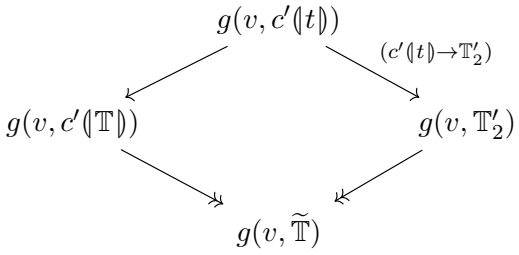
if $\beta = \eta$ then $\delta_0 = \alpha$ and the equality holds because both renamings $\{\alpha/\beta\}\{\delta_0/\gamma'\}$ and $\{\eta/\gamma'\}\{\alpha/\beta\}$ coincide with the unique renaming $\{\alpha/(\beta,\gamma')\}$[23].

For the latter equality, we have:

if $\gamma' = \eta'$: then $\delta'_2 = \eta$ and thus $\delta_2 = \delta_0$. Remark that it must be $\beta \neq \eta'$, because otherwise $\beta = \gamma'$ which is impossible. This means that $\delta'_1 = \eta'$. But then $\delta_1 = \delta_0$, so we are done.

if $\gamma' \neq \eta'$: then $\delta'_2 = \eta'$ and thus $\delta_2 = \delta'_1$. Now if $\beta \neq \eta'$ then $\delta'_1 = \eta'$ and thus $\delta_1 = \eta'$; if $\beta = \eta'$ then $\delta'_1 = \alpha$ and thus $\delta_1 = \delta^{\delta_0}_{\alpha}(\gamma') = \alpha$, because it cannot be $\gamma' = \alpha$. If we read what we just found about δ_1, it precisely says that $\delta_1 = \delta'_1$ so we are done.

2. Case $c = \mu\alpha.{}_{\beta}|c'|$ with either $c'(\!|t|\!)$ *not* a μ-abstraction, or $c = \lambda x.c'$. Then the reduction $c(\!|t|\!) \to \mathbb{T}_2$ can only be performed via a reduction $c'(\!|t|\!) \to \mathbb{T}'_2$. The diagram for both cases has the same shape; using the notation of Section 4.3, let's call in both cases $c =: g(v, c')$ (where v is either x if g is a λ-abstraction, or (α,β) if g is a μ-abstraction). Now the result is given by the following diagram:



where a sum $\widetilde{\mathbb{T}}$ s.t. $c'(\!|\mathbb{T}|\!) \twoheadrightarrow \widetilde{\mathbb{T}} \twoheadleftarrow \mathbb{T}'_2$ is given by inductive hypothesis on c'.

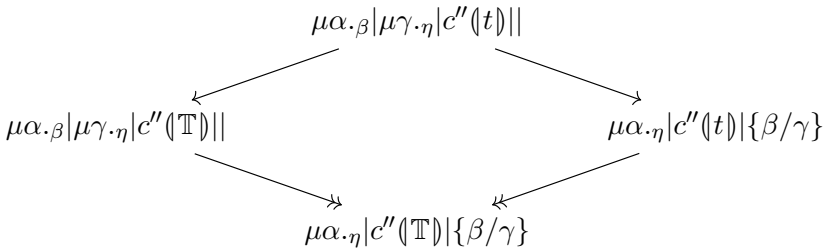
3. Case $c = \mu\alpha.{}_{\beta}|c'|$, with $c'(\!|t|\!)$ a μ-abstraction. Then either $c' = \xi$ and $t = \mu\gamma.{}_{\eta}|t'_0|$, or $c' = \mu\gamma.{}_{\eta}|c''|$.

   In the first case, since by hypothesis $t \to^{+}_{\text{base}^{\text{r}}} \mathbb{T}$, it must be $t = \mu\gamma.{}_{\eta}|\mu\gamma'.{}_{\eta'}|t'||$ and $\mathbb{T} = \mu\gamma.{}_{\eta'}|t'|\{\eta/\gamma'\}$. Therefore, the reduction $c(\!|t|\!) \to_{\text{r}} \mathbb{T}_2$ (on a different redex than the one of $t \to^{+}_{\text{base}^{\text{r}}} \mathbb{T}$) can only be performed either by reducing t', or by reducing the ρ-redex $...{}_{\beta}|\mu\gamma.\,...|$, or if $t' = \mu\widetilde{\gamma}.{}_{\widetilde{\eta}}|\widetilde{t}|$ and we reduce the ρ-redex $...{}_{\eta'}|\mu\widetilde{\gamma}.\,...|$. The first and second situation of the previous list have been already treated in the case $c = \xi$ (with the notation used there, it corresponds to the two diagrams of the subcase $t = \mu\gamma.{}_{\alpha}|\mu\beta.{}_{\eta}|s||$ and $\mathbb{T} = \mu\gamma.{}_{\eta}|s|\{\alpha/\beta\}$.). Also the third situation corresponds to the exact same case just mentioned, because the external $\mu\alpha.{}_{\beta}|...|$ is not modified in neither reductions.

   In the second case, then the reduction $c(\!|t|\!) \to_{\text{r}} \mathbb{T}_2$ can only be performed either by reducing $c''(\!|t|\!)$, or by reducing the ρ-redex $c(\!|t|\!)$. In the first situation the diagram follows easily by inductive hypothesis as in the previous case, and in the second one the diagram is the

[23] We mean here that both β and γ' get renamed with α.

following:



thanks to Remark 4.4.28 and Lemma 4.4.41.

4. Case $c = c'[\vec{u}]$. Then $c\langle\!|t|\!\rangle = c'\langle\!|t|\!\rangle[\vec{u}]$ and the reduction $c\langle\!|t|\!\rangle \to_{\mathrm{r}} \mathbb{T}_2$ can only be performed either by reducing $c'\langle\!|t|\!\rangle$, or reducing an element of $[\vec{u}]$, or the in case $c'\langle\!|t|\!\rangle$ is a λ-abstraction or a μ-abstraction and we reduce the $\lambda\mu$-redex $c\langle\!|t|\!\rangle$. In the first case one can easily use inductive hypothesis; in the second case one can easily write the diagram; let us see the third and fourth case.

   If $c'\langle\!|t|\!\rangle$ is a λ-abstraction, then either $c' = \xi$ and $t = \lambda x.t'$, or $c' = \lambda x.c''$. But the first case is impossible, because by hypothesis $t \to^+_{\mathrm{base}^{\mathrm{r}}} \mathbb{T}$, so t cannot be a λ-abstraction; In the second case the diagram corresponds to the first diagram of the case 1 (with the notations used there, take $s := c''\langle\!|t|\!\rangle$ and $\mathbb{S} := c''\langle\!|\mathbb{T}|\!\rangle$).

   If $c'\langle\!|t'|\!\rangle$ is a μ-abstraction, then either $c' = \xi$ and $t = \mu\alpha.{}_\beta|t'|$, or $c' = \mu\alpha.{}_\beta|c''|$. But in the first situation, since by hypothesis $t \to^+_{\mathrm{base}^{\mathrm{r}}} \mathbb{T}$, it must be $t = \mu\alpha.{}_\beta|\mu\gamma.{}_\eta|t''||$ and $\mathbb{T} = \mu\alpha.{}_\eta|t''|\{\beta/\gamma\}$, and this situation has already been treated in the case $c = \xi$ (with the notations used there, it corresponds to the subcase $t = (\mu\alpha.{}_\beta|s|)[\vec{u}]$ and $\mathbb{T} = \mu\alpha.\langle{}_\beta|s|\rangle^+_\alpha[\vec{u}]$, where we consider the reduction of the ρ-redex[24]). In the second case the diagram corresponds to the third diagram of the case 1 (with the notations used there, take $s := c''\langle\!|t|\!\rangle$ and $\mathbb{S} := c''\langle\!|\mathbb{T}|\!\rangle$).

5. Case $c = v[c', \vec{u}]$. Then $c\langle\!|t|\!\rangle = v[c'\langle\!|t|\!\rangle, \vec{u}]$ and the reduction $c\langle\!|t|\!\rangle \to_r \mathbb{T}_2$ can only be performed either by reducing $c'\langle\!|t|\!\rangle$, or reducing v, or reducing an element of $[\vec{u}]$, or in the case v is a λ-abstraction or a μ-abstraction and we reduce the $\lambda\mu$-redex $c\langle\!|t|\!\rangle$. In the first case one can trivially use the inductive hypothesis as done in the previous case; in the second and third case one can easily write the diagram; let's look at the fourth case.

   If v is a λ-abstraction, say $v = \lambda x.v'$, then the diagram corresponds to the second diagram of the case 1 (with the notations used there, take $w := c'\langle\!|t|\!\rangle$ and $\mathbb{W} := c'\langle\!|\mathbb{T}|\!\rangle$).

   If v is a μ-abstraction, say $v = \mu\alpha.{}_\beta|v'|$, then the diagram corresponds to the fourth diagram of the case 1 (with the notations used there, take $w := c'\langle\!|t|\!\rangle$ and $\mathbb{W} := c'\langle\!|\mathbb{T}|\!\rangle$).

□

Corollary 4.4.43. *The reduction $\to^+_{\mathrm{r}}$ is confluent in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$. Furthermore, every $\mathbb{T} \in \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ has exactly one $\to^+_{\mathrm{r}}$-normal form in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$.*

Proof. The confluence is an immediate application of Newman Lemma (Lemma 2.1.1), thanks to the strong normalisation (Remark 4.4.26) and Proposition 4.4.42. The existence and uniqueness of the normal form immediately follows from the confluence and the strong normalisation. □

[24] The diagrams are the last four of that subcase.

Now that we have the confluence (in particular, the fact that every term has at most one $\to_{\mathrm{r}}^{+}$-normal form) of the qualitative resource calculus (that is, of $(\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle, \to_{\mathrm{r}}^{+})$), we can prove the local confluence of the qualitative one, that is, of $(2\langle\lambda\mu^{\mathrm{r}}\rangle, \to_{\mathrm{r}})$. In order to do so, we introduce a reduction $\Rightarrow \subseteq \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ which $\to_{\mathrm{r}}^{+}$-reduces in one step all the occurrences of a term t of a quantitative sum in the same way.

In the following, $\mathrm{supp}(\mathbb{T}) \in 2\langle\lambda\mu^{\mathrm{r}}\rangle$ is the *support* of a sum $\mathbb{T} \in \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$, that is, the set of all its addends (with no coefficients)[25].

Definition 4.4.44. *The reduction $\Rightarrow \subseteq \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ is defined as the contextual closure of the reduction in Figure 4.3 (where $m \in \mathbb{N}$, $t \in \lambda\mu^{\mathrm{r}}$ and $\mathbb{T}, \mathbb{S} \in \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$).*

$$\frac{t \to_{\mathrm{r}}^{+} \mathbb{T} \qquad t \notin \mathrm{supp}(\mathbb{S})}{m\,t + \mathbb{S} \Rightarrow m\,\mathbb{T} + \mathbb{S}}$$

Figure 4.3: The reduction $\Rightarrow \subseteq \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle \times \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$

Lemma 4.4.45. *1.* $\Rightarrow \subseteq \twoheadrightarrow_{\mathrm{r}}^{+}$.

2. If $\mathbb{T} \to_{\mathrm{r}} \mathbb{S}$ in $2\langle\lambda\mu^{\mathrm{r}}\rangle$, then for all $m_t \in \mathbb{N}$ (with $t \in \mathbb{T}$), we have $\sum_{t\in\mathbb{T}} m_t t \Rightarrow \mathbb{S}'$ (in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$), with $\mathrm{supp}(\mathbb{S}') = \mathbb{S}$.

Proof. Both properties simply follow from the definition of $\Rightarrow$. □

Corollary 4.4.46. *The reduction $\to_{\mathrm{r}}$ in $2\langle\lambda\mu^{\mathrm{r}}\rangle$ is locally confluent.*

Proof. Let $\mathbb{T}_1 \ {}_{\mathrm{r}}\!\leftarrow t \to_{\mathrm{r}} \mathbb{T}_2$ in $2\langle\lambda\mu^{\mathrm{r}}\rangle$. Since we know that $\to_{\mathrm{r}}$ is strongly normalising (Corollary 4.4.15), there are (in $2\langle\lambda\mu^{\mathrm{r}}\rangle$) reductions $\mathbb{T}_1 \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_1$ and $\mathbb{T}_2 \twoheadrightarrow_{\mathrm{r}} \mathbb{S}_2$, with $\mathbb{S}_1, \mathbb{S}_2$ r-normal. Therefore by Lemma 4.4.45(2), we have (in $\mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$) reductions $t \Rightarrow \cdots \Rightarrow \mathbb{S}_1'$ and $t \Rightarrow \cdots \Rightarrow \mathbb{S}_2'$, for some $\mathbb{S}_1', \mathbb{S}_2' \in \mathbb{N}\langle\lambda\mu^{\mathrm{r}}\rangle$ s.t. $\mathrm{supp}(\mathbb{S}_i') = \mathbb{S}_i$. But then, since $\mathbb{S}_i$ is r-normal, $\mathbb{S}_i'$ must be $\to_{\mathrm{r}}^{+}$-normal. Now because of Corollary 4.4.43, it must be $\mathbb{S}_1' = \mathbb{S}_2'$, and therefore $\mathbb{S}_1 = \mathrm{supp}(\mathbb{S}_1') = \mathrm{supp}(\mathbb{S}_2') = \mathbb{S}_2$. Hence, we found a common reduct of $\mathbb{T}_1, \mathbb{T}_2$. □

We can now finally infer the confluence of the reduction $\to_{\mathrm{r}}$.

Corollary 4.4.47 (Confluence)**.** *The reduction $\to_{\mathrm{r}}$ is confluent on $2\langle\lambda\mu^{\mathrm{r}}\rangle$.*

Proof. It is an immediate application of Newman Lemma (Lemma 2.1.1), thanks to Corollary 4.4.15 and Corollary 4.4.46. □

4.4.1 Qualitative Taylor expansion

We define the qualitative Taylor expansion of $\lambda\mu$-calculus following Definition 4.3.2. That is, we define the map $\mathcal{T} : \lambda\mu \to \mathcal{P}(\lambda\mu^{\mathrm{r}})$ as:

$$\mathcal{T}(x) := \{x\} \qquad \mathcal{T}(\lambda x.M) := \{\lambda x.t \mid t \in \mathcal{T}(M)\} \qquad \mathcal{T}(\mu\alpha._{\beta}|M|) := \{\mu\alpha._{\beta}|t| \mid t \in \mathcal{T}(M)\}$$

$$\mathcal{T}(MN) := \{t[\vec{u}] \mid t \in \mathcal{T}(M),\ [\vec{u}] \in !\,\mathcal{T}(N)\}.$$

[25] That is, $\mathrm{supp}(\mathbb{T})$ is $\mathbb{T}$ when considered with an idempotent sum.

The same definition, as usual, also endows $\lambda\mu$ with the partial preorder $\leq$ given by the inclusion of Taylor normal forms.

Remark that the Monotonicity property (Theorem 4.3.4) is then true for $\lambda\mu$-calculus, that is: for C a k-context, the map $C : \lambda\mu \times \cdots \times \lambda\mu \longrightarrow \lambda\mu$ is monotone w.r.t. $\leq$.

As usual, we need a Lemma saying that Taylor expansion behaves well w.r.t. substitutions.

Lemma 4.4.48. *One has:*

$$\begin{array}{lcl} \mathcal{T}(M\{\alpha/\beta\}) & = & \mathcal{T}(M)\{\alpha/\beta\} \\ \mathcal{T}(M\{N/x\}) & = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{\vec{u}\in\, !\mathcal{T}(N)} t\langle[\vec{u}]/x\rangle \\ \mathcal{T}((M)_\alpha N) & = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{\vec{u}\in\, !\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}]. \end{array}$$

Proof. (1). Straightforward induction on M.

(2). Induction on M. Nothing changes w.r.t. the proof one does in λ-calculus, the only new case is $M = \mu\beta.{}_\alpha|P|$ but it is done straightforwardly exactly as the case $M = \lambda x.P$.

(3). Induction on M.

Case $M = x$:

$$\mathcal{T}((M)_\alpha N) = \mathcal{T}((x)_\alpha N) = \{x\} = \langle x\rangle_\alpha 1 = \bigcup_{\vec{u}\in\, !\mathcal{T}(N)} \langle x\rangle_\alpha[\vec{u}] = \bigcup_{t\in\mathcal{T}(M),\vec{u}\in\, !\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}].$$

Case $M = \lambda x.P$:

$$\begin{array}{lcll} \mathcal{T}((M)_\alpha N) & = & \mathcal{T}(\lambda x.(P)_\alpha N) & \\ & = & \{\lambda x.s \mid s \in \mathcal{T}((P)_\alpha N)\} & \\ & = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \lambda x.(\langle p\rangle_\alpha[\vec{u}]) & \textit{(by inductive hypothesis)} \\ & = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \langle \lambda x.p\rangle_\alpha[\vec{u}] & \\ & = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}]. & \end{array}$$

Case $M = \mu\beta.{}_\gamma|P|$ (with $\beta, \gamma \neq \alpha$):

$$\begin{array}{lcll} \mathcal{T}((M)_\alpha N) & = & \mathcal{T}(\mu\beta.{}_\gamma|(P)_\alpha N|) & \\ & = & \{\mu\beta.{}_\gamma|s| \mid s \in \mathcal{T}((P)_\alpha N)\} & \\ & = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \mu\beta.{}_\gamma|\langle p\rangle_\alpha[\vec{u}]| & \textit{(by inductive hypothesis)} \\ & = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \langle \mu\beta.{}_\gamma|p|\rangle_\alpha[\vec{u}] & \\ & = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{[\vec{u}]\in\, !\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}]. & \end{array}$$

Case $M = \mu\beta.{}_\alpha|P|$ (with $\beta \neq \alpha$):

$$\begin{array}{rcl}
\mathcal{T}((M)_\alpha N) & = & \mathcal{T}(\mu\beta.{}_\alpha|((P)_\alpha N)N|) \\
& = & \left\{\mu\beta.{}_\alpha|v[\vec{w}]| \mid [\vec{w}] \in !\,\mathcal{T}(N),\ v \in \bigcup_{p\in\mathcal{T}(P),\vec{q}\in\,!\,\mathcal{T}(N)} \langle p\rangle_\alpha[\vec{q}]\right\} \textit{ (by ind. hyp.)} \\
& = & \left\{\mu\beta.{}_\alpha|v[\vec{w}]| \mid [\vec{w}] \in !\,\mathcal{T}(N),\ v \in \langle p\rangle_\alpha[\vec{q}],\ p \in \mathcal{T}(P),\ [\vec{q}] \in !\,\mathcal{T}(N)\right\} \\
& = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \mu\beta. \sum_{\substack{([\vec{w}],[\vec{q}]) \\ w.c.\ of\ [\vec{u}]}} {}_\alpha|(\langle p\rangle_\alpha[\vec{q}])[\vec{w}]| \\
& = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \mu\beta.\langle{}_\alpha|p|\rangle_\alpha[\vec{u}] \qquad \textit{(by Figure 4.2)} \\
& = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \langle\mu\beta.{}_\alpha|p|\rangle_\alpha[\vec{u}] \\
& = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}].
\end{array}$$

Case $M = PQ$:

$$\begin{array}{rcl}
\mathcal{T}((M)_\alpha N) & = & \mathcal{T}(((P)_\alpha N)((Q)_\alpha N)) \\
& = & \bigcup_{n\in\mathbb{N}} \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{q}]\in\,!\,\mathcal{T}(Q)} \bigcup_{[\vec{s}^0],\ldots,[\vec{s}^n]\in\,!\,\mathcal{T}(N)} \left\{v[w_1,\ldots,w_n] \mid v \in \langle p\rangle_\alpha[\vec{s}^0],\ w_i \in \langle q_i\rangle_\alpha[\vec{s}^i]\right\} \\
& & \textit{(by inductive hypothesis)} \\
& = & \bigcup_{n\in\mathbb{N}} \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{q}]\in\,!\,\mathcal{T}(Q)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \sum_{\substack{([\vec{s}^0],\ldots,[\vec{s}^n]) \\ w.c.\ of\ [\vec{u}]}} (\langle p\rangle_\alpha[\vec{s}^0])[\langle q_1\rangle_\alpha[\vec{s}^1],\ldots,\langle q_n\rangle_\alpha[\vec{s}^n]] \\
& = & \bigcup_{p\in\mathcal{T}(P)} \bigcup_{[\vec{q}]\in\,!\,\mathcal{T}(Q)} \bigcup_{[\vec{u}]\in\,!\,\mathcal{T}(N)} \langle p[\vec{q}]\rangle_\alpha[\vec{u}] \qquad \textit{(by Figure 4.2)} \\
& = & \bigcup_{t\in\mathcal{T}(M)} \bigcup_{\vec{u}\in\,!\,\mathcal{T}(N)} \langle t\rangle_\alpha[\vec{u}].
\end{array}$$

□

We prove that Assumption 3 holds.

Proposition 4.4.49. *Assumption 3 holds in λμ-calculus. That is, if $M \to_{\text{base}} N$, then:*

1. *for all $s \in \mathcal{T}(M)$ there exist $\mathbb{T} \subseteq \mathcal{T}(N)$ s.t. $s \twoheadrightarrow_{\text{r}} \mathbb{T}$*
2. *for all $s' \in \mathcal{T}(N)$ there is $s \in \mathcal{T}(M)$ s.t. $s \twoheadrightarrow_{\text{r}} s' + \mathbb{T}$ for some sum $\mathbb{T} \subseteq \mathcal{T}(N)$.*

Therefore, Proposition 4.3.5 lifts the statement to all $\to$.

Proof. We have three subcases, corresponding to the three cases of the base-reduction.

Subcase $M = (\mu\alpha.{}_\beta|P|)Q,\ N = \mu\alpha.(_\beta|P|)_\alpha Q$.

(1). If $s \in \mathcal{T}((\mu\alpha.{}_\beta|P|)Q)$ then $s = (\mu\alpha.{}_\beta|p|)[\vec{q}]$ for $p \in \mathcal{T}(P)$ and $[\vec{q}] \in !\,\mathcal{T}(Q)$. So $s \to_{\text{r}} \mu\alpha.\langle{}_\beta|p|\rangle_\alpha[\vec{q}] \subseteq \mathcal{T}(\mu\alpha.(_\beta|P|)_\alpha Q)$ thanks to Lemma 4.4.48.

(2). If $s' \in \mathcal{T}(\mu\alpha.(_\beta|P|)_\alpha Q)$ then thanks to Lemma 4.4.48, $s' \in \mu\alpha.\langle{}_\beta|p|\rangle_\alpha[\vec{q}]$ for $p \in \mathcal{T}(P)$ and $[\vec{q}] \in !\,\mathcal{T}(Q)$. So $\mathcal{T}((\mu\alpha.{}_\beta|P|)Q) \ni (\mu\alpha.{}_\beta|p|)[\vec{q}] \to_{\text{r}} \mu\alpha.\langle{}_\beta|p|\rangle_\alpha[\vec{q}] \ni s'$, and $\mu\alpha.\langle{}_\beta|p|\rangle_\alpha[\vec{q}] \subseteq \mathcal{T}(\mu\alpha.(_\beta|P|)_\alpha Q)$.

Subcase $M = \mu\gamma.{}_\alpha|\mu\beta.{}_\eta|P||,\ N = \mu\gamma.{}_\eta|P|\{\alpha/\beta\}$. (1) and (2) are straightforward using Lemma 4.4.48 as above.

Subcase $M = (\lambda x.P)Q$, $N = P\{Q/x\}$. (1) and (2) are straightforward using Lemma 4.4.48 as above.

□

Let us now prove Assumption 4.
We need the following:

Lemma 4.4.50. *Let P, Q be $\lambda\mu$-terms, $p, p' \in \mathcal{T}(P)$ and $[\vec{d}], [\vec{d'}] \in !\mathcal{T}(Q)$. Then $p = p'$ and $[\vec{d}] = [\vec{d'}]$ are entailed by any of the following three conditions:*

1. *if $p\langle[\vec{d}]/x\rangle \cap p'\langle[\vec{d'}]/x\rangle \neq \emptyset$*
2. *if $\langle p\rangle_\gamma[\vec{d}] \cap \langle p'\rangle_\gamma[\vec{d'}] \neq \emptyset$*
3. *if $\langle_\eta|p|\rangle_\gamma[\vec{d}] \cap \langle_\eta|p'|\rangle_\gamma[\vec{d'}] \neq \emptyset$[26].*

Proof. 1. Induction on P, similar to as it is done in [ER08].

2. Induction on P. Let us see the case $P = \mu\alpha._\beta|M|$, the other cases being similar. Then $p = \mu\alpha._\beta|s|$ and $p' = \mu\alpha._\beta|s'|$, with $s, s' \in \mathcal{T}(M)$. Let $h \in \langle p\rangle_\gamma[\vec{d}] \cap \langle p'\rangle_\gamma[\vec{d'}]$. By Remark 4.4.5 (chosing $\alpha \neq \gamma$) we have two subcases.

   Subcase $\beta \neq \gamma$: then $h = \mu\alpha._\beta|h_0|$ with $h_0 \in \langle s\rangle_\gamma[\vec{d}] \cap \langle s'\rangle_\gamma[\vec{d'}]$. We can easily conclude by inductive hypothesis.

   Subcase $\beta = \gamma$: then $h = \mu\alpha._\gamma|h_0[\vec{v}]|$, with $h_0 \in \langle s\rangle_\gamma[\vec{w}] \cap \langle s'\rangle_\gamma[\vec{w}']$ and where $([\vec{v}], [\vec{w}])$ and $([\vec{v}], [\vec{w}'])$ are w.c. respectively of $[\vec{d}]$ and of $[\vec{d'}]$. Thus by inductive hypothes we have $s = s'$, i.e. $p = p'$, and also $[\vec{w}] = [\vec{w}']$. Finally, $[\vec{d}] = [\vec{w}] * [\vec{v}] = [\vec{w}'] * [\vec{v}] = [\vec{d'}]$.

3. Let $\alpha \neq \gamma$ and fresh. Then $\langle_\eta|p|\rangle_\gamma[\vec{d}] \cap \langle_\eta|p'|\rangle_\gamma[\vec{d'}] \neq \emptyset$ iff $\mu\alpha.\langle_\eta|p|\rangle_\gamma[\vec{d}] \cap \mu\alpha.\langle_\eta|p'|\rangle_\gamma[\vec{d'}] \neq \emptyset$. Using Figure 4.2 the latter interparagraph is $\langle\mu\alpha._\eta|p|\rangle_\gamma[\vec{d}] \cap \langle\mu\alpha._\eta|p'|\rangle_\gamma[\vec{d'}]$, so it is of the shape considered by (2), and $\mu\alpha._\eta|p|, \mu\alpha._\eta|p'| \in \lambda\mu$. Therefore (2) gives $\mu\alpha._\eta|p| = \mu\alpha._\eta|p'|$, i.e. $p = p'$, as well as $[\vec{d}] = [\vec{d'}]$. □

The previous statement may seem strange at first sight, because one would expect (3) (which is the item used in the proof of Theorem 4.4.51, together with (1)) to be an indutive step of (2). However, (3) is *not* an inductive step of (2), simply because $_\eta|p| \notin \lambda\mu^{\mathrm{r}}$. This is due to the fact that we are in $\lambda\mu$-calculus and not in Saurin's $\Lambda\mu$-calculus. One could be then tempted to state it in $\Lambda\mu$-calculus, so only with item (2). In this case, since p would be in $\Lambda\mu$ and not just in $\lambda\mu$, (3) would in fact be an inductive step of (2), but still this is not what we need: in fact when we use (3) in the present chapter, we want p to be in $\lambda\mu$, something which is not guaranteed if we state Lemma 4.4.50 for $\Lambda\mu$.

Now we can prove Assumption 4.

Theorem 4.4.51 (Non-interference property for $\lambda\mu$-calculus)**.** *Assumption 4 holds for the $\lambda\mu$-calculus. That is, for all $t, s \in \mathcal{T}(M)$ s.t. $t \neq s$, we have $\mathrm{nf}_{\mathrm{r}}(t) \cap \mathrm{nf}_{\mathrm{r}}(s) = \emptyset$.*

Proof. By induction on $\mathbf{ms}(t)$ we prove that for all $s \in \lambda\mu^{\mathrm{r}}$, if $t, s \in \mathcal{T}(M)$ for some $M \in \lambda\mu$, and if there is $h \in \mathrm{nf}_{\mathrm{r}}(t) \cap \mathrm{nf}_{\mathrm{r}}(s)$, then $t = s$.

Case $\mathbf{ms}(t) = (1, 0, 1)$. Then (Corollary 4.4.19) t is a variable, thus M is the same variable and therefore $s = t$.

[26] Notice that, even if this set is *not* a set of $\lambda\mu$-terms, item (3) still makes perfect sense: it is just the intersection of two sets, the sets of *named terms* built according to Figure 4.2.

Case $\mathbf{ms}(t) > (1,0,1)$. By Lemma 4.4.2, M has shape:

$$M = \lambda\vec{x}_1\mu\alpha_{1\cdot\beta_1}|\dots\lambda\vec{x}_k\mu\alpha_{k\cdot\beta_k}|RQ_1\dots Q_n||$$

for R either a variable, or a λ-redex or a μ-redex. Since the series of λ and μ abstractions (with their respective namings) will play no role in the following, in this proof we shorten $\lambda\vec{x}_1\mu\alpha_{1\cdot\beta_1}|\dots\lambda\vec{x}_k\mu\alpha_{k\cdot\beta_k}|\dots||$ to just $\vec{\lambda\mu}|\dots|$. So: $t = \vec{\lambda\mu}|t'[\vec{u}^{\,1}]\dots[\vec{u}^{\,n}]||$ and $s = \vec{\lambda\mu}|s'[\vec{v}^{\,1}]\dots[\vec{v}^{\,n}]||$ for $t', s' \in \mathcal{T}(R)$ and $[\vec{u}^{\,i}], [\vec{v}^{\,i}] \in !\,\mathcal{T}(Q_i)$.

We have now three subcases depending on the shape of R.

Subcase R variable, say $R = x$. Then $t' = s' = x$. Wlog $n \geq 1$, otherwise it is trivial that $t = s$. Now say $[\vec{u}^{\,i}] =: [u^i_1, \dots, u^i_{m_i}]$ and $[\vec{v}^{\,i}] =: [v^i_1, \dots, v^i_{m'_i}]$ for $i = 1, \dots, n$. By confluence we have $h \in \mathrm{nf_r}(\vec{\lambda\mu}|x\,\mathrm{nf_r}([\vec{u}^{\,1}])\dots\mathrm{nf_r}([\vec{u}^{\,n}])|)$, so $h \in \mathrm{nf_r}(\vec{\lambda\mu}|x[\vec{d}^{\,1}]\dots[\vec{d}^{\,n}]|)$ for some $d^i_j \in \mathrm{nf_r}(u^i_j)$. Similarly, we get: $h \in \mathrm{nf_r}(\vec{\lambda\mu}|x[\vec{d}^{\,\prime 1}]\dots[\vec{d}^{\,\prime n}]|)$ for some $d'^{\,i}_j \in \mathrm{nf_r}(v^i_j)$. So it must be $m_i = m'_i$ $(i = 1, \dots, n)$ and:

$$h = \vec{\lambda\mu}'|x[d^1_1, \dots, d^1_{m_1}]\cdots[d^n_1, \dots, d^n_{m_n}]|$$

for some head $\vec{\lambda\mu}'$, some $d^i_j \in \mathrm{nf_r}(u^i_j) \cap \mathrm{nf_r}(v^i_{\sigma_i(j)})$ and permutations σ_i on m_i elements. But $u^i_j, v^i_j \in \mathcal{T}(Q_i)$ and $\mathbf{ms}(u^i_j) < \mathbf{ms}(t)$ since u^i_j is a strict subterm of t. So we can apply the inductive hypothesis to each u^i_j and obtain $u^i_j = v^i_j$. Hence, $t = s$.

Subcase $R = (\lambda y.P)N$. It is the same argument as the following subcase, so we skip it.

Subcase $R = (\mu\gamma.{}_\eta|P|)N$. Then $t' = (\mu\gamma.{}_\eta|p|)[\vec{d}]$ and $s' = (\mu\gamma.{}_\eta|p'|)[\vec{d}']$ for $p, p' \in \mathcal{T}(P)$ and $[\vec{d}], [\vec{d}'] \in !\,\mathcal{T}(N)$. By confluence on $\lambda\mu^{\mathrm{r}}$ we have:

$$\mathrm{nf_r}(t) = \mathrm{nf_r}(\vec{\lambda\mu}|(\mu\gamma.\langle_\eta|p|\rangle_\gamma[\vec{d}])[\vec{u}^{\,1}]\dots[\vec{u}^{\,n}]|).$$

So there is $h_1 \in \mu\gamma.\langle_\eta|p|\rangle_\gamma[\vec{d}]$ s.t. $h \in \mathrm{nf_r}(\widetilde{h}_1)$ where: $\widetilde{h}_1 := \vec{\lambda\mu}|h_1[\vec{u}^{\,1}]\dots[\vec{u}^{\,n}]|$. Analogously we find a $h_2 \in \mu\gamma.\langle_\eta|p'|\rangle_\gamma[\vec{d}']$ s.t. $h \in \mathrm{nf_r}(\widetilde{h}_2)$, where: $\widetilde{h}_2 := \vec{\lambda\mu}|h_2[\vec{v}^{\,1}]\dots[\vec{v}^{\,n}]|$. By Lemma 4.4.48 we have $h_1, h_2 \in \mathcal{T}(\mu\gamma.({}_\eta|P|)_\gamma N)$ and so $\widetilde{h}_1, \widetilde{h}_2 \in \mathcal{T}(\vec{\lambda\mu}|(\mu\gamma.({}_\eta|P|)_\gamma N)Q_1\cdots Q_n|)$. This and the fact that $h \in \mathrm{nf_r}(\widetilde{h}_1) \cap \mathrm{nf_r}(\widetilde{h}_2)$ mean that $\widetilde{h}_1$ satisfies both the hypotheses of the inductive hypothesis. Moreover, since $t' \to_{\mu^{\mathrm{r}}} h_1 + \mathbb{T}$ for some sum $\mathbb{T}$, then $\mathrm{m}(h_1) < \mathrm{m}(t')$. And since the number of μ's is constant under μ-reduction, $\deg_\mu(t') = \deg_\mu(h_1)$. Therefore we can apply Lemma 4.4.18 and obtain:

$$\mathrm{m}(\widetilde{h}_1) = \mathrm{m}(\vec{\lambda\mu}|h_1[\vec{u}^{\,1}]\dots[\vec{u}^{\,n}]|) < \mathrm{m}(\vec{\lambda\mu}|t'\,[\vec{u}^{\,1}]\dots[\vec{u}^{\,n}]|) = \mathrm{m}(t).$$

So $\mathbf{ms}(\widetilde{h}_1) < \mathbf{ms}(t)$ and we can safely apply the inductive hypothesis obtaining $\widetilde{h}_1 = \widetilde{h}_2$. Looking at the definition of $\widetilde{h}_1, \widetilde{h}_2$, we get $h_1 = h_2$ as well as $[\vec{u}^{\,i}] = [\vec{v}^{\,i}]$ $(i = 1, \dots, n)$. But now we have:

$$\mu\gamma.\langle_\eta|p|\rangle_\gamma[\vec{d}] \ni h_1 = h_2 \in \mu\gamma.\langle_\eta|p|\rangle_\gamma[\vec{d}']$$

so Lemma 4.4.50 gives $p = p'$ and $[\vec{d}] = [\vec{d}']$, i.e. $t' = s'$. If we now look at the shape of t, s, this last information together with $[\vec{u}^{\,1}] = [\vec{v}^{\,1}], \dots, [\vec{u}^{\,n}] = [\vec{v}^{\,n}]$, precisely means $t = s$. □

The previous result was first proved by Ehrhard and Regnier in [ER08] for the λ-calculus. It is known[27] that it fails in MELL. A natural question is what is the threshold, between λ-calculus and MELL, where this property starts failing? This is a non-trivial question to which one does not have an answer yet, and it is important also because somehow linked to the possibility of defining a *coherence* on resource terms for which Taylor expansion is a maximal clique. We will mention this question in the conclusive Section 4.5.

[27] Private communication by Tortora de Falco, based on some work of Mazza, Pagani and Guerrieri.

Proposition 4.4.52. *Assumption 5 holds in λμ-calculus. That is, if* $\mathcal{T}(M) \ni t \to_{\text{base}} \mathbb{T}'$ *then there is* $N \in \lambda\mu$ *s.t.* $\mathbb{T}' \subseteq \mathcal{T}(N)$ *and* $M \to N$.

Proof. We have the three base-case reductions:

Case $t = (\mu\alpha.{}_\beta|p|)[\vec{q}]$ and $\mathbb{T}' = \mu\alpha.\langle_\beta|p|\rangle_\alpha[\vec{q}]$. So it must be $M = (\mu\alpha.{}_\beta|P|)Q$ with $p \in \mathcal{T}(P)$ and $[\vec{q}] \in !\,\mathcal{T}(Q)$. Now thanks to Lemma 4.4.48 we can take $N := \mu\alpha.({}_\beta|P|)_\alpha Q$.

Case $t = (\lambda x.p)[\vec{q}]$ and $\mathbb{T}' = p\langle[\vec{q}]/x\rangle$, or case $t = \mu\gamma.{}_\alpha|\mu\beta.{}_\eta|p||$ and $\mathbb{T}' = \mu\gamma.{}_\eta|p|\{\alpha/\beta\}$. Exactly as above, thanks to Lemma 4.4.48. □

In particular thus, from Section 4.3 we have that:

Corollary 4.4.53. . *The λμ-calculus satisfies Corollary 4.3.14.*

We want to mention that it is possible to prove Corollary 4.4.53 also by adapting the big-step reduction (Definition 3.3.22), following the way we did for λ-calculus.

Definition 4.4.54 (Big-step reduction)**.** *The* big-step reduction $\Downarrow \subseteq \lambda\mu \times \lambda\mu$ *is the contextual closure of the rules given in Figure 4.4.*

The big-step reduction for λμ-calculus has the same rules of Figure 3.2, with the following two modifications.

1- The rule for λ is replaced by the following rule:

$$\frac{t \Downarrow t' \quad \vec{\lambda\mu}|\cdot| \Downarrow \vec{\lambda\mu}'|\cdot|\{*\}}{\vec{\lambda\mu}|t| \Downarrow \vec{\lambda\mu}'|t'|\{*\}}$$

where by $\vec{\lambda\mu}|\cdot| \Downarrow \vec{\lambda\mu}'|\cdot|\{*\}$ *we mean that is* $\vec{\lambda\mu}'$ *obtained from* $\vec{\lambda\mu}'$ *by reducing all the ρ-redexes (since the ρ-reduction is confluent, there is only one such head* $\vec{\lambda\mu}'$*.)*

2- We add the following rule:

$$\frac{t \Downarrow \mu\alpha.{}_\beta|t'| \qquad u_1 \Downarrow v_1 \quad \cdots \quad u_k \Downarrow v_k \qquad w[\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t'' \qquad w \in \mu\alpha.\langle_\beta|t'|\rangle_\alpha[\vec{v}]}{t\,[\vec{u}]\,[\vec{s}^1]\cdots[\vec{s}^h] \Downarrow t''}$$

Figure 4.4: Base cases of the big-step reduction for λμ-calculus

One can straightforwardly reproduce the exact same proofs for the analogues of Lemma 3.3.24, Lemma 3.3.25 and Corollary 3.3.26.

Thus, we get again Corollary 4.4.53 (it is the analogue of Corollary 3.3.26).

In those proofs, one needs to use $\mathbf{ms}(\cdot)$ instead of $\mathbf{sz}(\cdot)$.

4.4.2 The λμ-theory $\mathrm{NF}\mathcal{T}$

By Corollary 4.3.6 we know that the equivalence $M =_\tau N$ defined by $\mathrm{NF}\mathcal{T}(M) = \mathrm{NF}\mathcal{T}(N)$ is a λμ-theory, and it is clearly non-trivial ($\mathtt{I} \neq_\tau \emptyset =_\tau \Omega$).

In λ-calculus, $=_\tau$ is the λ-theory equating Böhm trees. In particular, it is sensible (i.e. it equates all unsolvables). We will see (Corollary 4.4.61) that in our case it is still sensible.

Definition 4.4.55. *A $\lambda\mu$-term M is a* head normal form *(*hnf *for short) iff there are no ρ-redexes in its head (remember Lemma 4.4.2) and it has a head variable. We define the exact same notion for $\lambda\mu^{\mathrm{r}}$.*

Definition 4.4.56. *The* head reduction *is the partial function $\mathrm{H} : \lambda\mu \to \lambda\mu$ obtained defining $\mathrm{H}(M)$ via the following algorithm:*

1. *ρ-reduce the leftmost ρ-redex in the head of M, if there is one*
2. *otherwise, $\lambda\mu$-reduce the head redex of M, if there is one*
3. *otherwise, $\mathrm{H}(M)$ is not defined.*

$\mathrm{H}(M)$ is not *defined iff M is a hnf. We say that* head reduction starting on M terminates *iff there is a (necessarily unique) $n \geq 0$ s.t. $\mathrm{H}^n(M)$ is a hnf. Here we mean as usual that $\mathrm{H}^0(M) := M$.*

We extend these definitions to resource terms by setting $\mathrm{H}(t) := \emptyset$ whenever t is a hnf. Moreover, we set $\mathrm{H}^0(t) := t \in 2\langle\Lambda^{\mathrm{r}}\rangle$ and, for $n \geq 0$[28]*:*

$$\mathrm{H}^{n+1}(t) := \sum_{t_1 \in \mathrm{H}(t)} \sum_{t_2 \in \mathrm{H}(t_1)} \cdots \sum_{t_{n+1} \in \mathrm{H}(t_n)} t_{n+1} \quad \in 2\langle\Lambda^{\mathrm{r}}\rangle.$$

It is clear that we have $\mathrm{H}^1(t) = \mathrm{H}(t)$, and that $\mathrm{H}^{n+1}(t) = \sum_{t' \in \mathrm{H}(t)} \mathrm{H}^n(t')$.

Remark 4.4.57. *Observe that if t is a hnf (in particular, if it is normal), then $t \not\to_{\mathrm{r}} \mathrm{H}(t)$, because by definition $\mathrm{H}(t) = 0$. On the contrary, if t is not a hnf, then by definition of H (it reduces the leftmost redex), we have $t \to_{\mathrm{r}} \mathrm{H}(t)$. In conclusion, we have: t is not hnf iff $t \to_{\mathrm{r}} \mathrm{H}(t)$. Observe that of course it is possible that $t \to_{\mathrm{r}} \mathrm{H}(t) = 0$, for instance take $t = (\lambda x.x[x])[y]$.*

Lemma 4.4.58. *If s only contains empty bags (if any) and $s \in \mathrm{nf}_{\mathrm{r}}(t)$, then $s \in H^n(t)$ for some $n \geq 0$.*

Proof. If t is a hnf, we can easily conclude: since any eventual bag of s is empty, and since reductions cannot erase non-empty bags (because *linear*), the fact that $s \in \mathrm{nf}_{\mathrm{r}}(t)$ entails that already t contains only empty bags. But in a hnf the reduction can only take place inside some bag, so it actually must be $s = t$. Therefore, $s \in s = t = \mathrm{H}^0(t)$ and we are done taking $n := 0$.

So we are left with the case in which t is *not* hnf. In this case we know that $t \to_{\mathrm{r}} \mathrm{H}(t)$. By confluence, $\mathrm{H}(t) \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(t)$. But since $s \in \mathrm{nf}_{\mathrm{r}}(t)$, there is some $t_1 \in \mathrm{H}(t)$ s.t. $s \in \mathrm{nf}_{\mathrm{r}}(t_1)$. Now we can reason as in the beginning, splitting in two cases: either t_1 is a hnf, in which case we reason exactly as in the first four lines of the proof: by linearity we get $s = t_1 \in \mathrm{H}(t)$, and we are done taking $n := 1$. Or t_1 is not a hnf. In this case we can reason again as before, obtaining a $t_1 \to_{\mathrm{r}} \mathrm{H}(t_1)$, $\mathrm{H}(t_1) \twoheadrightarrow_{\mathrm{r}} \mathrm{nf}_{\mathrm{r}}(t_1)$ and a $t_2 \in \mathrm{H}(t_1)$ s.t. $s \in \mathrm{nf}_{\mathrm{r}}(t_2)$. The reader can look at Figure 4.5. We can keep going with the same splits: if t_2 is hnf, we are done taking $n := 2$; if t_2 is not hnf, we obtain a new t_3 as before. Now, the generation of such a new t_{i+1} from the previously generated t_i cannot continue forever: we claim that there must be some $m \in \mathbb{N}$ for which t_m is hnf. If this is the case, the proof is concluded because we can take $n := m$ as already mentioned. To see that such an m does exist, one simply remarks that at each time we have $t_i \to_{\mathrm{r}} \mathrm{H}(t_i)$ and $t_{i+1} \in \mathrm{H}(t_i)$. But the well-founded measure $\mathrm{m}(\cdot)$ is strictly decreasing along reductions, which precisely means that we obtain the strictly decreasing sequence:

$$\mathrm{m}(t) > \mathrm{m}(t_1) > \mathrm{m}(t_2) > \cdots$$

Therefore, it must terminate (because the order is well-founded) on some $\mathrm{m}(t_m)$, for some $m \in \mathbb{N}$ (in Figure 4.5 we have the case $m = 3$), and we are done as already explained. □

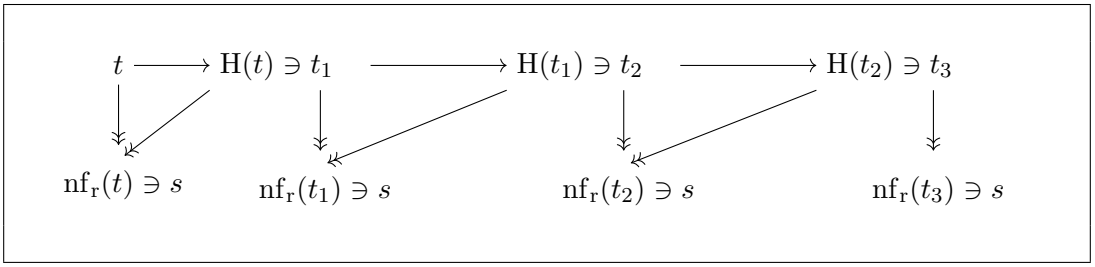


Figure 4.5: Schema of the proof of Lemma 4.4.58

Set $\mathrm{H}(\mathcal{T}(M)) := \bigcup_{t\in\mathcal{T}(M)} \mathrm{H}(t) \subseteq \lambda\mu^{\mathrm{r}}$. The following lemma states that there is a commutation between the Taylor expansion and the head reduction.

Lemma 4.4.59 (Commutation between Taylor expasion and Head-reduction)**.** *If $M \in \lambda\mu$ with* $\mathrm{H}(M)$ *defined, we have:*

$$\mathcal{T}(\mathrm{H}(M)) = \mathrm{H}(\mathcal{T}(M)).$$

Proof. We have to show that $\mathcal{T}(\mathrm{H}(M)) = \bigcup_{t\in\mathcal{T}(M)} \mathrm{H}(t)$. By Lemma 4.4.2 we know that:

$$M = \lambda\vec{x_1}.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x_k}.\mu\alpha_k.{}_{\beta_k}|RQ_1\dots Q_n||$$

with the condition that either there is a ρ-redex in the head of M, or there is no such ρ-redex and R is *not* a variable (thus R is either a λ-redex or a μ-redex). We have just said in a different fashion that M is *not* a hnf. Now let us show the two inclusions.

($\subseteq$). Take $s \in \mathcal{T}(\mathrm{H}(M))$. We have three cases:

Case in which there is a ρ-redex in the head of M. Therefore there is also a leftmost ρ-redex $\dots{}_{\beta_i}|\mu\alpha_{i+1}.\dots|$ in the head of M. Then:

$$\mathrm{H}(M) = \lambda\vec{x_1}.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x_i}.\mu\alpha_i.{}_{\beta_{i+1}}|\dots\lambda\vec{x}_k\mu\alpha_k.{}_{\beta_k}|R\vec{Q}||\{\beta_i/\alpha_{i+1}\}|.$$

So $s = \lambda\vec{x_1}.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x_i}.\mu\alpha_i.{}_{\beta_{i+1}}|\dots\lambda\vec{x}_k\mu\alpha_k.{}_{\beta_k}|r[\vec{q^1}]\dots[\vec{q^n}]||\{\beta_i/\alpha_{i+1}\}|$ for $r \in \mathcal{T}(R)$ and $[\vec{q^i}] \in !\,\mathcal{T}(Q_i)$. But $\mathcal{T}(M) \ni \lambda\vec{x_1}.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x_k}.\mu\alpha_k.{}_{\beta_k}|r[\vec{q^1}]\dots[\vec{q^n}]|| =: t$, and thus $s = \mathrm{H}(t)$, i.e. $s \in \bigcup_{t\in\mathcal{T}(M)} \mathrm{H}(t)$.

Case there are no ρ-redexes and $R = (\mu\gamma.{}_\eta|P|)D$. Then $\mathrm{H}(M) = \vec{\lambda\mu}.\ |\ (\mu\gamma.({}_\eta|P|)_\gamma D)\vec{Q}\ |$. So by Lemma 4.4.48 $s \in \vec{\lambda\mu}.\ |\ (\mu\gamma.\langle{}_\eta|p|\rangle_\gamma[\vec{d}])[\vec{q^1}]\dots[\vec{q^n}]\ |$ for a $p \in \mathcal{T}(P)$, $[\vec{d}] \in !\,\mathcal{T}(D)$ and $[\vec{q^i}] \in !\,\mathcal{T}(Q_i)$. So $s \in \mathrm{H}(t)$, with $t := \lambda\vec{x_1}.\mu\alpha_1.{}_{\beta_1}|\dots\lambda\vec{x_k}.\mu\alpha_k.{}_{\beta_k}|(\mu\gamma.{}_\eta|p|)[\vec{d}][\vec{q^1}]\dots[\vec{q^n}]|| \in \mathcal{T}(M)$, i.e. $s \in \bigcup_{t\in\mathcal{T}(M)} \mathrm{H}(t)$.

Case there are no ρ-redexes and $R = (\lambda x.P)D$. Exactly as above using Lemma 4.4.48.

($\supseteq$). One can follow the exact same kind of argument as before: the fact that Taylor expansion preserves the structure of the term, plus Lemma 4.4.48, is what makes us able to transport one step of the head reduction from terms to resource terms. □

[28] Here we write the sum symbol just to stress the fact that those are finite sums, but remember that they are just sets (so the sum is the union).

The following proposition shows that in $\lambda\mu$-calculus one can define a notion of *solvability* which is analogous to the one in λ-calculus.

Proposition 4.4.60. *For $M \in \lambda\mu$, the following are equivalent:*

1. *$M =_{\lambda\mu\rho} H$ with H hnf*
2. *Head reduction starting on M terminates*
3. $\mathrm{NF}\mathcal{T}(M) \neq \emptyset$.

Proof. (1⇒2). By confluence M and H have a common $\lambda\mu\rho$-redex M_0. Since H is a hnf, M_0 is too. Let s_0 be the unique resource $\lambda\mu$-term in $\mathcal{T}(M_0)$ s.t. all its bags (if any) are empty. This term clearly exists. Note that, by construction, s_0 is $\lambda\mu\rho$-normal. By repeatedly applying Proposition 4.4.49 one can check that we obtain an $s \in \mathcal{T}(M)$ s.t. $s_0 \in \mathrm{nf_r}(s)$. Now, by Lemma 4.4.58, $s_0 \in \mathrm{H}^n(s)$ for some $n \geq 0$. By repeatedly applying Lemma 4.4.59, we find that $s_0 \in \mathrm{H}^n(\mathcal{T}(M)) = \mathcal{T}(\mathrm{H}^n(M))$. And being s_0 a hnf, so it must be $\mathrm{H}^n(M)$.

(2⇒3). We know that $=_\tau$ is a $\lambda\mu$-theory, so if $M \twoheadrightarrow H$ with H hnf (this is the hypothesis that head-reduction starting from M terminates), we have $M =_\tau H$, and it is immediate that $\mathrm{NF}\mathcal{T}(H) \neq 0$ because H is hnf.

(3⇒1). If $\mathrm{NF}\mathcal{T}(M) \neq 0$ there is $t \in \mathcal{T}(M)$ s.t. $\mathrm{nf_r}(t) \neq 0$. By Corollary 4.4.53, $M \twoheadrightarrow N$ for some $N \in \lambda\mu$ s.t. $\mathrm{nf_r}(t) \subseteq \mathcal{T}(N)$. So $\mathcal{T}(N)$ contains at least a hnf, and thus N must be a hnf too. □

We call $M \in \lambda\mu$ *solvable* iff it satisfies any of the previous equivalent conditions of Lemma 4.4.60. Otherwise, M is called *unsolvable*.

Corollary 4.4.61. *The $\lambda\mu$-theory $=_\tau$ is* sensible *(that is, it equates all unsolvable terms).*

Proof. Immediate by Lemma 4.4.60. □

4.4.3 Stability and Perpendicular Lines Property

Stability We have all the necessary assumptions in order to immediately have that the Stability property holds in $\lambda\mu$-calculus.

Remembering the notation of Definition 4.3.15, Theorem 4.3.16 takes the form:

Theorem 4.4.62 (Stability)**.** *Let C be an n-ary $\lambda\mu$-context and fix non-empty bounded $\mathcal{X}_1, \ldots, \mathcal{X}_n \subseteq \lambda\mu^{\mathrm{r}}$. For all $M_1, \ldots, M_n \in \lambda\mu$ s.t. $M_i =_\tau \bigcap_{N \in \mathcal{X}_i} N$ (for $i = 1, \ldots, n$) we have:*

$$C(\!|M_1, \ldots, M_n|\!) =_\tau \bigcap_{N_1 \in \mathcal{X}_1} \cdots \bigcap_{N_n \in \mathcal{X}_n} C(\!|N_1, \ldots, N_n|\!).$$

We immediately obtain the non implementability of the following parallel-or, for which we use the usual encoding of couples as $(M, N) := \lambda z.zMN$.

Corollary 4.4.63 (No parallel-or in $\lambda\mu$-calculus)**.** *There is* no *$\mathtt{Por} \in \lambda\mu$ s.t. for all $M, N \in \lambda\mu$,*

$$\begin{cases} \mathtt{Por}(M, N) & =_\tau & \mathtt{True} & \text{if } M \neq_\tau \Omega \text{ or } N \neq_\tau \Omega \\ \mathtt{Por}(M, N) & =_\tau & \Omega & \text{if } M =_\tau N =_\tau \Omega. \end{cases}$$

Proof. Exactly the same already shown for λ-calculus. □

The perpendicular Lines Property We already discussed the PLP in the setting of λ-calculus. We are going to prove that the exact same statement holds in λμ-calculus, that is, if a term $\lambda z_1 \dots z_n.F \in \lambda\mu$, seen as the function $\vec{M} \in \lambda\mu^n/_{=_\tau} \longrightarrow (\lambda\vec{z}.F)\vec{M} \in \lambda\mu/_{=_\tau}$, is constant on n "perpendicular lines", then it is constant everywhere. Another way to see it is as a weak form of sequentiality. The proof is a simple readaptation of the one already given for λ-calculus.

The following lemma (and its proof) is the analogue of Lemma 3.3.12.

Lemma 4.4.64. *If $\vec{\lambda\mu}|.xM_1 \cdots M_k| =_\tau \vec{\lambda\mu}|.yN_1 \cdots N_{k'}|$ then $n = n'$, $k = k'$, $x = y$ and $M_i =_\tau N_i$ for all $i = 1, \dots, k$.*

The following is the version of Lemma 3.3.12 for the current framework, whose proof follows the same lines.

Lemma 4.4.65. *Fix $\vec{z} := z_1, \dots z_n$ distinct variables and let $t \in \lambda\mu^{\mathrm{r}}$. Suppose that:*

i. $\mathrm{nf}_{\mathrm{r}}(t) \neq 0$

ii. there is $F \in \lambda\mu$ s.t. $t \in \mathcal{T}(F)$

iii. there are $\{M_{ij}\}_{1 \leq i \neq j \leq n} \subseteq \lambda\mu$ s.t. the function mapping $\vec{M} \in \lambda\mu^n/_{=_\tau}$ to $(\lambda\vec{z}.F)\vec{M} \in \lambda\mu/_{=_\tau}$ is constant on the following "perpendicular lines" of $\lambda\mu^n/_{=_\tau}$:

$$\begin{array}{l} l_1 = \{(Z,\ \ M_{12},\ \dots\dots,\ M_{1n}) \mid Z \in \lambda\mu\} \\ l_2 = \{(M_{21},\ Z,\ \ \dots\dots,\ M_{2n}) \mid Z \in \lambda\mu\} \\ \qquad\qquad \ddots \\ l_n = \{(M_{n1},\ \dots,\ M_{n(n-1)},\ Z) \mid Z \in \lambda\mu\}. \end{array} \tag{4.1}$$

Then $\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0$.

Proof. Induction on the size $\mathbf{ms}(t)$ of $t \in \lambda\mu^{\mathrm{r}}$.

Case $\mathbf{ms}(t) = (1, 0, 1)$. Then t is a variable (Corollary 4.4.19). If $t = z_i$ for some i then the i-th line of (4.1) gives the contradiction:

$$N_i =_\tau (\lambda\vec{z}.z_i)M_{i1} \cdots M_{i(i-1)} Z M_{i(i+1)} \cdots M_{in} =_\tau Z$$

for all $Z \in \lambda\mu$. Hence, it must be $\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0$.

Case $\mathbf{ms}(t) > (1, 0, 1)$. By (i) there is $u \in \mathrm{nf}_{\mathrm{r}}(t)$. Being u normal, it has shape: $u = \vec{\lambda\mu}|y[\vec{u}^1] \dots [\vec{u}^m]|$ for some $m \geq 0$, some variable y, some normal bags $[\vec{u}^j]$, and where we have shorten, as before, a series $\lambda\vec{x}_1\mu\alpha_{1\cdot\beta_1}| \dots \lambda\vec{x}_k\mu\alpha_{k\cdot\beta_k}| \dots ||$ of λ and μ abstraction by just $\vec{\lambda\mu}|\dots|$. By (ii) $t \in \mathcal{T}(F)$, so that by Theorem 4.3.14 there is $Q \in \lambda\mu$ s.t. $F \twoheadrightarrow_{\lambda\mu\rho} Q$ and $u \in \mathcal{T}(Q)$. So Q must have shape: $Q = \vec{\lambda\mu}|yQ_1 \cdots Q_m|$ for some Q_j's in $\lambda\mu$ s.t. $[\vec{u}_j] \in !\mathcal{T}(Q_j)$ for all $j = 1, \dots, m$. Now there are two possibilities: either $y = z_i$ for some $i = 1, \dots, n$, either $y \neq z_i$ for all i.

Suppose $y = z_i$. Then, for $\vec{q} := q_1, \dots, q_m$ fresh variables, we can chose $Z := \lambda\vec{q}.\mathtt{True} \in \lambda\mu$ (or $Z := \mathtt{True}$ if $m = 0$) in the i-th line l_i of (4.1), and since by (iii) $\lambda\vec{z}.F$ is constant (mod $=_\tau$) on l_i, we can compute its value as:

$$\begin{array}{rcl} (\lambda\vec{z}.F)M_{i1} \cdots M_{i(i-1)}\,(\lambda\vec{q}.\mathtt{True})\,M_{i(i+1)} \cdots M_{in} & =_\tau & Q\{M_{i1}/z_1, \dots, (\lambda\vec{q}.\mathtt{True})/z_i, \dots, M_{in}/z_n\} \\ & =_\tau & \vec{\lambda\mu}|(\lambda\vec{q}.\mathtt{True})\widetilde{Q_{i1}} \cdots \widetilde{Q_{im}}| \\ & =_\tau & \vec{\lambda\mu}|\mathtt{True}| \end{array}$$

where we set $\widetilde{Q_{ij}} := Q_j\{M_{i1}/z_1, \dots, (\lambda\vec{q}.\mathtt{True})/z_i, \dots, M_{in}/z_n\}$. The first equality holds because $F \twoheadrightarrow_{\lambda\mu\rho} Q$ and $=_\tau$ is finer than $=_{\lambda\mu\rho}$ (Corollary 4.3.6), and the second equality

holds because $y = z_i$. In the same way, choosing $Z := \lambda\vec{q}.\texttt{False} \in \lambda\mu$ in l_i we find that the value (mod $=_\tau$) of $\lambda\vec{z}.F$ on l_i is $\vec{\lambda\mu}|\texttt{False}|$. But this is impossible because $\texttt{True} \neq_\tau \texttt{False}$.

Therefore, it must be $y \neq z_i$ for all i. Note that wlog $m \geq 1$ (indeed if $m = 0$, from the fact that $y \neq z_i$ for all i we already get $\deg_{z_i}(u) = 0$ and, as $u \in \mathrm{nf}_{\mathrm{r}}(t)$ and in $\lambda\mu^{\mathrm{r}}$ one cannot erase non-empty bags, we are done). Now fix $i \in \{1, \ldots, n\}$ and $Z', Z'' \in \lambda\mu$. Similarly as before, choosing $Z := Z'$ in l_i and using what we found so far, putting $Q'_{ij} := Q_j\{M_{i1}/z_1, \ldots, Z'/z_i, \ldots, M_{in}/z_n\}$, since $\lambda\vec{z}.F$ is constant (mod $=_\tau$) on l_i, we can compute its value as:

$$\begin{array}{rcl}(\lambda\vec{z}.F)M_{i1}\ldots M_{i(i-1)}Z'M_{i(i+1)}\ldots M_{in} & =_\tau & Q\{M_{i1}/z_1, \ldots, Z'/z_i, \ldots, M_{in}/z_n\} \\ & =_\tau & \vec{\lambda\mu}|yQ'_{i1}\ldots Q'_{im}|\end{array}$$

where the last equality holds since y is *not* one of the z_i's. Choosing Z'' instead of Z' and putting Q''_{ij} the same as Q'_{ij} but with Z'' instead of Z', one has that the value (mod $=_\tau$) of $\lambda\vec{z}.F$ on l_i is $\vec{\lambda\mu}|yQ''_{i1}\ldots Q''_{im}|$. So we have $\vec{\lambda\mu}|yQ'_{i1}\ldots Q'_{im}| = \vec{\lambda\mu}|yQ''_{i1}\ldots Q''_{im}|$, and Lemma 4.4.64 entails that:

$$\begin{array}{c}Q'_{i1} =_\tau Q''_{i1} \\ \vdots \\ Q'_{im} =_\tau Q''_{im}.\end{array}$$

But by construction it is:

$$Q'_{ij} =_\tau (\lambda\vec{z}.Q_j)M_{i1}\ldots M_{i(i-1)}Z'M_{i(i+1)}\ldots M_{in}$$

$$Q''_{ij} =_\tau (\lambda\vec{z}.Q_j)M_{i1}\ldots M_{i(i-1)}Z''M_{i(i+1)}\ldots M_{in}.$$

So if we remember that Z', Z'' were generic in $\lambda\mu$, the previous equalities $Q'_{ij} = Q''_{ij}$ precisely say that $\lambda\vec{z}.Q_j$ is constant on the line l_i. And since this holds for all $i = 1, \ldots, n$, we have just found that $\lambda\vec{z}.Q_j$ satisfies (iii). And since we have equalities $Q'_{ij} = Q''_{ij}$ for all $j = 1, \ldots, m$, we have that each $\lambda\vec{z}.Q_1, \ldots, \lambda\vec{z}.Q_k$ satisfies (iii). We can now comfortably apply the induction hypothesis on any $s \in [\vec{u}^{\,j}]$. In fact, as $[\vec{u}^{\,j}]$ is normal, $\mathrm{nf}_{\mathrm{r}}(s) \neq 0$, i.e. (i); as $[\vec{u}^{\,j}] \in !\,\mathcal{T}(Q_j)$, we have $s \in \mathcal{T}(Q_j)$, i.e. (ii); and we just found that $\lambda\vec{z}.Q_j$ satisfies (iii); finally, s is a strict subterm of $u \in \mathrm{nf}_{\mathrm{r}}(t)$, thus (Corollary 4.4.19) $\mathbf{ms}(s) < \mathbf{ms}(u) \leq \mathbf{ms}(t)$. Therefore, the inductive hypothesis gives $\deg_{z_1}(s) = \cdots = \deg_{z_n}(s) = 0$. Since this is true for all s in all $[\vec{u}^{\,j}]$, $j = 1, \ldots, m$, we get $\deg_{z_1}(u) = \cdots = \deg_{z_n}(u) = 0$. And now $u \in \mathrm{nf}_{\mathrm{r}}(t)$ entails $\deg_{z_1}(t) = \cdots = \deg_{z_n}(t) = 0$. □

Theorem 4.4.66 (Perpendicular Lines Property)**.** *Suppose that for some fixed* $\{M_{ij}\}_{1\leq i\neq j\leq n}$, $\{N_i\}_{1\leq i\leq n} \subseteq \lambda\mu$*, the system of equations:*

$$\left\{\begin{array}{llcl}(\lambda z_1\ldots z_n.F) & Z\ M_{12}\ \ldots\ldots\ M_{1n} & =_\tau & N_1 \\ (\lambda z_1\ldots z_n.F) & M_{21}\ Z\ \ldots\ldots\ M_{2n} & =_\tau & N_2 \\ & \ddots & \vdots & \\ (\lambda z_1\ldots z_n.F) & M_{n1}\ \ldots\ M_{n(n-1)}\ Z & =_\tau & N_n\end{array}\right.$$

holds for all $Z \in \lambda\mu$*. Then:*

$$(\lambda z_1\ldots z_n.F)Z_1\ldots Z_n =_\tau N_1$$

for all $Z_1, \ldots, Z_n \in \lambda\mu$*.*

Proof. It follows from Lemma 4.4.65 as done in [BM20]. □

PLP immediately entails the non implementability of the following parallel-or, a result which is known as folklore via arguments involving stable models: here we proved it solely via Taylor expansion.

Corollary 4.4.67 (No Parallel-or in $\lambda\mu$-calculus)**.** *There is* no $\mathtt{Por}' \in \lambda\mu$ *s.t. for all* $Z \in \lambda\mu$ *one has:*

$$\begin{cases} \mathtt{Por}'\ \mathtt{True}\, Z =_\tau \mathtt{True} \\ \mathtt{Por}'\ Z\ \mathtt{True} =_\tau \mathtt{True} \\ \mathtt{Por}'\ \mathtt{False}\, \mathtt{False} =_\tau \mathtt{False}. \end{cases}$$

4.5 Conclusive comments

In [Lau04] Laurent studies the mathematics of (untyped) $\lambda\mu$-calculus via its denotational semantics; in this chapter we did it by developing a theory of program approximation based on Linear Logic resources. In particular, we proved that the approximation theory satisfies the "non-interference property", that it induces a sensible $\lambda\mu$-theory, and that it can be used as a tool in order to obtain the Stability property, the Perpendicular lines property, and thus the impossibility of parallel computations in the language. A first natural question immediately arises:

1. Can Taylor expansion allow to find new interesting properties that are *not* satisfied by the λ-calculus, but that are enjoyed by the $\lambda\mu$-calculus due to the presence of continuations (such as `callcc`)? It is usually a difficult task to conjecture *new* properties (interesting from a mathematical or programming point of view) of a language, and we do not have an answer to such question.

For future investigations, we believe that it would be interesting to integrate this approach with the *differential* extension of $\lambda\mu$-calculus defined in [Vau07b], in order to explore quantitative properties as well.

The following two questions are maybe the most significant:

2. Does it makes sense to introduce Böhm trees for the $\lambda\mu$-calculus? For instance, for the call-by-value λ-calculus, the Taylor expansion has provided in [KMP20] invaluable guidance for finding a meaningful notion of trees satisfying Ehrhard and Regnier's *commutation formula*; the same methodology could maybe be applied here. However, in [DP01] it is shown that $\lambda\mu$-calculus does *not* enjoy Böhm's separation property. David and Py's counterexample could hence be an indication that, instead, Böhm trees are not a "good" notion for $\lambda\mu$-calculus. The best way of proceeding would be, in that case, to consider Saurin's $\Lambda\mu$-calculus [Sau12] (see the end of Section 4.2). It was introduced precisely to satisfy Böhm's property and, as a matter of fact, in [Sau12] Saurin proposes a definition of Böhm trees for his $\Lambda\mu$-calculus. This certainly constitutes an interesting starting point.

3. Does $\Lambda\mu$-calculus admit all the constructions of the present chapter? On one hand, many constructions we did in this chapter seem possible also in Saurin's calculus (for instance, Definition 4.4.6, but remember also the discussion in the line just above that definition), on the other hand we used the fact that the number of μ's in a term is the same as named subterms, for instance, in Remark 4.4.9. In general, one could wonder which one, between $\lambda\mu$ and $\Lambda\mu$, should be the "canonical lambda-mu-calculus": from a proof-theoretical perspective $\lambda\mu$-calculus precisely corresponds to Parigot's CD-derivations, but $\Lambda\mu$-calculus satisfies more wishful properties (Böhm separation). Moreover, in [Sau10], Saurin adapts usual techniques of λ-calculus to $\Lambda\mu$-calculus: he studies the notion of solvability, proves a standardization theorem and studies more in detail the notion of Böhm trees. A very interesting future direction of research would be, hence, to develop a theory of resource approximation for Saurin's calculus, and study its relation with his theory of "Böhm approximation".

In any case, we regard the fact that the Taylor expansion works so nicely in $\lambda\mu$-calculus – and this regardless of a notion of Böhm trees – as a *a posteriori* confirmation of the high power of this form of approximation.

There are at least two other interesting points in relation with strictly related areas:

4. In order to perform a deeper logical analysis, one should consider translations into Linear Logic. It is known from [Lau03] that $\lambda\mu$-calculus translates into polarized proof nets. Taylor expansion for proof-nets is possible, but the construction can be complex: in fact one of the interests in *directly* defining a Taylor expansion for a certain "λ-calculus style" programming language (as we did for $\lambda\mu$-calculus, and as one does for λ-calculus) is precisely to avoid that complexity. In our case we have just shown that, at the end of the day, the theory of resource approximation for $\lambda\mu$-calculus can be developed with essentially the same methodology as in λ-calculus. This leads to asking what makes a Taylor expansion "easy", and should be considered in relation to the already mentioned "non-interference property" (Theorem 4.4.51) and the possibility of the existence of a coherence relation for which $\mathcal{T}(M)$ is a clique. This motivates an investigation of the complexity of the definition of a Taylor expansion of a programming language/proof system, which may be related to the notion of connectedness of proof-nets, whose study starts in [GPdF16]. Such question should be considered in relation with the so-called problem of the "inversion of Taylor expansion" [GPdF19, GPdF20] and the problem of "injectivity" of denotational models (in particular, the relational one) for Linear Logic.

5. As we have mentioned in Section 4.2, the $\lambda\mu$-calculus is not the only way of extending the Curry-Howard correspondence to classical logic. Another notable one is the already mentioned *Krivine's classical realizability*, which is a "machine to extract computational content from proofs + axioms" (for almost all mathematics, such as the one formalizable in ZF+AC, see [Kri20]). There are translations between $\lambda\mu$-calculus and Krivine's calculus, and *vice-versa*, so developing differential tools on $\lambda\mu$-calculus might be related to developing them in Krivine's setting: can one move the resource approximation and Taylor expansion from $\lambda\mu$-calculus to Krivine's realizability? What do the mathematical properties we found for $\lambda\mu$-calculus (such as Stability and PLP) say for Krivine's realizability? The same questions hold for the other systems extending the correspondence to classical logic. We believe that an investigation of those questions would be of a major interest.

Chapter 5

A miscellany of meditations

5.1 Plan of the Chapter

This final chapter is more peculiar than the rest of the manuscript and presents some reflections more than some results. It is organized in two geometrical "meditations" and one philosophical one, for a total of three sections:

1. Section 5.2.1 is a brief methodological discussion on the nature of the mathematics of λ-calculus, in particular with respect to the Scott topology.

2. Section 5.2.2 contains the presentation of a possible direction of future research, which is still incomplete and we only started to explore. The main observation is that the exact same notion of *coherent space*, very well known in linear logic, exists also in combinatorics/geometry under a different name. In particular, topologists study this kind of structures (the abstract simplicial complexes, in general) by means of (simplicial) homology: whence the idea of doing the same, but from a logically oriented perspective. As we will explain, this raises one main non-trivial question, which we tried (but did not clearly succeed) to resolve.

3. Section 5.3 is not of a mathematical content but of a philosophical one. We will present our viewpoint on the question of the "foundations of mathematics" – the question that in some sense gave birth to mathematical logic. Our reflection is mainly influenced by the crucial insights that computer science has made possible, and by the vision of Girard on logic.

5.2 Geometrical meditations

5.2.1 What about topology for λ-calculus ?

We want to discuss, in this section, the particular status of λ-calculus as a mathematical object. Indeed, the kind of mathematics one develops in this discipline is very different from the "standard" one[1]. Thus, it is sometimes a misconception that λ-calculus does not enjoy a rich mathematical theory. This manuscript itself, which only considers a very small piece of the great number of works in λ-calculus, together with areas such as combinatorial algebras, λ-theories, denotational models etc, show that this misconception is false.

Before further doing, let us give in the following pages some examples showing in which sense the mathematics related to it is particular.

[1] A situation somehow annoying, for the "working λ-calculist" as well as for the popularization of the discipline.

A first example is what Barendregt in [Bar84] calls a "pathological algebraic structure". He refers to the fact that term-algebras of λ-theories never enjoy some natural properties "standard" algebraic object usually satisfy; namely, associativity of the product, and recursivity.

To that, we would add also another kind of "pathological" situation: one can associate each λ-theory with a group, called its *Church group*, and always in [Bar84, Chapter 21] it is proved that, for the more standard λ-theories[2] these groups all trivialize, and for the more complex λ-theories[3] these groups all come from some difficult constructions, which makes it not clear what one could use them for[4]. So λ-calculus presents the unusual situation of an object which one is not able to study with the usual techniques (it is not associative nor recursive) and with which one is not even able to associate an interesting algebraic object (usually, a group).

There is another situation in which one is very tempted to consider some groups associated with an object, and this is when one has a topological space (the groups being the homotopy groups). Now, it is well known (see for instance [Bar84]) that Λ can be endowed with a non-trivial topology, called the Scott topology[5]. This is possible since the set $\mathfrak{B}$ of Böhm-like-trees is a "coherent algebraic cpo" (see [AC98] for these known notions), and thus comes with its Scott topology of cpo's. One can immediately transfer it to Λ: the *Scott topology* on Λ is the smallest one that makes the map:

$$\mathrm{BT} : M \in \Lambda \to \mathrm{BT}(M) \in \mathfrak{B}$$

continuous (where the cpo[6] $\mathfrak{B}$ is endowed with its Scott topology).

The opens of the topology are thus the sets of shape:

$$\mathrm{BT}^{-1}(\mathcal{O} \cap \mathfrak{B}_\Lambda)$$

for $\mathcal{O}$ open in $\mathfrak{B}$. For instance, the set $\mathrm{SOL} \subseteq \Lambda$ of the solvables is open, since $\mathrm{SOL} = \mathrm{BT}^{-1}\{A \in \mathfrak{B}_\Lambda \mid A \neq \bot\}$ and $\{A \in \mathfrak{B} \mid A \neq \bot\}$ is open in $\mathfrak{B}$.

It is known that Scott-topology on Λ is not T_0. This is simply because any two terms $M \neq N$ s.t. $\mathrm{BT}(M) = \mathrm{BT}(N)$ cannot be separated by an open. When one has a non-T_0 space X, one usually wants to consider a T_0 space which is "essentially" the same as X. This is always possible by performing the simple *Kolmogorov quotient* $\mathrm{KQ}(X)$ of X, which is defined as the quotient space of X under the identification of all the points which have exactly the same open neighbours[7]. In the Scott-topology on Λ, indistinguishable terms are exactly those who have the same Böhm trees, so $\mathrm{KQ}(\Lambda) = \Lambda/_{=_\mathcal{B}}$. It is worth adding here that not only $\Lambda/_{=_\mathcal{B}}$ is in bijection with $\mathfrak{B}_\Lambda$ – the latter has to be seen as the set of the canonical representatives of the equivalence classes of the former – but one can actually see by a simple computation that they are homeomorphic. In particular, thus, $\mathfrak{B}_\Lambda$ is T_0 and Λ is *not* homeomorphic to $\mathfrak{B}_\Lambda$.

However, the obtained space still fails to be, e.g. Hausdorff. This is another kind of pathological situation, intended as being far from the usual spaces one is used to.

Remark 5.2.1. *As a side remark, notice that using the Continuity theorem 3.5.34, together with the two following general properties of Kolmogorov quotients:*

[2] Namely, for $=_\lambda, \mathcal{H}$ and $=_\mathcal{B}$.

[3] Namely, the Hilbert-Post completion $\mathcal{H}^*$ of $\mathcal{H}$, or the ones obtained from $=_\lambda$ and $\mathcal{H}$ by considering the η or ω-rules.

[4] To the best of our knowledge, these groups do not constitute nowadays an active topic of research.

[5] Actually, at least two: the Visser topology, and the Scott topology – which Barendregt calls the "tree topology",but we will lighten terminologies and call it "Scott". Here we only consider the Scott one, because it is the most useful one.

[6] Equivalently, one can substitute $\mathfrak{B}$ with $\mathfrak{B}_\Lambda$ (equipped with the subspace topology) in the definition.

[7] Obviously $\mathrm{KQ}(X)$ is T_0 by construction. There is also a bijection between the topology on $\mathrm{KQ}(X)$ and that on X, so that the spaces are really "essentially the same".

1. *for topological spaces X, Y and continuous $f : X \to Y$, if $x \in X$ and $y \in X$ are identified in* $\mathrm{KQ}(X)$ *then also $f(x)$ and $f(y)$ are identified in* $\mathrm{KQ}(Y)$.

2. $\mathrm{KQ}(X_1 \times \cdots \times X_k) = \mathrm{KQ}(X_1) \times \cdots \times \mathrm{KQ}(X_k)$.

one finds again the Monotonicity of Böhm trees on multi-hole contexts[8]*. In fact by Continuity of contexts, one can apply* (1) *to $C : \Lambda \times \cdots \times \Lambda \to \Lambda$ and obtain that if $(M_1, \dots, M_k)$ is identified to $(N_1, \dots, N_n)$ in* $\mathrm{KQ}(\Lambda \times \cdots \times \Lambda) = \Lambda/_{=_\mathcal{B}} \times \cdots \times \Lambda/_{=_\mathcal{B}}$*, then $C(\!|M_1, \dots, M_n|\!)$ is identified to $C(\!|N_1, \dots, N_n|\!)$ in $\Lambda/_{=_\mathcal{B}}$.*

One can give the following notion.

Definition 5.2.2. *A point x in a topological space X is a* compactification point *iff x admits exactly one open neighbourhood in X (which, thus, must be all X).*

The following are the typical topological features of Λ.

Proposition 5.2.3. 1. *The sets $O_{M,k} := \{N \in \Lambda \text{ s.t. } N \geq M^{(k)}\}$ for $M \in \Lambda$ and $k \in \mathbb{N}$, form a basis for the Scott topology. Therefore, Λ is second countable.*

2. *The set* $\mathrm{NF} \subseteq \Lambda$ *of all the λ-nf is dense in Λ.*

3. *M is λ-normalizable iff M is isolated.*

4. *M is unsolvable iff M is a compactification point.*

5. *Any term algebra $\Lambda/_\mathcal{T}$ of a λ-theory $\mathcal{T}$ (and in general any quotient of Λ) contains a compactification point (with the quotient topology).*

Points 1-4 are well known and already appear in [Bar84]. Point 5 is not written there, but it is trivial from point 4 since containing a compactification point is preserved by quotients. Indeed, let Ω be a compactification point of a space X, and let $\sim$ be an equivalence on X. Let $q : x \in X \to [x] \in X/\!\sim$ be the quotient map, which is surjective and continuous. Now if $\mathcal{O} \ni [\Omega]$ is open in $X/\!\sim$, then $q^{-1}\mathcal{O} \ni \Omega$ and it is open in X. Being Ω a compactification point, $q^{-1}\mathcal{O} = X$, and thus $\mathcal{O} \supseteq qX = X/\!\sim$. So $[\Omega]$ is a compactification point of $\in X/\!\sim$.

As already mentioned, now that we have a topology on Λ, a typical question would be: "what is its fundamental group $\pi_1(\Lambda, M_0)$?" As far as we know, this question does not appear in the literature. It may be because the answer is immediate: in fact, consider the following undergraduate exercise.

Proposition 5.2.4. *If a space X contains a compactification point then X is contractible.*

Proof. Let Ω be a compactification point of X and take the constantly equal to Ω map from X to itself. An homotopy between it and id_X is by definition a continuous map $F : X \times [0,1] \to X$ (w.r.t. product topology) s.t. $F(x,0) = x$ and $F(x,1) = \Omega$ for all $x \in X$. The following F does the work:

$$F(x,t) := \begin{cases} x & \text{if } t \in [0, \frac{1}{2}) \\ \Omega & \text{if } t \in [\frac{1}{2}, 1]. \end{cases}$$

In fact, we only have to show that it is continuous: let $O \neq X$ be open in X, and let us show that $F^{-1}O = O \times [0, \frac{1}{2})$. Being the latter an open in $X \times [0,1]$, this will conclude the proof. The inclusion $(\supseteq)$ is trivial by definition of F. For $(\subseteq)$, take $(x,t) \in F^{-1}O$. If $t \in [\frac{1}{2}, 1]$ then $O \ni F(x,t) = \Omega$ and, since Ω is a compactification point, $O = X$, which is not possible. So it must be $t \in [0, \frac{1}{2})$, and this entails $(x,t) = (F(x,t), t) \in O \times [0, \frac{1}{2})$. □

[8] However, in order to prove the Continuity theorem one needs, in the lemmas that one uses, the Monotonicity for 1-hole contexts. Since we already saw that this easily implies the Monotonicity for multi-hole contexts, this remark does not give an alternative proof of the Monotonicity.

Therefore, the properties of contractible spaces immediately answer to our question:

Corollary 5.2.5 (Fundamental group of λ-calculus)**.** *Λ, as well as any of its quotients, is contractible to a point. Hence, Λ, as well as any of its quotients (in particular any term algebra $\Lambda/_{\mathcal{T}}$ of a λ-theory $\mathcal{T}$) is simply connected, i.e. it is path-connected and have trivial fundamental group $\pi_1(\Lambda) = \pi_1(\Lambda/_{\mathcal{T}}) = \{*\}$. More generally, it is n-connected for all $n \in \mathbb{N}$, so all its homotopy groups are trivial (and the same for the quotients).*

Despite the striking banality, we find that Corollary 5.2.5 deserves to be written somewhere, because when having a topological space it is always good manner to know what its fundamental group is.

We see here another manifestation of the fact that (Scott)-topology on λ-calculus is "badly behaved", at least if one considers it from a geometrical perspective. In fact, the Scott-topology should be better understood from the perspective of order theory. However, the "pathology" of the geometry seems to be caused by the presence of compactification points, which are exactly the unsolvables, so one could ask: is the geometry of the space of solvables still trivial/pathological? Does, in general, Λ contain "interesting" subspaces? Even if our guess, and maybe the one of the most of the researchers, is that one *cannot* find any interesting "standard" geometry in Λ, this should still be supported by concrete results – of which our Corollary 5.2.5 is an example.

To the list of "pathological" properties of λ-calculus, we can also add the fact that continuity of the contexts functions is componentwise, and that any denotational model must embed the function space into the object space[9]. Longo imputes these last "pathologies" to the lack of attention to the notion of *space*:

> *The componentwise analysis of continuity, as Cartesian Closure of the intended categories, the weakness of the topology (T0 separation) ... are all symptoms of this "lack of attention" to physical space (and its cartesian dimension) which is typical of this theory. The sequentiality of (classical and intuitionistic) logical deduction is at the core of it. [...] Observe then that all its models force the strong isomorphic embedding that we considered as a paradigm of "being unrelated to physical space" (as dimension is a topological invariant): in any Cartesian Closed Category, a solution D of the equation $X = X \to X$ is such that $D \times D$ can be isomorphically embedded into D.*
> Giuseppe Longo – [Lon03]

Who says "space" says "geometry", and Longo observes that, indeed, while at the beginning of the XX^{th} century physics became "geometrical", mathematical logic – and thus (theoretical) computer science – became instead "arithmetical".

In conclusion, we think that λ-calculus (and its term-algebras, λ-theories, Church groups etc) only reflects the mathematical properties that the notion of computation shows up, when modelled as such a functional programming language. We suggest that one should, instead, approach the peculiarity of λ-calculus[10] as a point of interest: its real mathematical content has to be find somewhere else than, say, usual geometry; the fact that the computation, when expressed in this fundamental formalism, organizes in a certain kind of mathematics, different and less known than standard one, is interesting and actually gives motivation to investigate it.

[9] Of course this is, in some sense, the "essence" itself of λ-calculus, and comes from the fact that it is a *high order* programming language.

[10] And here we could really talk about all the discipline known as "computer science logic".

5.2.2 What about the homology of the space of proofs?

This section begins with the observation that a coherent space is a particular case of the well known geometrical notion of *abstract simplicial complex* (Remark 5.2.13). It contains some incomplete observations and ideas which we had the occasion to quickly discuss together with Ehrhard, Manzonetto and Tortora de Falco.

We start by recalling the notions of abstract simplicial complex and the simplicial homology (a standard reference being [Hat02]). Then, we explain in which sense we would like to bring this kind of techniques into the study of proofs in relation to denotational semantic; we will show how this raises the non-trivial question of choosing a right notion of morphisms in order to make the homology modules functorial; finally, we give some naive notions of such morphisms, which however fail, and we sketch a possible solution.

Interlude: abstract simplicial complexes and simplicial homology

Definition 5.2.6. *An* abstract simplicial complex *("asc", for short) X is the data of a set $|X|$ and a collection $\mathcal{S}X$ of finite non-empty subsets of $|X|$, such that:*

1. *$\{a\} \in X$ for all $a \in |X|$*
2. *X is an initial segment of $\mathscr{P}^*_{\text{fin}}(|X|)$ w.r.t. inclusion.*

Of course tha important point in the previous definition is 2), as an asc on a given web can be given either by declaring *all* its elements, or by declaring only its *non-singleton* elements.

We will always assume $|X|$ to be at most countable.

Terminology 5.2.7. *The set $|X|$ is called the* web *of X and its elements are the* vertices *of X. The elements of $\mathcal{S}X$ are the* simplices *of X. The dimension* $\dim x$ *of a simplex $x = \{a_0, \dots, a_k\}$ of X is k, and a k-dimensional simplex is called a k-simplex for short. The dimension* $\dim X$ *of X is* $\sup_{x \in \mathcal{S}X} \{\dim x\} \in \mathbb{N} \cup \{\infty\}$. *The subsets y of a simplex x of X are the* faces *of x. Vertices can be identifies with* 0*-simplices.* 1*-simplices are called the* edges *of X.* 2*-simplices are called the* triangles *of X.* 3*-simplices are called the* thetraedras *of X. The elements of $\mathscr{P}^*_{\text{fin}}(|X|) - \mathcal{S}X$ are called the* non-simplices *of X. We will use the usual notions of* standard n-simplex Δ^n*, and of S^n.*

Definition 5.2.8. *A sub-asc of an asc X is an asc X' with web $|X|$ s.t. $\mathcal{S}X' \subseteq \mathcal{S}X$.*
The k-skeleton $\text{Sk}_k(X)$ *of an asc X is the sub-asc of X whose simplices are the h-simplices of X with $h \leq k$. The* 1*-skeleton of X can be seen as a graph and it is called the* underlying graph *of X.*

Remark 5.2.9. *Any asc X can be "geometrically realized" as a topological space*[11] $\text{Geo}(X)$ *inside $\mathbb{R}^{\text{Card}(|X|)}$. It is a standard and well known construction, which justifies Terminology 5.2.7. For example, Δ^2 becomes the filled triangle on the vertices $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ in $\mathbb{R}^3$, S^1 becomes its "border" and so on. Remark that this geometric realization "abuses of space", in the sense that, for instance, $\text{Geo}(\Delta^2)$ is a surface and so it can be already embedded in $\mathbb{R}^2$. Furthermore, the construction depends on an arbitrary enumeration of the vertices and on an arbitrary choice of the embedding in $\mathbb{R}^n$, so it is not canonical*[12]*. But we will not worry about*

[11] Usually the geometrical realization is denoted with $|X|$. Here we used this notation for something completely different: the set of vertices, which we call the "web". We did so in order to stay within the tradition of the notations and terminologies used in the discipline of logic of programs.

[12] In fact, there is a standard modified version of the geometric realization, usually called "topological realization", which is canonical and realizes an asc X inside the free $\mathbb{R}$-vector space over $|X|$, endowed with a canonical topology (but it still "abuses of dimensions"). Of course this two realizations are homeomorphic.

these facts since we will only sporadically refer to it, and for the same reason we will not detail it.

The following is a standard construction.

Definition 5.2.10. *The* barycentric subdivision $\mathbf{bs}(X)$ *of an asc X is the asc with web $|\mathbf{bs}(X)|:=\mathcal{S}X$ and whose k-simplices are the $\subseteq$-totally-ordered $\{u_0,\dots,u_k\}\subseteq\mathcal{S}X$.*

One proves that $\mathrm{Geo}(\mathbf{bs}(X))\approx\mathrm{Geo}(X)$ (we mean homeomorphic).

Definition 5.2.11. *A map $f:|X|\to|Y|$ is said to be* simplicial *iff it sends simplices of X to simplices of Y. Asc and simplicial maps form a category, denoted ASC.*
Remark that if f is simplicial, then $\dim(fx)\leq\dim x$ for all simplex x of X. If the equality holds for all simplex x of X, then f is said to be rigid.

Definition 5.2.12. 1. *A* qualitative domain *(qd for short) is a asc X s.t. for all $D\subseteq\mathcal{S}X$ directed[13] w.r.t. inclusion, we have $\bigcup_{d\in D} d\in\mathcal{S}X$.*

2. *A asc X is said to be* flag *iff any minimal (w.r.t. inclusion) non-simplex has cardinality 2 (we say that it is a "missing edge").*

Theoretical computer scientists and logicians know very well the notion of flag complexes, as the following remark explains:

Remark 5.2.13 (Coherent spaces)**.** 1. *Flag complexes can be equivalently defined by requiring $\mathrm{Cl}(\mathrm{Sk}_1(X))\subseteq\mathcal{S}X$. Note that this means $\mathcal{S}X=\mathrm{Cl}(\mathrm{Sk}_1(X))$, since $\mathrm{Cl}(\mathrm{Sk}_1(X))\supseteq\mathcal{S}X$ holds for any asc.*
Equivalently, a flag complex can be defined by requiring that for all family $\{x_i\}_i$ of simplices, if $x_i\cup x_j$ is a simplex for all i,j, then $\bigcup_i x_i$ is a simplex.
Flag complexes are known in the realm of logic of programs as coherent spaces *(cs for short), and this is the terminology we will employ from now on.*

2. *It is easily seen that a cs is a qd, but the converse is in general false.*

3. *If G is a graph[14] with vertices W, then $\mathrm{Cl}(G)$ forms a cs (still denoted by $\mathrm{Cl}(G)$) with web W. Thus, $G=\mathrm{Sk}_1(\mathrm{Cl}(G))$ (since they are graphs with the exact same cliques). Remembering the definition of cs's, graphs and cs's are hence identified via the inverse bijections $\mathrm{Cl}(\cdot)$ and $\mathrm{Sk}_1(\cdot)$.*

Let us now recall the the notion of *(simplicial) homology*. It is a a powerful tool to study the geometry of asc's by looking at their "n-dimensional holes" via the means of a chain of modules. Its construction goes as follows.

Definition 5.2.14 (Oriented simplices)**.** *Let X be a asc. Let $\widetilde{\mathcal{C}}_kX$ be the free $\mathbb{Z}$-module generated by the set of the $(a_0,\dots,a_k)\in|X|^{k+1}$ such that $\{a_0,\dots,a_k\}\in\mathcal{S}X$.*
*Define the set $\mathcal{C}_kX$ of the k-*chains *of X as the set $\widetilde{\mathcal{C}}_kX$ quotiented under the identifications*

$$(a_0,\dots,a_k)=\epsilon(\sigma)(a_{\sigma(0)},\dots,a_{\sigma(k)})$$

$$(a_0,\dots,a_k)=0 \text{ if } a_i=a_j \text{ for some } i\neq j$$

where 0 is the empty sum and $\sigma\in\mathfrak{S}_{k+1}$ and $\epsilon(\sigma)=\pm1$ is the parity of σ. $\mathcal{C}_kX$ hereditates a $\mathbb{Z}$-module structure from $\widetilde{\mathcal{C}}_kX$. We still denote with $(a_0,\dots,a_k)\in\mathcal{C}_kX$ the image of $(a_0,\dots,a_k)\in\widetilde{\mathcal{C}}_kX$ under the quotient map. We denote with $\overrightarrow{\mathcal{S}_k}X$ the set of the k-chains of X with exactly one summand, that is, the elements of $\mathcal{C}_kX$ with shape $(a_0,\dots,a_k)$. They are called the oriented k-simplices *of X.*

[13] A poset D is said to be directed iff for all $x,y\in D$ there is $z\in D$ s.t. $x\leq z\geq y$.
[14] To be pendantic, here we identify a graph with its reflexive graph.

Each k-simplex x of X induces exactly two oriented k-simplices (except for $k = 0$, where it induces only one), and the choice of one of them intuitively corresponds to the choice of an orientation for x. So, concretely, a k-chain of X is a finite linear combination with coefficients in $\mathbb{Z}$ of k-simplices of X with a chosen orientation for each of them. That is, $\mathcal{C}_k X$ is free with basis the set $\overrightarrow{\mathcal{S}_k} X$ of the oriented k-simplices of X.

Proposition 5.2.15 (Boundary operator)**.** *Let X be an asc. There is*[15] *a unique* linear *map*

$$\partial_k : \mathcal{C}_k X \to \mathcal{C}_{k-1} X$$

called the k-boundary operator*, such that for all oriented k-simplex $(a_0, \dots, a_k) \in \overrightarrow{\mathcal{S}_k} X$ one has*

$$\partial_k(a_0, \dots, a_k) = \sum_{i=0}^{k} (-1)^i (a_0, \dots, a_{i-1}, a_{i+1}, \dots, a_k).$$

Proposition 5.2.16. *We have*

$$\partial_k \circ \partial_{k+1} = 0.$$

Definition 5.2.17 (Chain complexes)**.** *A* chain complex *on $\mathbb{Z}$ is the data of a sequence of $\mathbb{Z}$-modules M_k together with linear maps*

$$\dots \xrightarrow{\partial_{k+2}} M_{k+1} \xrightarrow{\partial_{k+1}} M_k \xrightarrow{\partial_k} M_{k-1} \xrightarrow{\partial_{k-1}} \dots$$

such that[16]

$$\partial^2 = 0.$$

We just saw that chain-modules $\mathcal{C}_k X$ and boundary operators ∂_k on an asc form a chain-complex[17] on $\mathbb{Z}$.

The following constructions are the starting point of homological algebra.

Definition 5.2.18 (Simplicial homology modules)**.** *Let $(M_k, \partial_k)_{k\in\mathbb{N}}$ be a chain complex. Elements of* $\mathrm{Im}(\partial_{k+1})$ *are called* k-boundaries *and elements of* $\mathrm{Ker}(\partial_k)$ *are called* k-cycles*. Both are sub-$\mathbb{Z}$-modules of M_k. By definition of chain complex, a k-boundary is a k-cycle, so the (abelian) group structure of* $\mathrm{Im}(\partial_{k+1})$ *forms a subgroup (thus, normal) of* $\mathrm{Ker}(\partial_k)$*, and so we can take the abelian quotient group:*

$$\mathcal{H}_k := \mathrm{Ker}(\partial_k)/\mathrm{Im}(\partial_{k+1})$$

which is called the k-homology group *of $(M_k, \partial_k)_{k\in\mathbb{N}}$. Its elements are called* k-homology calsses *on $\mathbb{Z}$ and are the cosets $\gamma + \mathrm{Im}(\partial_{k+1})$ for $\gamma \in \mathrm{Ker}(\partial_k)$.*
The group $\mathcal{H}_k$ inherits a structure of $\mathbb{Z}$-module via the multiplication in $\mathrm{Ker}(\partial_k)$*, it is known as the* k-homology *module of X.*

Definition 5.2.19 (The category of chain complexes)**.** *A* chain map *from a chain complexes $(M_k, \partial_k)_{k\in\mathbb{N}}$ to a chain complex $(M'_k, \partial'_k)_{k\in\mathbb{N}}$ is the data of a sequence of linear maps $\varphi_k : M_k \to M'_k$ commuting with borders, that is, such that one has the commutative diagrams:*

$$\begin{array}{ccc} M_{k+1} & \xrightarrow{\partial_{k+1}} & M_k \\ {\scriptstyle\varphi_{k+1}}\downarrow & & \downarrow{\scriptstyle\varphi_k} \\ M'_{k+1} & \xrightarrow[\partial'_{k+1}]{} & M'_k \end{array}$$

Chain complexes on $\mathbb{Z}$ with chain maps as morphisms form a category, denoted $\mathrm{ChainCpx}_{\mathbb{Z}}$.

[15] This is not immediatly obvious due to the quotient operated in $\mathcal{C}_k X$. One has to show that the same map when defined on $\widetilde{\mathcal{C}}_k X$ is well-defined on the quotient.

[16] Or equivalently, $\mathrm{Im}(\partial_{k+1}) \subseteq \mathrm{Ker}(\partial_k)$.

[17] Actually, one can prove that in a certain sense the boundary operator is, modulo the sign $\pm$, the only existing map forming a chain-complex with the sets of chains of an asc.

Remark 5.2.20. *Every chain map $\varphi = (\varphi_k)_{k\in\mathbb{N}}$ from a chain complexes $M = (M_k, \partial_k)_{k\in\mathbb{N}}$ to a chain complex $M' = (M'_k, \partial'_k)_{k\in\mathbb{N}}$ induces linear maps:*

$$\mathcal{H}_k\varphi : \mathcal{H}_k(M) \to \mathcal{H}_k(M')$$

setting for all k-cycles γ in M,

$$\mathcal{H}_k\varphi(\gamma + \mathrm{Im}(\partial_{k+1})) := \varphi_k(\gamma) + \mathrm{Im}(\partial'_{k+1}).$$

That is, we have a functor:

$$\mathcal{H}_k : \mathrm{ChainCpx}_{\mathbb{Z}} \to \mathrm{Modules}_{\mathbb{Z}}.$$

Remark 5.2.21. *Thanks to the previous remark, a functor F from a category $\mathcal{A}$ to the category* $\mathrm{ChainCpx}_{\mathbb{Z}}$ *induces, by composition, functors*[18]

$$\mathcal{H}_k : \mathcal{A} \to \mathrm{Modules}_{\mathbb{Z}}.$$

In algebraic topology, one often takes $\mathcal{A} = \mathrm{ASC}$ and considers the *k-homology $\mathbb{Z}$-modules* $\mathcal{H}_k(X)$ of X defined by the chain-complex $(\mathcal{C}_kX, \partial_k)_k$ of an asc X. The construction is the following:

Let $f : X \to Y$ a simplicial map. For all $k \in \mathbb{N}$ there is a unique *linear* map:

$$\mathcal{C}_kf : \mathcal{C}_kX \to \mathcal{C}_kY$$

such that for all $(a_0, \ldots, a_k) \in \overrightarrow{\mathcal{S}}_kX$ one has:

$$\mathcal{C}_kf(a_0, \ldots, a_k) = (f(a_0), \ldots, f(a_k)).$$

One can easily see that one has the following commutative diagrams:

$$\begin{array}{ccc} \mathcal{C}_kX & \xrightarrow{\partial_k^X} & \mathcal{C}_{k-1}X \\ {\scriptstyle \mathcal{C}_kf}\downarrow & & \downarrow{\scriptstyle \mathcal{C}_{k-1}f} \\ \mathcal{C}_kY & \xrightarrow[\partial_k^Y]{} & \mathcal{C}_{k-1}Y \end{array}$$

That is, the maps $\mathcal{C}_kf$ form a chain map $\mathcal{C}f$ from the chain complex $(\mathcal{C}_kX, \partial_k^X)_{k\in\mathbb{N}}$ to the chain complex $(\mathcal{C}_kY, \partial_k^Y)_{k\in\mathbb{N}}$. Furthermore, one can prove that we have actually defined a functor:

$$\mathcal{C} : \mathrm{ASC} \to \mathrm{ChainCpx}_{\mathbb{Z}}$$

setting $\mathcal{C}X := (\mathcal{C}_kX, \partial_k^X)_{k\in\mathbb{N}}$ and $\mathcal{C}f := (\mathcal{C}_kf)_{k\in\mathbb{N}}$ for a simplicial $f : X \to Y$.

This entails that, for all $k \in \mathbb{N}$, one has a functor:

$$\mathcal{H}_k : \mathrm{ASC} \to \mathrm{Modules}_{\mathbb{Z}}.$$

The reason why we gave all these details about this standard constructions is, in addition to recalling them, because we would like to render the homology construction functorial in the same sense as the above functor. However, we are led to change the category ASC, and we have to find the "right" notion of morphism.

[18] To be pedantic, we should write $\mathcal{H}_k \circ F$, but one just writes $\mathcal{H}_k$.

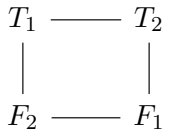
Getting denotational semantics involved

Coherent spaces are a particular case of asc's, therefore they can be realized geometrically and can be thought as concrete spaces inside $\mathbb{R}^n$. We can interpret LL within the coherent semantics, so a formula becomes a space inside $\mathbb{R}^n$ and its proofs, which are cliques of the coherent space, become simplices of the space.

Let us see an easy situation.

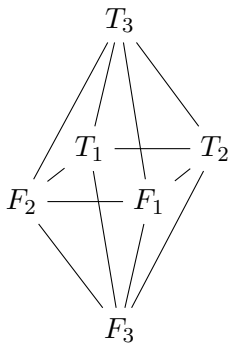
Take the usual encoding of booleans inside LL, that is, via the MALL formula $\mathrm{Bool} := 1 \oplus 1$. Write $\mathrm{Bool}^{\&n} := \mathrm{Bool} \,\&\, \cdots \,\&\, \mathrm{Bool}$ (n times).

Consider $\mathrm{Bool}^{\&2}$. If we call $\{T_1, F_1, T_2, F_2\}$ its web, then $[\![\mathrm{Bool}^{\&2}]\!]$ is given by the graph[19]:



In order to see it as an asc, we have to take the cliques of the graph as simplices. So when realized geometrically, we get the circle S^1 modulo homeomorphism.

Now consider $\mathrm{Bool}^{\&3}$. If we call $\{T_1, F_1, T_2, F_2, T_3, F_3\}$ its web, then $[\![\mathrm{Bool}^{\&3}]\!]$ is given by the graph:



The maximal cliques of this graph are the 8 "faces of the octaedron", so when realized geometrically we get an empty octaedron, which is homeomorphic to the sphere S^2. One can see that in general $\mathrm{Bool}^{\&n}$ is realized as S^{n-1} modulo homeomorphism.

Observe that in the previous examples, *all* the simplices of the asc $[\![\mathrm{Bool}^{\&n}]\!]$ are "witnessed" by a proof of $\vdash \mathrm{Bool}^{\&n}$, in the sense that any simplex is cointained in some proof. This is due to the fact that the 8 maximal cliques are exactly the interpretation of some proof. In fact, take to stay simple the case $n = 3$, and it is easy to see that in MALL there are exactly $2^3 = 8$ cut-free proofs of $\mathrm{Bool}^{\&3}$, which are the following:

$$\pi_{ijk} \quad := \quad \dfrac{\dfrac{\dfrac{\dfrac{}{\vdash 1}\mathrm{ax}}{\vdash \mathrm{Bool}} \oplus_i \quad \dfrac{\dfrac{}{\vdash 1}\mathrm{ax}}{\vdash \mathrm{Bool}} \oplus_j}{\vdash \mathrm{Bool}^{\&2}} \& \quad \dfrac{\dfrac{}{\vdash 1}\mathrm{ax}}{\vdash \mathrm{Bool}} \oplus_k}{\vdash \mathrm{Bool}^{\&3}} \&$$

for $i, j, k = 1, 2$. Now, it is easy to see that, setting $a_1^h = T_h$ and $a_2^h = F_h$ for $h = 1, 2, 3$, one has $[\![\pi_{ijk}]\!] = \{a_i^1, a_j^2, a_k^3\}$, that is, the 8 faces of the octaedron.

However, the fact that all the cliques are witnesses by a proof (in the previous sense) is due to the simple case we have considered.

[19] In the sense that this graph is its 1-skeleton. But since we are takling about cs's, it already gives all the information about its simplices (which are its cliques).

In fact, it is not always the case: take for instance "Gustave's formula":

$$G := (1 \oplus (1\&1)) \parr (1 \oplus (1\&1)) \parr (1 \oplus (1\&1))$$

of MALL. If we call $\{\bot, T, F\}$ the web of $[\![(1 \oplus (1\&1))]\!]$, then $[\![G]\!]$ has web $|G| = \{\bot, T, F\}^3$. Let's call $a_1 := (\bot, T, F), a_1 := (F, \bot, T), a_3 := (T, F, \bot) \in |G|$. Now, it is easily seen that there are MALL proofs $\pi_{1,2,3} :\vdash 1 \oplus (1\&1)$ s.t. $[\![\pi_1]\!] \supseteq \{a_2, a_3\}$, $[\![\pi_2]\!] \supseteq \{a_1, a_3\}$ and $[\![\pi_3]\!] \supseteq \{a_1, a_2\}$. So each of the previous 3 sets is an edge of $[\![G]\!]$, and the definition of coherence space forces thus $x := \{a_1, a_2, a_3\}$ to be a 2-simplex of $[\![G]\!]$, as one can in fact check. But one can also see that there is *no* MALL proof whose coherent semantics contains the whole x.

The property of being a cs can be thought as "synchronization" property: cs's are those asc for which, when one can "synchronize" a family of simplices (or of vertices) two by two, one can synchronize the whole family. From this point of view, the prototipical example of an asc which is not a cs, the simplicial cercle S^1, can be thought as a three agents process synchronising any two of them but failing to synchronize them all three together. The previous example of Gustave's formula shows that, however, if we think to the "synchronization" as given by the proofs of a formula, then the situation is not faithfully reflected by the coherent semantics.

In order to have a faithful representation of the situation, one would be tempted to take as simplices the interpretations of the proofs of a formula A; of course this fails to give an asc[20]. But the problem is easily solved by "down-closing it" and consider the sub-asc $[A]$ of $[\![A]\!]$ given by the web $|[A]| := |A|$ and simplices:

$$\mathcal{S}[A] := \{x \subseteq [\![\pi]\!] \mid \pi :\vdash A\}.$$

In $[A]$, every simplex comes from a proof in the lax sense that it is a face of the interpretation of one (i.e. it is contained in it).

In the previous examples we have thus seen that $\mathcal{S}[G] \subsetneq \mathcal{S}[\![G]\!]$, and $\mathcal{S}[\mathrm{Bool}^{\&n}] = \mathcal{S}[\![\mathrm{Bool}^{\&n}]\!]$. In the case of coherent semantics, to have $\mathcal{S}[A] = \mathcal{S}[\![A]\!]$ (that is, $[A] = [\![A]\!]$) is equivalent to the fact that every $\subseteq$-maximal clique[21] is the interpretation of a proof.

It is important to remark at this point some aspects:

1. In the case of coherent semantics $[\![A]\!]$ is already a geometrical object - it is an asc itself - but because it is a cs, it does not show up a faithful geometry w.r.t. proofs. The quite natural definition of $[A]$ "solves" this problem. Furthermore, the construction of $[A]$ makes sense if $[\![\cdot]\!]$ is any "webbed semantics" of LL, not just the coherent one. This two points suggest that $[A]$ could be the "canonical" geometrical object associated with webbed semantics.

2. The meaning of the asc $[A]$ is geometric, and we are not trying to construct a new denotational model of MALL. The idea is different: *from* a denotational interpretation $[\![A]\!]$ of a formula A we want to extract a geometrical object to study.

3. The asc $[A]$ is, of course, determined by the formula A, but only in the fact that (for a fixed system - here we took MALL) a formula determines its proofs, so $[A]$ is really talking about the proofs of A. More precisely it provides the geometrical organisation of the *interpretations* of the proofs of A as an asc. Concretely, the information of the simplices of $[A]$ comes from the proofs of A, and A alone would be not enough. In particular, the geometrical property of having or not "n-holes" (the "absence" of n-simplices), is determined by the existence or not of suited proofs. A more precise notation would therefore be something like $[\mathrm{Proofs}(A)]$, or if one wants to remember that the construction depends on a fixed denotational semantics for the proofs, we could write $[\mathrm{Proofs}_{[\![\cdot]\!]}(A)]$.

[20] Because it is not closed w.r.t. inclusion.

[21] That is, the "external" simplices if we think of it as a concrete space.

Of course we will only use the simpler $[A]$ but one has to remember that what we are really considering is the geometry of the *space of the proofs* of A.

We already said that a powerful tool to precisely study asc's is simplicial homology. A first natural request would be that homology modules be type-isomorphism invariants. Since homology, when induced by a functor, is a functor itself (Remark 5.2.21), if we can have this property, then we automatically obtain that whenever there is a type isomorphism there is an isomorphism of homologies. This is a basic property that we want to require.

However, we face a problem: what should we take as morphisms between asc's? In topology one takes simplicial maps, but in denotational semantics one typically works with relations instead of functions. It is not clear how to make homology a functor w.r.t. to a "relational based" notion of morphism.

In the following we quickly give some quite natural tries, which however all fail. Only the last one succeeds in giving raise to a functor, and thus we have the isomorphic homologies for isomorphic types (Corollary 5.2.26), but the price to pay is that we need to modify the space $[A]$ via a certain monad $\mathscr{I}$ (Definition 5.2.23). We will comment on that after Corollary 5.2.26 till the end of the section.

Relational semantics? The easiest way of interpreting LL is the relational semantics, so let us try with it.

Relational semantics interprets a type A as a set $[\![A]\!]$ and a proof $\pi :\vdash A \multimap B$ as a subset $[\![\pi]\!] \subseteq [\![A]\!] \times [\![B]\!]$. We can thus send A to the (easily checked) asc $[A] := \{x \subseteq [\![\rho]\!] \mid \rho :\vdash A\}$ with web $|A|$, and one easily sees that $[\![\pi]\!] \subseteq |A| \times |B|$ satisfies the following "simplicial property":

$$[\![\pi]\!] \cdot \sigma \in [B] \text{ for all } \sigma \in [A]$$

where $[\![\pi]\!] \cdot \sigma$ is the image of σ through the function $[\![\pi]\!] \cdot (.) : |A| \to [B]$ defined by $[\![\pi]\!] \cdot a := \{b \in |B| \mid (a, b) \in t\}$. The idea is to take as the new category from which start the constructions, the category RelASC of asc's as objects, and morphisms given by:

$$\mathrm{RelASC}(X, Y) := \{t \subseteq |X| \times |Y| \mid t \cdot \sigma \in Y \textit{ for all } \sigma \in X\}$$

where of course $t \cdot \sigma$ is defined analogously as before.
The natural choice for $\mathcal{C}_k t : \mathcal{C}_k X \to \mathcal{C}_k Y$ is then to extend by linearity the following association:

$$\mathcal{C}_k t(a_0, \ldots, a_k) := \sum_{(\vec{a},\vec{b}) \in t^{\otimes(k+1)}} (b_0, \ldots, b_k)$$

where $t^{\otimes(k+1)}$ is the relation[22] $\{(\vec{a}, \vec{b}) \in |X|^{k+1} \times |Y|^{k+1} \mid (a_i, b_i) \in t \textit{ for } i = 0, \ldots, k\}$.
This construction has two crucial problems: first, $(\mathcal{C}_k t)_k$ is not a chain map, and second, it is not functorial. Just one of them is enough to invalidate the construction we are looking for, but let us see the reason for both of them:

- Non chain map: it does not commute with borders. In fact, take $X = S^1$ (with web $\{a_0, a_1, a_2\}$)[23], $Y = \Delta^2$ (with web $\{b_0, b_1, b_2\}$), and the morphism t from X to Y given by the set $\{(a_0, b_0), (a_0, b_1), (a_1, b_1), (a_2, b_2)\}$. Now it is easily seen that $\partial_1(\mathcal{C}_1 t(a_0, a_2)) = 2(b_2) - (b_0) - (b_1)$ while $\mathcal{C}_0 t(\partial_1(a_0, a_2)) = (b_2) - (b_0) - (b_1)$. This would be enough, but just to mention it, let us say that also on (a_0, a_1) we don't have a commutation: $\partial_1(\mathcal{C}_1 t(a_0, a_1)) = (b_1) - (b_0)$ while $\mathcal{C}_0 t(\partial_1(a_0, a_1)) = -(b_0)$.

- Non functoriality: take asc's X,Y,Z with X containinig a vertex a, Y an edge $\{b, b'\}$ and Z a vertex c. Now take $s := \{(a, b), (a, b')\} \in \mathrm{RelASC}(X, Y)$ and $t := \{(b, c), (b', c)\} \in$

[22] Or, equivalently, we can require $b_0 \in t \cdot a_0, \ldots, b_k \in t \cdot a_k$.
[23] We could also take $X = \Delta^2$.

RelASC(Y,Z). If $\mathcal{C}$ is a functor from RelASC to ChainComplexes$_{\mathbb{Z}}$, then $\mathcal{C}_0(t\circ s)(a) = \mathcal{C}_0t(\mathcal{C}_0s(a))$. But one can check that $\mathcal{C}_0(t\circ s)(a) = (c)$ and $\mathcal{C}_0t(\mathcal{C}_0s(a)) = 2(c)$.

The problems seem to be related to the fact the relational sematics is qualitative and not quantitative. So let us try with a quantitative one.

Matrix semantics? Matrix semantics interprets a type A as the relational semantics, and a proof $\pi :\vdash A \multimap B$ as an integer matrix on $|A|\times|B|$. Here by "integer matrix on a cartesian product $S\times S'$ of sets" we mean a function $t : S\times S'\to\mathbb{N}$. We won't bother say "integer matrix" but just "matrix". Let us set some notations: we write t_{ab} for $t(a,b)$ and $t_a : S'\to\mathbb{N}$ for the curried function $t_a(b) := t_{ab}$.
So the idea is to take as MatixASC the category of asc's and morphims given by:

$$\text{MatixASC}(X,Y) := \{t \textit{ matrix on } X\times Y \mid \bigcup_{a\in\sigma}\text{support}(t_a)\in Y \textit{ for all } \sigma\in X\}.$$

The identities are the diagonally 1 matrices, and composition is matrix composition.
Now the natural choice for $\mathcal{C}_kt : \mathcal{C}_kX\to\mathcal{C}_kY$ is to extend by linearity the following association:

$$\mathcal{C}_kt(a_0,\dots,a_k) := \sum_{(b_0,\dots,b_k)}\left(\prod_{i=0}^{k}t_{a_ib_i}\right)(b_0,\dots,b_k).$$

But this fails to form a chain map. In fact, consider the following counterexample:

take $X = Y = S^1$ (or also Δ^1 would work) with vertices $\{a_0,a_1,a_2\}$, and take $t\in$ MatixASC(X,X) the following matrix:

	a_0	a_1	a_2
a_0	1	0	1
a_1	0	0	1
a_2	1	0	0

It is a simple verification that one has:

$$\begin{aligned}\mathcal{C}_1t(a_0,a_1) &= \det(t_{\restriction\{a_0,a_1\}\times\{a_0,a_1\}})(a_0,a_1)+\det(t_{\restriction\{a_0,a_1\}\times\{a_0,a_2\}})(a_0,a_2)\\ &\quad+\det(t_{\restriction\{a_0,a_1\}\times\{a_1,a_2\}})(a_1,a_2)\\ &= (a_1,a_2)\end{aligned}$$

so $\partial_1(\mathcal{C}_1t(a_0,a_1)) = (a_2)-(a_1)$. But one also can check that:

$$\mathcal{C}_0t(\partial_1(a_0,a_1)) = (t_{a_1a_0}-t_{a_0a_0})(a_0)+(t_{a_1a_1}-t_{a_0a_1})(a_1)+(t_{a_1a_2}-t_{a_0a_2})(a_2) = -(a_0)$$

and thus the $\mathcal{C}_kt$ do not commute with borders.

The problem seems to be related to the fact that we don't have any constraint on the morphisms. So let us try to impose them in the following way.

Relational semantics with coherent semantics? A possible attempt could be to "mix" together the relational and the matrix semantc, in the following sense:

Definition 5.2.22. *An* coherence asc X *is the data of an asc* $\underline{X} = (|X|,\mathcal{S}X)$ *and of a cs* $\text{cs}(X) = (|X|,\frown_X)$ *both on the same web* $|X|$*. It is easily checked that coherence asc's form a category* CohASC*, where the morphisms from* X *to* Y *are the* simplicial cliques *of the cs* $\text{cs}(X)\multimap\text{cs}(Y)$*, that is,* CohASC$(X,Y)$ *is the set of the* $t\subseteq|X|\times|Y|$ *such that:*

i) $t\cdot\sigma\in\mathcal{S}Y$ *for all* $\sigma\in\mathcal{S}Y$

ii) for all $(a,b),(a',b') \in t$*, if* $a \frown_X a'$ *then one has both* $b \frown_Y b'$ *and the implication: if* $b = b'$ *then* $a = a'$*.*

Composition is usual relational composition[24] *and the identities are the diagonal relations.*

The natural way is to define $\mathcal{C}$ as in the relational semantics case. But this fails again to be a chain map, because we can reproduce the same exact couter example of that case, just taking for X the asc $\underline{X} = S^1$ and coherence the trivial one (only the singletons are coherent with themselves)[25], and for Y take $\underline{Y} = \Delta^2$ with coherence given by $b_0 \frown b_1$ (plus the trivial coherences, that is, the one on the singletons). Now the same exact t of the relational semantics' case counterexample is a well defined simplicial clique from X to Y and the computations showing that $\mathcal{C}$ does not commute with the borders are exactly the same.

A possible solution: transform relations into maps via monads? Remember that we defined the category RelASC as the category whose objects are the asc's and whose morphisms are the *simplicial relations*. That is, the elements of $\mathrm{RelASC}(X,Y)$ are the $t \subseteq |X| \times |Y|$ s.t. $t \cdot \sigma \in \mathcal{S}Y$ for all $\sigma \in \mathcal{S}X$, where for a $\sigma \subseteq \mathscr{P}^*_{\mathrm{fin}}(|X|)$ we set:

$$t \cdot \sigma := \bigcup_{a \in \sigma} t^+(a)$$

and $t^+ : |X| \to \mathscr{P}(|Y|)$ is given by:

$$t^+(a) := \{b \in |Y| \mid (a,b) \in t\}.$$

Definition 5.2.23. *We define the following endofunctor* $\mathscr{I}$ *on* ASC*.*

If X *is an asc, then* $\mathscr{I}X$ *is the asc with web* $|\mathscr{I}X| := \mathcal{S}X$ *and* k*-simplices the sets* $\{u_0,\dots,u_k\} \subseteq^*_{\mathrm{fin}} \mathcal{S}X$ *s.t.*

$$\bigcup_{i=0}^{k} u_i \in \mathcal{S}X.$$

If $f : |X| \to |Y|$ *is a simplicial map from* X *to* Y*, then* $\mathscr{I}f : \mathcal{S}X \to \mathcal{S}Y$ *is given by the direct image:* $\mathscr{I}f(u) := fu$*.*

It is immediate to check that the previous definition makes sense.

Proposition 5.2.24. *The endofunctor* $\mathscr{I}$ *forms a monad on* ASC *with unit* $\varepsilon : \mathrm{id} \Rightarrow \mathscr{I}$ *and multiplication* $\mu : \mathscr{I}^2 \Rightarrow \mathscr{I}$ *whose components are respectively:*

$\varepsilon_X : |X| \to \mathcal{S}X$ *given by:*

$$\varepsilon_X(a) := \{a\}$$

$\mu_X : \mathcal{S}\mathscr{I}X \to \mathcal{S}X$ *given by:*

$$\mu_X(U) := \bigcup_{u \in U} u.$$

Proof. Firstly, let us see why $\varepsilon_X \in \mathrm{ASC}(X, \mathscr{I}X)$ and $\mu_X \in \mathrm{ASC}(\mathscr{I}^2X, \mathscr{I}X)$. For ε_X, take a simplex u of X. Then $\varepsilon_X u = \{\{a\} \mid a \in u\} \subseteq^*_{\mathrm{fin}} \mathcal{S}X$ and thus ε_X is simplicial because $\varepsilon_X u$ is a simplex of $\mathscr{I}X$ since $\bigcup_{a \in u} \{a\} = u \in \mathcal{S}X$. For μ_X, take a simplex $\mathbb{U}$ of $\mathscr{I}^2X$. Then $\mu_X \mathbb{U} =$

[24] Just remark that $(s \circ t) \cdot \sigma = s \cdot (t \cdot \sigma)$ to prove that composition preserves simpliciality.

[25] In particular, thus, we have $a_0 \smile a_1$.

$\{\bigcup_{u\in U} u \mid U \in \mathbb{U}\} \subseteq^*_{\text{fin}} \mathcal{S}X$ because $\mathbb{U}$ is finite and non-empty and $\bigcup_{u\in U} u \in \mathcal{S}X$ by definition of $\mathcal{S}\mathscr{I}X$. Thus μ_X is simplicial because $\mu_X\mathbb{U}$ is a simplex of $\mathscr{I}X$ since $\bigcup_{U\in\mathbb{U}}\bigcup_{u\in U} u = \bigcup_{u\in\bigcup_{U\in\mathbb{U}} U} u \in \mathcal{S}X$ by definition of $\mathcal{S}\mathscr{I}X$, because $\bigcup_{U\in\mathbb{U}} U \in \mathcal{S}\mathscr{I}X$ and $\mathbb{U} \in \mathcal{S}\mathscr{I}^2X$.

Secondly, fix $f \in \mathrm{ASC}(X,Y)$. Then ε_X is natural in X because of the equality $\mathscr{I}f(\varepsilon_X(a)) = \{f(a)\} = \varepsilon_Y(f(a))$ for $a \in |X|$, and μ_X is natural in X because of the equality

$$\mathscr{I}f(\mu_X(U)) = f\bigcup_{u\in U} u = \bigcup_{u\in U} fu = \mu_Y(\mathscr{I}f\,U) = \mu_Y(\mathscr{I}^2 f(U))$$

for $U \in \mathcal{S}\mathscr{I}X$.

Finally, let us check the three monad diagrams. For $u \in \mathcal{S}X$, the first diagram for the unit ε is given by the immediate equality $\mu_X(\varepsilon_{\mathscr{I}X}(u)) = u$, and the second diagram for ε is given by the immediate equality $\mu_X(\mathscr{I}\varepsilon_X(u)) = u$. The third diagram is the equality $\mu_X \circ \mathscr{I}\mu_X = \mu_X \circ \mu_{\mathscr{I}X}$. This is easily checked:

$$\mu_X(\mathscr{I}\mu_X(\mathbb{U})) = \mu_X(\mu_X\mathbb{U}) = \bigcup_{U\in\mathbb{U}}\bigcup_{u\in U} u = \bigcup_{u\in\bigcup_{U\in\mathbb{U}} U} u = \mu_X(\bigcup_{U\in\mathbb{U}} U) = \mu_X(\mu_{\mathscr{I}X}(\mathbb{U}))$$

where $\mathbb{U} \in \mathcal{S}\mathscr{I}^2X$. □

Proposition 5.2.25. *There is an equivalece of categories between* RelASC *and the Kleisli category* $\mathrm{ASC}_{\mathscr{I}}$ *of* $\mathscr{I}$*, which is given by the functor:*

$$(\cdot)^+ : \mathrm{RelASC} \to \mathrm{ASC}_{\mathscr{I}}$$

defined by $X^+ := X$ *and* $t^+ : |X| \to \mathcal{S}X$ *by* $t^+(a) := \{b \in |Y| \mid (a,b) \in t\}$ *for a simplicial relation* $t \subseteq |X| \times |Y|$.

Proof. It is easily checked that $(\cdot)^+$ is well-defined. Let's show that it $(\cdot)^+$ is functorial. The verification of $(id_X)^+ = \varepsilon_X$ is immediate. Remembering that the Kleisli composition $t^+ \circ_{\mathscr{I}} s^+$ of two Kleisli morphisms $t^+ \in \mathrm{RelASC}(Y, \mathscr{I}Z)$ and $s^+ \in \mathrm{RelASC}(X, \mathscr{I}Y)$ is by definition $\mu_Z \circ \mathscr{I}t^+ \circ s^+$, the verification of $(t \circ s)^+ = t^+ \circ_{\mathscr{I}} s^+$ is the following simple computation:

$$\begin{array}{rcl}(t^+ \circ_{\mathscr{I}} s^+)(a) & = & \mu_Z(\mathscr{I}t^+(s^+(a))) \\ & = & \bigcup_{b\in s^+(a)} t^+(b) \\ & = & \{c \in |Z| \mid \exists b \in |Y| \text{ s.t. } (a,b) \in s, (b,c) \in t\} \\ & = & (t \circ s)^+(a).\end{array}$$

Finally, let us show that $(\cdot)^+$ gives an equivalence of categories. It is immediate that it is faithful and essentailly surjective. To show that it is full, it is immediate to see that if $g \in \mathrm{ASC}_{\mathscr{I}}(X,Y)$ then $g = \underline{g}^+$ where $\underline{g} \in \mathrm{RelASC}(X,Y)$ is the graph relation of g. □

We have thus the following situation (where we call $\mathrm{Mod}_{\mathbb{Z}}$ the category of $\mathbb{Z}$-modules):

$$\mathrm{RelASC} \xrightarrow[\cong]{(\cdot)^+} \mathrm{ASC}_{\mathscr{I}} \xrightarrow{R_{\mathscr{I}}} \mathrm{ASC} \xrightarrow{\mathcal{C}} \mathrm{ChainCpx}_{\mathbb{Z}} \xrightarrow{\mathcal{H}_k} \mathrm{Mod}_{\mathbb{Z}}$$

where $R_{\mathscr{I}}$ is part of the Kleisli adjunction $L_{\mathscr{I}} \dashv R_{\mathscr{I}}$. That is, $R_{\mathscr{I}}X := \mathscr{I}X$ and $R_{\mathscr{I}}f := \mu_Y \circ \mathscr{I}f$.

Therefore, we immediately have the desired property that we have been looking:

Corollary 5.2.26. *If A and B are isomorphic types in MALL, then $\mathcal{H}_k(\mathscr{I}[A]) \cong \mathcal{H}_k(\mathscr{I}[B])$ for all $k \in \mathbb{N}$.*

A brief discussion is needed here. We declared that our interest is the study of the geometry of $[A]$, which being an asc is quite natural to tackle by regarding its homology. Because of our constraint to take simplicial relations as morphisms instead of simplicial maps, we find ourself able to talk in a satisfactory (functorial) way about the modified asc $\mathscr{I}[A]$. The question is thus what is the relation between $\mathscr{I}X$ and X, or at least between the geometry of $\mathscr{I}X$ and that of X, for whatever "geometry" could mean.

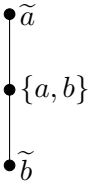
For instance:

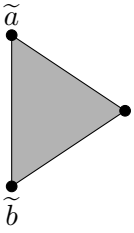
Proposition 5.2.27. *If X is non-trivial (trivial being a point),* $\mathbf{bs}(X)$ *is a* strict *sub-asc of $\mathscr{I}X$.*

Proof. The inclusion is clear: if σ is a simplex of $\mathbf{bs}(X)$ then (since σ is totally ordered w.r.t. inclusion) $\bigcup_{u\in\sigma} u$ coincides with some $u \in \sigma$, which is a simplex of X by definition of $\mathbf{bs}(X)$.
To see that it is a strict sub-asc, take the segment Δ^1 (call its web $\{a, b\}$). Then $\widetilde{a} := \{a\}$ and $\widetilde{b} := \{b\}$ are simplices of Δ^1 s.t. $\widetilde{a} \cup \widetilde{b} \in \mathcal{S}\Delta^1$, thus $\{\widetilde{a}, \widetilde{b}\}$ is a simplex of $\mathscr{I}\Delta^1$. But it is not a simplex of $\mathbf{bs}(\Delta^1)$ because $\widetilde{a} \not\subseteq \widetilde{b}$ nor the other way around. If now an asc X is not a point, then it contains at least a Δ^1 as an edge, so $\mathscr{I}X$ strictly contains $\mathbf{bs}(X)$. □

It is instructive to visualize some examples:

$\Delta^1 =$ 

$\mathbf{bs}(\Delta^1) =$ 

$\mathscr{I}\Delta^1 =$ 

$= \Delta^2$

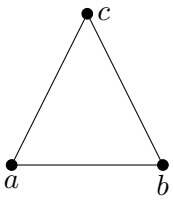
We immediately remark something: $\mathscr{I}X$ makes the dimension grow. This means that in general, the geometric realisations are not homeomorphic:

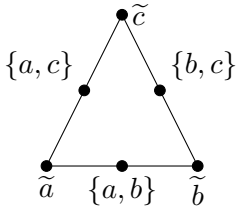
$$\mathrm{Geo}(\mathscr{I}X) \not\approx \mathrm{Geo}(X).$$

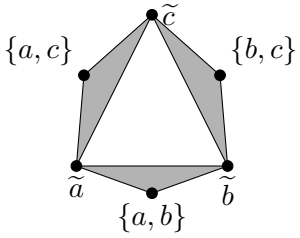
One can easily see that in general we have:

$$\mathscr{I}\Delta^n = \Delta^{(2^{n+1}-1)}.$$

The following is an interesting example too:

$S^1 =$ 

$\mathbf{bs}(S^1) =$ 

$\mathscr{I}S^1 =$ 

We remark something about all the previous examples (both the cases $X = \Delta^n$ and $X = S^1$): even if $\mathbf{bs}(X) \neq \mathscr{I}X$ and even if their geometric realizations are not homeomorphic, we still

have that $\mathrm{Geo}(\mathscr{I}X)$ *retracts onto* $\mathrm{Geo}(\mathbf{bs}(X)) \approx \mathrm{Geo}(X)$. Therefore, in this very simple cases, the homology of X and of $\mathscr{I}X$ remains the same.

However, it is not clear what happens in more complex cases. We think that, if one had $\mathcal{H}_k(\mathscr{I}X) \neq \mathcal{H}_k(X)$ in general, this should be possible to show by considering $X = S^2$. If the inequality above does hold, then it would require to understand if the study of this geometrical object $\mathscr{I}X$ still makes sense (does it have some geometrical interest? Does it carries some interesting computational/logical properties?). On the contrary, if our conjecture is false, then the homology is invariant under $\mathscr{I}$, and so we have achieved our first goal of constructing a space ($[A]$) whose homology is invariant under type isomorphism.

5.3 Philosophical meditations: what about the foundations of mathematics?

We organize our reflection in three steps:

- First, we give a quick presentation of the traditional viewpoint under which mathematical logic is usually considered. In particular, we highlight the points of unsatisfactoriness in it, that can be found in the notion of truth and coherence, when one charges them with a too important foundational relevance.

- Then, we present (our understanding of) Girard's philosophical proposal known as "transcendental syntax", which precisely moves from the mentioned unsatisfactoriness in order to find a deeper point of view, incorporating ideas from computer science.

- Last, we present our proposal organization for some "foundational activity", a central idea being Kant's transcendental method.

5.3.1 The traditional ideology and the questions of logic

Tarski and the problem of truth Mathematical logic is for the most understood from the point of view of analytical philosophy[26], as that was the one of two of the main father founders of discipline: Frege and Russel. A central role is played by the dichotomy syntax/semantics, which is understood as "language/reality". Logic is supposed to extrapolate the formal aspects of reality, as it can be seen in Russel's famous quote:

> *[...] logic is concerned with the real world just as truly as zoology, though with its more abstract and general features.*
> — Russel, Introduction to Mathematical Philosophy (1919)

The mean which controls the distinction syntax/semantics is identified with the notion of *truth*, that is, accordance with reality. This vision is accompanied by a clear and unambiguous distinction between "subject/object", which is another way to usually look at the dichotomy syntax/semantics, and this philosophical point of view is close to what is usually called *essentialism*. This approach is reflected, for instance, in the so-called philosophical logics, and in general in the so-called Tarski's definition of truth – which is just the name for a simple mathematical construction. This approach (which constitutes the main philosophical conception in most of logic departments) has shown to be valid, from a certain point of view: typically, model theory has proved to be a valid branch of mathematics, with its own problems and its own mathematical reason of being; we also find the very idea of "interpreting syntax in some mathematical structure", even if in a slightly different taste, behind the notion of denotational

[26] We are referring here to the well known distinction analytical/continental philosophy.

semantics of programs. The arguably most important achievement of the "essentialist" point of view is the notion of Turing machines[27], which constitute the first model of computer.

However, considered as a philosophical position on the question of foundations, we tend to consider this traditional approach quite limited, for basically the following two reasons:

1. The notion of truth playing the role of ultimate justification is of dubious foundational relevance, as expressed by Girard's doubts:

   > *La vérité est de fait une notion rebelle à toute analyse, car elle commute à toutes les opérations logiques ($A \wedge B$ est vrai ssi A et B le sont, $\neg A$ est vrai ssi A n'est pas vrai etc.) et en particulier toute justification des principes mathématiques par leur vérité est grandement suspecte.*
   > — Girard, [Gir00]

   What is achieved in this way is in fact a form of *translation* from a language to another, so that the relation is more language/language than language/reality. Remark that "translations" are useful: for example, programs are compiled[28] in executables, and this exactly means translating a program from a language to another. Another value of the dominant point of view is that of the "justification" of the different formal systems. By "justification" we mean the arguments supporting the fact that some formal system is worth the study, or not, just like one does in applied mathematics when deciding what model to use for a specific situation. Typically, one justifies predicate calculus as "abstract zoology of boolean operations on non-empty sets"[29]; or we can justify modal logics as a way of formally dealing with our concepts of necessity and possibility, and similarly for other philosophical logics; we can apply temporal logics in formal verification; we can also say that linear logic is the logic "of automatic coffee machines"[30].

   But we would say that this approach places this disciplines more in the realm of applied mathematics – for it provides a mathematical model of some phenomenon – rather than answering some foundational quest. In general, we tend to agree with Dummett's conception:

   > *If to know the meaning of a mathematical statement is to grasp its use; if we learn the meaning by learning its use, and our knowledge of its meaning is a knowledge which we must be capable of manifesting by the use we make of it: then the notion of truth, considered as a feature which each mathematical statement determinately possesses or determinately lacks, independently of ours means of recognising its truth-value, cannot be the central notion for a theory of the meanings of mathematical statements [...]*
   > — Dummett [Dum75]

2. In general, this approach proposes a quite simplistic vision on the "logical activity", on the lines of the – nowadays surpassed – vision of the physics of the absolute space and time, and of the ultimate "objective" reality (in contraposition with the "subjective" reality of the relativistic world and the "non-reality" of the quantic world). One could reasonably expect that, like physics, also logic needs a renewing of its foundational paradigm. As a matter of fact, a "truth-based viewpoint" is not able to satisfactorily take into account the activity one does in some parts of proof-theory/computer science (mainly in relation

[27] Turing's analysis of the (human) computer is based on the distinction object (datas) / subject (the program, or the programmer).

[28] Or interpreted, but this is essentially the same.

[29] This is supported by the soundness and completeness theorems with non-empty Tarski semantics.

[30] We refer here to the resource interpretation of linear logic.

with linear logic). It is interesting to notice, as Abrusci does[31], that a purely essentialist vision is also in discrepancy with today's state of computer science, which is much more a matter of nets (typically, internet) and *interaction* between entities (the hosts of the net) which can play both the role of subject and object, rather than a matter of single-machine-computation on data as in Turing's conception – dating back to before computers existed. There is also another hint: Kant's analysis of (transcendental) logic shows how relevant it is to carefully consider the subject's "action" on the object. And to consider the action of the subject on the object we are studying exactly means to reconsider the duality syntax/semantics.

In conclusion, the traditional approach (and all its "products") are *not* "wrong"; they should simply not be considered as providing a satisfactory foundational paradigm, and their interest should be measured only with respect to the concrete results they allow to find in the respective areas, just like one does for other mathematical modelings. We believe that what we need is not a negation of the traditional point of view, just like XX^{th} century physics is not the negation of the physics before it: it is its surpass, in the Hegelian sense of *aufhebung*.

Hilbert and the problem of coherence In Hilbert's formalistic vision of mathematics, mathematics is divided in two parts: the finitist one and the ideal one. The two are linked via the fascinating idea that the ideal mathematics is to finitist mathematics as points to infinity in projective geometry are to standard geometry, or as imaginary numbers are to the reals: it is there, basically for convention and utility, in order to "complete" the discipline and make it "better functioning". Also, in Hilbert's vision, finitist mathematics is nothing but a meaningless game of symbols. This raises the question of which one, among all possible games of symbols (that is, formal systems), one should use in order to represent mathematics; Hilbert's answer was basically "anyone which is coherent" (and powerful enough).

Let us say we take PA (first order Peano's arithmetic) as formal system. Now Hilbert's program could be resumed as follows:

1. An assumption: if PA does not derive $\bot$, then arithmetic is coherent.

2. First goal: reduce the coherence of all mathematics to the one of arithmetic.

3. Second goal: prove, *with an argument formalizable in PA*[32], that PA does not derive $\bot$.

If such a program was achieved, by (3) and (1) arithmetic would be coherent, and by (2) so would be all mathematics. The point here is that this program aims at an *absolute* certainty on the coherence of mathematics, and the absolute character should be assured by the fact that the proof can be carried on in PA. In particular, an argument which is not formalizable inside PA would not give absolute certainty.

Moreover, Hilbert's vision was based on his famous "*Wir müssen wissen, wir werden wissen*", consisting on the following assumptions:

4 All non-finitary argument admits a formalization in PA.

5 All (and only) arithmetical truths are provable inside PA.

6 There is an algorithm which, for all formula of PA, decides whether it is true or false.

[31] See his intervention in `https://www.youtube.com/watch?v=r3pDktwmtzg`.

[32] Le us say that a property P admits a proof formalizable in PA when there is a formula P of PA which "expresses P" and which is derivable inside PA; here with "P expresses P" we mean that $\mathbb{N} \vDash P$ iff P holds.

Each of the previous points 1-6 motivated, in a sense, the very birth of mathematical logic as well as all its most important developments. However, as it is well known, his program fails and his vision is wrong: Gödel's second incompleteness theorem says that, if PA is really coherent, then such a proof as in (3) cannot exist[33]. Gödel's second incompleteness theorem, together with Gentzen's proof of the coherence of PA based on transfinite induction, says that (4) is false. Gödel's first incompleteness theorem says that (5) is false. Turing's and Church's undecidability of the Halting Problem, or the undecidability of first order predicate calculus, say that (6) is false.

We are left with (1), which is generally accepted, and (2) which, at least for some interesting parts of mathematics, is completed. But we have lost the "absolute" certainty of the coherence of mathematics and of arithmetic.

Furthermore, Gödel's results also imply that Hilbert's suggestion that any (sufficiently powerful) coherent system would do to represent mathematics, is false. Coherence loses thus its role of the only "light" of logic: not only it is necessary reduced to relative coherence – and so cannot be established once and for all –, but it is not even reliable for what concerns arithmetic; there are in fact *coherent* theories which derive *false* arithmetical statements, such as $T + \neg G_T$ (where G_T is a Gödel's formula for a coherent theory T under the hypothesis of the theorems), deriving the arithmetically false statement $\neg G_T$.

Therefore, we tend to consider the question of the coherence of "all" mathematics as less important than one would expect. On one side, even if we know that *there are* contradictions in informal mathematics[34], those never appear in the practice; also, there are interesting formal theories which are proven to be coherent (and/or complete). On the other side, from a mathematical point of view, modern developments of proof theory show that the internal structure of the systems is already a valid object of study, and this even if the system is contradictory. Of course, coherence still remains an important question in some circumstances: for example, one would hope the kernel of a proof assistant to be coherent.

The questions of logic We have seen how the two main possible sources of foundational certainty – truth and coherence – on which mathematical logic (and thus mathematics) has been linked to in the last century, and which still constitute the main point of view of the most of mathematicians and logicians, actually fail in their foundational role.

What are, then, the questions of logic? Were the foundational questions of mathematical logic only the ones of truth and coherence? The answer is of course, no.

Take for instance Kant's question:

> *How is mathematics possible?*

It certainly is an interesting foundational question, and we would say it belongs to logic. Since it deserves a philosophical investigation rather than a mathematical one, we will omit it in the following pages.

Other interesting questions – which sound extremely modern today – were expressed by the neo-Kantian Mouburg school:

> *How do we know that the logical laws are true? How do we know that the concepts that they isolate are the fundamental concepts of logic, and how do they [Frege and Russel] know that their basic laws are in fact properly basic?*
> — Natorp, 1910

[33] We mean a proof of the formula "naturally" expressing, in PA, the coherence of PA.
[34] See the ones mentioned in Section 1.3.

> *In virtue of what do the logical laws have content, as opposed to being just empty symbols?*
> — Cassirer, 1907

As the name "neo-Kantian" suggests, with no surprise they proposed a transcendental approach to those questions.

The two previous questions suggest a point of view surprisingly close to the one of Girard, both in the fact of believing in the relevance of a transcendental approach, and in the fact of questioning the status of the logical "laws" in themselves, *a priori* from the notion of truth. For instance, today we can nicely answer to some of those questions: we now know that modus ponens is in fact *not* properly basic, due to linear logic's decomposition $A \to B = (!A) \multimap B$. We also we know that, for say MLL, logical "laws" have content in virtue of graph theoretical properties (such as the Danos-Regnier criterion), and *this* is what differentiates them from empty symbols.

Remark that, from the traditional point of view, those questions essentially trivialize: everything reduces to saying that logical laws are true *because* they preserve truth[35], and that what differentiates them from being empty symbols is precisely the fact that they preserve truth.

Last, let us mention an important notion, not central in the traditional viewpoint: the very notion of proof. Some important questions about this notions are:

> *In which sense a proof is a finite object? How is it possible that a finite object gives us certainty about the infinite? This certainty cannot be absolute (due to incompleteness). So it is relative: relative to what?*
> - Girard [Gir12]

Traditionally, proofs do not appear with the same level of "foundational importance" as truth or coherence, because they are only seen *bureaucratic* artefacts in order to find out (when such bureaucratic artefact can be constructed) if the value of a proposition is "1" or "0". A more exciting way of saying that the logical laws preserve truth, would be saying that a proof is a "way of propagating truth", and this all of a sudden sounds much more modern if we state it as "a proof is like a circuit in which some information propagates". The study of proofs then becomes then the study of the properties of such a circuit[36], or the study of the propagation itself[37], or, thanks to the Curry-Howard correspondence, the study of it as a computational object.

It would not be wise to say that those are the only questions of logic. Nevertheless, we believe they are some mathematically and philosophically important questions of logic and – employing a recurrent slogan of Girard – they really shift the logical interest *from the "rules of logic" to the "logic of rules"*.

5.3.2 Girard's trascendental syntax: between Kant and computer science

Girard has, in the last 20 years, proposed a sort of new conceptual setting for conceptually organising logic (and in particular proof theory). We find it of relevant foundational value. As already mentioned, we assist to a change of the perspective on the whole discipline:

> *[...] we are led to consider the opposition between syntax and semantics as an obsolete legacy of the XIXth century : in particular the traditional logical issues of logic – completeness and soundness – should be completely revisited.*
> - Girard [Gir99]

[35] This is still the way things are presented to students in the introductory courses.
[36] Such as in proof-nets, which are indeed circuits in the same way Feynman diagrams are: see [BP11].
[37] Such as in Geometry of Interaction.

How should we handle, then, the questions of logic?

> *My thesis is that the meaning of logical rules is to be found in the well-hidden geometrical structure of the rules themselves : typically, negation should not be interpreted by ⟨NO⟩, but by the exchange between Player and Opponent.*
> - Girard [Gir99]

And with which methodology?

> *[...] l'expression magique conditions de possibilité, si l'on préfère transcendantalisme, est la clef qui ouvre les serrures obstruées par un bon siècle de décervelage scientiste.*
> - Girard [Gir18]

Thus, as it was already clear to the neo-Kantians one century ago:

> *[...] il est nécesssaire de comprendre la part de nous-mêmes dans les objets que nous étudions, d'où l'invocation de Kant. [...] Bien sûr, il ne faut pas compter sur Immanuel pour répondre à notre place : rappelons que ces questions n'avaient aucun sens précis il y a 25 ans.*
> - Girard [Gir11]

The program he launched, called "transcendental syntax", is meant to be both philosophical and technical. However, we believe that the core of the matter has to be found first of all in the philosophical aspects, which are those we are going to develop in this section[38].

Girard's knitting The basic idea is to organise the logical activity in some "cases", reminiscent of the Kantian table of judgements. Let us say that we are dealing with "entities" (in an explicitly vague sense) that can "interact" (in an explicitly vague sense) with each other[39]. There are four ways of describing such a situation:

- Via an *ascertainment*[40]: that is, a self-contained description of each of the entities.
- Via a *performance*: that is, a self-contained description of how the interaction of each entity goes.
- Via an *usine*: that is, organising the entities w.r.t. to behaviours which are possible to check in an effective manner.
- Via an *usage*: that is, giving all the possible interactions.

For example, in a net of machines, one may say that: an ascertainment is a description of the hardware of each machine; a performance is a description of the software of each machine; an usine is given by the routing tables; an usage is a description of all the possible "states" of the net.

We can group the four previous modes as follows:

[38] Girard's texts are often as fascinating as difficult to understand, so what follows is our conception of the matter.

[39] The image of a network of computers here is useful.

[40] "Constat" in french. "Ascertainment" is the most similarly meaning word in English we could find.

1. An analytic description is one in which we give an *internal* description of each entity (ascertainment and performance). That is, one considers the system as the mere addition of entities.

   > *De ce point de vue, A et ∼A ne sont que des ⟨⟨lieux⟩⟩ où pourront, éventuellement, se manifester les partenaires. S'il s'agissait* [he is talking about the formula $A⅋ {\sim}A$] *d'une rallonge physique, il faudrait imaginer un fil nu que l'on branche au moyen d'épissures. Rien ne nous empêche donc de mettre l'appareil du côté A et la source du côté ∼ A. Avec, en perspective, de réjouissants incidents: court-circuits, électrocutions et incendies.*
   > - Girard [Gir21]

2. A synthetic description is one in which we give an *external* description of each entity (usine and usage). That is, one considers the system as a whole new entity (of a different kind).

   > *C'est le mode d'emploi, le surmoi qui régule le ⟨⟨ça⟩⟩ analytique. En logique, ce surmoi est représente par des énonces, des textes pédants, des règles qui restreignent, in fine, les branchements licites.*
   > - Girard [Gir21]

One could ask why use Kant's terminology. The answer is that, in a sense, this it is related to Kant's table of judgments[41], with his famous definitions:
- analytic judgements are those for whom all the concepts present in the predicate already appear in the subject;
- synthetic judgements are those which are not analytic.

From a different viewpoint, we can also consider the distinction syntax/semantics as an instance of the distinction analytic/synthetic: syntax is just "raw information", while semantics is raw syntax plus some meaning.

The typical ascertainment is the result of a computation, considered together with the program which did it[42] (or also a program which is "not meant to be executed"). It cannot be questioned, nor taken to mean something, it is just there and that is it[43].

A program "meant to be executed" can be regarded as a performance, since it describes the way it manipulates (read: "interacts with") its argument.

The types in a typed computation can be regarded as usine.

The typical usage is the infinity of all the possible ways one can use a program.

Girard usually resumes this as saying that "analytic is the space of the meaningless", and "synthetic is the meaning".

We would say that a useful everyday-life example is provided by *dialogues*: take for instance a correspondence between Abelard and Heloise; in answering to Abelard's question, Heloise sends analytic information; in sending the question to Heloise, Abelard also fixes a *format for answers* he is expecting the answer to respect; in receiving Heloise's answer, Abelard structures it (if possible) inside the fixed format. For instance, if Abelard asks Heloise if she knows the

[41] Sentences of shape "subject + predicate".

[42] This is what Girard means when he says that "tout est sur la table, *y compris la table*".

[43] Girard often makes the example of a computation for $27 + 37 = 999$, with the program of multiplication loaded inside $+$.

time, he expects the answer to be of the form: "yes, it is ...", or of the form: "no, sorry". If her answer is different, this raises an error (typically caught by Abelard repeating the question).

This example helps understand what Girard means when he says that "analytic corresponds to answers" and "synthetic to questions".

Typically, when the notions of typed/untyped make sense, one has: "analytic = untyped", "synthetic = typed". For example, this is the case for the duality proof-structures/proof-nets, or untyped λ-calculus/simply typed λ-calculus.

The synthetic usually comes with an element, the *format*. It typically corresponds to formal systems, and its nature is particular. In fact consider Berry's paradox[44]: the usual solution to the contradiction consists in preventing it to happen by using two different meanings of the verb "to define": when we say that b is "defined" by those 12 words English sentence that is the *informal* meaning of "to define" we are using; when inside the English sentence we say that b cannot be "defined" in less than 12 English words, that is some fixed formal meaning we are using, which does not include the informal one. Now, we can regard the paradox as saying that if we want to "rigorously ask questions" (which is the heart of what we do in mathematics), then some format is necessary (if we do not want contradictions); but most importantly, the fact that the intuitive meaning of "defined" cannot be included in the format, indicates that a "well-behaving format" cannot answer everything (here we lose the answer to the question "is b defined in the intuitive sense?"). This is not just a pointless remark: this aspect of Berry's paradox can be seen as a primitive manifestation of the phenomenon of incompleteness. In fact, the systematic existence of an undecidable formula inside a certain "well-behaving" formal system, can be understood as the fact that not only outside of the format – as for the intuitive notion of "defined" – but also inside the format, the presence of the format itself prohibits one to answer any question (even if one can ask it). This is why Girard often compares the format to a turtle shell: necessary for surviving, but uncomfortable.

Let us synthesize all we said with the "conceptual" table in Figure 5.1, which has to be thought of as a sort of *refinement* of Kant's renowned table, with the new ingredient of computer science (which of course Kant did not dispose of). The collocation of the different modes in the explicit/implicit column is obtained thinking of the explicit as something *effectively* given, while implicit is the rest. This corresponds to Kant's *a posteriori/a priori* in some sense.

	Explicit	Implicit
Analytic	Ascertainment	Performance
Synthetic	Usine	Usage

	A posteriori	A priori
Analytic	/	Analytic a priori
Synthetic	Synthetic a posteriori	Synthetic a priori

Figure 5.1: Girard's knitting and Kant's table

Remark, however, that Girard's knitting should not be applied to a theory of knowledge, as Kant's table. The prototypical situation of application is linear logic, since all those considerations were first made possible by linear logic. In general, its typical realm of application, and of all the transcendental syntax which rounds around it, is the theory of proofs and programs. We will clarify that in the next Section 5.3.3.

[44] See Section 1.3.

There is a natural question: *can the implicit be always effectively reduced to the explicit?* This specializes in the following two questions:

- Can any performance be effectively given by an ascertainment?
- Does it always exist an usine which guarantees the usage?

Remark that, in the second question, we are asking for an adequacy of the usine w.r.t. usage, i.e. that usine *guarantees* (or justifies) the possibility of usage. If we now remember that usage is the synthetic a priori, we see that from this point of view a major preoccupation of logic is that of the justification of the synthetic a priori, just like in Kant it was the major one for the theory of knowledge. However its justification, that is its *conditions of possibility*, now is given (when possible) by the usine, corresponding to the synthetic a posteriori, and which gives only sufficient conditions by no means necessary; this a point of departure with respect to Kant[45].

Looking at some old questions through this new perspective

- In some cases, one can explicit a performance. And in some cases there is adequacy between usine and usage: for example, a simply typed λ-term of type $\mathbf{Int} \to \mathbf{Int}$, when applied to any **Int** necessary computes a normal form of type **Int**. Analogously for proof nets (for which the usine is the correctness criterion – which is decidable). However, Turing's 1936 result and Gödel's incompleteness theorems say that, in general, the previously mentioned questions of the analytic and of the synthetic have negative answers.

- What does the notation "$A \vdash B$" mean? This could look like as a naive "beginner student" question, as one immediately answers: "it means that under the hypothesis A, one gets B", that is, it is implication. So one could ask why bother introducing a new symbol in addition to "$\to$", but after a certain time one becomes accustomed with this notation and stops questioning it: it is just a notation for expressing the hypotheses, and it turns out that it is the same thing of implication. Of course this is not false[46]. But we find it not satisfying; a more exciting way of dealing with this question is to say that, conceptually speaking, $A \vdash B$ is the *implicit* version of the implication, while $A \to B$ is the *explicit* version of it. This comes with the fact that the cut-rule, which brings dynamics/computation – so it represents the implicit – applies on $A \vdash B$, while $A \to B$ is just a static piece of information (typically used to "eliminate" all the hypothesis after a proof is concluded).

- The analytical implicit and explicit are "relative". For instance, one can regard a program both as an analytic piece of information (as a result), or as synthetic (when it is meant to be executed). For example, in the sequent calculus LJ, the relativity of the analytic implicit/explicit manifests as the fact that *performances can be stalled*. This does not mean that performances can be systematically *reduced* to ascertainments (we saw that this is impossible), but that we can choose to consider the performance given by $A \vdash B$ as

[45] There is nothing strange in this departure: Kant is talking about theory of knowledge, we are talking about logic.

[46] It supported by:
- the so-called "deduction theorem" in Hilbert-style systems, which precisely says that $A \vdash B$ iff $\vdash A \to B$.
- the introduction rule for implication in natural deduction, which says exactly the same thing of the deduction theorem, but with the considerable intellectual advantage of simply posing it as a definition and not as a result to find.
- soundness and completeness theorems (in first order): B is a logical consequence of A by definition iff all models of A are models of B. Then $A \vdash B$ exactly corresponds to logical consequence.

the ascertainment given by $A \to B$. So the deduction theorem/arrow-introduction-rule:

$$\frac{A \vdash B}{\vdash A \to B}$$

makes one pass from the *executable* $A \vdash B$ (which is *waiting* for its argument – a proof of A) to the *non-executable*[47] $\vdash A \to B$ (which is just a *factual statement*).

5.3.3 A shift of the foundational question

In the last 60 or so years the field of mathematical logic has known a great polarization of its branches: on one side, the tradition related to model theory and set theory; on the other side the one related to proof-theory and the notion of computation/programs. This two traditions became technically, conceptually and even spatially[48] distant. We feel the need a conceptual reorganization of the discipline, particularly when related to the "question of foundations". By "the questions of foundations" we do not mean to tell now which is a "foundational theory" in some sense, question on which we tend to agree with Maddy's pluralist spirit:

> *Once we reject the idea that the choice of a fundamental theory to do these foundational jobs is a matter of determining the 'true mathematical ontology', once we focus instead on the literal mathematical content of our decisions, we come to see that we can and should allow some wiggle room for both pure and applied mathematicians to work in well-motivated variants of the fundamental theory.*
> — Maddy [Mad19]

The discussions in the previous pages indicate, to us, that we should operate what we would call a true *shift of the foundational paradigm*, in the sense that the "quest of foundations" should instead be thought of as an *organic mathematical study, as well as philosophical organization* of the aspects of logic and mathematics. From our point of view, the state of art about the foundational reflections on mathematics should be mentally organized in the following points:

- Fundamental mathematical logic
  - Formal mathematics
  - Proof Theory
- Transcendental mathematical logic.

By "fundamental mathematical logic" we do *not* mean "more important than others": we simply mean "who refers to foundational questions" and not just to other mathematical aspects. The contents of the item "fundamental mathematical logic" is mainly mathematical, in opposition with the content of the "Transcendental mathematical logic", which is philosophical.

A very important point is that the above classification is *not* meant to be a "classification of the disciplines", but instead of the *approaches* a researcher can have when dealing with foundational questions. That is, one can work in formal mathematics or proof theory without having a foundational point of view at all: this is actually the most common situation. So the classification above must be seen as a conceptual organization of the different topics a typical nowadays foundational reflection could include.

Let us give some details.

[47] Think of Linux's "chmod -x". In LJ we can also pass from the non-executable implication to the executable one: the previous rule is reversible; think of Linux's "chmod +x".

[48] The first tradition is mainly done in mathematics departments, while the second one in computer science departments.

Formal mathematics Inside "formal mathematics" we intend all which is related to formalising mathematics – in real computers as well as in formal yet still not computerized languages (such as formal theories). Typically, this includes works in proof assistants (for which we share Buzzard's enthusiastic vision on the "digitalization of mathematics"[49]) and the relative theoretical issues (such as Voevodsky's Univalent Foundations [Uni13] and similar topics). But we can also include reverse mathematics, set theory or model theory when one mentally considers them as formalizations of (parts of) mathematics.

For instance, ZFC set theory has always been the most thought of as concerned with "formalizing mathematics". However it is nowadays mainly accepted that, as a "concrete" formalization of mathematics, set theory is not ideal. This is basically because the encoding is possible only in theory – the system is too "low level" (think of programming everything in assembly) – and it is full of arbitrary implementation choices which have as side effect to make the formal theory able to ask and answer "forbidden" questions, for example "$2 \in 3$", or worse:

> *What's even worse, is that in theory the set M could genuinely represent more than one mathematical object. Perhaps somebody writing the library on natural numbers decided to define 3 to be the set $\{0, 1, 2\}$, and more generally they defined n to be the set $\{0, 1, 2, \dots, n-1\}$. Then someone doing topology defined a topology on a set X to be a set of subsets of X satisfying some axioms. It then turns out that 3 is a topology on 2 (check it! It's really true!).*
> - Buzzard, in: `https://xenaproject.wordpress.com/2020/04/30/the-invisible-map/`

Its weaknesses of course do not mean that formal set theory is not valuable or that it is not interesting as a mathematical discipline. They just mean that it should be freed from much of its usual philosophical/foundational charge. Furthermore, if one wants to *really* formalize mathematics, then type theories behave much better. For example, Voevodsky proposes to substitute ZFC with, basically, Homotopy Type Theory plus his Univalence Axiom. Actually, insisting to consider ZFC as a true formal foundation of mathematics is charging set theory (or, better, set theorists) of a responsibility which is not demanded. It is true that, at the beginning (more than one century ago), ZF(C) was thought of as a real foundation of mathematics (but always in a purely theoretical way, computers not even existing), but soon set theorists became more interested in the mathematics of the "actual infinite[50]":

> *[...] moving rather far from the foundational issues, as much as set theorists have understood the privileged position of the subject as regards to its ability to formulate mathematics, many in general consider that the subject of foundations is not worth pursuing and are much more interested in the mathematics of infinity. [...] Therefore, not only the working set theorists of today do not see themselves as providing the unique ontological foundation of mathematics, but they do not believe in such a unique foundation and do not claim it even for their own subject.*
> — Mirna Dzamonja [Dža17]

Proof theory By "proof theory" we mean everything mathematical which is related to the mathematical study of "mathematical reasoning". It is here that one studies proofs as mathematical objects. Nowadays the major activity in this item is usually done in relation with computer science. As already mentioned, it is important to keep in mind that subjects in this item are not limited to foundational issues: for example, one can work in the area of the abstract theory of programming languages, and yet not have any foundational interest.

[49] See `https://xenaproject.wordpress.com/2021/01/21/formalising-mathematics-an-introduction/`.
[50] In contraposition with the mathematics of the infinite in power, which is arithmetic.

We think that the activity one carries on in this item can be nicely organized via Girard's transcendental syntax approach (in its philosophical form explained in Section 5.3.2): it is in transcendental syntax that one can produce questions, interpret answers etc. In addition to that, we agree with Girard's "descending levels" description of the different approaches one can have when dealing with proofs:

> In level -1 we see proofs as mere "bureaucratic" artefacts, the only thing which matters being its existence. This means that we are not really interested in the mathematics of proofs, but more in the notion of *provability* inside some formal system.
>
> In level -2 we see proofs as programs and/or morphisms. This is where Curry-Howard-Lambek like correspondences typically happen. Historically, the first idea of this level can be traced back in the so called "Brower-Heyting-Kolmogorov interpretation" of proofs.
>
> In level -3 we see proofs as interactive processes, typically in a sort of "dialogue". Typical examples of this point of view are Game Semantics, Ludics, Geometry of Interaction, but also Krivine's classical realizability.

A very nice and clear explanation of these levels can be found in [Tro07].

Trascendental mathematical logic While the technical work one does in the previous two items is mathematical, here we mean it to be philosophical. The questions we mean to investigate are those who are left, from the previous items, with respect to the nature of mathematics. Let us take as "definition" the following:

> *What is maths? I think it can basically be classified into four types of thing. There are definitions, true/false statements, proofs, and ideas.*
> - Buzzard, in
> `https://xenaproject.wordpress.com/2020/06/20/mathematics-in-type-theory/`

Identifying Buzzard's expression "true/false statements" with "mathematical language in general", we can say that we are only left with the nature of "definitions"[51], and the nature of "ideas", which we will both refer to with the word *intuitions*.

We choose the name "transcendental mathematical logic" because our idea is to develop arguments in a "Kantian style", in order to treat the status of mathematical *concepts*. Let us just mention the points which we believe important in such a reflection:

- First of all, one has to describe how mathematics is created in our mind[52]. Our feeling is that a good way of proceeding is by considering Kant's theory of knowledge – "from empirical sensations to concepts by applying some transcendental categories of logic[53]").
- It is evidence that mathematical statements can be organized in the table[54]:

Quantity	Quality	Relation
$\forall,\exists$	$\neg$	$\wedge,\vee,\rightarrow$

[51] In the sentence we quoted, Buzzard was thinking mostly about a formal account of definitions, while here we intend it from a philosophical point of view.

[52] We mean here from a philosophical point of view, not from a neuroscience-like one.

[53] Here we explicitly change Kant's original expression "intellect" to "logic", in order to stress the fact that we are looking for an account of the logical/mathematical activity, not knowledge in general.

[54] Here the names Quantity, Quality and Relation are just there to preserve a continuity with Kant's terminology.

It is therefore quite tempting to consider the transcendental forms of logic as the rules of, for instance, the formal system NK (or LK, etc). But we know that we can modify these systems: typically, a deeper logical analysis is provided via the tool of linear logic and obtaining thus the following table:

Quantity	Quality	Relation	Modality
$\forall,\exists$	$()^{\perp}$	$\otimes,\&,\parr,\oplus,\multimap$	!,?

which comes together with the proof rules given by LL sequent calculus, or LL proof nets. This is a novelty w.r.t. Kant's, where the transcendental forms are necessary and cannot be "decomposed".

One can now see why the name "transcendental syntax" is appropriate for denoting the conceptual organization of the study of mathematical reasoning: it can be seen as the (mathematical and philosophical) study of the transcendental forms of the logic.

- The study of *intuition* in mathematics. By that we mean the fact that in mathematics one usually follows two "parallel" paths: we have an intuitive argument in mind and we follow it in order to create a formal one. These two paths are of a *different* nature, since one is in our mind and basically formulated in an informal language, while the other is on the paper and formulated in such a way that it could be translated in a formal language. The fact of reproducing, in a a formal way, intuitive considerations is not always a trivial act, as it can be for simple mathematical concepts. For instance, consider what Buzzard says:

  > *I also have a picture in my head of an overconvergent modular form defined on a neighbourhood of the ordinary locus on a p-adic modular curve. This picture informed several papers I wrote earlier this century with Richard Taylor, Frank Calegari, and others. I was once privileged to be invited to speak in the number theory seminar at Orsay in Paris, and Jean-Pierre Serre was in the audience. I drew one of these pictures of mine on the board and Serre interrupted! He asked what the picture meant. I had drawn a picture of a compact Riemann surface of genus 3 and was drawing discs and annuli on the Riemann surface. [...] However, my Annals of Mathematics paper with Taylor and my follow-up Journal of the AMS single-author paper (which I was lecturing on at the time) were all evidence that my way of thinking about things, the pictures in my head, really could be translated down into rigorous mathematics, even though they were in some sense meaningless. They were effective guides. [...] In short, I knew 'how far one could push the picture' in some sense - which bits of it to take seriously.*
  > — Buzzard, in
  > `https://xenaproject.wordpress.com/2021/01/21/formalising-mathematics-an-introduction/`

- If the creation of mathematical concepts follows the same process as Kant's theory of knowledge, then it must be explained how mathematics is different from metaphysics. In fact, both mathematical and metaphysical concepts would appear from the application of some transcendental forms to objects which are not necessary directly derived by experience – even if both of them are necessary generated by it. We believe that the solution is to consider again Kant, in particular its idea of *schematism*: mathematical concepts can be "schematized", while metaphysical ones cannot. Here we mean a sort of revisited notion of (what we understand as) Kantian schematism. In particular, one has to stress the idea of a schema as a "rule"; today, we would be tented to say a *program*. Better said, what makes mathematics not metaphysics, is the possibility for it to be formalized into a programming language (such as the usual formal systems, PA, ZF, System F, HoTT, ...); whence, the possibility of actually applying the transcendental forms not to the intuitions one has in mind, but on the concrete formal language (which, appearing on the paper, is an object of direct experience).

Let us make a last consideration.

Observe that, once mathematics is formalized, it looks like an "explicitation" of the theorems from the axioms through purely formal manipulations; let us call this the *analytical moment*. In the creation of these explicitations one is guided by intuition, in what one could call the *intuitionistic moment*. A crucial situation, in this "dialogue" between intuition and formalities, is when one stumbles upon an undecidable question, and one has thus to *decide* if to take it as a new axiom, or not; let us call this the *synthetic moment*. In the analytic moment one can only obtain theorems which are either provable or disprovable inside the formal system one has decided to stay in. That is, only *decidable* formulas (w.r.t. the fixed system). Here the use of the terms "analytic" and "synthetic" should *not* be seen in the Girardian sense, but in the Kantian one: in a sense, the information given by a decidable formula is already "contained" in the axioms of the formal system, while the one of an undecidable formula is not.

From this point of view, the undecidability of interesting formal systems (that is, incompleteness) is the heart of the "mathematical progress"[55], and the mathematics activity has the following alternating shape[56]:

Synthetic moment; Analytic moment; Synthetic moment; Analytic moment; ...

We recently found out that this point of view is quite close to the one expressed by the physician Müller in [Mül21].

A natural question appears: how should one decide if to take or not an undecidable formula? Evidence from the mathematical practice indicates that the solution is usually made according to: either utility (in the sense of considering the results one can obtain with it); or experiments (typically, this occurs when one does applied mathematics). Remark that this is quite far from a Platonist view of mathematics, in which, being mathematical objects "really there" together with their true or false value, one cannot "choose" about an undecidable formula, but only "discover" if it is true or false[57]. On the contrary, this vision is much more on the lines of, for instance, Poincaré's conventionalism.

We can thus make another[58] "Kantian-like table":

	A posteriori	A priori
Analytic moment	/	Proofs
Synthetic moment	Experiments	Utility

Here the distinction *a priori/a posteriori* is with respect to experience. It is interesting to note that, as Kant's epistemological problem was how to justify his *synthetic a priori*, and as Girard's logical problem is how to justify his *synthetic a priori* (the usage), here the problem is again how to justify the a *synthetic a priori* moment (and it is handled considering the utility).

[55] Quite close to the famous Poincaré's quote from "Science et méthode": *C'est par la logique qu'on démontre, c'est par l'intuition qu'on invente.*

[56] Famous examples of undecidable (with respect to different formal systems) formulas which make the progress of mathematics go, are: Euclid's V postulate w.r.t. the first four; induction axiom w.r.t. the other Peano's axioms; axiom of choice w.r.t. ZF; the continuum hypothesis w.r.t. ZFC.

[57] A nice discussion which is, in a certain sense, precisely about the distinction between these two approaches, can be found in [Deh07] in relation with the works of Woodin about the Continuum Hypothesis.

[58] The last one!

Bibliography

[AC98] Roberto M Amadio and Pierre-Louis Curien. *Domains and lambda-calculi*. Number 46. Cambridge University Press, 1998.

[Acc17] Beniamino Accattoli. (in)efficiency and reasonable cost models. In Sandra Alves and Renata Wasserman, editors, *12th Workshop on Logical and Semantic Frameworks, with Applications, LSFA 2017, Brasília, Brazil, September 23-24, 2017*, volume 338 of *Electronic Notes in Theoretical Computer Science*, pages 23–43. Elsevier, 2017.

[Bar84] Hendrik Pieter Barendregt. *The lambda calculus - its syntax and semantics*, volume 103 of *Studies in logic and the foundations of mathematics*. North-Holland, 1984.

[BCS09] R. Blute, J. Cockett, and R. Seely. Cartesian differential categories. *Theory and Applications of Categories*, pages 622–672, 2009.

[BEM12] Antonio Bucciarelli, Thomas Ehrhard, and Giulio Manzonetto. A relational semantics for parallelism and non-determinism in a functional setting. *Annals of Pure and Applied Logic*, 163(7):918–934, 2012. The Symposium on Logical Foundations of Computer Science 2009.

[Ben74] Edward A Bender. Partitions of multisets. *Discrete Mathematics*, 9(4):301–311, 1974.

[Ber78] Gérard Berry. Stable models of typed lambda-calculi. In Giorgio Ausiello and Corrado Böhm, editors, *Automata, Languages and Programming, Fifth Colloquium, Udine, Italy, July 17-21, 1978, Proceedings*, volume 62 of *Lecture Notes in Computer Science*, pages 72–89. Springer, 1978.

[BM20] Davide Barbarossa and Giulio Manzonetto. Taylor subsumes Scott, Berry, Kahn and Plotkin. *Proc. ACM Program. Lang.*, 4(POPL):1:1–1:23, 2020.

[Bon07] Denis Bonnay. Règles et signification: le point de vue de la logique classique. *Logique, dynamique et cognition*, pages 213–231, 2007.

[Bou93] Gérard Boudol. The lambda-calculus with multiplicities (abstract). In Eike Best, editor, *CONCUR '93, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 23-26, 1993, Proceedings*, volume 715 of *Lecture Notes in Computer Science*, pages 1–6. Springer, 1993.

[BP11] Richard Blute and Prakash Panangaden. Proof nets as formal Feynman diagrams. *Lecture Notes in Physics*, 813:437–466, 01 2011.

[Cho19] Jules Chouquet. Taylor expansion, finiteness and strategies. In Barbara König, editor, *Proceedings of the Thirty-Fifth Conference on the Mathematical Foundations of Programming Semantics, MFPS 2019, London, UK, June 4-7, 2019*, volume 347 of *Electronic Notes in Theoretical Computer Science*, pages 65–85. Elsevier, 2019.

[CL19] J. R. B. COCKETT and J.-S. LEMAY. Integral categories and calculus categories. *Mathematical Structures in Computer Science*, 29(2):243–308, 2019.

[CT20] Jules Chouquet and Christine Tasson. Taylor expansion for call-by-push-value. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic, CSL 2020, January 13-16, 2020, Barcelona, Spain*, volume 152 of *LIPIcs*, pages 16:1–16:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[Cur41] Haskell B. Curry. The paradox of Kleene and Rosser. *Transactions of the American Mathematical Society. 50 (3): 454-516*, 1941.

[Dav82] Martin Davis. Why Gödel didn't have Church's thesis. *Information and Control*, 54(1):3–24, 1982.

[dC18] Daniel de Carvalho. Execution time of λ-terms via denotational semantics and intersection types. *Math. Struct. Comput. Sci.*, 28(7):1169–1203, 2018.

[Deh07] Patrick Dehornoy. Au-delà du forcing : la notion de vérité essentielle en théorie des ensembles. *Logique, dynamique et cognition*, pages 147–169, 2007.

[DJS97] Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx. A new deconstructive logic: Linear logic. *J. Symb. Log.*, 62(3):755–807, 1997.

[DP01] René David and Walter Py. Lambda-mu-calculus and Böhm's theorem. *J. Symb. Log.*, 66(1):407–413, 2001.

[Dum75] Michael Dummett. The philosophical basis of intuitionistic logic. In *Studies in Logic and the Foundations of Mathematics*, volume 80, pages 5–40. Elsevier, 1975.

[Dža17] Mirna Džamonja. Set theory and its place in the foundations of mathematics: A new look at an old question. *Journal of Indian Council of Philosophical Research*, 34(2):415–424, 2017.

[EdV08] Jörg Endrullis and Roel C. de Vrijer. Reduction under substitution. In Andrei Voronkov, editor, *Rewriting Techniques and Applications, 19th International Conference, RTA 2008, Hagenberg, Austria, July 15-17, 2008, Proceedings*, volume 5117 of *Lecture Notes in Computer Science*, pages 425–440. Springer, 2008.

[Ehr02] Thomas Ehrhard. On Köthe sequence spaces and linear logic. *Mathematical Structures in Computer Science*, 12(5):579â€“623, 2002.

[Ehr16] Thomas Ehrhard. An introduction to differential linear logic: proof-nets, models and antiderivatives. *CoRR*, abs/1606.01642, 2016.

[ER03] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theor. Comput. Sci.*, 309(1-3):1–41, 2003.

[ER06a] Thomas Ehrhard and Laurent Regnier. Böhm trees, Krivine's machine and the Taylor expansion of lambda-terms. In Arnold Beckmann, Ulrich Berger, Benedikt Löwe, and John V. Tucker, 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, 2006.

[ER06b] Thomas Ehrhard and Laurent Regnier. Differential interaction nets. *Theor. Comput. Sci.*, 364(2):166–195, 2006.

[ER08] Thomas Ehrhard and Laurent Regnier. Uniformity and the Taylor expansion of ordinary lambda-terms. *Theor. Comput. Sci.*, 403(2-3):347–372, 2008.

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

[Gir96] Jean-Yves Girard. Proof-nets: The parallel syntax for proof-theory. In *Logic and Algebra*, pages 97–124. Marcel Dekker, 1996.

[Gir99] Jean-Yves Girard. On the meaning of logical rules I: syntax versus semantics. In *Computational logic*, pages 215–272. Springer, 1999.

[Gir00] Jean-Yves Girard. *Du pourquoi au comment: la théorie de la démonstration de 1950 à nos jours.* Les mathématiques 1950-2000, pp. 515-545, Birkhauser, 2000.

[Gir11] Jean-Yves Girard. La syntaxe transcendantale, manifeste. Unpublished note, 2011.

[Gir12] Jean-Yves Girard. Transcendental syntax 2.0. *Notes du cours donné à "Logique et Interaction 2012"*, 2012.

[Gir18] Jean-Yves Girard. Titres et travaux. Unpublished note, 2018.

[Gir21] Jean-Yves Girard. Schrödinger's cut. Unpublished note, 2021.

[GLT89] J.Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types.* Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, 1989.

[GPdF16] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Computing connected proof(-structure)s from their Taylor expansion. In Delia Kesner and Brigitte Pientka, editors, *1st International Conference on Formal Structures for Computation and Deduction, FSCD 2016, June 22-26, 2016, Porto, Portugal*, volume 52 of *LIPIcs*, pages 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.

[GPdF19] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Proof-net as graph, Taylor expansion as pullback. In Rosalie Iemhoff, Michael Moortgat, and Ruy J. G. B. de Queiroz, editors, *Logic, Language, Information, and Computation - 26th International Workshop, WoLLIC 2019, Utrecht, The Netherlands, July 2-5, 2019, Proceedings*, volume 11541 of *Lecture Notes in Computer Science*, pages 282–300. Springer, 2019.

[GPdF20] Giulio Guerrieri, Luc Pellissier, and Lorenzo Tortora de Falco. Glueability of resource proof-structures: Inverting the Taylor expansion. In Maribel Fernández and Anca Muscholl, editors, *28th EACSL Annual Conference on Computer Science Logic, CSL 2020, January 13-16, 2020, Barcelona, Spain*, volume 152 of *LIPIcs*, pages 24:1–24:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[Gri90] Timothy Griffin. A formulae-as-types notion of control. In Frances E. Allen, editor, *Conference Record of the Seventeenth Annual ACM Symposium on Principles of Programming Languages, San Francisco, California, USA, January 1990*, pages 47–58. ACM Press, 1990.

[Hat02] Allen Hatcher. *Algebraic topology.* Cambridge University Press, Cambridge, 2002.

[Her95] Hugo Herbelin. *Séquents qu'on calcule: de l'interprétation du calcul des séquents comme calcul de lambda-termes et comme calcul de stratégies gagnantes. (Computing with sequents: on the interpretation of sequent calculus as a calculus of lambda-terms and as a calculus of winning strategies)*. PhD thesis, Paris Diderot University, France, 1995.

[Hug06] Dominic J.D. Hughes. Proofs without syntax. *Annals of Mathematics*, 164(3):1065–1076, 2006.

[Hyl76] Martin Hyland. A Syntactic Characterization of the Equality in Some Models for the Lambda Calculus. *Journal of the London Mathematical Society*, s2-12(3):361–370, 02 1976.

[JR12] Bart Jacobs and Jan Rutten. An introduction to (co)algebra and (co)induction. In Davide Sangiorgi and Jan J. M. M. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, volume 52 of *Cambridge tracts in theoretical computer science*, pages 38–99. Cambridge University Press, 2012.

[KMP20] Emma Kerinec, Giulio Manzonetto, and Michele Pagani. Revisiting call-by-value Böhm trees in light of their Taylor expansion. *Log. Methods Comput. Sci.*, 16(3), 2020.

[KR35] S. C. Kleene and J. B. Rosser. The inconsistency of certain formal logics. *Annals of Mathematics*, 36(3):630–636, 1935.

[Kri09] Jean-Louis Krivine. Realizability in classical logic. *Panoramas et synthèses*, 27:197–229, 2009.

[Kri20] Jean-Louis Krivine. A program for the full axiom of choice, 2020.

[Las99] S.B. Lassen. Bisimulation in untyped lambda calculus:: Böhm trees and bisimulation up to context. *Electronic Notes in Theoretical Computer Science*, 20:346–374, 1999. MFPS XV, Mathematical Foundations of Progamming Semantics, Fifteenth Conference.

[Lau99] Olivier Laurent. Polarized proof-nets: Proof-nets for LC. In Jean-Yves Girard, editor, *Typed Lambda Calculi and Applications, 4th International Conference, TLCA'99, L'Aquila, Italy, April 7-9, 1999, Proceedings*, volume 1581 of *Lecture Notes in Computer Science*, pages 213–227. Springer, 1999.

[Lau03] Olivier Laurent. Polarized proof-nets and $\lambda\mu$-calculus. *Theor. Comput. Sci.*, 290(1):161–188, 2003.

[Lau04] Olivier Laurent. On the denotational semantics of the untyped lambda-mu calculus. Unpublished note, January 2004.

[LL19a] Ugo Dal Lago and Thomas Leventis. On the Taylor Expansion of Probabilistic lambda-terms. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction (FSCD 2019)*, volume 131 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 13:1–13:16, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[LL19b] Ugo Dal Lago and Thomas Leventis. On the Taylor expansion of probabilistic lambda-terms. In Herman Geuvers, editor, *4th International Conference on Formal Structures for Computation and Deduction, FSCD 2019, June 24-30, 2019,*

Dortmund, Germany, volume 131 of *LIPIcs*, pages 13:1–13:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[Lon03] Giuseppe Longo. Some topologies for computations. *Invited Lecture, proceedings of Géométrie au XX siècle, 1930 - 2000 (2001), Hermann, Paris, 2004.*, 2003.

[LZ12] Ugo Dal Lago and Margherita Zorzi. Probabilistic operational semantics for the lambda calculus. *RAIRO Theor. Informatics Appl.*, 46(3):413–450, 2012.

[Mad19] Penelope Maddy. What do we want a foundation to do? In Stefania Centrone, Deborah Kant, and Deniz Sarikaya, editors, *Reflections on the Foundations of Mathematics: Univalent Foundations, Set Theory and General Thoughts*, pages 293–311, Cham, 2019. Springer International Publishing.

[Man09] Giulio Manzonetto. A general class of models of $\mathcal{H}^*$. In Rastislav Královic and Damian Niwinski, editors, *Mathematical Foundations of Computer Science 2009, 34th International Symposium, MFCS 2009, Novy Smokovec, High Tatras, Slovakia, August 24-28, 2009. Proceedings*, volume 5734 of *Lecture Notes in Computer Science*, pages 574–586. Springer, 2009.

[Maz21] Damiano Mazza. An axiomatic notion of approximation for programming languages and machines, 2021. Unpublished note.

[MPV17] Damiano Mazza, Luc Pellissier, and Pierre Vial. Polyadic approximations, fibrations and intersection types. *Proc. ACM Program. Lang.*, 2(POPL), December 2017.

[MR14] Giulio Manzonetto and Domenico Ruoppolo. Relational graph models, taylor expansion and extensionality. In Bart Jacobs, Alexandra Silva, and Sam Staton, 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, 2014.

[Mül21] Markus P. Müller. Undecidability and unpredictability: Not limitations, but triumphs of science. In Anthony Aguirre, Zeeya Merali, and David Sloan, editors, *Undecidability, Uncomputability, and Unpredictability*, pages 5–16, Cham, 2021. Springer International Publishing.

[OA20] Federico Olimpieri and Lionel Vaux Auclair. On the taylor expansion of lambda-terms and the groupoid structure of their rigid approximants. *CoRR*, abs/2008.02665, 2020.

[Par91] Michel Parigot. Free deduction: An analysis of "computations" in classical logic. In Andrei Voronkov, editor, *Logic Programming, First Russian Conference on Logic Programming, Irkutsk, Russia, September 14-18, 1990 - Second Russian Conference on Logic Programming, St. Petersburg, Russia, September 11-16, 1991, Proceedings*, volume 592 of *Lecture Notes in Computer Science*, pages 361–380. Springer, 1991.

[Par92] Michel Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In Andrei Voronkov, editor, *Logic Programming and Automated Reasoning,International Conference LPAR'92, St. Petersburg, Russia, July 15-20, 1992, Proceedings*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[Plo74] Gordon D. Plotkin. The lambda-calculus is omega-incomplete. *J. Symb. Log.*, 39(2):313–317, 1974.

[Plo75] Gordon D. Plotkin. Call-by-name, call-by-value and the lambda-calculus. *Theor. Comput. Sci.*, 1(2):125–159, 1975.

[Py98] Walter Py. Confluence en lambda-mu-calcul. *Ph.D.Thesis, Université de Savoie*, 1998.

[Sau10] Alexis Saurin. Standardization and Böhm Trees for $\Lambda\mu$-Calculus. In Matthias Blume, Naoki Kobayashi, and Germán Vidal, editors, *Functional and Logic Programming, 10th International Symposium, FLOPS 2010, Sendai, Japan, April 19-21, 2010. Proceedings*, volume 6009 of *Lecture Notes in Computer Science*, pages 134–149. Springer, 2010.

[Sau12] Alexis Saurin. Böhm theorem and Böhm trees for the $\Lambda\mu$-calculus. *Theoretical Computer Science*, 435:106 – 138, 2012. Functional and Logic Programming.

[SB99] Richard Statman and Henk Barendregt. Applications of Plotkin-terms: Partitions and morphisms for closed terms. *J. Funct. Program.*, 9(5):565–575, 1999.

[Sco93] Dana S. Scott. A type-theoretical alternative to ISWIM, CUCH, OWHY. *Theoretical Computer Science*, 121(1):411–440, 1993.

[Sel03] Peter Selinger. From continuation passing style to Krivine's abstract machine. Manuscript, 23 pages, 2003.

[TAO17] Takeshi Tsukada, Kazuyuki Asada, and C.-H. Luke Ong. Generalised species of rigid resource terms. In *2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017.

[Tro07] Samuel Tronçon. Interaction et signification. *Logique, dynamique et cognition*, pages 119–145, 2007.

[TS19] Andrea Aler Tubella and Lutz Straßburger. Introduction to Deep Inference. Lecture, August 2019.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[Var15] Moshe Y. Vardi. Why doesn't ACM have a SIG for theoretical computer science? *Commun. ACM*, 58(8):5, July 2015.

[Vau07a] Lionel Vaux. *λ-calcul différentiel et logique classique : interactions calculatoires. (Differential λ-calculus and classical logic : their computational interactions)*. PhD thesis, University of the Mediterranean, Marseille, France, 2007.

[Vau07b] Lionel Vaux. The differential $\lambda\mu$-calculus. *Theor. Comput. Sci.*, 379(1-2):166–209, 2007.

[Vau19] Lionel Vaux. Normalizing the Taylor expansion of non-deterministic λ-terms, via parallel reduction of resource vectors. *Log. Methods Comput. Sci.*, 15(3), 2019.

[vD80] Dirk van Daalen. The language theory of Automath. *Ph.D. thesis. Technical University Eindhoven*, 1980.

[Wad76] Christopher P. Wadsworth. The relation between computational and denotational properties for Scott's $\mathcal{D}_\infty$-models of the lambda-calculus. *SIAM J. Comput.*, 5(3):488–521, 1976.

[Wad03] Philip Wadler. Call-by-value is dual to call-by-name. In Colin Runciman and Olin Shivers, editors, *Proceedings of the Eighth ACM SIGPLAN International Conference on Functional Programming, ICFP 2003, Uppsala, Sweden, August 25-29, 2003*, pages 189–201. ACM, 2003.

List of Figures

Index