Game semantics and realizability for classical logic

Valentin Blot

THÈSE

en vue de l'obtention du grade de
Docteur de l'Université de Lyon, délivré par l'École Normale Supérieure de Lyon

Discipline : Informatique

Laboratoire de l'Informatique du Parallélisme
École Doctorale InfoMaths (ED512)

Game semantics and realizability for classical logic

---

présentée et soutenue publiquement le vendredi 7 novembre 2014 par

Valentin Blot

Directeur : Olivier Laurent
Encadrant : Colin Riba

Devant la commission d'examen formée de :

Pierre-Louis Curien	CNRS, Université Paris Diderot - Paris 7	*Président*
Pierre Hyvernat	Université de Savoie	*Examinateur*
Olivier Laurent	CNRS, École Normale Supérieure de Lyon	*Directeur*
Andrzej Murawski	University of Warwick	*Rapporteur*
Colin Riba	École Normale Supérieure de Lyon	*Encadrant*
Thomas Streicher	Technische Universität Darmstadt	*Rapporteur*

Abstract

This thesis investigates two realizability models for classical logic built on HO game semantics. The main motivation is to have a direct computational interpretation of classical logic, arithmetic and analysis with programs manipulating a higher-order store.

Relaxing the innocence condition in HO games provides higher-order references, and dropping the well-bracketing of strategies reveals the CPS of HO games and gives a category of continuations in which we can interpret Parigot's lambda-mu calculus. This permits a direct computational interpretation of classical proofs from which we build two realizability models.

The first model is orthogonality-based, as the one of Krivine. However, it is simply-typed and first-order. This means that we do not use a second-order coding of falsity, and extraction is handled by considering realizers with a free mu-variable. We provide a bar-recursor in this model and prove that it realizes the axiom of dependent choice, relying on two consequences of the CPO structure of the games model: every function on natural numbers (possibly non computable) exists in the model, and every functional on sequences is Scott-continuous. Usually, bar-recursion is used to intuitionistically realize the double negation shift and consequently the negative translation of the axiom of choice. Here, we directly realize the axiom of choice in a classical setting.

The second model relies on winning conditions and is very specific to the games model. A winning condition is a set of positions in a game which satisfies some coherence properties, and a strategy realizes a formula if all its positions are winning.

Résumé

Cette thèse étudie deux modèles de réalisabilité pour la logique classique construits sur la sémantique des jeux HO, interprétant la logique, l'arithmétique et l'analyse classiques directement par des programmes manipulant un espace de stockage d'ordre supérieur.

La non-innocence en jeux HO autorise les références d'ordre supérieur, et le non parenthésage révèle la CPS des jeux HO et fournit une catégorie de continuations dans laquelle interpréter le lambda-mu calcul de Parigot. Deux modèles de réalisabilité sont construits sur cette interprétation calculatoire directe des preuves classiques.

Le premier repose sur l'orthogonalité, comme celui de Krivine, mais il est simplement typé et au premier ordre. En l'absence de codage de l'absurdité au second ordre, une mu-variable libre dans les réaliseurs permet l'extraction. Nous définissons un bar-récurseur et prouvons qu'il réalise l'axiome du choix dépendant, utilisant deux conséquences de la structure de CPO du modèle de jeux: toute fonction sur les entiers (même non calculable) existe dans le modèle, et toute fonctionnelle sur des séquences est Scott-continue. La bar-récursion est habituellement utilisée pour réaliser intuitionnistiquement le « double negation shift » et en déduire la traduction négative de l'axiome du choix. Ici, nous réalisons directement l'axiome du choix dans un cadre classique.

Le second, très spécifique au modèle de jeux, repose sur des conditions de gain: des ensembles de positions d'un jeu munis de propriétés de cohérence. Un réaliseur est alors une stratégie dont les positions sont toutes gagnantes.

Acknowledgements

Tout d'abord un grand merci à Colin pour ces trois années d'encadrement. Tu as su être extrêmement disponible, notamment dans les moments difficiles que vit tout thésard, et je mesure pleinement l'effort fourni pour me faire comprendre patiemment des concepts complexes malgré ma novicité scientifique. Je pense que cette première expérience d'encadrement est un franc succès et j'espère que tu auras l'occasion d'en faire profiter encore beaucoup d'autres après moi. Merci également à Olivier, tout d'abord pour ton rôle de directeur administratif, mais aussi et surtout pour être allé régulièrement au-delà de ce seul rôle en me faisant profiter de ton expertise scientifique, de la sémantique des jeux au lambda-mu calcul. Cette présence a sans aucun doute été un vrai plus tout au long de mon travail.

I would like to warmly thank my referees. Thomas, you gave very relevant remarks on my work. Every exchange we had allowed me to consider a new point of view and stimulated my scientific thirst. I thank you for the interest you showed regarding to my work. Andrzej, thank you for the time you spent reading my thesis, witnessed by your very accurate understanding of my explanations, despite their intricacy. I was honoured to have the opportunity to present you my work.

Merci à Pierre-Louis d'avoir accepté de faire partie de mon jury. Ton large spectre de connaissances apporte assurément une grande valeur à l'évaluation de mon travail, et j'admire ta capacité à rester malgré tout toujours accessible. Pierre, c'est un honneur de te compter parmi les membres de mon jury et de profiter de ton expertise, et merci pour ta permanente ouverture à la discussion.

Je remercie également tous les membres du labo, ma présence au LIP restera une expérience inoubliable. Merci aux plumeux, Alexandre pour nos nombreuses discussions scientifiques ou non et pour son éternelle bonne humeur, Daniel ce sacré boute-en-train, Pierre le sémanticien des jeux improvisé career advisor, Damien et nos échanges de scies, Patrick pour son organisation sans faille, et tous les autres membres. Merci à Catherine pour sa gentillesse et son efficacité, à Damien et à tout le secrétariat qui fait un travail remarquable. Merci à JMpega, Paul « fin en humour...» Brunet, Maxime, Matthieu, Gaupy, Adeline, Sébastien, Thanos, Mickaël, Anupam, Simon.

Je remercie très fort ma famille. Bien que j'aie été par le passé très injuste avec vous, papa et maman, vous êtes toujours restés fiers de moi et votre soutien m'a sans cesse aidé à avancer et à m'accomplir pleinement. Merci à Flo pour l'extrême bienveillance dont tu as toujours fait preuve envers moi, merci à Chris pour ta toujours très communicative bonne humeur, merci à Dine pour ton omniprésence malgré l'éloignement géographique, et merci à Antoine et Victor. Merci également à Kader, Blandine et Hugues, et merci à Inès, Elijah, Naël et [...], pour être là (ou bientôt là), simplement. Merci à Camille pour tous les moments partagés, passés, présents et à venir, les bons et les plus difficiles, grâce à toi je me sens exister un peu plus chaque jour. Merci également à Val pour me supporter sans répit depuis maintenant presque 28 ans.

Merci à Aurèle et aux parisiens, et merci au gang des lyonnais, Sam, Ju, Olivier, Grovaro, pour notre année d'agreg inoubliable et pour tout le reste. Merci enfin à Blanche, Nor & Guigui Grosmouly, Guilhem, Séb, Nath, Pauline, Martin et tous ceux que j'oublie.

Contents

Chapter 1

Introduction

This thesis investigates the computational content of classical logic, arithmetic and the axiom of choice. In order to extract computational content from proofs we use the techniques of realizability, that is a formalization of the Brouwer-Heyting-Kolmogorov interpretation. Moreover, this computational content is exploited in the model of Hyland-Ong-Nickau game semantics which provides control features (used to handle classical reasoning) and higher-order references (which could be later exploited to provide new realizers).

1.1 Logic

Logic finds its very first roots in ancient Egypt and Babylonia. Later on, it was mainly developed in ancient Greece, India and China around the 5th century BC. In the field of geometry, the Pythagoreans developed a notion of basic true propositions (which we now call axioms) from which one can formally derive all true propositions of a system. The principle of reductio ad absurdum (if we can derive an impossible conclusion from a given proposition then this proposition is false) also appeared in the same era. In political debates, Sophists exploited the perception of logic as a universal principle in order to persuade an audience by hiding false logical principles behind an apparently trivial statement. Most of what we know from the logic at that time comes from Plato's work. Aristotle studied for twenty years at Plato's Academy and made a significant work on logic, introducing the syllogism. The most famous example of a syllogism is: "All men are mortal, Socrates is a man, therefore Socrates is mortal." However, without a formal system one can easily build false syllogisms, for example: "what is rare is expensive, a cheap horse is rare, therefore a cheap horse is expensive." This illustrates that logic admits no "common-sense" argument and logic is empty without formalism.

During post-classical history, advances were made in the Middle-east and in Europe, mainly following the lines of Aristotelian logic, the philosophical branch of logic reaching a peak with "The Science of Logic" by Hegel in 1816. In the 17th century the field of symbolic logic emerged, with Leibniz arguing on the importance of symbolic notations in mathematics, and more specifically in calculus. This idea appeared to be fundamental in modern logic, for example in the works of Turing and Gödel in the 1930's. This led to a revival of logic, which was no more a philosopher's concern, but more and more an area of mathematics. During the 19th century, Boole and De Morgan followed this way and undertook an algebraization of logic. They were followed by Peirce who took as an axiom the formula $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$ (which is now called Peirce's law). This formula allows classical reasoning just as the law of excluded middle, and plays a crucial role in the computational interpretation of classical logic.

The concept of quantification ($\exists$ and $\forall$) first appeared in Frege's Begriffsschrift [Fre79] in

1879, who used a two-dimensional notation for these. This concept allowed him to build an axiomatic system which is still at the heart of modern logic, with rules of modus ponens (from A and $A \Rightarrow B$ one can deduce B), generalization (if $A \Rightarrow B\,(x)$ holds and the variable x does not appear in A, then $A \Rightarrow \forall x\, B\,(x)$ holds) and substitution (from $\forall x\, A\,(x)$ one can derive $A\,(t)$ for any t). His work remained confidential until Russell shed the lights on it. Indeed, in 1874, Cantor had developed in [Can74] the first set theory (which we call now naive set theory), and proved that the sets of natural numbers and real numbers were not equinumerous, even though these are both infinite. Cantor later reproved this fact in [Can92] using a diagonal argument, which turned out to be a very fruitful technique which was later used by Turing in computability theory and by Gödel in proof theory. However, the most striking use of the diagonal argument is that of Russell, who used it to prove in 1901 the inconsistency of Cantor's own set theory (more precisely, of the axiomatization of his set theory in Frege's formal system). Russell's paradox relies on the fact that in naive set theory, one can define from any given property the set of elements that satisfy this property. The property of not being an element of ourself then gives a set x such that x belongs to x and x does not belong to x, hence the contradiction. This discovery put a cat among the pigeons, and two solutions emerged, which are still present in modern logic. The first solution came in 1908 from Zermelo [Zer08], who defined the axiom of separation: from a given property, we cannot construct the set of elements satisfying it (otherwise we would get Russell's paradox, and therefore inconsistency), however if we also have a fixed set E, then it is consistent to construct the set of elements of E for which the property holds. Zermelo set theory was improved later and became Zermelo-Frænkel set theory, which is a formal system in which most of mathematics can be formalized. The second solution came from a joint work of Russell and Whitehead who wrote between 1910 and 1913 the Principia Mathematica [RW13]. Russell's paradox is avoided in the Principia Mathematica by considering a hierarchy: a set is one level above its elements. Because of this hierarchy, it makes no sense to consider a set which belongs to itself. This concept of a hierarchy is still present in Martin-Löf's type theory and in the proof assistant Coq.

The foundational crisis of the beginning of 20th century was the trigger to theoretical computer science. In order to put an end to this crisis, Hilbert's program proposes to find a fixed set of axioms from which one could derive every theorem of mathematics. Hilbert believed that this set of axioms would be those of basic arithmetic, more or less Peano's axioms. Hilbert's program made sense only if one could prove that this basic set of axioms is consistent, and this was the subject of the second problem among the 23 that Hilbert formulated during the international congress of mathematicians in 1900 in Paris. In 1931, Gödel's incompleteness theorems [Göd31] put an end to this by stating that any effectively presentable consistent logical system which formalizes arithmetic contains undecidable formulas: formulas which cannot be proved to be true nor false, and the consistency of arithmetic is such a formula. Using a diagonal argument, Gödel proved that there exists no formal proof of the consistency of arithmetic inside the formal system of arithmetic. Gödel is probably the most influential logician of the 20th century, his main results being the completeness theorem of first-order logic [Göd29] (a formula is provable if and only if it is true in every model), the incompleteness of arithmetic [Göd31] (consistency of arithmetic cannot be proved in arithmetic) and the consistency of the axiom of choice and the continuum hypothesis with respect to Zermelo-Frænkel set theory [Göd40] (there exists a model of Zermelo-Frænkel set theory, the constructible universe, in which both hold).

After Gödel's answer to this quest for a universal system, new research areas emerged. First, in 1934-1935, Gentzen defined in [Gen35] two formal systems for writing mathematical proofs: natural deduction and sequent calculus. The former is useful to write formally mathematical proofs. The later, which is an equivalent but symmetric version of natural deduction, is hard to use for formalizing proofs. However, sequent calculus is particularly fit for proving Gentzen's

famous cut-elimination theorem (from any formal proof, one can eliminate all the intermediary lemmas and get a "minimal" proof). On another side, Church and Turing defined (independently) in 1936 the λ-calculus [Chu36] and the Turing machines [Tur37] to give a formal definition of the notion of algorithm. It appeared quickly that the two notions coincide (the same functions on natural numbers can be computed in the two models). Church and Turing used their models of computability to provide a negative answer to Hilbert's Entscheidungsproblem (finding an algorithm which decides for any property if it is true or not) that Hilbert formulated in 1928. Later, many other models of computation were proved equivalent to Church's and Turing's ones, which led to the Church-Turing thesis: every function which can be algorithmically computed (which is an informal notion) can be computed by (equivalently) a λ-term or a Turing Machine.

Proof theory and computability theory continued to evolve with discoveries of strong links between the two. First, in 1958, Curry observed in [CF58] that the axioms of Hilbert-style deduction systems for propositional logic correspond exactly to the types of the combinators of Schönfinkel's combinatory logic (a variable-free reformulation of Church's λ-calculus). In 1969, Howard went further and observed an exact correspondence between Gentzen's natural deduction and Church's simply typed λ-calculus (manuscript published in [How80]). This correspondence is now known as Curry-Howard isomorphism. In 1990, Griffin extended this isomorphism to classical logic in [Gri90] by typing the `call/cc` operator of the functional language Scheme with Peirce's law. The `call/cc` operator allows to manipulate continuations and it is similar to the exception handling of modern programming languages, though the precise mechanisms are different, as explained in [RT99]. Building on this extension of the isomorphism, several calculi with control features were defined, for example with Parigot's $\lambda\mu$-calculus [Par92] and Curien and Herbelin's $\overline{\lambda}\mu\widetilde{\mu}$-calculus [CH00]. These calculi allow the programmer to manipulate contexts, through context variables in $\lambda\mu$-calculus, and by explicitly writing them in $\overline{\lambda}\mu\widetilde{\mu}$-calculus.

1.2 Realizability

Realizability is a technique introduced by Kleene in [Kle45] in order to give a formal account of the so-called Brouwer-Heyting-Kolmogorov interpretation of proofs. To each formula we associate a set of realizers, which are programs which behave like a proof of the formula. The important thing is that we do not look at how these programs are written, but only at their behavior. Many notions of realizability exist, for different logics and using different sets of realizers, see [Tro98] for a survey.

Realizability should be put in parallel with the Curry-Howard isomorphism described in the previous section. Under this isomorphism, a proof corresponds exactly to one typed program, and the type of this program is derived from the formula. In a realizability model, the adequacy lemma states that every proof of a formula, seen as a program, is a realizer of that formula. However, the strength of realizability is that a realizer is not required to be such a proof. In some realizability models, a realizer may even be untypable. Under this view, realizability may be seen as a very loose notion of typing, the only requirement being that combining different realizers preserves truth.

Kleene's realizability model for Heyting arithmetic *HA* (the intuitionistic variant of Peano arithmetic *PA*) uses the set of recursive functions as realizers (using Gödel's encoding of these in the set of natural numbers). Using this model, he obtained the existence property: from a proof of an existential statement one can extract a concrete witness which satisfies the property. Another example is Kreisel's modified realizability [Kre59] for HA^ω, Heyting arithmetic in finite types, in which the realizers are typed programs. For a long time extraction of computational content from proofs was only considered for intuitionistic, constructive logic.

Using concepts very close to those of realizability, Gödel defined in [Göd58] the Dialectica interpretation of *HA* into system T (an extension of primitive recursive functions to higher-order types). A comparison with realizability can be found in [Oli08], by decomposing the interpretation through linear logic.

In [Gri90], Tim Griffin found out that the `call/cc` operator of the functional language Scheme could be typed with a classical logic principle: the law of Peirce. This led Krivine to build in [Kri01, Kri03, Kri09] a realizability model for classical logic so as to extract computational content from classical proofs.

1.3 Game semantics

1.3.1 Historical overview

A programming language is usually defined by first its syntax (what is a well-formed program) and second its operational semantics (how these programs compute) which is usually defined by an evaluation relation between programs. If moreover we fix a particular set of programs that we call values, we can define the observational equivalence of programs as: M and N are observationally equivalent if for any context C and value v, $C[M]$ evaluates to v if and only if $C[N]$ evaluates to v. When one defines a model of such a programming language, a very strong property is that of full abstraction: two programs are observationally equivalent if and only if their interpretations in the model are equal. For a deeper overview of the full abstraction problem for PCF, we recommend to the reader Curien's paper [Cur07], from which this introduction is inspired. For a domain-theoretic approach to the problem, we also recommend the book [Str06].

The quest for a fully abstract model of the functional language PCF started with the seminal work of Plotkin [Plo77] and Milner [Mil77]. Plotkin showed that the continuous model was fully abstract for an extension of PCF with a "parallel or" operator adding concurrent features to PCF, while Milner showed the uniqueness (up to isomorphism) of a fully abstract model of PCF, and constructed it by quotienting the syntax. The goal was then to present this unique model in a way that is as far from the syntax as possible.

A stepping stone was then Berry and Curien's model of sequential algorithms [BC82], which was proved to be fully abstract for an extension of PCF with a "catch" operator adding the possibility to change the control flow of a program.

Game semantics then appeared in the 90's, at the same time in [HO00, Nic94, AJM00], as a solution to the problem. The full abstraction property of these models is obtained from a model which satisfies the definability property by quotienting it with the observational equivalence relation (which is non-computable). For this reason, the strength of the games models is rather their definability property than their full abstraction with regard to PCF. These models are much more away from the syntax than that of Milner, even if not as much as the continuous model.

Later on, Loader's result of undecidability of finitary PCF [Loa01] proved that in fact one could not hope for an effective fully abstract model of this functional language. However, if we add references to PCF, the effective full abstraction result can be recovered for a version of Idealized Algol [AM96].

1.3.2 The concepts of game semantics

Eliminating the "parallel or" from the model requires to distinguish between programs that would evaluate their arguments in different orders. This leads to the necessity to take into account sequentiality in the model. For this reason, game semantics is based on the notion of play, which

is a sequence of moves representing an interaction between a program and its environments. The programs are then interpreted as strategies, which are sets of plays.

The plays constituting a strategy have in fact a very special structure: they are walks in the Böhm tree of the corresponding program, a Böhm tree being a representation of a normal form. The most technical part of game semantics is the handling of composition. Indeed, since a strategy can be seen as a normal form, composition of strategies amounts to compute a normal form from two normal forms, and the combinatorics of beta-reduction is transposed into composition.

A walk in a Böhm tree can be seen as an exploration of the tree by an environment (the opponent), the tree being described step-by-step by the program (the player). Since the tree is fixed in advance by the syntax of the program, the description of it by the player must not depend on the particular way the opponent explores it, but only on the particular branch that opponent is exploring. This condition is what game semanticists call "innocence" of strategies. Indeed, relaxing innocence provides a model of languages with references, and references can be interpreted as the possibility for a Böhm tree to change during the computation.

The model obtained is still a little too general: player may observe for example that opponent asks for a value but, instead of providing it, it may answer to a move appearing lower in the branch under exploration. This behavior must be understood as a kind of exception (though technically it is a syntactically scoped exception, like the `call/cc` operator of Scheme). Indeed, if we see the current branch as the call stack, it amounts to exit the current function without giving any answer, and go back lower in the stack. In order to forbid this, strategies of the fully abstract model of PCF are required to be "well-bracketed": a question must not be answered until all intermediary ones have been answered.

Finally, considering only innocent, well-bracketed and compact strategies, one obtains the definability result: any strategy is the image of a PCF program. Once one has the soundness and definability results, a simple quotient by the observational equivalence in the model leads to the full abstraction result.

1.3.3 The extensions of game semantics

In this thesis we are interested in providing a model of classical reasoning. As observed by Griffin, the `call/cc` operator of Scheme can be typed with the law of Peirce. Under the lights of the Curry-Howard isomorphism, this means that the computational content of classical reasoning lies in the notion of control. As seen earlier, this syntactic control, which is not present in PCF, was forbidden in the games model by requiring the strategies to be well-bracketed. Therefore, we consider strategies which may not be well-bracketed, providing us with a model of PCF+call/cc due to [Lai97].

On the other side, the innocence constraint can also be relaxed: a strategy may memorize which branches are under exploration by opponent, and in which order. This indeed was proved to give a model of a programming language with references (see [AHM98]). However, if we completely forget about innocence, the product type $\mathcal{A} \times \mathcal{B}$ of arenas $\mathcal{A}$ and $\mathcal{B}$ contains strategies which may benefit from the possibility to exchange information between $\mathcal{A}$ and $\mathcal{B}$, and therefore such a strategy is not entirely defined by its computations on $\mathcal{A}$ together with its computations on $\mathcal{B}$, and we do not have the uniqueness property of the pair of two strategies. In fact we get a symmetric monoidal closed category and not a cartesian closed one. If we want a model of lambda (and lambda-mu) calculus with $\beta\eta$-equivalence, we can not drop entirely the innocence condition. However, it is known that if we only require the strategy to be blind to the particular computation currently performed, then we get back the cartesian product. This condition is known as single threadedness, and allows strategies to remember what has been asked by opponent and in which

order, while making sure that this knowledge is reset each time a new computation starts.

1.4 Outline

Using ideas from both Kreisel's and Krivine's realizability, we devise in this thesis two notions of classical realizability in which realizers are typed programs. The logical system under consideration is classical analysis CA^ω (as it is called in [Koh08]), that is Peano arithmetic in finite types PA^ω augmented with the axiom of dependent choice. The weaker axiom of countable choice can then be used to recover the axiom scheme of full comprehension over numbers, and it was the main motivation for the study of CA^ω. Many theorems of analysis can be formalized in this system (see e.g. [Koh08]). Another possibility for formalizing theorems of analysis is to work in second-order logic, where comprehension is a feature of the logic itself, and this is the path followed by Krivine [Kri09]. The main challenge into building a realizability model of classical analysis is that we need to provide a computational interpretation for the axiom of choice. Indeed, the axiom of choice is validated in every intuitionistic realizability model, but when it comes to classical realizability it is known to raise some difficulties (see e.g. [BBC98]). The usual route is to perform a negative translation from classical to intuitionistic logic, and then to provide an intuitionistic realizer of the translation of the axiom of choice. This realizer can for instance be obtained using a variant of Spector's bar-recursion [Spe62] due to Berger and Oliva [BO06] which realizes the double-negation shift. Indeed, a variant of Gödel's negative translation can turn a proof of $PA^\omega \vdash A$ into a proof of $HA^\omega \vdash A^\neg$, where $A^\neg$ is inductively defined by adding a double negation in front of every atomic or existential formula (disjunction can be coded). Then the formula $DNS \Rightarrow A \Rightarrow A^\neg$ is provable in intuitionistic logic, where $DNS \equiv \forall x^\iota \neg\neg A \Rightarrow \neg\neg \forall x^\iota A$ is the principle of double-negation-shift. Therefore a realizability model for HA^ω can be turned into a realizability model of CA^ω by providing a realizer of DNS, which is done using bar-recursion in [BO05, BBC98]. Here we prove that Berger and Oliva's bar-recursor, when used in our classical realizability model, directly realizes the axiom of choice.

We present in chapter 2 a version of Gentzen's multi-conclusioned sequent calculus with a distinguished conclusion. Proofs in that calculus will then be interpreted as programs in chapter 3. We define a relativization predicate which will be used in the realizability models to realize the induction scheme and the axiom of choice. The relativized variables are the ones that are manipulated by the realizers. Then we define the theories of Peano arithmetic in finite types PA^ω and classical analysis in finite types CA^ω, by extending PA^ω with the axiom of dependent choice. We also define their relativized versions $PA^{\omega \mathrm{r}}$ and $CA^{\omega \mathrm{r}}$. Finally we give a quick reminder of first-order models.

In chapter 3 we present Parigot's $\lambda\mu$-calculus [Par92], an extension of λ-calculus with control primitives allowing for an interpretation of classical principles. We also present the translation of proofs of $CA^{\omega \mathrm{r}}$ into $\lambda\mu$-calculus. Then we describe Selinger's categories of continuations [Sel01], the interpretation of $\lambda\mu$-calculus in these, and the connection of this interpretation with the continuation-passing-style translation of $\lambda\mu$-calculus. Then we present the programming language μPCF as a particular $\lambda\mu$-theory. As an example, we encode Gödel's system T extended with control features of $\lambda\mu$-calculus into μPCF and describe the interpretation of $PA^{\omega \mathrm{r}}$ in it. Finally, we give an introduction to bar-recursion as a recursion operator on well-founded trees, we define formally Berger and Oliva's bar-recursion [BO06] in μPCF and we use it to give an interpretation of $CA^{\omega \mathrm{r}}$ in μPCF.

Chapter 4 is devoted to game semantics. We first give a technical introduction to the concepts of game semantics, then we present formally arenas, strategies, and their cartesian-closed structure. We focus on single-threaded, non-innocent, unbracketed strategies (the unbracketed-

ness providing a model for control features). Finally, we recall that the category $\mathcal{G}$ of arenas and strategies is a category of continuations and we describe the interpretation of μPCF in $\mathcal{G}$.

In chapter 5 we define our first realizability model. In this model, realizers of an implication are defined by a Kleene arrow in the category of continuations $\mathcal{G}$ (a realizer of $A \Rightarrow B$ is a strategy which maps realizers of A to realizers of B). Then, in order to prove the adequacy lemma, the set of realizers of a given formula is proved to be the orthogonal of a more primitive set of counter-realizers. In Krivine's realizability model [Kri09], the set of counter-realizers is primitive and the set of realizers is then defined to be its orthogonal. In particular, the realizers of an implication are not Kleene realizers. However, their η-expansion are, and moreover, a realizer of $A \Rightarrow B$ is a Kleene realizer if we take the reduction relation to be closed not only by anti-evaluation, but also by direct evaluation. We prove the adequacy lemma for the axioms of $CA^{\omega \mathrm{r}}$ using bar-recursion for the axiom of dependent choice. The realization of the axiom of choice uses two particular aspects of the HON games model: the fact that it contains all (in particular non-computable) functions on natural numbers, and the fact that every function on sequences expressible in it is continuous for the product topology on sequences. Finally, we prove the extraction result for Π^0_2 formulas derivable in CA^{ω}.

Our second realizability model is defined in chapter 6. In this model, each formula has an associated winning condition, and a strategy realizes a formula if its plays are winning for the associated winning condition. We first define positions as desequentialized plays, relying on Boudes' thick subtrees [Bou09]. Then we define winning conditions as sets of positions which are closed under weakening of opponent and strengthening of player. We then define for each formula of $PA^{\omega \mathrm{r}}$ a winning condition on the associated arena, define the realizability relation and prove the adequacy lemma for $PA^{\omega \mathrm{r}}$. We give ideas about the realization of the axiom of choice in this setting by discussing about a bar recursor which would use references. Finally we prove the extraction result: from every Π^0_2 formula provable in PA^{ω} we can extract an algorithm computing the function that it represents.

Chapter 2

Logic

We define in this chapter the logical system under which we will work through the thesis. First, we define the general case of classical multisorted first-order logic (handling classical reasoning by the use of multi-conclusioned sequents), then we describe the case of logics with equality and the relativization of proofs that we need for the realizability interpretation. We describe the case of Peano arithmetic and its extension with the axiom of dependent choice, and finally we recall some basic definitions about models of classical logic.

2.1 Classical multisorted first-order logic

The logical framework that we use is multisorted first-order logic, where the sorts are fixed to be the types of simply typed λ-calculus (see e.g. [TVD88]). We build from a set of base sorts ι the set of sorts:

$$T, U \quad ::= \quad \iota \mid T \to U$$

which are used for the individuals of the logic. We fix a set of sorted individual constants (ranged over by c^T) from which we build the set of individuals of the logic:

$$t^T, u^U \quad ::= \quad \mathsf{c}^T \mid x^T \mid \left(t^{T \to U} u^T\right)^U$$

We also fix a set of sorted predicates (ranged over by P) from which we define the formulas of the logic:

$$A, B \quad ::= \quad P\left(t_1^{T_1}, ..., t_n^{T_n}\right) \mid \bot \mid A \Rightarrow B \mid A \wedge B \mid \forall x^T A$$

Definition 2.1. *A signature Σ is a set of base sorts together with a set of sorted constant individuals and a set of sorted predicates.*

The negation is defined as $\neg A \stackrel{\Delta}{=} A \Rightarrow \bot$. We choose to have only negative connectives, since the interpretation of our logic in the category of games is based on a negative call-by-name continuation-passing-style translation. We define the positive connectives using the negative ones: $A \vee B \stackrel{\Delta}{=} \neg(\neg A \wedge \neg B)$ and $\exists x^T A \stackrel{\Delta}{=} \neg \forall x^T \neg A$. It is well-known that with this coding of positive connectives with negative ones, a formula A is provable in our system if and only if the formula obtained by replacing every $P(t_1, ..., t_n)$ with $\neg\neg P(t_1, ..., t_n)$ in A is provable in its intuitionistic restriction. Therefore, if in some axiomatic system one can intuitionistically prove $\neg\neg P(t_1, ..., t_n) \Rightarrow P(t_1, ..., t_n)$, then the system presented here becomes equivalent to its intuitionistic version.

Definition 2.2. *A theory on a given signature Σ is a set $\mathcal{A}x$ of closed formulas (axioms) written on the language defined by Σ.*

A context Γ or Δ is a finite unordered sequence of formulas and a sequent is of the form:

$$\Gamma \vdash A \mid \Delta$$

Such a sequent should be interpreted as: the conjunction of the formulas of Γ implies the disjunction of A and of the formulas of Δ. If Δ is empty we will simply write $\Gamma \vdash A$. The set of derivable sequents of a given theory is defined from $\mathcal{A}x$ using the following rules:

$$\frac{}{\Gamma, A \vdash A \mid \Delta} \qquad \frac{}{\Gamma \vdash A \mid \Delta}\ {\scriptstyle (A \in \mathcal{A}x)} \qquad \frac{\Gamma \vdash A \mid \Delta, A}{\Gamma \vdash \bot \mid \Delta, A} \qquad \frac{\Gamma \vdash \bot \mid \Delta, A}{\Gamma \vdash A \mid \Delta}$$

$$\frac{\Gamma, A \vdash B \mid \Delta}{\Gamma \vdash A \Rightarrow B \mid \Delta} \qquad \frac{\Gamma \vdash A \Rightarrow B \mid \Delta \qquad \Gamma \vdash A \mid \Delta}{\Gamma \vdash B \mid \Delta}$$

$$\frac{\Gamma \vdash A \mid \Delta \qquad \Gamma \vdash B \mid \Delta}{\Gamma \vdash A \wedge B \mid \Delta} \qquad \frac{\Gamma \vdash A_1 \wedge A_2 \mid \Delta}{\Gamma \vdash A_i \mid \Delta}$$

$$\frac{\Gamma \vdash A \mid \Delta}{\Gamma \vdash \forall x^T A \mid \Delta}\ {\scriptstyle (x^T \notin \mathrm{FV}(\Gamma, \Delta))} \qquad \frac{\Gamma \vdash \forall x^T A \mid \Delta}{\Gamma \vdash A\{t^T/x^T\} \mid \Delta}$$

In this system one can for example derive in any theory the following sequents:

$$\vdash \bot \Rightarrow A \qquad \vdash \neg(\neg A) \Rightarrow A \qquad \vdash ((A \Rightarrow B) \Rightarrow A) \Rightarrow A$$

The weakening rule:

$$\frac{\Gamma \vdash A \mid \Delta}{\Gamma' \vdash A \mid \Delta'}\ {\scriptstyle \Gamma \subseteq \Gamma', \Delta \subseteq \Delta'}$$

is admissible and will be used without mentioning it. The left contraction rule is derivable and the right one is admissible:

$$\frac{\dfrac{\Gamma, A, A \vdash B \mid \Delta}{\Gamma, A \vdash A \Rightarrow B \mid \Delta} \qquad \dfrac{}{\Gamma, A \vdash A \mid \Delta}}{\Gamma, A \vdash B \mid \Delta} \qquad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash A \mid B, B, \Delta}{\Gamma \vdash A \mid A, B, B, \Delta}\ {\scriptstyle B,B,\Delta \subseteq A,B,B,\Delta}}{\Gamma \vdash \bot \mid A, B, B, \Delta}}{\Gamma \vdash B \mid A, B, \Delta}}{\Gamma \vdash \bot \mid A, B, \Delta}}{\Gamma \vdash A \mid B, \Delta}$$

so we will also use them implicitly. If a sequent $\vdash A$ is derivable in a theory $\mathcal{A}x$, then since $\mathcal{A}x$ may contain infinitely many formulas we write to avoid confusion:

$$\mathcal{A}x \mathrel{|\!\sim} A$$

2.2 Equational theories

We say that a theory is an equational theory if it contains for each sort T an inequality predicate $\neq_T$ between terms of sort T (for which we use infix notation), and the following axioms for reflexivity and the Leibniz scheme:

$$\textbf{(refl)} \quad \forall x\,(x = x) \qquad \textbf{(Leib)} \quad \forall \vec{z} \forall x \forall y\,(\neg A \Rightarrow A\{y/x\} \Rightarrow x \neq y)$$

where $t = u \stackrel{\Delta}{=} \neg(t \neq u)$. As in [Kri09], we use a primitive inequality rather than equality, because inequality is a negative predicate as all other basic connectives of our logic, therefore we can easily realize the Leibniz scheme. Moreover, since we are in classical logic, the set of provable sequents is unchanged.

2.3 The relativization predicate

In the following we will consider two kinds of quantifications: the uniform and the relativized ones. From the point of view of provability in the two theories PA^ω and CA^ω considered in this thesis, this will not change anything. Indeed, we will define for PA^ω and CA^ω (which have uniform quantifications only) their relativized versions $PA^{\omega\mathrm{r}}$ and $CA^{\omega\mathrm{r}}$ (which have both uniform and relativized quantifications). Then we will describe a mapping from PA^ω (resp. CA^ω) to $PA^{\omega\mathrm{r}}$ (resp.$CA^{\omega\mathrm{r}}$), and we will perform our realizability interpretation in the relativized theories.

Relativization is a technique that was already used in Krivine's models [Kri94, Miq11], and its utility will appear more clearly in the realizability interpretation. A realizer of $\forall x\, A$ (uniform quantification) must be a realizer of $A\{a/x\}$ for every instance a of x, whereas a realizer of $\forall^{\mathrm{r}} x\, A$ (relativized quantification) must turn any instance a of x into a realizer of $A\{a/x\}$. The uniform quantifiers suffice to realize the rules of first-order logic and Leibniz equality, but when it comes to Peano arithmetic, and more particularly to the axiom scheme of induction, the recursor of Gödel's system T needs to know on which particular natural number to recurse. A simple and intuitive example arises when we use the induction scheme to perform case-analysis: we prove on one hand $A\{\mathsf{0}\}$, and on the other hand $A\{n\}$ for $n \neq \mathsf{0}$, so by case-analysis we get $\forall^{\mathrm{r}} x\, A$ (note the relativized quantification). Then the corresponding program must read the natural number, test if it is zero, and then branch to the corresponding program. One solution would have been to relativize all quantifiers, however relativizing only when it is necessary gives a better understanding of the realizability interpretation and simpler extracted programs.

We explain now how we map proofs of a theory into proofs of a relativized version of the theory. Fix a theory $\mathcal{A}x$ on a signature Σ. Let Σ' be the signature obtained by adding to Σ a predicate symbol for each base sort ι of Σ:

$$(\!|t^\iota|\!)$$

We lift it to every sort with the following syntactic sugar:

$$(\!|t^{T\to U}|\!) \triangleq \forall x^T\, (\!|x^T|\!) \Rightarrow (\!|t\, x^U|\!)$$

First, since $\Sigma \subseteq \Sigma'$, $\mathcal{A}x$ can be considered as a theory on Σ'. We also define syntactic sugar for relativized quantifications:

$$\forall^{\mathrm{r}} x^T\, A \triangleq \forall x^T\, \big((\!|x^T|\!) \Rightarrow A\big) \qquad \exists^{\mathrm{r}} x^T\, A \triangleq \neg\forall^{\mathrm{r}} x^T\, \neg A$$

and we define inductively the relativization A^{r} (on the signature Σ') of a formula A (on the signature Σ):

$$P\left(t_1^{T_1}, ..., t_n^{T_n}\right)^{\mathrm{r}} \triangleq P\left(t_1^{T_1}, ..., t_n^{T_n}\right) \qquad \bot^{\mathrm{r}} \triangleq \bot$$

$$(A \Rightarrow B)^{\mathrm{r}} \triangleq A^{\mathrm{r}} \Rightarrow B^{\mathrm{r}} \qquad (A \wedge B)^{\mathrm{r}} \triangleq A^{\mathrm{r}} \wedge B^{\mathrm{r}} \qquad \left(\forall x^T\, A\right)^{\mathrm{r}} \triangleq \forall^{\mathrm{r}} x^T\, A^{\mathrm{r}}$$

This translation is extended to contexts the obvious way, so we write Γ^{r} and Δ^{r}. The following lemma then gives the mechanism of translation from a uniform system into a relativized one.

Lemma 2.1. *If $\mathcal{A}x^{\mathrm{r}}$ is a theory on Σ' such that:*

- *for any constant individual c^T of Σ, $\mathcal{A}x^{\mathrm{r}} \vdash\!\!\sim (\!|\mathsf{c}^T|\!)$*
- *for each $A \in \mathcal{A}x$, $\mathcal{A}x^{\mathrm{r}} \vdash\!\!\sim A^{\mathrm{r}}$*
- *for each sort T of Σ, there is a closed term t^T*

then for any closed formula A on Σ:

$$\mathcal{A}x \vdash\sim A \qquad \Longrightarrow \qquad \mathcal{A}x^{\mathrm{r}} \vdash\sim A^{\mathrm{r}}$$

Proof. We translate inductively every proof in the theory $\mathcal{A}x$ (on the signature Σ) of a sequent:

$$\Gamma \vdash A \mid \Delta$$

to a proof in $\mathcal{A}x^{\mathrm{r}}$ (on the signature Σ') of a sequent:

$$(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}$$

where $\mathrm{FV}(\Gamma, A, \Delta) \subseteq \vec{x}^{\vec{T}}$. The translation of a proof $\vdash A$ in $\mathcal{A}x$ for A a closed formula is then a proof in $\mathcal{A}x^{\mathrm{r}}$ of a sequent $(\!|\vec{x}^{\vec{T}}|\!) \vdash A^{\mathrm{r}}$. Indeed, some dummy variables may appear during the translation, without appearing free in A (this is due to our system not satisfying the subformula property). In order to eliminate these, we use the last hypothesis: from $(\!|\vec{x}^{\vec{T}}|\!) \vdash A^{\mathrm{r}}$ we can derive $\vdash \forall^{\mathrm{r}} \vec{x}^{\vec{T}} A^{\mathrm{r}}$, and then $\vdash (\!|\vec{t}^{\vec{T}}|\!) \Rightarrow A^{\mathrm{r}}$ where $\vec{t}^{\vec{T}}$ are closed individuals obtained from the last hypothesis. Finally, using lemma 2.2 below, we can combine the proof of $\vdash (\!|\vec{t}^{\vec{T}}|\!) \Rightarrow A^{\mathrm{r}}$ with proofs of $(\!|\vec{t}^{\vec{T}}|\!)$ and get a proof of $\vdash A^{\mathrm{r}}$.

The translation of the axiom rule follows from the hypotheses of the lemma, the identity rule is translated as follows, with $\vec{x}^{\vec{T}} = \mathrm{FV}(\Gamma, A, \Delta)$:

$$\frac{}{\Gamma, A \vdash A \mid \Delta} \quad \rightsquigarrow \quad \frac{}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}}, A^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

The rules for logical connectives are translated trivially:

$$\frac{\Gamma \vdash A \mid \Delta, A}{\Gamma \vdash \bot \mid \Delta, A} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}, A^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash \bot \mid \Delta^{\mathrm{r}}, A^{\mathrm{r}}}$$

$$\frac{\Gamma \vdash \bot \mid \Delta, A}{\Gamma \vdash A \mid \Delta} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash \bot \mid \Delta^{\mathrm{r}}, A^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

$$\frac{\Gamma, A \vdash B \mid \Delta}{\Gamma \vdash A \Rightarrow B \mid \Delta} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}}, A^{\mathrm{r}} \vdash B^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \Rightarrow B^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

$$\frac{\Gamma \vdash A \Rightarrow B \mid \Delta \qquad \Gamma \vdash A \mid \Delta}{\Gamma \vdash B \mid \Delta} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \Rightarrow B^{\mathrm{r}} \mid \Delta^{\mathrm{r}} \qquad (\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash B^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

$$\frac{\Gamma \vdash A \mid \Delta \qquad \Gamma \vdash B \mid \Delta}{\Gamma \vdash A \wedge B \mid \Delta} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}} \qquad (\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash B^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \wedge B^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

$$\frac{\Gamma \vdash A_1 \wedge A_2 \mid \Delta}{\Gamma \vdash A_i \mid \Delta} \quad \rightsquigarrow \quad \frac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash {A^{\mathrm{r}}}_1 \wedge {A^{\mathrm{r}}}_2 \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash {A^{\mathrm{r}}}_i \mid \Delta^{\mathrm{r}}}$$

where in the cases of introduction of implication and conjunction, the two premises can get the same relativized variables on the left by applying the (admissible) left weakening rule when necessary. The translation of the introduction of universal quantification is given by:

$$\frac{\Gamma \vdash A \mid \Delta}{\Gamma \vdash \forall x^T A \mid \Delta}\ \left(x^T \notin \mathrm{FV}(\Gamma, \Delta)\right) \quad \rightsquigarrow \quad \frac{\dfrac{(\!|x^T|\!), (\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash (\!|x^T|\!) \Rightarrow A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash \forall x^T \left((\!|x^T|\!) \Rightarrow A^{\mathrm{r}}\right) \mid \Delta^{\mathrm{r}}}\ \left(x^T \notin \mathrm{FV}\left((\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}}, \Delta^{\mathrm{r}}\right)\right)$$

and preserves the fact that all the free variables of a sequent are relativized in the context. Finally, in order to translate the elimination of the universal quantification we must prove that we can lift relativization to all constructs on individuals. This can be obtained using the hypothesis of the lemma requiring that for every individual constant c^T of Σ, $\mathcal{A}x^{\mathrm{r}} \vdash\sim (\!|\mathsf{c}^T|\!)$. Indeed, we have the following lemma:

Lemma 2.2. *Suppose that for every individual constant c^T of Σ, $\mathcal{A}x^{\mathrm{r}} \vdash\sim (\!|\mathsf{c}^T|\!)$. Let t^T be an individual of the logic with free variables $\vec{x}^{\vec{V}}$. We have:*

$$\mathcal{A}x^{\mathrm{r}} \vdash\sim \forall^{\mathrm{r}}\vec{x}^{\vec{V}}\, (\!|t^T|\!)$$

Proof. We prove it by induction on t^T. If t^T is some c^T, then this is an assumption of the lemma. If t^T is a variable x^T, then $\forall^{\mathrm{r}}x^T\, (\!|x^T|\!) \equiv \forall x^T \left((\!|x^T|\!) \Rightarrow (\!|x^T|\!)\right)$, which is trivially derivable, and if t^T is $u^{U\to T}\, v^U$, then we get $(\!|\vec{x}^{\vec{V}}|\!) \vdash \forall y^U \left((\!|y^U|\!) \Rightarrow (\!|(u\, y)^T|\!)\right)$ and $(\!|\vec{x}^{\vec{V}}|\!) \vdash (\!|v^U|\!)$ from the induction hypotheses, so we obtain $(\!|\vec{x}^{\vec{V}}|\!) \vdash (\!|(u\, v)^T|\!)$ and $\vdash \forall^{\mathrm{r}}\vec{x}^{\vec{V}}\, (\!|(u\, v)^T|\!)$. □

Now we can describe the translation of the elimination rule of the universal quantifier:

$$\frac{\Gamma \vdash \forall x^T A \mid \Delta}{\Gamma \vdash A\left\{t^T/x^T\right\} \mid \Delta} \quad\rightsquigarrow\quad \frac{\dfrac{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash \forall^{\mathrm{r}}x^T\, A^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash (\!|t^T|\!) \Rightarrow A\,\{t/x\}^{\mathrm{r}} \mid \Delta^{\mathrm{r}}} \qquad \dfrac{\vdots}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash (\!|t^T|\!) \mid \Delta^{\mathrm{r}}}}{(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash A\,\{t/x\}^{\mathrm{r}} \mid \Delta^{\mathrm{r}}}$$

where we can suppose without loss of generality that $\mathrm{FV}\left(t^T\right) \subseteq \vec{x}^{\vec{T}}$ (using the left weakening rule if necessary), and $(\!|\vec{x}^{\vec{T}}|\!), \Gamma^{\mathrm{r}} \vdash (\!|t^T|\!) \mid \Delta^{\mathrm{r}}$ is easily derivable using $\forall^{\mathrm{r}}\vec{y}^{\vec{U}}\, (\!|t^T|\!)$ (with $\vec{y}^{\vec{U}} = \mathrm{FV}\left(t^T\right) \subseteq \vec{x}^{\vec{T}}$) from lemma 2.2. □

2.4 Peano arithmetic in finite types: PA^ω

Peano arithmetic in finite types, PA^ω, is an extension of Peano arithmetic in which we can write any term of Gödel's system T and quantify over functions of any type.

The signature Σ_{PA^ω} of Peano arithmetic in finite types contains a single base sort ι, the following sorted constants:

$$\begin{array}{llll} \mathsf{s} & \text{of sort} \quad (T\to U\to V)\to (T\to U)\to T\to V & \mathsf{k} & \text{of sort} \quad T\to U\to T \\ \mathsf{0} & \text{of sort} \quad \iota & \mathsf{S} & \text{of sort} \quad \iota\to\iota \end{array}$$

$$\mathsf{rec} \quad \text{of sort} \quad T\to(\iota\to T\to T)\to\iota\to T$$

PA^ω is an equational theory as defined above, and the only predicate symbol is inequality:

$$t^T \neq u^T$$

Now we give the axioms of PA^ω. The first two axioms are those of equational theories:

$$(\boldsymbol{refl}) \quad \forall x^T\, (x =_T x) \qquad (\boldsymbol{Leib}) \quad \forall\vec{z}^{\vec{U}}\, \forall x^T\, \forall y^T\, (\neg A \Rightarrow A\,\{y/x\} \Rightarrow x \neq_T y)$$

PA^ω also contains the definitional axioms of s, k and rec:

$$\begin{array}{ll} (\boldsymbol{\Delta s}) \quad \forall x\, \forall y\, \forall z \;\; \mathsf{s}\, x\, y\, z = x\, z\, (y\, z) & (\boldsymbol{\Delta rec0}) \quad \forall x\, \forall y \;\; \mathsf{rec}\, x\, y\, \mathsf{0} = x \\ (\boldsymbol{\Delta k}) \quad \forall x\, \forall y \;\; \mathsf{k}\, x\, y = x & (\boldsymbol{\Delta recS}) \quad \forall x\, \forall y\, \forall z \;\; \mathsf{rec}\, x\, y\, (\mathsf{S}\, z) = y\, z\, (\mathsf{rec}\, x\, y\, z) \end{array}$$

and finally the non-confusion axiom and the axiom scheme of induction for every formula A with free variables among $x^\iota, \vec{y}^{\vec{T}}$:

$$(\boldsymbol{Snz}) \quad \forall x \;\; \mathsf{S}\, x \neq \mathsf{0} \qquad (\boldsymbol{ind}) \quad \forall\vec{y}\, (A\,\{\mathsf{0}/x\} \Rightarrow \forall x\, (A \Rightarrow A\,\{\mathsf{S}\, x/x\}) \Rightarrow \forall x\, A)$$

2.4.1 Relativized PA^ω: $PA^{\omega\mathrm{r}}$

We now define the relativized version $PA^{\omega\mathrm{r}}$ of PA^ω, using lemma 2.1. First, $\Sigma_{PA^{\omega\mathrm{r}}}$ is Σ_{PA^ω} augmented with a predicate symbol $(\!|t^\iota|\!)$. The axioms of $PA^{\omega\mathrm{r}}$ are those of PA^ω where the induction scheme is replaced by:

$$(\boldsymbol{ind^r}) \quad \forall \vec{y}\,(A\,\{\mathsf{0}/x\} \Rightarrow \forall^{\mathrm{r}} x\,(A \Rightarrow A\,\{\mathsf{S}\,x/x\}) \Rightarrow \forall^{\mathrm{r}} x\,A)$$

plus the axioms $(\!|\mathsf{0}^\iota|\!)$ and $(\!|\mathsf{S}^{\iota\to\iota}|\!)$. In order to use lemma 2.1 we need the following lemmas:

Lemma 2.3.

$$PA^{\omega\mathrm{r}} \vdash\!\sim (\!|\mathsf{s}|\!) \qquad PA^{\omega\mathrm{r}} \vdash\!\sim (\!|\mathsf{k}|\!) \qquad PA^{\omega\mathrm{r}} \vdash\!\sim (\!|\mathsf{rec}|\!)$$

Proof. The formulas $(\!|\mathsf{s}|\!)$ and $(\!|\mathsf{k}|\!)$ are provable using the axioms of equality and respectively $(\boldsymbol{\Delta s})$ and $(\boldsymbol{\Delta k})$. Indeed, $(\!|\mathsf{s}|\!)$ is the following formula:

$$\forall x^{T\to U\to V}\,\big((\!|x|\!) \Rightarrow \forall y^{T\to U}\,((\!|y|\!) \Rightarrow \forall z^{\sigma}\,((\!|z|\!) \Rightarrow (\!|\mathsf{s}\,x\,y\,z|\!)))\big)$$

which is equivalent in first-order logic to:

$$\forall x^{T\to U\to V} \forall y^{T\to U} \forall z^{T}\,((\!|x|\!) \Rightarrow (\!|y|\!) \Rightarrow (\!|z|\!) \Rightarrow (\!|\mathsf{s}\,x\,y\,z|\!))$$

using $(\boldsymbol{\Delta s})$ and $(\boldsymbol{Leib})$ this is equivalent to:

$$\forall x^{T\to U\to V} \forall y^{T\to U} \forall z^{T}\,((\!|x|\!) \Rightarrow (\!|y|\!) \Rightarrow (\!|z|\!) \Rightarrow (\!|x\,z\,(y\,z)|\!))$$

and since $(\!|x|\!)$ and $(\!|y|\!)$ are the following formulas:

$$\forall x_1^T\,\big((\!|x_1|\!) \Rightarrow \forall x_2^U\,((\!|x_2|\!) \Rightarrow (\!|x\,x_1\,x_2|\!))\big) \qquad \forall y_1^T\,((\!|y_1|\!) \Rightarrow (\!|y\,y_1|\!))$$

we can instantiate these with $x_1 \equiv z$, $x_2 \equiv y\,z$ and $y_1 \equiv z$ to obtain a proof of $(\!|\mathsf{s}|\!)$. The case of $(\!|\mathsf{k}|\!)$ is similar.

Similarly, $(\!|\mathsf{rec}|\!)$ is provable using the axioms of equality, $(\boldsymbol{\Delta rec0})$, $(\boldsymbol{\Delta recS})$ and $(\boldsymbol{ind^r})$. $(\!|\mathsf{rec}|\!)$ is equivalent in first-order logic to:

$$\forall x^T \forall y^{\iota\to T\to T}\,\big((\!|x|\!) \Rightarrow \forall^{\mathrm{r}} y_1^\iota \forall y_2^T\,((\!|y_2|\!) \Rightarrow (\!|y\,y_1\,y_2|\!)) \Rightarrow \forall^{\mathrm{r}} z^\iota\,(\!|\mathsf{rec}\,x\,y\,z|\!)\big)$$

if we instantiate y_2 with $\mathsf{rec}\,x\,y\,y_1$ it is sufficient to prove:

$$\forall x^T \forall y^{\iota\to T\to T}\,((\!|x|\!) \Rightarrow \forall^{\mathrm{r}} y_1^\iota\,((\!|\mathsf{rec}\,x\,y\,y_1|\!) \Rightarrow (\!|y\,y_1\,(\mathsf{rec}\,x\,y\,y_1)|\!)) \Rightarrow \forall^{\mathrm{r}} z^\iota\,(\!|\mathsf{rec}\,x\,y\,z|\!))$$

which is equivalent, using $(\boldsymbol{\Delta rec0})$, $(\boldsymbol{\Delta recS})$ and $(\boldsymbol{Leib})$, to:

$$\forall x^T \forall y^{\iota\to T\to T}\,((\!|\mathsf{rec}\,x\,y\,\mathsf{0}|\!) \Rightarrow \forall^{\mathrm{r}} y_1^\iota\,((\!|\mathsf{rec}\,x\,y\,y_1|\!) \Rightarrow (\!|\mathsf{rec}\,x\,y\,(\mathsf{S}\,y_1)|\!)) \Rightarrow \forall^{\mathrm{r}} z^\iota\,(\!|\mathsf{rec}\,x\,y\,z|\!))$$

which is an instance of $(\boldsymbol{ind^r})$ with the formula $(\!|\mathsf{rec}\,x\,y\,z|\!)$ □

Lemma 2.4. *For any $A \in PA^\omega$, $PA^{\omega\mathrm{r}} \vdash\!\sim A^{\mathrm{r}}$.*

Proof. For $(\boldsymbol{\Delta s})$, $(\boldsymbol{\Delta k})$, $(\boldsymbol{\Delta rec0})$, $(\boldsymbol{\Delta recS})$ and $(\boldsymbol{Snz})$ it follows from the fact that the formula $\forall x\,A \Rightarrow \forall^{\mathrm{r}} x\,A$ is derivable in first-order logic. For $(\boldsymbol{ind})$ it comes from this and the fact that the following formula:

$$\forall^{\mathrm{r}} \vec{y}\,(A^{\mathrm{r}}\,\{\mathsf{0}/x\} \Rightarrow \forall^{\mathrm{r}} x\,(A^{\mathrm{r}} \Rightarrow A^{\mathrm{r}}\,\{\mathsf{S}\,x/x\}) \Rightarrow \forall^{\mathrm{r}} x\,A^{\mathrm{r}})$$

is a consequence of $(\boldsymbol{ind^r})$. □

Lemma 2.5. *For every sort T on Σ_{PA^ω}, there is a closed individual t^T.*

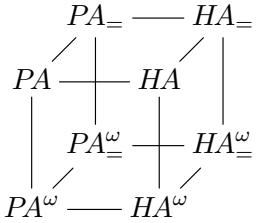
Proof. By induction on T. If $T = \iota$, then $\mathsf{0}^\iota$ is such an individual. If $T = U \to V$, then by induction hypothesis there is some closed t^V, and so $\mathsf{k}^{V\to U\to V}\, t^V$ is a closed individual of sort $T = U \to V$. □

Using these three lemmas, it follows from lemma 2.1 that for any closed formula A on the signature Σ_{PA^ω}:

$$PA^\omega \vdash\!\sim A \qquad \Longrightarrow \qquad PA^{\omega\mathrm{r}} \vdash\!\sim A^{\mathrm{r}}$$

2.4.2 Relation to usual theories

We discuss in this section about the relation between our version of PA^ω, in which the inequality predicate is atomic, and other intuitionistic or classical systems. First, in this section we will consider the following systems:



where the P/H part corresponds to Peano and Heyting arithmetic, the latter being the intuitionistic version of the former, the ω part corresponds to the possibility of quantifying over higher type variables, and the $=$ part corresponds to having an atomic predicate for equality while the absence of $=$ indicates a primitive inequality. For the systems based on a primitive equality we refer to [TVD88]. In the case of primitive equality, the Leibniz scheme is:

$$\forall \vec{z^U} \forall x^T \forall y^T \, (x = y \Rightarrow A \Rightarrow A\,\{y/x\})$$

while in the case of primitive inequality we define it is as in section 2.2:

$$\forall \vec{z^U} \forall x^T \forall y^T \, (\neg A \Rightarrow A\,\{y/x\} \Rightarrow x \neq y)$$

Similarly, in $PA_=$ and $HA_=$ the injectivity of successor is:

$$\forall x^\iota \forall y^\iota \, (\mathsf{S}\, x = \mathsf{S}\, y \Rightarrow x = y)$$

while in PA and HA we define it to be the following formula, that we add also to the axioms of HA^ω:

$$\forall x^\iota \forall y^\iota \, (x \neq y \Rightarrow \mathsf{S}\, x \neq \mathsf{S}\, y)$$

It is well known that $HA_= \vdash\!\sim \forall x^\iota \forall y^\iota \, (\neg\neg x = y \Rightarrow x = y)$, so in our system with only negative connectives, $HA_=$ and $PA_=$ prove exactly the same formulas. It is quite easy to prove also that $HA \vdash\!\sim \forall x^\iota \forall y^\iota \, (\neg\neg x \neq y \Rightarrow x \neq y)$, so HA and PA also prove exactly the same formulas (again, in the particular case of our system with only negative connectives). On the other side, despite the equivalence of provability between PA and HA, the associated proof terms are very different. Indeed, the proof of $HA \vdash\!\sim \forall x^\iota \forall y^\iota \, (\neg\neg x \neq y \Rightarrow x \neq y)$ is obtained by a double induction on x and y, while in PA, a proof of $PA \vdash\!\sim \forall x^\iota \forall y^\iota \, (\neg\neg x \neq y \Rightarrow x \neq y)$ can also be obtained through classical logic and its computational content relies then on the control features of languages for classical logic (like $\lambda\mu$-calculus). This difference is reflected by the fact that, in the case of relativized theories, $PA^{\omega\mathrm{r}}$ is capable of proving $\forall x^\iota \forall y^\iota \, (\neg\neg x \neq y \Rightarrow x \neq y)$ while $HA^{\omega\mathrm{r}}$ can only

prove $\forall^{\mathrm{r}} x^{\iota} \forall^{\mathrm{r}} y^{\iota} \left(\neg\neg x \neq y \Rightarrow x \neq y\right)$. When we switch to higher type equalities, things become much more complicated. Indeed, $HA^{\omega}_{=}$ fails to prove $\forall x^{T} \forall y^{T} \left(\neg\neg x = y \Rightarrow x = y\right)$, since we cannot perform induction on higher type variables, so the systems $HA^{\omega}_{=}$ and $PA^{\omega}_{=}$ are not equivalent anymore, even at the level of provability. For classical systems PA^{ω} and $PA^{\omega}_{=}$, a formula with equalities and inequalities is provable in PA^{ω} (through the encoding $(x = y) \equiv (x \neq y \Rightarrow \bot)$) if and only if it is provable in $PA^{\omega}_{=}$ (through the encoding $(x \neq y) \equiv (x = y \Rightarrow \bot)$). However, for intuitionistic systems HA^{ω} and $HA^{\omega}_{=}$ things are less clear since double negations cannot be eliminated on atomic formulas. In particular, we did not investigate the relation between the system HA^{ω} with primitive inequalities and $HA^{\omega}_{=}$ or $PA^{\omega}/PA^{\omega}_{=}$.

2.5 The axiom of choice

2.5.1 Classical analysis: CA^{ω}

Classical analysis CA^{ω} (in the sense of [Koh08]) is PA^{ω} augmented with the axiom scheme of dependent choice:

$$(\boldsymbol{DC}) \quad \forall \vec{u}^{\vec{U}} \left(\forall x^{\iota} \, \forall y^{T} \, \exists z^{T} \, A \;\; \Rightarrow \;\; \exists y^{\iota \to T} \, \forall x^{\iota} \, A \left\{y \, x / y, y \left(\mathsf{S} \, x\right) / z\right\}\right)$$

where A is any formula on $\Sigma_{CA^{\omega}} = \Sigma_{PA^{\omega}}$ with free variables among $x^{\iota}, y^{T}, z^{T}, \vec{u}^{\vec{U}}$.

As proved in [Koh08], in PA^{ω} this particular form of dependent choice implies both countable choice:

$$\forall \vec{u}^{\vec{U}} \left(\forall x^{\iota} \, \exists y^{T} \, A \;\; \Rightarrow \;\; \exists z^{\iota \to T} \, \forall x^{\iota} \, A \left\{z \, x / y\right\}\right)$$

where A is any formula on $\Sigma_{CA^{\omega}} = \Sigma_{PA^{\omega}}$ with free variables among $x^{\iota}, y^{T}, \vec{u}^{\vec{U}}$, and the more usual version of dependent choice:

$$\forall \vec{u}^{\vec{U}} \left(\forall x^{T} \, \exists y^{T} \, A \;\; \Rightarrow \;\; \forall v^{T} \exists z^{\iota \to T} \left(z \, \mathsf{0} = v \wedge \forall x^{\iota} \, A \left\{z \, x / x, z \left(\mathsf{S} \, x\right) / y\right\}\right)\right)$$

where A is any formula on $\Sigma_{CA^{\omega}} = \Sigma_{PA^{\omega}}$ with free variables among $x^{T}, y^{T}, \vec{u}^{\vec{U}}$. Remark that if we unfold the definitions of $\exists z^{T}$ and $\exists y^{\iota \to T}$, $(\boldsymbol{DC})$ becomes:

$$\forall \vec{u} \left(\forall x \, \forall y \, \neg \forall z \, \neg A \;\; \Rightarrow \;\; \neg \forall y \, \neg \forall x \, A \left\{y \, x / y, y \left(\mathsf{S} \, x\right) / z\right\}\right)$$

2.5.2 Relativized CA^{ω}: $CA^{\omega \mathrm{r}}$

As we did for PA^{ω}, we define the relativized version $CA^{\omega \mathrm{r}}$ of CA^{ω}. First, the signature is the same as $\Sigma_{PA^{\omega \mathrm{r}}}$: it is $\Sigma_{CA^{\omega}} = \Sigma_{PA^{\omega}}$ augmented with a predicate symbol $(\!|t^{\iota}|\!)$. The axioms are those of $PA^{\omega \mathrm{r}}$ plus the following version of dependent choice, $(\boldsymbol{DC}^{\boldsymbol{r}})$:

$$\forall \vec{u}^{\vec{U}} \forall^{\mathrm{r}} v^{T} \left(\forall^{\mathrm{r}} x^{\iota} \forall^{\mathrm{r}} y^{T} \neg \forall z^{T} \neg \left((\!|z|\!) \wedge A^{\mathrm{r}}\right) \Rightarrow \neg \forall y^{\iota \to T} \neg \left((\!|y \, \mathsf{0}|\!) \wedge \forall^{\mathrm{r}} x^{\iota} \left((\!|y \left(\mathsf{S} \, x\right)|\!) \wedge A^{\mathrm{r}} \left\{y \, x / y, y \left(\mathsf{S} \, x\right) / z\right\}\right)\right)\right)$$

where A is any formula on $\Sigma_{CA^{\omega}} = \Sigma_{PA^{\omega}}$ with free variables among $x^{\iota}, y^{T}, z^{T}, \vec{u}^{\vec{U}}$. In order to use lemma 2.1, we need to prove that $(\boldsymbol{DC})^{\mathrm{r}}$ is derivable in $CA^{\omega \mathrm{r}}$. Indeed, $(\boldsymbol{DC}^{\boldsymbol{r}})$ is different from $(\boldsymbol{DC})^{\mathrm{r}}$ which is:

$$\forall^{\mathrm{r}} \vec{u}^{\vec{U}} \forall^{\mathrm{r}} v^{T} \left(\forall^{\mathrm{r}} x^{\iota} \, \forall^{\mathrm{r}} y^{T} \, \neg \forall^{\mathrm{r}} z^{T} \, \neg A^{\mathrm{r}} \Rightarrow \neg \forall^{\mathrm{r}} y^{\iota \to T} \, \neg \forall^{\mathrm{r}} x^{\iota} \, A^{\mathrm{r}} \left\{y \, x / y, y \left(\mathsf{S} \, x\right) / z\right\}\right)$$

First, $\forall^{\mathrm{r}} z^{T} \, \neg A^{\mathrm{r}} \equiv \forall z^{T} \left((\!|z|\!) \Rightarrow A^{\mathrm{r}} \Rightarrow \bot\right)$ and $\forall z^{T} \neg \left((\!|z|\!) \wedge A^{\mathrm{r}}\right) \equiv \forall z^{T} \left((\!|z|\!) \wedge A^{\mathrm{r}} \Rightarrow \bot\right)$, so since $(A \Rightarrow B \Rightarrow C) \Leftrightarrow (A \wedge B \Rightarrow C)$ is a propositional tautology we get:

$$\vdash \forall^{\mathrm{r}} z^{T} \, \neg A^{\mathrm{r}} \Leftrightarrow \forall z^{T} \neg \left((\!|z|\!) \wedge A^{\mathrm{r}}\right)$$

Moreover:

$$\forall^{\mathrm{r}} y^{\iota\to T}\, \neg\forall^{\mathrm{r}} x^{\iota}\, A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}$$
$$\equiv \forall y^{\iota\to T}\left(\forall x^{\iota}\left(\left(\!\left|x\right|\!\right) \Rightarrow \left(\!\left|y\, x\right|\!\right)\right) \Rightarrow \forall x^{\iota}\left(\left(\!\left|x\right|\!\right) \Rightarrow A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}\right) \Rightarrow \bot\right)$$

$$\forall y^{\iota\to T}\, \neg\left(\left(\!\left|y\, \mathsf{0}\right|\!\right) \wedge \forall^{\mathrm{r}} x^{\iota}\left(\left(\!\left|y\left(\mathsf{S}\, x\right)\right|\!\right) \wedge A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}\right)\right)$$
$$\equiv \forall y^{\iota\to T}\left(\left(\!\left|y\, \mathsf{0}\right|\!\right) \wedge \forall x^{\iota}\left(\left(\!\left|x\right|\!\right) \Rightarrow \left(\!\left|y\left(\mathsf{S}\, x\right)\right|\!\right) \wedge A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}\right) \Rightarrow \bot\right)$$

so since again $(A \Rightarrow B \Rightarrow C) \Leftrightarrow (A \wedge B \Rightarrow C)$ is a propositional tautology, $\forall x\,(A \wedge B) \Leftrightarrow \forall x\, A \wedge \forall x\, B$ is provable in first-order logic, and $\forall^{\mathrm{r}} x^{\iota}\, A \Leftrightarrow A\left\{\mathsf{0}/x\right\} \wedge \forall^{\mathrm{r}} x^{\iota}\, A\left\{\mathsf{S}\, x/x\right\}$ is provable using induction (actually only case-analysis) we get:

$$(\boldsymbol{ind^r}) \vdash \forall^{\mathrm{r}} y^{\iota\to T}\, \neg\forall^{\mathrm{r}} x^{\iota}\, A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}$$
$$\Leftrightarrow \forall y^{\iota\to T}\, \neg\left(\left(\!\left|y\, \mathsf{0}\right|\!\right) \wedge \forall^{\mathrm{r}} x^{\iota}\left(\left(\!\left|y\left(\mathsf{S}\, x\right)\right|\!\right) \wedge A^{\mathrm{r}}\left\{y\, x/y, y\left(\mathsf{S}\, x\right)/z\right\}\right)\right)$$

so finally since $\forall x\, A \Rightarrow \forall^{\mathrm{r}} x\, A$ is derivable in first-order logic, $(\boldsymbol{DC})^{\mathrm{r}}$ is derivable from $(\boldsymbol{DC^r})$ in $CA^{\omega\mathrm{r}}$. Therefore we can apply lemma 2.1 to get for any A on the signature $\Sigma_{CA^{\omega}}$:

$$CA^{\omega} \mathrel{|\!\sim} A \qquad \Longrightarrow \qquad CA^{\omega\mathrm{r}} \mathrel{|\!\sim} A^{\mathrm{r}}$$

2.6 Models of classical multisorted first-order logic

In this section we define what is a model of a given multisorted first-order theory. We fix a signature Σ.

Definition 2.3. *A Σ-structure $\mathcal{M}$ is given by:*

- *a set $T^{\mathcal{M}}$ for each sort T of Σ*
- *an application function from $(T \to U)^{\mathcal{M}} \times T^{\mathcal{M}}$ to $U^{\mathcal{M}}$*
- *an element $\mathsf{c}^{\mathcal{M}} \in T^{\mathcal{M}}$ for each individual constant c^{T} of Σ*
- *a set $P^{\mathcal{M}} \subseteq {T_1}^{\mathcal{M}} \times ... \times {T_n}^{\mathcal{M}}$ for each sorted predicate $P\left(t_1^{T_1}, ..., t_n^{T_n}\right)$ of Σ*

Using the application function, we can extend the interpretation to individuals with parameters: if t is an individual of sort T with free variables $\vec{x}^{\vec{U}}$ and if $\vec{a}$ are elements of $\vec{U}^{\mathcal{M}}$, then $(t\left\{\vec{a}/\vec{x}\right\})^{\mathcal{M}}$ is an element of $T^{\mathcal{M}}$.

As usual in model theory, for any Σ-structure $\mathcal{M}$ and any closed formula A on Σ we can define when $\mathcal{M}$ validates A, written $\mathcal{M} \vDash A$. Now we fix a theory $\mathcal{A}x$ and we define what is a model of $\mathcal{A}x$:

Definition 2.4. *If $\mathcal{M}$ is a Σ-structure, then $\mathcal{M}$ is a model of $\mathcal{A}x$ if for any $A \in \mathcal{A}x$:*

$$\mathcal{M} \vDash A$$

In the particular case of equational theories we fix the interpretation of the $\neq_T$ predicate to be the following set of pairs of elements of $T^{\mathcal{M}}$:

$$\neq_T{}^{\mathcal{M}} \quad \triangleq \quad \left\{(a,b) \in T^{\mathcal{M}} \times T^{\mathcal{M}} \,\middle|\, a \neq b\right\}$$

and in the case of $PA^{\omega \mathrm{r}}$ and $CA^{\omega \mathrm{r}}$ we fix the interpretation of the relativization predicate at the unique base type ι, $(\!|.|\!)$, to be the following set of elements of $\iota^{\mathcal{M}}$:

$$(\!|.|\!)^{\mathcal{M}} \quad \triangleq \quad \left\{\mathsf{0}^{\mathcal{M}}; (\mathsf{S}\,\mathsf{0})^{\mathcal{M}}; (\mathsf{S}\,(\mathsf{S}\,\mathsf{0}))^{\mathcal{M}}; (\mathsf{S}\,(\mathsf{S}\,(\mathsf{S}\,\mathsf{0})))^{\mathcal{M}}; \ldots\right\}$$

Chapter 3

Typed languages

In this chapter, we define the programming language to which we map classical proofs, and define the categorical model of this language. First, we describe $\lambda\mu$-calculus as a typed language and give its equational theory under call-by-name semantics before defining how we map classical proofs in this language. Then we describe categories of continuations as a model of call-by-name $\lambda\mu$-calculus, we explain how the coproduct completion of certain categories can be used to provide a category of continuations, and we present the interpretation of $\lambda\mu$-calculus in categories of continuations and its connection with that of its continuation-passing-style translation in the underlying cartesian "almost" closed category. Finally, we describe the particular case of μPCF which is the language in which we interpret Peano arithmetic and the axiom of choice trough the bar-recursion operator, for which we give an intuitive meaning as a recursion operator for well-founded trees.

3.1 $\lambda\mu$-calculus

The $\lambda\mu$-calculus is an extension of the λ-calculus introduced by Parigot in [Par92] in order to represent and evaluate classical proofs. In $\lambda\mu$-calculus, there is another kind of variables along standard λ-variables: the μ-variables. Just like the λ-variables are bound by the construct $\lambda x.M$, the μ-variables (which will be written $\alpha, \beta, ...$) are bound by the new construct of $\lambda\mu$-calculus: $\mu\alpha.M$. The other new construct, $[\alpha]\ M$ should be understood as a mean for M to get arguments which are passed to the enclosing $\mu\alpha.N$.

To be more explicit, if N is a $\lambda\mu$-term with free variable x, then $(\mu\alpha.N\,\{[\alpha]\ M/x\})\,P$ reduces to $\mu\alpha.N\,\{[\alpha]\ M\,P/x\}$. The argument P given to the term bound by $\mu\alpha$ gets passed over to M because of the $[\alpha]$ in front of it. The $[\alpha]$ may also appear multiple times. If x and y are free variables of N, then $(\mu\alpha.N\,\{[\alpha]\ M_1/x, [\alpha]\ M_2/y\})\,P$ reduces to $\mu\alpha.N\,\{[\alpha]\ M_1\,P/x, [\alpha]\ M_2\,P/y\}$. The arguments received by a $\mu\alpha....$ are passed over to all the subterms preceded by $[\alpha]$. This behavior may also be obtained by replacing $\mu\alpha.N\,\{[\alpha]\ M/x\}$ with $\lambda\vec{z}.N\,\{M\,\vec{z}/x\}$, the length of $\vec{z}$ being fixed by the type of α. This is almost what happens when a $\lambda\mu$-term is CPS-translated to a λ-term. The CPS translation also needs to introduce new types in order to translate the base types to arrow types returning the empty type.

$\lambda\mu$-calculus provides a direct interpretation of classical proofs just like λ-calculus is an interpretation of the intuitionistic ones. In intuitionistic sequent calculus, sequents are of the form $A_1, ..., A_n \vdash A$, and should be interpreted as the formula $A_1 \wedge ... \wedge A_n \Rightarrow A$. The interpretation of a proof of such a sequent in λ-calculus is then a λ-term with free variables $k_1, ..., k_n$, where k_i represents the possibility to use the hypothesis A_i in the proof. In classical sequent calculus, sequents are of the form $A_1, ..., A_n \vdash B_1, ..., B_m$, and should be interpreted as the formula

$A_1 \wedge ... \wedge A_n \Rightarrow B_1 \vee ... \vee B_m$. The interpretation of a proof of such a sequent in $\lambda\mu$-calculus is then a $\lambda\mu$-term with free λ-variables $x_1, ..., x_n$, and free μ-variables $\alpha_1, ..., \alpha_m$, x_i representing again the possibility to use the hypothesis A_i, but α_i representing the possibility to produce (part of) a proof of B_i. Under this view, a $\lambda\mu$-term may provide many proofs at the same time, each proof being given step-by-step.

In the original version of [Par92], the terms were restricted to λ-abstractions, applications and $\mu\alpha.\,[\beta]\ M$ for some term M. This syntax was extended in [dG94] by allowing terms $\mu\alpha.M$ and $[\alpha]\ M$. This extension of syntax was used in [Sau05] together with a well-chosen set of reduction rules (this calculus being called $\Lambda\mu$-calculus) to recover the separation property that failed in Parigot's $\lambda\mu$-calculus, as shown in [DP01]. Later on in [AHS07], Parigot's version was related to minimal classical logic, and De Groote's to full classical logic. Here we use the version of [Sel01] (but without disjunction types), which is in the lines of [dG94, Sau05]. We recommend [HS09] for an historical overview of the different versions of $\lambda\mu$-calculus and how they relate to each other.

3.1.1 Type system

The types of $\lambda\mu$-calculus are those of simply typed λ-calculus with a fixed set of base types (ranged over by ι) together with a product type and and a distinguished empty type 0:

$$T, U \quad ::= \quad \iota \mid T \to U \mid T \times U \mid 0$$

The empty type 0 is used to type terms $[\alpha]\ M$. Indeed, $[\alpha]\ M$ means that M will get the arguments that are received by the enclosing $\mu\alpha$. However, it cannot return anything, it is only a consumer. The only way this kind of term returns something is when it appears in $\mu\alpha.\,[\alpha]\ M$ and α does not appear free in M. In that case it is equivalent to M.

In addition to the base types, $\lambda\mu$-calculus is parameterized with a fixed set of typed constants: $\mathcal{C}st = \{\mathsf{c} : T, ...\}$. These two parameters form a signature of $\lambda\mu$-calculus:

Definition 3.1 ($\lambda\mu$ signature)**.** *A $\lambda\mu$ signature is a set of base types together with a set of typed constants.*

From a $\lambda\mu$ signature, we will now define the derivable typing judgements, which are presented as sequents of the form:

$$x_1 : T_1, ..., x_n : T_n \vdash M : T \mid \alpha_1 : U_1, ..., \alpha_m : U_m$$

where the free λ-variables of M are among $x_1, ..., x_n$ and its free μ-variables are among $\alpha_1, ..., \alpha_m$. The $x_1 : T_1, ..., \alpha_n : T_n$ part will be called the λ-context, and $\alpha_1 : U_1, ..., \alpha_m : U_m$ the μ-context. If the μ-context is empty, then we will simply write:

$$x_1 : T_1, ..., x_n : T_n \vdash M : T$$

Once a signature of $\lambda\mu$-calculus is given, the derivable typing judgements are defined as follows:

$$\frac{}{\vec{x} : \vec{T}, x : T \vdash x : T \mid \vec{\alpha} : \vec{U}} \qquad \frac{}{\vec{x} : \vec{T} \vdash \mathsf{c} : T \mid \vec{\alpha} : \vec{U}}\ (\mathsf{c}{:}T \in \mathcal{C}st)$$

$$\frac{\vec{x} : \vec{T}, x : T \vdash M : U \mid \vec{\alpha} : \vec{U}}{\vec{x} : \vec{T} \vdash \lambda x.M : T \to U \mid \vec{\alpha} : \vec{U}} \qquad \frac{\vec{x} : \vec{T} \vdash M : T \to U \mid \vec{\alpha} : \vec{U} \qquad \vec{x} : \vec{T} \vdash N : T \mid \vec{\alpha} : \vec{U}}{\vec{x} : \vec{T} \vdash MN : U \mid \vec{\alpha} : \vec{U}}$$

$$\frac{\vec{x} : \vec{T} \vdash M : T \mid \vec{\alpha} : \vec{U} \qquad \vec{x} : \vec{T} \vdash N : U \mid \vec{\alpha} : \vec{U}}{\vec{x} : \vec{T} \vdash \langle M, N \rangle : T \times U \mid \vec{\alpha} : \vec{U}} \qquad \frac{\vec{x} : \vec{T} \vdash M : T_1 \times T_2 \mid \vec{\alpha} : \vec{U}}{\vec{x} : \vec{T} \vdash \mathfrak{p}_i\, M : T_i \mid \vec{\alpha} : \vec{U}}$$

$$\frac{\vec{x} : \vec{T} \vdash M : U \mid \vec{\alpha} : \vec{U}, \alpha : U}{\vec{x} : \vec{T} \vdash [\alpha]M : 0 \mid \vec{\alpha} : \vec{U}, \alpha : U} \qquad \frac{\vec{x} : \vec{T} \vdash M : 0 \mid \vec{\alpha} : \vec{U}, \alpha : U}{\vec{x} : \vec{T} \vdash \mu\alpha.M : U \mid \vec{\alpha} : \vec{U}}$$

The left and right weakening rules are admissible in that type system, and we use them without explicitly mentioning it. Here are some examples of derivable typing judgements.

$$\vdash \lambda x.\mu\alpha.x : 0 \to T \qquad \vdash \lambda y.\mu\alpha.y\,(\lambda x.[\alpha]\,x) : ((T \to 0) \to 0) \to T$$

$$\vdash \lambda y.\mu\alpha.[\alpha]\,y\,(\lambda x.\mu\beta.[\alpha]\,x) : ((T \to U) \to T) \to T$$

The logical counterparts of these terms are respectively the ex falso formula $\bot \Rightarrow A$, the double-negation elimination $((A \Rightarrow \bot) \Rightarrow \bot) \Rightarrow A$ and the law of Peirce $((A \Rightarrow B) \Rightarrow A) \Rightarrow A$, these last two being classical principles. Remark that in the first judgement we write $\mu\alpha.x$, and in the second example we write $\lambda x.[\alpha]\,x$. This is a consequence of using the extended syntax of [dG94, Sau05], where the empty type is handled as any other. The third term will be denoted `cc`, since it corresponds to the `call/cc` operator of the Scheme programming language which, as observed by Griffin in [Gri90], can be typed with the law of Peirce.

3.1.2 $\lambda\mu$ theories

We follow [Sel01] and define $\lambda\mu$-calculus as an equational theory. We only consider here the case of call-by-name semantics. The axioms of the call-by-name $\lambda\mu$-calculus are the following:

$$\begin{array}{llll}
(\beta_\to) & (\lambda x.M)\,N = M\,\{N/x\} & (\beta_\times) & \mathfrak{p}_i\,\langle M_1, M_2\rangle = M_i \\
(\eta_\to) & \lambda x.M\,x = M \qquad (x \notin \mathrm{FV}\,(M)) & (\eta_\times) & \langle \mathfrak{p}_1\,M, \mathfrak{p}_2\,M\rangle = M \\
(\zeta_\to) & (\mu\alpha.M)\,N = \mu\alpha.M\,\{[\alpha]\ _\,N / [\alpha]\ _\} & (\zeta_\times) & \mathfrak{p}_i\,(\mu\alpha.M) = \mu\alpha.M\,\{[\alpha]\ \mathfrak{p}_i\,_ / [\alpha]\ _\}
\end{array}$$

$$\begin{array}{ll}
(\beta_0) & [\alpha]\ \mu\beta.M = M\,\{\alpha/\beta\} \\
(\eta_0) & \mu\alpha.\,[\alpha]\ M = M \qquad (\alpha \notin \mathrm{FV}\,(M)) \\
(\zeta_0) & \mu\alpha.M = M\,\{\,_\,/\,[\alpha]\ _\} \qquad (\alpha : 0)
\end{array}$$

where in each equation the two terms are typed with the same type. From these axioms we define the notion of $\lambda\mu$ theory:

Definition 3.2 ($\lambda\mu$ theory)**.** *A $\lambda\mu$ theory is a set $\mathcal{T}$ of equations between typed terms of the same type (with free variables of the same type) which contains the axioms of call-by-name $\lambda\mu$-calculus and is a congruence (contextually-closed equivalence relation).*

We use here a slightly different set of axioms from that of Selinger [Sel01]. Our β_0 and η_0 equations are the β_μ and η_μ equations of Selinger, and we replace his $\beta_\bot$ (which is $[\alpha]\ M = M$ if $M : 0$) with our ζ_0. However, the two systems are equivalent:

Lemma 3.1. *Under the contextual closures of β_0 and η_0 (the β_μ and η_μ of Selinger), the equation $\beta_\bot$ of Selinger is equivalent to ζ_0.*

Proof. Suppose $\beta_\bot$ holds, let $M : 0$ and α be a μ-variable of type 0. Using the contextual closure of $\beta_\bot$, we have $M = M\,\{\,\_\,/\,[\alpha]\ \_\}$, and again by $\beta_\bot$ we get $M = [\alpha]\ M\,\{\,\_\,/\,[\alpha]\ \_\}$. Then by contextual closure $\mu\alpha.M = \mu\alpha.\,[\alpha]\ M\,\{\,\_\,/\,[\alpha]\ \_\}$, which is equal to $M\,\{\,\_\,/\,[\alpha]\ \_\}$ using η_0, since α does not appear free in $M\,\{\,\_\,/\,[\alpha]\ \_\}$.

Conversely, suppose ζ_0 holds, let $M : 0$, α be a μ-variable of type 0 and β be a μ-variable of type 0 which does not appear free in M. Using ζ_0 we have $M = \mu\beta.M$, so by contextual closure $[\alpha]\ M = [\alpha]\ \mu\beta.M$, so by β_0 we get $[\alpha]\ M = M\,\{\alpha/\beta\}$, but $M\,\{\alpha/\beta\} = M$ since β does not appear free in M, and we get finally $[\alpha]\ M = M$. □

Replacing Selinger's $\beta_\perp$ allows us to have a more symmetric calculus, with a set of three equations for each type constructor, the ζ equations representing the transmission of the context of a $\mu\alpha.M$ to the subterms $[\alpha]\ N$.

3.1.3 Interpreting classical logic in $\lambda\mu$-calculus

In this section we first interpret every formula A as a type A^* of $\lambda\mu$-calculus, and then every proof of a sequent $\Gamma \vdash A \mid \Delta$ as a typing derivation of a term $\Gamma^* \vdash M : A^* \mid \Delta^*$. This interpretation is the first component of our realizability interpretation of classical logic.

Fix a first-order signature Σ and a $\lambda\mu$ signature which has as set of base types the set of base sorts of Σ (so any sort on Σ is a type on the $\lambda\mu$ signature). Fix also for each predicate P of Σ a type P^* of $\lambda\mu$-calculus. We extend this interpretation to every formula on Σ the following way:

$$\perp^* = 0 \qquad (A \Rightarrow B)^* = A^* \to B^* \qquad (A \wedge B)^* = A^* \times B^* \qquad \left(\forall x^T\, A\right)^* = A^*$$

This interpretation is then extended to contexts: Γ^* (resp. Δ^*) is a context of λ-variables (resp. μ-variables) with types B^* for B in Γ (resp. Δ). We now describe how to interpret proofs in a first-order theory $\mathcal{A}x$ as typing derivations in $\lambda\mu$-calculus. Choose for each $A \in \mathcal{A}x$ a closed $\lambda\mu$-term M_A of type A^* which is a parameter of the interpretation. We interpret each first-order proof as a typing derivation in $\lambda\mu$-calculus the following way:

$$\left(\frac{}{\Gamma, A \vdash A \mid \Delta}\right)^* = \frac{}{\Gamma^*, x : A^* \vdash x : A^* \mid \Delta^*}$$

$$\left(\frac{}{\Gamma \vdash A \mid \Delta}\,{\scriptstyle (A \in \mathcal{A}x)}\right)^* = \Gamma^* \vdash M_A : A^* \mid \Delta^*$$

$$\left(\frac{\Gamma, A \vdash B \mid \Delta}{\Gamma \vdash A \Rightarrow B \mid \Delta}\right)^* = \frac{\Gamma^*, x : A^* \vdash M : B^* \mid \Delta^*}{\Gamma^* \vdash \lambda x.M : A^* \to B^* \mid \Delta^*}$$

$$\left(\frac{\Gamma \vdash A \Rightarrow B \mid \Delta \qquad \Gamma \vdash A \mid \Delta}{\Gamma \vdash B \mid \Delta}\right)^* = \frac{\Gamma^* \vdash M : A^* \to B^* \mid \Delta^* \qquad \Gamma^* \vdash N : A^* \mid \Delta^*}{\Gamma^* \vdash MN : B^* \mid \Delta^*}$$

$$\left(\frac{\Gamma \vdash A \mid \Delta \qquad \Gamma \vdash B \mid \Delta}{\Gamma \vdash A \wedge B \mid \Delta}\right)^* = \frac{\Gamma^* \vdash M : A^* \mid \Delta^* \qquad \Gamma^* \vdash N : B^* \mid \Delta^*}{\Gamma^* \vdash \langle M, N\rangle : A^* \times B^* \mid \Delta^*}$$

$$\left(\frac{\Gamma \vdash A_1 \wedge A_2 \mid \Delta}{\Gamma \vdash A_i \mid \Delta}\right)^* = \frac{\Gamma^* \vdash M : A_1^* \times A_2^* \mid \Delta^*}{\Gamma^* \vdash \mathfrak{p}_i\, M : A_i^* \mid \Delta^*}$$

$$\left(\frac{\Gamma \vdash A \mid \Delta}{\Gamma \vdash \forall x^T A \mid \Delta}\,{\scriptstyle (x^T \notin \mathrm{FV}(\Gamma, \Delta))}\right)^* = \Gamma^* \vdash M : A^* \mid \Delta^*$$

$$\left(\frac{\Gamma \vdash \forall x^T A \mid \Delta}{\Gamma \vdash A\left\{t^T/x^T\right\} \mid \Delta}\right)^* = \Gamma^* \vdash M : A^* \mid \Delta^*$$

$$\left(\frac{\Gamma \vdash A \mid \Delta, A}{\Gamma \vdash \perp \mid \Delta, A}\right)^* = \frac{\Gamma^* \vdash M : A^* \mid \Delta^*, \alpha : A^*}{\Gamma^* \vdash [\alpha]M : 0 \mid \Delta^*, \alpha : A^*}$$

$$\left(\frac{\Gamma \vdash \perp \mid \Delta, A}{\Gamma \vdash A \mid \Delta}\right)^* = \frac{\Gamma^* \vdash M : 0 \mid \Delta^*, \alpha : A^*}{\Gamma^* \vdash \mu\alpha.M : A^* \mid \Delta^*}$$

In the particular case of equational theories, we fix:

$$\neq_T{}^* \quad \triangleq \quad 0$$

$$M_{(\boldsymbol{refl})} \triangleq \lambda x.x : 0 \to 0 \qquad M_{(\boldsymbol{Leib})} \triangleq \lambda x.x : (A^* \to 0) \to A^* \to 0$$

and in the case of a relativized theory we fix:

$$(\!|.^\iota|\!)^* \quad \triangleq \quad \iota$$

for each base sort ι of Σ. Since $(\!|t^{T\to U}|\!)$ is defined as $\forall x^T \left((\!|x^T|\!) \Rightarrow (\!|(t\,x)^U|\!) \right)$, we have by induction on T:

$$(\!|.^T|\!)^* \triangleq T$$

3.2 Categories of continuations

Categories of continuations are to call-by-name $\lambda\mu$-calculus what cartesian closed categories are to λ-calculus, in the sense that if we fix a signature, there is a one-to-one correspondence between $\lambda\mu$ theories and categories of continuations together with an interpretation of the signature. In [Ong96], Ong defines $\lambda\mu$ categories by reformulating the syntax of $\lambda\mu$-calculus in categorical terms. Later on, Hofmann and Streicher proved the soundness and completeness of categories of continuations with respect to $\lambda\mu$-calculus, providing the first abstract version of $\lambda\mu$-categories. Later on, Selinger axiomatized these in [Sel01] under the name of control categories, proving that they are equivalent to categories of continuations, and therefore sound and complete with respect to call-by-name $\lambda\mu$-calculus. Moreover, he proved that the categorical dual of control categories are sound and complete with respect to call-by-value $\lambda\mu$-calculus. Here we are only interested in the call-by-name version, and we use the model of categories of continuations.

Definition 3.3 (Category of continuations)**.** *Let $\mathcal{C}$ be a distributive category, that is, a category with finite products and coproducts such that the canonical distributivity morphisms from $A \times B + A \times C$ to $A \times (B + C)$ are isomorphisms, and let $R \in Ob(\mathcal{C})$ be a fixed object such that all exponentials R^A for $A \in Ob(\mathcal{C})$ exist. Then the full subcategory $R^{\mathcal{C}}$ of $\mathcal{C}$ consisting of the objects R^A for $A \in \mathcal{C}$ is called a category of continuations.*

As observed in [LRS93], a category of continuations $R^{\mathcal{C}}$ is in particular a cartesian closed category:

Lemma 3.2. *If $R^{\mathcal{C}}$ is a category of continuations, then $R^{\mathcal{C}}$ is cartesian, and if $A \in Ob(\mathcal{C})$ and $R^B \in Ob\left(R^{\mathcal{C}}\right)$, then $R^{A\times B}$ defines an exponential in $\mathcal{C}$ of R^B by A. Consequently, $R^{\mathcal{C}}$ is cartesian closed, the exponential of R^B by R^A being $R^{R^A\times B}$.*

Proof. We define the terminal object as $R^{\mathbf{0}} \in Ob\left(R^{\mathcal{C}}\right)$, where $\mathbf{0}$ is the initial object of $\mathcal{C}$, the product of two objects $R^A, R^B \in Ob\left(R^{\mathcal{C}}\right)$ as $R^{A+B} \in Ob\left(R^{\mathcal{C}}\right)$, where $+$ is the coproduct of $\mathcal{C}$. The cartesian structure comes easily from the facts that the terminal object of $R^{\mathcal{C}}$ is isomorphic to the terminal object of $\mathcal{C}$, the product of two objects in $R^{\mathcal{C}}$ is isomorphic to their product in $\mathcal{C}$, and the universal properties translate from $\mathcal{C}$ to $R^{\mathcal{C}}$ since it is a full subcategory.

Let now $A \in Ob(\mathcal{C})$ and $R^B \in Ob\left(R^{\mathcal{C}}\right)$. We define an evaluation morphism from $R^{A\times B} \times A$ to R^B by currying (in $\mathcal{C}$):

$$\left(R^{A\times B} \times A\right) \times B \xrightarrow{\simeq} R^{A\times B} \times (A \times B) \xrightarrow{\mathbf{ev}} R$$

and the currying of $\phi : C \times A \to R^B$ is obtained by currying (again in $\mathcal{C}$):

$$C \times (A \times B) \xrightarrow{\simeq} (C \times A) \times B \xrightarrow{\phi \times \mathbf{Id}_B} R^B \times B \xrightarrow{\mathbf{ev}} R$$

the above evaluation and currying in $\mathcal{C}$ being correct since $\mathcal{C}$ has all exponentials R^D. □

Take care that we have two (isomorphic) terminal objects, one in $\mathcal{C}$ and one in $R^{\mathcal{C}}$, and two (isomorphic) products of R^A and R^B, again one in $\mathcal{C}$ and one in $R^{\mathcal{C}}$. To avoid confusion and without loss of generality we will suppose that they are equal: $R^{\mathbf{0}} = \mathbf{1}$ and $R^{A+B} = R^A \times R^B$. For the same reason, we also suppose $R = R^{\mathbf{1}}$.

3.2.1 Classical disjunction in categories of continuations.

We could have added a primitive connective $\vee$ for the disjunction in the logic, with the following rules:

$$\frac{\Gamma \vdash A \vee B \mid \Delta, A, B}{\Gamma \vdash \perp \mid \Delta, A, B} \qquad \frac{\Gamma \vdash \perp \mid \Delta, A, B}{\Gamma \vdash A \vee B \mid \Delta}$$

and then interpret these logical rules by adding the following typing rules to $\lambda\mu$-calculus:

$$\frac{\vec{x} : \vec{T} \vdash M : U_1 \parr U_2 \mid \vec{\alpha} : \vec{U}, \beta_1 : U_1, \beta_2 : U_2}{\vec{x} : \vec{T} \vdash [\beta_1, \beta_2]\ M : 0 \mid \vec{\alpha} : \vec{U}, \beta_1 : U_1, \beta_2 : U_2} \qquad \frac{\vec{x} : \vec{T} \vdash M : 0 \mid \vec{\alpha} : \vec{U}, \beta_1 : U_1, \beta_2 : U_2}{\vec{x} : \vec{T} \vdash \mu\,(\beta_1, \beta_2)\,.M : U_1 \parr U_2 \mid \vec{\alpha} : \vec{U}}$$

These rules are present in [Sel01], however we choose here to keep things simple and stick to the usual $\lambda\mu$-calculus without disjunction types. Nevertheless we still use the binoidal functor $\parr$ in categories of continuations to interpret multi-conclusioned sequents. Following [Sel01] we write: $R^A \parr R^B \stackrel{\Delta}{=} R^{A \times B}$ for the interpretation of the classical disjunction between R^A and R^B, $i : R \to R^A$ for the interpretation of the right weakening rule and $\nabla : R^A \parr R^A \to R^A$ for the interpretation of the right contraction rule.

Another interesting fact about categories of continuations is that we can define a functor from $\mathcal{C}^{op}$ to $R^{\mathcal{C}}$ which maps $A \in Ob(\mathcal{C})$ to $R^A \in Ob\left(R^{\mathcal{C}}\right)$, and $\phi : A \to B$ to $R^\phi : R^B \to R^A$ which is the currying of:

$$R^B \times A \xrightarrow{\mathbf{Id}_{R^B} \times \phi} R^B \times B \xrightarrow{\mathbf{ev}} R$$

The morphism $i : R \to R^A$ corresponds to $R^{(*_A)}$ (where $*_A$ is the unique morphism from A to the terminal object $\mathbf{1}$) and the morphism $\nabla : R^A \parr R^A \to R^A$ corresponds to $R^{\mathbf{pair}(\mathbf{Id}_A, \mathbf{Id}_A)}$ (remember that $R^A \parr R^A = R^{A \times A}$). Also, in particular, if $A \simeq B$ in $\mathcal{C}$, then $R^A \simeq R^B$ in $R^{\mathcal{C}}$ (and in $\mathcal{C}$). Through this functor, the cocartesian structure of $\mathcal{C}$ translates to the cartesian structure of $R^{\mathcal{C}}$.

3.2.2 Coproduct completion of a category

We give here an example of a category of continuations. We start from a cartesian closed category with countable products $\mathcal{D}$, we apply a countable coproduct completion (see e.g. [AM97]) to get a countably distributive category $\mathcal{C} = \mathrm{Fam}(\mathcal{D})$, and we prove that for a certain class of objects $R \in Ob(\mathcal{C})$, the categories $R^{\mathcal{C}}$ are categories of continuations. We perform the construction in the countable case because it will be useful in the next chapter, however the same results can be obtained in the finite case: if $\mathcal{D}$ is cartesian closed and if we apply a finite coproduct completion and choose well R in this completion we also obtain a category of continuations.

We fix now a cartesian closed category with countable products $\mathcal{D}$. The objects of $\mathrm{Fam}\,(\mathcal{D})$ are countable families of objects of $\mathcal{D}$:

$$\{A_i \,|\, i \in I\}$$

where I is any countable set. Then a morphism from $\{A_i \,|\, i \in I\}$ to $\{B_j \,|\, j \in J\}$ is given by a function $f : I \to J$ together with a family of morphisms $\phi_i : A_i \to B_{f(i)}$. The composition of two morphisms:

$$\left(f : I \to J, \left\{\phi_i : A_i \to B_{f(i)} \,\middle|\, i \in I\right\}\right) : \{A_i \,|\, i \in I\} \to \{B_j \,|\, j \in J\}$$

$$\left(g : J \to K, \left\{\psi_j : B_j \to C_{g(j)} \,\middle|\, j \in J\right\}\right) : \{B_j \,|\, j \in J\} \to \{C_k \,|\, k \in K\}$$

is the following morphism:

$$\left(g \circ f : I \to K, \left\{\phi_i ; \psi_{f(i)} : A_i \to C_{g \circ f(i)} \,\middle|\, i \in I\right\}\right) : \{A_i \,|\, i \in I\} \to \{C_k \,|\, k \in K\}$$

Associativity of composition is easy to prove. The identity $\mathbf{Id}_{\{A_i \,|\, i \in I\}}$ is defined as follows:

$$(Id_I : I \to I, \{\mathbf{Id}_{A_i} : A_i \to A_i \,|\, i \in I\}) : \{A_i \,|\, i \in I\} \to \{A_i \,|\, i \in I\}$$

Neutrality of the identity is also easy. As expected, the coproduct completion of a category $\mathcal{D}$ has all countable coproducts:

Lemma 3.3. $\mathrm{Fam}\,(\mathcal{D})$ *has all countable coproducts*

Proof. Let $(\{A_{i,j} \,|\, j \in J_i\})_{i \in I}$ be a countable family of objects of $\mathrm{Fam}\,(\mathcal{D})$. We define the coproduct of this family as:

$$\sum_{i \in I} \{A_{i,j} \,|\, j \in J_i\} = \left\{ A_{i,j} \,\middle|\, (i,j) \in \sum_{i \in I} J_i \right\}$$

which is an object of $\mathrm{Fam}\,(\mathcal{D})$, since the disjoint sum of the J_i is countable if I and the J_i are countable. The morphism $\mathbf{in}_{i_0}$ for $i_0 \in I$ is defined as:

$$\mathbf{in}_{i_0} = \left(\begin{array}{ll} J_{i_0} \longrightarrow \displaystyle\sum_{i \in I} J_i & , \quad \left\{ \mathbf{Id}_{A_{i_0,j}} : A_{i_0,j} \to A_{i_0,j} \,\middle|\, j \in J_{i_0} \right\} \\ j \longmapsto (i_0, j) & \end{array} \right) \\ : \{A_{i_0,j} \,|\, j \in J_{i_0}\} \to \sum_{i \in I} \{A_{i,j} \,|\, j \in J_i\}$$

The sum of an I-indexed family of morphisms from $\{A_{i,j} \,|\, j \in J_i\}$ to $\{B_k \,|\, k \in K\}$:

$$\left(f_i : J_i \longrightarrow K, \left\{\phi_{i,j} : A_{i,j} \to B_{f_i(j)} \,\middle|\, j \in J_i\right\}\right)_{i \in I}$$

is given by:

$$\left(\begin{array}{ll} \displaystyle\sum_{i \in I} J_i \longrightarrow K & , \quad \left\{ \phi_{i,j} : A_{i,j} \to B_{f_i(j)} \,\middle|\, (i,j) \in \sum_{i \in I} J_i \right\} \\ (i,j) \longmapsto f_i\,(j) & \end{array} \right) \\ : \sum_{i \in I} \{A_{i,j} \,|\, j \in J_i\} \to \{B_k \,|\, k \in K\}$$

The universal property is straightforward from the definitions. The empty sum and initial object $\mathbf{0}$ is just the empty family $\{\}$. □

The coproducts completion of a category with countable products has finite products:

Lemma 3.4. *If $\mathcal{D}$ has countable products, then* $\mathrm{Fam}(\mathcal{D})$ *has finite products*

Proof. The proof is obtained by distributing the categorical product on the index sets, relying on the fact that the finite product of countable families is still a countable family. The product of a finite (possibly empty with $n = 0$) family of objects:

$$(\{A_{i,j} \mid j \in J_i\})_{1 \leq i \leq n}$$

is defined as:

$$\prod_{1 \leq i \leq n} \{A_{i,j} \mid j \in J_i\} = \left\{ \prod_{1 \leq i \leq n} A_{i,j_i} \;\middle|\; (j_i)_{1 \leq i \leq n} \in \prod_{1 \leq i \leq n} J_i \right\}$$

which is an object of $\mathrm{Fam}(\mathcal{D})$, since the finite product of the J_i is countable if the J_i are countable. The morphism $\mathbf{proj}_{i_0}$ for $i_0 \in I$ is defined as:

$$\mathbf{proj}_{i_0} = \left(\begin{array}{c} \prod_{1 \leq i \leq n} J_i \longrightarrow J_{i_0} \\ (j_i)_{1 \leq i \leq n} \longmapsto j_{i_0} \end{array} \quad , \quad \left\{ \mathbf{proj}_{i_0} : \prod_{1 \leq i \leq n} A_{i,j_i} \to A_{i_0,j_{i_0}} \;\middle|\; (j_i)_{1 \leq i \leq n} \in \prod_{1 \leq i \leq n} J_i \right\} \right)$$
$$: \prod_{1 \leq i \leq n} \{A_{i,j} \mid j \in J_i\} \to \{A_{i_0,j} \mid j \in J_{i_0}\}$$

The product of a J-indexed family of morphisms from $\{A_i \mid 1 \leq i \leq n\}$ to $\{B_{j,k} \mid k \in K_j\}$:

$$\left(f_j : I \longrightarrow K_j, \left\{ \phi_{i,j} : A_i \to B_{j,f_j(i)} \;\middle|\; 1 \leq i \leq n \right\} \right)_{j \in J}$$

is given by:

$$\left(\begin{array}{c} \{1; ...; n\} \longrightarrow \prod_{j \in J} K_j \\ i \longmapsto (f_j(i))_{j \in J} \end{array} \quad , \quad \left\{ (\phi_{i,j})_{j \in J} : A_i \to \prod_{j \in J} B_{j,f_j(i)} \;\middle|\; 1 \leq i \leq n \right\} \right)$$
$$: \{A_i \mid 1 \leq i \leq n\} \to \prod_{j \in J} \{B_{j,k} \mid k \in K_j\}$$

Here again, the universal property is straightforward from the definitions and the universal property in $\mathcal{D}$. It is easy to see that the empty product and terminal object of $\mathrm{Fam}(\mathcal{D})$ is the singleton family $\{\mathbf{1}\}$. □

The product and coproduct structures of $\mathrm{Fam}(\mathcal{D})$ make it a countably distributive category:

Lemma 3.5. *If $\mathcal{D}$ has binary products, then* $\mathrm{Fam}(\mathcal{D})$ *is a countably distributive category*

Proof. Let $\{A_i \mid i \in I\}$ be an object of $\mathrm{Fam}(\mathcal{D})$ and $(\{B_{j,k} \mid k \in K_j\})_{j \in J}$ be a J-indexed family of objects of $\mathrm{Fam}(\mathcal{D})$. On one hand we have:

$$\begin{aligned} \sum_{j \in J} (\{A_i \mid i \in I\} \times \{B_{j,k} \mid k \in K_j\}) &= \sum_{j \in J} \{A_i \times B_{j,k} \mid (i,k) \in I \times K_j\} \\ &= \left\{ A_i \times B_{j,k} \;\middle|\; (j,i,k) \in \sum_{j \in J} (I \times K_j) \right\} \end{aligned}$$

$$= \left\{ A_i \times B_{j,k} \,\middle|\, (i,j,k) \in I \times \sum_{j \in J} K_j \right\}$$

and on the other hand:

$$\{A_i \mid i \in I\} \times \sum_{j \in J} \{B_{j,k} \mid k \in K_j\} = \{A_i \mid i \in I\} \times \left\{ B_{j,k} \,\middle|\, (j,k) \in \sum_{j \in J} K_j \right\}$$
$$= \left\{ A_i \times B_{j,k} \,\middle|\, (i,j,k) \in I \times \sum_{j \in J} K_j \right\}$$

Therefore, the two objects are the same and the identity is an isomorphism between the two. □

The following property gives a class of objects $R \in \mathrm{Fam}(\mathcal{D})$ such that $R^{\mathrm{Fam}(\mathcal{D})}$ is a category of continuations.

Lemma 3.6. *If $\mathcal{D}$ is cartesian closed and has countable products, then* $\mathrm{Fam}(\mathcal{D})$ *has all exponential of singleton families*

Proof. The exponential of $\{A\}$ by $\{B_i \mid i \in I\}$ is defined as:

$$\{A\}^{\{B_i \mid i \in I\}} = \left\{ \prod_{i \in I} A^{B_i} \right\}$$

A morphism from $\{C_j \mid j \in J\} \times \{B_i \mid i \in I\}$ to $\{A\}$ is just given by a family of morphisms:

$$\phi = \{\phi_{i,j} : C_j \times B_i \to A \mid (i,j) \in I \times J\} : \{C_j \mid j \in J\} \times \{B_i \mid i \in I\} \to \{A\}$$

since there is only one function to the singleton set. Then the currying of such a morphism is given by:

$$\mathbf{\Lambda}(\phi) = \left\{ (\mathbf{\Lambda}(\phi_{i,j}))_{i \in I} : C_j \to \prod_{i \in I} A^{B_i} \,\middle|\, j \in J \right\} : \{C_j \mid j \in J\} \to \{A\}^{\{B_i \mid i \in I\}}$$

since again the codomain is a singleton family. Finally, the evaluation morphism **ev** is defined as:

$$\mathbf{ev} = \left\{ \left(\mathbf{proj}_{i_0} \times \mathbf{Id}_{B_{i_0}} \right); \mathbf{ev} : \prod_{i \in I} A^{B_i} \times B_{i_0} \to A \,\middle|\, i_0 \in I \right\} : \{A\}^{\{B_i \mid i \in I\}} \times \{B_i \mid i \in I\} \to \{A\}$$

The universal property is then an easy consequence of the universal property in $\mathcal{D}$ and the definitions. □

From this lemma it follows that for any $A \in Ob(\mathcal{D})$, the category $\{A\}^{\mathrm{Fam}(\mathcal{D})}$ is a category of continuations.

3.2.3 Interpretation of $\lambda\mu$-calculus in categories of continuations

We describe here the interpretation of call-by-name $\lambda\mu$-calculus in a category of continuations as defined in [Sel01]. We fix a signature of $\lambda\mu$-calculus and a category of continuations $R^{\mathcal{C}}$. To each type T of $\lambda\mu$-calculus we associate an object $[\![T]\!] \in Ob(\mathcal{C})$ of continuations of type T, and an object $[T] = R^{[\![T]\!]} \in Ob\left(R^{\mathcal{C}}\right)$ of computations of type T. The objects $[\![\iota]\!] \in Ob(\mathcal{C})$ where ι is a base type of the signature are parameters of the interpretation, and we define inductively:

$$[\![T \to U]\!] = [T] \times [\![U]\!] \in Ob(\mathcal{C}) \qquad [\![T \times U]\!] = [\![T]\!] + [\![U]\!] \in Ob(\mathcal{C})$$
$$[\![0]\!] = \mathbf{1} \qquad [T] = R^{[\![T]\!]} \in Ob\left(R^{\mathcal{C}}\right)$$

We have in particular $[T \times U] = R^{[\![T]\!]+[\![U]\!]} = R^{[\![T]\!]} \times R^{[\![U]\!]} = [T] \times [U]$ where $\times$ is the cartesian product in $R^{\mathcal{C}}$ defined above, $[T \to U] = R^{[T]\times[\![U]\!]} = R^{R^{[\![T]\!]}\times[\![U]\!]} = \left(R^{[\![U]\!]}\right)^{R^{[\![T]\!]}} = [U]^{[T]}$, using the definition of the exponential in $R^{\mathcal{C}}$ given above, and $[0] = R^{\mathbf{1}} = R$.

Once we have an interpretation of types in $R^{\mathcal{C}}$, we define the interpretation of typed $\lambda\mu$-terms such that a term:

$$x_1 : T_1, ..., x_n : T_n \vdash M : V \mid \alpha_1 : U_1, ..., \alpha_m : U_m$$

is interpreted as a morphism in $R^{\mathcal{C}}$:

$$[x_1 : T_1, ..., x_n : T_n \vdash M : V \mid \alpha_1 : U_1, ..., \alpha_m : U_m] : [T_1] \times ... \times [T_n] \to [V] \parr [U_1] \parr ... \parr [U_m]$$

where $[T_1] \times ... \times [T_n]$ associates to the left and $[V] \parr [U_1] \parr ... \parr [U_m]$ associates to the right. In order to do that, we suppose given for each constant $\mathsf{c} : T \in \mathcal{C}st$ of the signature a morphism in $R^{\mathcal{C}}$:

$$[\mathsf{c} : T] : \mathbf{1} \to [T]$$

which is again a parameter of the interpretation. These parameters are summarized in the following definition:

Definition 3.4 (Interpretation)**.** *Given a signature and a category of continuations $R^{\mathcal{C}}$, an interpretation of $\lambda\mu$-calculus is given by an object $[\![\iota]\!] \in \mathcal{C}$ for each base type ι of the signature and a morphism $[\mathsf{c} : T] : \mathbf{1} \to [T]$ in $R^{\mathcal{C}}$ for each constant $\mathsf{c} : T \in \mathcal{C}st$ of the signature.*

We now have all necessary material to interpret every typed $\lambda\mu$-term as a morphism in $R^{\mathcal{C}}$. The interpretation of typed $\lambda\mu$-terms is almost identical to the interpretation of λ-calculus in a cartesian closed category (since as shown in the previous section, $R^{\mathcal{C}}$ is cartesian closed). The first difference is that we must be able to carry over the μ-context, so we want to build from $\phi : [T] \to [U]$ a morphism $\phi \parr [V] : [T] \parr [V] \to [U] \parr [V]$. The second difference is that in order to interpret the introduction rules for $\mu\alpha.M$ and $[\alpha]\ M$, we also need to have canonical morphisms from $[0] \parr \left[\vec{U}\right] \parr [T]$ to $[T] \parr \left[\vec{U}\right]$ and from $[T] \parr \left[\vec{U}\right] \parr [T]$ to $[0] \parr \left[\vec{U}\right] \parr [T]$. These requirements are axiomatized in [Sel01], to which we refer for the full definition of the interpretation, and the proof that the axioms of call-by-name $\lambda\mu$-calculus are sound under this interpretation.

Since call-by-name $\lambda\mu$-calculus is the internal language of categories of continuations (as shown in [Sel01]), we can apply $\lambda\mu$-calculus constructs on morphisms of $R^{\mathcal{C}}$ through the use of $\lambda\mu$-terms with parameters in $R^{\mathcal{C}}$. For example, if $\phi : \prod_{j\in J} R^{A_j} \to R^B \parr \left(\parrop_{k\in K} R^{D_k}\right)$ and $\psi : \prod_{j\in J} R^{A_j} \to R^{R^B\times C} \parr \left(\parrop_{k\in K} R^{D_k}\right)$, then $\psi\,\phi : \prod_{j\in J} R^{A_j} \to R^C \parr \left(\parrop_{k\in K} R^{D_k}\right)$, where formally $\psi\,\phi$ is the term with parameters $(y\,x)\,\{\phi/x, \psi/y\}$.

3.2.4 Connection with the call-by-name CPS translation of $\lambda\mu$-calculus

Another interesting thing about the interpretation of $\lambda\mu$-calculus into a category of continuations is the exact correspondence with the interpretation of its call-by-name CPS translation in the underlying cartesian "R-closed" category, as stressed in [HS02]. The target of such a translation is a simply-typed λ-calculus $\lambda^{R\times+}$ with product and sum types, a particular base type R, and the function types being restricted to $T \to R$. This particular λ-calculus can be interpreted in $\mathcal{C}$ by interpreting the product type as the product in $\mathcal{C}$, the sum type as the coproduct, and using the fact that function types are of the form $T \to R$, so having all exponentials R^A in $\mathcal{C}$ is enough.

To be more precise, $\lambda^{R\times+}$ has one base type $\tilde{\iota}$ for each base type ι of $\lambda\mu$-calculus and another particular base type R. From these we build the types:

$$T, U \quad ::= \quad \widetilde{\iota} \mid T \to R \mid T \times U \mid 1 \mid T + U$$

The arrow types are syntactically restricted to be of the form $T \to R$. We map every type T of $\lambda\mu$-calculus to a type $\widetilde{T}$ of $\lambda^{R\times+}$ (each base type ι being obviously mapped to $\widetilde{\iota}$) as follows:

$$\widetilde{T \to U} = \left(\widetilde{T} \to R\right) \times \widetilde{U} \qquad \widetilde{T \times U} = \widetilde{T} + \widetilde{U} \qquad \widetilde{0} = 1$$

$\lambda^{R\times+}$ also has one constant $\widetilde{\mathsf{c}} : \widetilde{T} \to R$ for each constant $\mathsf{c} : T$ of the source language. We also suppose given for each λ-variable $x : T$ of the source language a variable $\widetilde{x} : \widetilde{T} \to R$ in the target language, and for each μ-variable $\alpha : U$ of the source language a variable $\widetilde{\alpha} : \widetilde{U}$ in the target language. A typed $\lambda\mu$-term:

$$x_1 : T_1, ..., x_n : T_n \vdash M : T \mid \alpha_1 : U_1, ..., \alpha_m : U_m$$

will then be translated to a typed λ-term:

$$\widetilde{x_1} : \widetilde{T_1} \to R, ..., \widetilde{x_n} : \widetilde{T_n} \to R, \widetilde{\alpha_1} : \widetilde{U_1}, ..., \widetilde{\alpha_m} : \widetilde{U_m} \vdash \widetilde{M} : \widetilde{T} \to R \tag{3.1}$$

Before defining the translation, we give the typing rules of $\lambda^{R\times+}$:

$$\frac{}{\vec{k} : \vec{T}, k : T \vdash k : T} \qquad \frac{}{\vec{k} : \vec{T} \vdash \widetilde{\mathsf{c}} : \left(\widetilde{T} \to R\right)}\ (\widetilde{\mathsf{c}} : \left(\widetilde{T} \to R\right) \in \mathcal{C}st)$$

$$\frac{\vec{k} : \vec{T}, k : T \vdash M : R}{\vec{k} : \vec{T} \vdash \lambda k.M : T \to R} \qquad \frac{\vec{k} : \vec{T} \vdash M : T \to R \qquad \vec{k} : \vec{T} \vdash N : T}{\vec{k} : \vec{T} \vdash MN : R}$$

$$\frac{}{\Gamma \vdash * : 1} \qquad \frac{\vec{k} : \vec{T} \vdash M : T \qquad \vec{k} : \vec{T} \vdash N : U}{\vec{k} : \vec{T} \vdash \langle M, N \rangle : T \times U} \qquad \frac{\vec{k} : \vec{T} \vdash M : T_1 \times T_2}{\vec{k} : \vec{T} \vdash \mathfrak{p}_i\, M : T_i}$$

$$\frac{\Gamma \vdash M : T_i}{\Gamma \vdash \mathsf{in}_i\, M : T_1 + T_2} \qquad \frac{\Gamma \vdash M : T_1 + T_2 \qquad \Gamma, k : T_1 \vdash N_1 : U \qquad \Gamma, k : T_2 \vdash N_2 : U}{\Gamma \vdash \mathsf{case}\, M\, \{\mathsf{in}_1\, k \mapsto N_1 \mid \mathsf{in}_2\, k \mapsto N_2\} : U}$$

The typing rules for the arrow type are restricted to the case $T \to R$. The translation of a variable x or a constant c is $\widetilde{x}$ or $\widetilde{\mathsf{c}}$ as defined above, and the remaining part is as follows:

$$\widetilde{\lambda x.M} = \lambda k.\left(\lambda \widetilde{x}.\widetilde{M}\right)(\pi_1\, k)\,(\pi_2\, k) \qquad \widetilde{M\,N} = \lambda k.\widetilde{M}\left\langle \widetilde{N}, k \right\rangle$$

$$\widetilde{\langle M, N \rangle} = \lambda k.\mathsf{case}\, k \left\{\mathsf{in}_1\, l \mapsto \widetilde{M}\, l \mid \mathsf{in}_2\, l \mapsto \widetilde{N}\, l\right\} \qquad \widetilde{\mathfrak{p}_i\, M} = \lambda k.\widetilde{M}\, \mathsf{in}_i\, k$$

$$\widetilde{\mu\alpha.M} = \lambda\widetilde{\alpha}.\widetilde{M} * \qquad \widetilde{[\alpha]\ M} = \lambda k.\widetilde{M}\,\widetilde{\alpha}$$

where k is always a fresh variable. We define the interpretation of $\lambda^{R\times+}$ in $\mathcal{C}$ by first giving an object $[\![T]\!]$ of $\mathcal{C}$ for each type T:

$$[\![\iota]\!] = [\![\iota]\!] \qquad [\![T \to R]\!] = R^{[\![T]\!]} \qquad [\![T \times U]\!] = [\![T]\!] \times [\![U]\!] \qquad [\![1]\!] = \mathbf{1} \qquad [\![T + U]\!] = [\![T]\!] + [\![U]\!]$$

The notation $[\![\_]\!]$ may seem misleading, but it is on purpose, since one can easily see that if T is a type of $\lambda\mu$-calculus, then $\left[\!\left[\widetilde{T}\right]\!\right] = [\![T]\!]$. Since the only function types of our λ-calculus are of the form $T \to R$ and since $\mathcal{C}$ has all exponentials R^A, we can interpret $\widetilde{M}$ in $\mathcal{C}$ the same way we would interpret simply typed λ-calculus in a cartesian closed category. The term $\widetilde{M}$ of (3.1) is interpreted as:

$$\left[\!\left[\widetilde{M}\right]\!\right] : R^{[\![\widetilde{T_1}]\!]} \times ... \times R^{[\![\widetilde{T_n}]\!]} \times \left[\!\left[\widetilde{U_1}\right]\!\right] \times ... \times \left[\!\left[\widetilde{U_m}\right]\!\right] \to R^{[\![\widetilde{T}]\!]}$$

which is, by the above observation that $\left[\!\left[\widetilde{T}\right]\!\right] = [\![T]\!]$:

$$\left[\!\left[\widetilde{M}\right]\!\right] : R^{[\![T_1]\!]} \times ... \times R^{[\![T_n]\!]} \times [\![U_1]\!] \times ... \times [\![U_m]\!] \to R^{[\![T]\!]}$$

Now, by currying we obtain:

$$\mathbf{\Lambda}\left(\left[\!\left[\widetilde{M}\right]\!\right]\right) : R^{[\![T_1]\!]} \times ... \times R^{[\![T_n]\!]} \to R^{[\![T]\!]\times[\![U_1]\!]\times...\times[\![U_m]\!]}$$

and we have the following result:

$$\mathbf{\Lambda}\left(\left[\!\left[\widetilde{M}\right]\!\right]\right) = [M]$$

where the brackets on the left represent the interpretation of $\lambda^{R\times+}$ in the cartesian "R-closed" category $\mathcal{C}$, and the brackets on the right represent the interpretation of $\lambda\mu$-calculus in the category of continuations $R^{\mathcal{C}}$.

On the equational side, $\lambda^{R\times+}$ has the following set of equations, where the two terms are of the same type:

$$\left(\beta^{\lambda}_{\to}\right) \quad (\lambda k.M)\, N = M\,\{N/k\} \qquad \left(\eta^{\lambda}_{\to}\right) \quad \lambda k.M\, k = M \quad (k \notin \mathrm{FV}\,(M))$$

$$\left(\beta^{\lambda}_{\times}\right) \quad \mathfrak{p}_i\,\langle M_1, M_2\rangle = M_i \qquad \left(\eta^{\lambda}_{\times}\right) \quad \langle \mathfrak{p}_1\, M, \mathfrak{p}_2\, M\rangle = M$$

$$\left(\beta^{\lambda}_{+}\right) \quad \mathsf{case}\ (\mathsf{in}_i\, M) \begin{Bmatrix} \mathsf{in}_1\, k \mapsto N_1 \\ \mathsf{in}_2\, k \mapsto N_2 \end{Bmatrix} = N_i\,\{M/k\} \qquad \left(\eta^{\lambda}_{+}\right) \quad \mathsf{case}\, M \begin{Bmatrix} \mathsf{in}_1\, k \mapsto \mathsf{in}_1\, k \\ \mathsf{in}_2\, k \mapsto \mathsf{in}_2\, k \end{Bmatrix} = M$$

$$\left(\eta^{\lambda}_{1}\right) \quad * = M$$

Since these equations are typed, M is of type 1 in $\left(\eta^{\lambda}_{1}\right)$. If M and N are $\lambda\mu$-terms of the same type, then $M = N$ holds using the equations $(\beta_\to)$, $(\eta_\to)$, $(\zeta_\to)$, $(\beta_\times)$, $(\eta_\times)$, $(\zeta_\times)$, (β_0), (η_0) and (ζ_0) of section 3.1.2 if and only if $\widetilde{M} = \widetilde{N}$ holds using the equations $\left(\beta^{\lambda}_{\to}\right)$, $\left(\eta^{\lambda}_{\to}\right)$, $\left(\beta^{\lambda}_{\times}\right)$, $\left(\eta^{\lambda}_{\times}\right)$, $\left(\beta^{\lambda}_{+}\right)$, $\left(\eta^{\lambda}_{+}\right)$ and $\left(\eta^{\lambda}_{1}\right)$.

Just as we did for $\lambda\mu$-calculus and categories of continuations, we use terms of $\lambda^{R\times+}$ with parameters in $\mathcal{C}$. For example, if $\zeta : \prod_{j\in J} A_i \to B$ and $\varphi : \prod_{j\in J} A_i \to R^B$, then $\varphi\,\zeta : \prod_{j\in J} A_i \to R$, where formally $\varphi\,\zeta$ is the term with parameters $(y\,x)\,\{\zeta/x, \varphi/y\}$.

3.2.5 Interactions between $\mathcal{C}$ and $R^{\mathcal{C}}$

We will also extend the $\lambda\mu$-terms with parameters of section 3.2.3 by allowing the substitution of terms of $\lambda^{R\times+}$ with parameters in $\mathcal{C}$ (defined in section 3.2.4) for μ-variables of $\lambda\mu$-terms

with parameters in $R^{\mathcal{C}}$. For example, if $\phi : \prod_{j\in J} R^{A_j} \to R^B \parr \left(\parr_{k\in K} R^{C_k}\right)$ in $R^{\mathcal{C}}$ and $\zeta : \left(\prod_{j\in J} R^{A_j}\right) \times \left(\prod_{k\in K} C_k\right) \to B$ in $\mathcal{C}$, then $[\zeta]\ \phi : \prod_{j\in J} R^{A_j} \to R^{\mathbf{1}} \parr \left(\parr_{k\in K} R^{C_k}\right)$ in $R^{\mathcal{C}}$. It can be shown that as expected:

$$\left[\left\langle \widetilde{\phi}, \zeta \right\rangle\right]\ \psi = [\zeta]\ \psi\, \phi \qquad \text{if} \begin{cases} \phi : \displaystyle\prod_{j\in J} R^{A_j} \to R^B \parr \left(\parr_{k\in K} R^{D_k}\right) \\ \psi : \displaystyle\prod_{j\in J} R^{A_j} \to R^{R^B \times C} \parr \left(\parr_{k\in K} R^{D_k}\right) \\ \zeta : \left(\displaystyle\prod_{j\in J} R^{A_j}\right) \times \left(\displaystyle\prod_{k\in K} D_k\right) \to C \end{cases}$$

$$[*]\ \phi = \phi \qquad \text{if } \phi : \prod_{j\in J} R^{A_j} \to R^{\mathbf{1}} \parr \left(\parr_{k\in K} R^{D_k}\right)$$

$$[\mathsf{in}_i\, \zeta]\ \phi = [\zeta]\ \mathfrak{p}_i\, \phi \qquad \text{if} \begin{cases} \phi : \displaystyle\prod_{j\in J} R^{A_j} \to \left(R^{B_1} \times R^{B_2}\right) \parr \left(\parr_{k\in K} R^{D_k}\right) \\ \zeta : \left(\displaystyle\prod_{j\in J} R^{A_j}\right) \times \left(\displaystyle\prod_{k\in K} D_k\right) \to B_i \end{cases}$$

This possibility of having terms of $\lambda^{R\times+}$ inside the brackets of $\lambda\mu$-terms is closely related to the $\overline{\lambda}\mu\widetilde{\mu}$-calculus of [Her95, CH00], extended to handle products (using sums of $\lambda^{R\times+}$).

3.3 μPCF

Programming language for Computable Functions (PCF) is a functional programming language described by Plotkin in [Plo77]. It is based on Scott's Logic for Computable Functions (LCF), which was presented in [Sco93]. The language contains constants for natural numbers and general recursion. It is probably the simplest example of a Turing-complete higher-order language. Here we consider an extension of PCF to primitively handle control operators, by presenting PCF as a $\lambda\mu$-theory. In [OS97] the authors define a call-by-value semantics for $\lambda\mu$-calculus, and they illustrate it with $\mu\mathrm{PCF}_V$, a call-by-value version of PCF with control. Later on, Laird defines in [Lai99] its call-by-name version, which is the version we use here. The choice of this language is justified on one hand by our will to get computational content directly from classical proofs, and on the other hand by the fact that the games model can be seen primitively as a model of μPCF.

3.3.1 Type constants

In his original formulation of PCF, Plotkin considered two base types: one for boolean and one for natural numbers. Since it is easy to encode booleans into natural numbers (for example, false is zero and true is any other natural number), we will only consider here one base type, the type of natural numbers: ι.

3.3.2 Constants

Since we do not have a type for booleans, we factorize the conditional taking a boolean and the "test-to-zero" with a single conditional $\mathsf{if0}$ branching depending on its first argument being zero

or not. In order to manipulate natural numbers we need the predecessor and successor functions, and to be able to write any computable function we take a general fixed point operator. The constants of μPCF are then:

$$\overline{\mathsf{n}} : \iota \qquad \mathsf{succ} : \iota \to \iota \qquad \mathsf{pred} : \iota \to \iota \qquad \mathsf{if}_0 : \iota \to T \to T \qquad \mathsf{Y} : (T \to T) \to T$$

It will also be useful to have a canonical term on each type so we define:

$$\Omega \stackrel{\Delta}{=} \mathsf{Y}\,(\lambda x.x) : T$$

This term represents non-termination, or "undefined".

3.3.3 Equations

Now we need to describe how these constructs relate to each other. Since we will work in a model of μPCF, we are not interested into reduction, but only into equality. A reduction relation can easily be defined, and the corresponding equivalence relation on ground types is the equality defined here (see for example [HO00]). Since we are in the context of $\lambda\mu$-calculus, we also need the equations ruling the interaction between the constants and the μ operator. The equations ruling the behavior of pred and succ are standard (since we work with natural numbers, the predecessor of 0 is 0):

$$\mathsf{succ}\,\overline{\mathsf{n}} = \overline{\mathsf{n}+1} \qquad \mathsf{succ}\,(\mu\alpha.M) = \mu\alpha.M\,\{[\alpha]\ \mathsf{succ}\,(\_) \,/\, [\alpha]\ \_\}$$

$$\mathsf{pred}\,\overline{0} = \overline{0} \qquad \mathsf{pred}\,\overline{\mathsf{n}+1} = \overline{\mathsf{n}} \qquad \mathsf{pred}\,(\mu\alpha.M) = \mu\alpha.M\,\{[\alpha]\ \mathsf{pred}\,(\_) \,/\, [\alpha]\ \_\}$$

For the conditional if_0, it is not much a surprise that the defining equations are:

$$\mathsf{if}_0\,\overline{0}\,M\,N = M \qquad \mathsf{if}_0\,\overline{\mathsf{n}+1}\,M\,N = N \qquad \mathsf{if}_0\,(\mu\alpha.M)\ N\,P = \mu\alpha.M\,\{[\alpha]\ \mathsf{if}_0\,(\_)\ N\,P \,/\, [\alpha]\ \_\}$$

and the equation for fixed point operator Y is also standard:

$$\mathsf{Y}\,M = M\,(\mathsf{Y}\,M)$$

3.4 System T

System T was introduced by Gödel in [Göd58] in order to give a consistency proof of Heyting arithmetic (and therefore of Peano arithmetic by double-negation translation). This system can be equivalently formulated as a system of primitive recursive functionals, which is an extension of primitive recursive functions to higher types. It is strictly more powerful than primitive recursion, since for example the Ackermann's function is expressible in system T.

System T has one base type for natural numbers, product and function types, constants for 0 and successor, and a recursion operator of type $T \to (\iota \to T \to T) \to \iota \to T$ for any type T. Restricting the type of the recursor to $T = \iota$ gives back the usual primitive recursive functions.

Since μPCF contains constants for every natural number, successor, predecessor and general recursion, it is easy to encode system T in it. Indeed, if we define:

$$\mathsf{rec} \stackrel{\Delta}{=} \lambda xy.\mathsf{Y}\,(\lambda zu.\mathsf{if}_0\,u\,x\,(y\,(\mathsf{pred}\,u)\,(z\,(\mathsf{pred}\,u))))$$

Then it is easy to derive:

$$\mathsf{rec} : T \to (\iota \to T \to T) \to \iota \to T$$

and in order to prove that it implements Gödel's recursor we must prove that it satisfies the corresponding equations:

Lemma 3.7. *Let $M : T$ and $N : \iota \to T \to T$. We have:*

$$\mathsf{rec}\, M\, N\, \overline{0} = M$$

and for any $n \in \mathbb{N}$:

$$\mathsf{rec}\, M\, N\, \overline{n+1} = N\, \overline{n}\, (\mathsf{rec}\, M\, N\, \overline{n})$$

Proof. This follows easily from the definition of rec and the equations of μPCF for pred, if_0 and Y. □

Therefore in the following we will consider system T as a subsystem of μPCF.

3.4.1 Interpreting $PA^{\omega \mathrm{r}}$ in system T + Ω

In order to interpret $PA^{\omega \mathrm{r}}$ in system T, we first need to provide a term M_A of type A^* for each $A \in PA^{\omega \mathrm{r}}$:

$$M_{(\boldsymbol{\Delta s})} \triangleq \lambda x.x : 0 \to 0 \qquad M_{(\boldsymbol{\Delta k})} \triangleq \lambda x.x : 0 \to 0 \qquad M_{(\boldsymbol{\Delta rec0})} \triangleq \lambda x.x : 0 \to 0$$

$$M_{(\boldsymbol{\Delta recS})} \triangleq \lambda x.x : 0 \to 0 \qquad M_{(\boldsymbol{Snz})} \triangleq \Omega : 0$$

$$M_{(\boldsymbol{ind^r})} \triangleq \mathsf{rec} : B^* \to (\iota \to B^* \to B^*) \to \iota \to B^*$$

for every formula B with free variables among $x^\iota, \vec{y^T}$. Finally, concerning the relativization axioms:

$$M_{(\!|\mathsf{s}|\!)} \triangleq \lambda xyz.x\, z\, (y\, z) : (T \to U \to V) \to (T \to U) \to T \to V$$

$$M_{(\!|\mathsf{k}|\!)} \triangleq \lambda xy.x : T \to U \to T$$

$$M_{(\!|\mathsf{0}|\!)} \triangleq \overline{0} : \iota$$

$$M_{(\!|\mathsf{S}|\!)} \triangleq \mathsf{succ} : \iota \to \iota$$

$$M_{(\!|\mathsf{rec}|\!)} \triangleq \mathsf{rec} : T \to (\iota \to T \to T) \to \iota \to T$$

3.5 Bar recursion

Bar recursion is an operator which can be seen as recursion on well-founded trees. It was first introduced by Spector in [Spe62] to extend Gödel's dialectica interpretation to Heyting arithmetic augmented with the axiom of countable choice. A variant of this operator was studied in [Koh90], and another more uniform operator which is very similar to bar recursion was introduced in [BBC98] and used in a realizability setting. A version in which the well-foundedness of trees is implicit was proposed in [BO05] under the name of modified bar recursion. For a comparison between these different forms of bar recursion and other similar principles we refer the reader to [Pow13, Pow14].

Here we consider a slight extension of the modified bar recursion of [BO05] taking benefits of control operators to allow an arbitrary return type. However, we will see that this operator is not total in the general case, a sufficient condition being that the return type is a base type, in which case our bar recursion becomes equivalent to the modified bar recursion of [BO05]. We did not investigate further to find a less restrictive condition.

In this section, we first recall general results about well-founded recursion, then we present recursion on natural numbers as a particular instance of it, and we introduce the bar recursion operator as the equivalent of system T's recursor, but for recursion over well-founded trees rather than natural numbers. Finally, we define formally the bar recursor in μPCF and give an illustration of its non-totality in the general case.

3.5.1 Well-founded induction and recursion

We recall here the basic principles of well-founded induction and recursion.

Definition 3.5 (Well-founded relation)**.** *A (binary) relation R on a set E is well-founded if every non-empty subset of E has a minimal element, that is:*

$$\forall F \subseteq E, F \neq \emptyset \Rightarrow \exists m \in F, \forall f \in F, \neg (f \, R \, m)$$

In the following we will write $R^{-1}(e) = \{f \in E \mid f \, R \, e\}$. With this notation, R is well-founded if and only if:

$$\forall F \subseteq E, F \neq \emptyset \Rightarrow \exists m \in F, R^{-1}(m) = \emptyset$$

Using the axiom of dependent choice, this condition is equivalent to the non-existence of infinite sequences of decreasing elements:

$$\forall (e_n)_{n \in \mathbb{N}} \in E^{\mathbb{N}}, \exists n \in \mathbb{N}, \neg (e_{n+1} \, R \, e_n)$$

The first property of well-founded relations is well-founded induction. If we want to prove a property P for every element e of a set E endowed with a well-founded relation R, then we can suppose without loss of generality that P holds for every $f \in R^{-1}(e)$ in order to prove $P(e)$:

Lemma 3.8 (Well-founded induction)**.** *Let E be a set, R a well-founded relation on E and P a property on E. The following holds:*

$$\forall e \in E \left(\left(\forall f \in R^{-1}(e), P(f) \right) \Rightarrow P(e) \right) \Rightarrow \forall e \in E, P(e)$$

The second fundamental property of well-founded relations is that similarly to well-founded induction, if we want to define a function ψ on a set E endowed with a well-founded relation R, then we can make the definition of $\psi(e)$ depend on $\psi(f)$ for $f \in R^{-1}(e)$:

Lemma 3.9 (Well-founded recursion)**.** *Let R be a well-founded relation on a set E, X be a set and:*

$$\varphi : \left(\sum_{e \in E} X^{R^{-1}(e)} \right) \to X$$

then there exists a unique function $\psi : E \to X$ such that:

$$\forall e \in E, \psi(e) = \varphi \left(e, \psi_{|R^{-1}(e)} \right)$$

The function φ represents the definition of $\psi(e)$ which depends on e and on the restriction of ψ to $R^{-1}(e)$.

3.5.2 Recursion on natural numbers

The usual recursion on natural numbers can be described in terms of well-founded recursion. Indeed, if we define:

$$S = \{(n, n+1) \mid n \in \mathbb{N}\}$$

it is easy to see that S is well-founded. The well-founded recursion theorem states that in order to define a function from natural numbers to a set X, it is sufficient to provide a function:

$$\varphi : \left(\sum_{n \in \mathbb{N}} X^{S^{-1}(n)} \right) \to X$$

In the particular case of natural numbers, there are two possibilities: either $n = 0$, in which case $S^{-1}(n)$ is empty, or $n \neq 0$, in which case $S^{-1}(n)$ is the singleton set $\{n-1\}$. Therefore, we can provide instead of φ a value $x_0 \in X$ together with a function $\varphi_0 : \mathbb{N} \times X \to X$. φ can then be recovered by:

$$\varphi(n, f) = \begin{cases} x_0 \text{ if } n = 0 \\ \varphi_0(n, f(n-1)) \text{ if } n \neq 0 \end{cases}$$

If we look at it from the point of view of types, then the recursion theorem tells us that from an element a of type T and an element b of type $\iota \to T \to T$ we can build an element c of type $\iota \to T$ such that for any $\overline{\mathsf{n}}$:

$$c\,\overline{\mathsf{n}} = \begin{cases} a \text{ if } n = 0 \\ b\,\overline{\mathsf{n}}\,(c\,\overline{\mathsf{n}-1}) \text{ otherwise} \end{cases}$$

which are the exact equational definitions of $\mathsf{rec}\,a\,(\lambda x.b\,(\mathsf{succ}\,x))$. System T's recursor is then the computational interpretation of well-founded recursion on natural numbers.

3.5.3 Recursion on well-founded trees

There are several ways to define trees. Here we choose to define a tree as a prefix-closed set of words on a given alphabet:

Definition 3.6 (Tree)**.** *If A is a non-empty set (the alphabet), a tree on A is a non-empty set $T \subseteq A^*$ of finite words on A such that:*

$$u \in T \wedge v \sqsubseteq u \Rightarrow v \in T$$

where $\sqsubseteq$ denotes the prefix ordering.

In this section we will only be interested into a particular kind of trees: fully-branching trees. These are trees T on A such that:

$$\forall u \in T\,(\exists a \in A, ua \in T \Rightarrow \forall a \in A, ua \in T)$$

This means that if $u \in T$, then either u is a leaf, or every extension of u with a single element is in T.

We now explain how we can construct trees using functions of a special kind. Consider a function $f : A^{\mathbb{N}} \to X$, and suppose that this function is continuous:

$$\forall \alpha \in A^{\mathbb{N}}, \exists n \in \mathbb{N}, \forall \beta \in A^{\mathbb{N}}\,(\forall i < n, \beta_i = \alpha_i \Rightarrow f(\beta) = f(\alpha)) \tag{3.2}$$

This notion of continuity is obtained when we give $A^{\mathbb{N}}$ the product topology of the discrete topology on A, and X the discrete topology. This notion has a particularly intuitive interpretation: $f : A^{\mathbb{N}} \to X$ is continuous if and only if the result of $f(\alpha)$ depends only on a finite number of values of α.

Let define the function giving the modulus of continuity for f:

$$\begin{aligned} \mathrm{mod}_f : A^{\mathbb{N}} &\longrightarrow \mathbb{N} \\ \alpha &\longmapsto \min\left\{ n \in \mathbb{N} \,\middle|\, \forall \beta \in A^{\mathbb{N}}\,(\forall i < n, \beta_i = \alpha_i \Rightarrow f(\beta) = f(\alpha)) \right\} \end{aligned}$$

Then this function defines a fully-branching tree:

$$T_f = \left\{ \alpha_0\alpha_1...\alpha_i \,\middle|\, \alpha \in A^{\mathbb{N}} \wedge -1 \leq i < \mathrm{mod}_f(\alpha) \right\}$$

Easily, T_f is a tree. For the fully-branchingness of T_f remark that:

$$\forall \alpha, \beta \in A^{\mathbb{N}} \left(\forall i < \mathrm{mod}_f(\alpha), \beta_i = \alpha_i \Rightarrow \mathrm{mod}_f(\beta) = \mathrm{mod}_f(\alpha) \right)$$

Indeed, if $a_0...a_ib \in T_f$, then there is some $\beta \in A^{\mathbb{N}}$ such that $\beta_0...\beta_i\beta_{i+1} = a_0...a_ib$ and $\mathrm{mod}_f(\beta) > i+1$. Suppose now that some $a_0...a_ic \notin T_f$, then $\alpha = a_0...a_iccc...$ is such that $\alpha_0...\alpha_i\alpha_{i+1} = a_0...a_ic$ so it must verify $\mathrm{mod}_f(\alpha) \leq i+1$ (otherwise $a_0...a_ic \in T_f$). Since $\beta_0...\beta_i = a_0...a_i = \alpha_0...\alpha_i$ and $\mathrm{mod}_f(\alpha) \leq i+1$ then we have by the above remark $\mathrm{mod}_f(\alpha) = \mathrm{mod}_f(\beta) > i+1$, hence the contradiction.

If $f : A^{\mathbb{N}} \to X$ is continuous, then T_f also has the property that for any $\alpha \in A^{\mathbb{N}}$:

$$\alpha_0\alpha_1...\alpha_{\mathrm{mod}_f(\alpha)} \notin T_f$$

from the above remark. This means that there is no infinite branch. We will now connect this notion to that of well-founded relations.

We can define a binary relation P on A^* by:

$$P = \{(ux, u) \mid u \in A^*, x \in A\}$$

This relation is of course ill-founded on A^* since $A \neq \emptyset$ (consider $...\,P\,aa\,P\,a\,P\,\epsilon$ for any $a \in A$). A well-founded tree is a tree on which this relation is well-founded:

Definition 3.7 (Well-founded tree)**.** *If A is a set and $T \subseteq A^*$ is a tree on A, then T is well-founded if $P \cap T^2$ is a well-founded relation on T.*

Then using the axiom of dependent choice, a tree T is well-founded if and only if there is no $\alpha \in A^{\mathbb{N}}$ such that:

$$\forall n \in \mathbb{N}, \alpha_0\alpha_1...\alpha_n \in T$$

Therefore, if $f : A^{\mathbb{N}} \to X$ is continuous, then T_f is a well-founded fully-branching tree.

With this definition we now describe how to perform well-founded recursion on well-founded trees defined by continuous functions. Let $f : A^{\mathbb{N}} \to X$ be continuous. The recursion theorem states that in order to define a function from T_f to X, it is sufficient to provide a function:

$$\varphi : \left(\sum_{u \in T_f} X^{\left(P \cap T_f^2\right)^{-1}(u)} \right) \to X$$

Since T_f is fully-branching and for any $u \in T_f$:

$$\left(P \cap T_f^2\right)^{-1}(u) = P^{-1}(u) \cap T_f$$

this set is either empty (u is a leaf), or equal to $P^{-1}(u)$ (we will call it an internal node of T_f), in which case it is in one-to-one correspondence with A. Moreover, if u is a leaf, then u is uniquely described by any infinite sequence $\alpha \sqsupseteq u$, since in that case u can be recovered by taking the longest prefix of α which is in T_f. Therefore, if we provide:

$$\varphi_0 : A^* \times X^A \to X$$

we can then define:

$$\varphi(u, g) = \begin{cases} f(\alpha) \text{ for some } \alpha \sqsupseteq u & \text{if } u \text{ is a leaf} \\ \varphi_0(u, a \mapsto g(ua)) & \text{if } u \text{ is an internal node} \end{cases}$$

This definition is correct since if $u \in T_f$ is a leaf, then any $\alpha, \beta \in A^{\mathbb{N}}$ extending u are such that $f(\alpha) = f(\beta)$. By recursion we obtain a function $\psi : T_f \to X$ which is easily extended to A^* by putting for $u \notin T_f$: $\psi(u) = f(\alpha)$ for some $\alpha \in A^{\mathbb{N}}$ extending u. Again this is correct since by definition of T_f, for any infinite $\alpha, \beta \sqsupseteq u$ we have $f(\alpha) = f(\beta)$.

If we put all this together we have:

$$f : A^{\mathbb{N}} \to X \text{ continuous} \qquad \varphi_0 : A^* \times X^A \to X$$

and we build $\psi : A^* \to X$ such that for any $u \in A^*$:

$$\psi(u) = \begin{cases} \varphi_0(u, a \mapsto \psi(ua)) & \text{if } u \text{ is an internal node} \\ f(\alpha) \text{ for some } \alpha \sqsupseteq u & \text{otherwise} \end{cases}$$

The bar recursion operator builds ψ from φ_0 and f, as we will explain in the next section.

3.5.4 Bar recursion in μPCF

We now describe how we can encode bar recursion in μPCF. First we introduce some notations that will get their formal definition in the next paragraph. Suppose we have a type for lists of elements of type T: $T^\diamond$. The empty list is denoted $\epsilon : T^\diamond$, if u is an element of type $T^\diamond$ and a is an element of type T, then $u * a : T^\diamond$ denotes the appending of a at the end of u, and $u \mathbin{@} a : \iota \to T$ denotes the infinite sequence starting with u, and for which the subsequent values are all equal to a.

If F is an element of type $(\iota \to T) \to U$ which is a program, then if F gives an answer for some input α, the computation was finite and therefore F only looked at a finite part of α. Therefore, F is continuous in the sense of (3.2), and F defines a tree T_F as described above. If we have also an element Φ of type $T^\diamond \to (T \to U) \to U$ then recursion on T_F must provide an element Ψ of type $T^\diamond \to U$ such that for any element u of type $T^\diamond$:

$$\Psi\, u = \begin{cases} \Phi\, u\, (\lambda x.\Psi\, (u * x)) & \text{if } u \text{ is an internal node of } T_F \\ F\, (u \mathbin{@} \Omega) & \text{otherwise} \end{cases}$$

The question is: how do we know if u is an internal node? u being an internal node means that there exist two infinite extensions of u for which F's computation is different. Therefore, if u is an internal node then F looks at the Ω part of $u \mathbin{@} \Omega$. Conversely, if u is not an internal node, then all the computations of F on infinite extensions of u are the same, which means that F only looks at the u part of such extensions. Now we use the control possibilities of μPCF to capture the fact that F looks at the Ω part of $u \mathbin{@} \Omega$, and when it happens (if it happens) cancel the computation of F and launch the computation of some other a of type U. This is done by the following term:

$$\mu\alpha.\, [\alpha]\ F\, (u \mathbin{@} \mu\beta.\, [\alpha]\ a)$$

where α, β are two distinct fresh μ-variables. What happens during the computation of that term is that if F only looks at the u part of $u \mathbin{@} \mu\beta.\, [\alpha]\ a$, then the term is equal to $\mu\alpha.\, [\alpha]\ F\, (u \mathbin{@} \Omega) = F\, (u \mathbin{@} \Omega)$, but if F looks at the $\mu\beta.\, [\alpha]\ a$ part of $u \mathbin{@} \mu\beta.\, [\alpha]\ a$, then since β does not appear free in $[\alpha]\ a$, the computation of $\mu\beta.\, [\alpha]\ a$ will never return anything so the computation of F stops. However the result of the computation of a will be given back directly as the result of $\mu\alpha.\, [\alpha]\ F\, (u \mathbin{@} \mu\beta.\, [\alpha]\ a)$, so it becomes equal to a. Now, since we want $\Psi\, u$ to be $\Phi\, u\, (\lambda x.\Psi\, (u * x))$ when u is an internal node of T_F, we simply replace a with $\Phi\, u\, (\lambda x.\Psi\, (u * x))$ and obtain:

$$\Psi\, u = \mu\alpha.\, [\alpha]\ F\, (u \mathbin{@} \mu\beta.\, [\alpha]\ \Phi\, u\, (\lambda x.\Psi\, (u * x)))$$

then the two cases are handled transparently. Indeed, fix some $u : T^\diamond$. Either F does not look at the extension of u so we have:

$$\Psi\, u = \mu\alpha.\, [\alpha]\ F\,(u \mathbin{@} \Omega) = F\,(u \mathbin{@} \Omega)$$

which is correct since u is not an internal node, or F looks at the extension of u so the $\mu\beta$ stops the computation of F and we get:

$$\Psi\, u = \Phi\, u\,(\lambda x.\Psi\,(u * x))$$

which is also correct since u is an internal node.

What we just described is what we obtain when we apply the bar recursion operator to Φ and F. The bar recursion operator, is therefore a term:

$$\mathsf{barrec} : (T^\diamond \to (T \to U) \to U) \to ((\iota \to T) \to U) \to T^\diamond \to U$$

satisfying the equation:

$$\mathsf{barrec}\,\Phi\, F\, u = \mu\alpha.\, [\alpha]\ F\,(u \mathbin{@} \mu\beta.\, [\alpha]\ \Phi\, u\,(\lambda x.\mathsf{barrec}\,\Phi\, F\,(u * x)))$$

In the following we will explain formally how we define lists and the bar recursor in μPCF.

Encoding lists in μPCF

In order to define bar recursion, we first need to encode lists and operations on lists in μPCF. We choose to represent a list by a natural number (the size of the list) together with a (partial) function on natural numbers. Therefore we define the type of lists as:

$$T^\diamond \stackrel{\Delta}{=} \iota \times (\iota \to T)$$

As stated above, the first component of a list is its size. We introduce the following notation:

$$|M| \stackrel{\Delta}{=} \mathfrak{p}_1\, M$$

It is easy to derive $|M| : \iota$ from $M : T^\diamond$. We also define the empty list:

$$\epsilon \stackrel{\Delta}{=} \langle \overline{0}, \lambda x.\Omega \rangle$$

for which it is easy to derive $\epsilon : T^\diamond$ and $|\epsilon| = \overline{0}$, and we give a notation to access elements in a list:

$$M\,\wr N \wr \stackrel{\Delta}{=} \mathfrak{p}_2\, M\, N$$

We can derive from $M : T^\diamond$ and $N : \iota$ that $M\,\wr N \wr : T$ and $\epsilon\,\wr N \wr = \Omega$. In order to define extensions of lists, we need subtraction on natural numbers and tests of equality and strict ordering. Subtraction is defined by:

$$\mathsf{sub} \stackrel{\Delta}{=} \mathsf{Y}\,(\lambda zxy.\mathsf{if}_0\, y\, x\,(z\,(\mathsf{pred}\, x)\,(\mathsf{pred}\, y)))$$

for which we can derive $\mathsf{sub} : \iota \to \iota \to \iota$ and prove (by induction on n) that for any $n, m \in \mathbb{N}$:

$$\mathsf{sub}\,\overline{\mathsf{m}}\,\overline{\mathsf{n}} = \begin{cases} \overline{\mathsf{m} - \mathsf{n}} \text{ if } n \leq m \\ \overline{0} \text{ otherwise} \end{cases}$$

and test for equality and strict ordering of natural numbers are defined by:

$$\mathsf{if}_= \triangleq \lambda xyuv.\mathsf{if}_0\,(\mathsf{sub}\,x\,y)\,(\mathsf{if}_0\,(\mathsf{sub}\,y\,x)\,u\,v)\,v \qquad \mathsf{if}_< \triangleq \lambda xyuv.\mathsf{if}_0\,(\mathsf{sub}\,y\,x)\,v\,u$$

and we derive $\mathsf{if}_= : \iota \to \iota \to T \to T \to T$, $\mathsf{if}_< : \iota \to \iota \to T \to T \to T$ and:

$$\mathsf{if}_=\,\overline{\mathsf{m}}\,\overline{\mathsf{n}}\,M\,N = \begin{cases} M \text{ if } m = n \\ N \text{ otherwise} \end{cases} \qquad \mathsf{if}_<\,\overline{\mathsf{m}}\,\overline{\mathsf{n}}\,M\,N = \begin{cases} M \text{ if } m < n \\ N \text{ otherwise} \end{cases}$$

We are now able to define the extension of a list by a single element:

$$M * N \triangleq \langle \mathsf{succ}\,|M|\,, \lambda x.\mathsf{if}_=\,x\,|M|\,N\,(M\,\wr x \wr)\rangle$$

it is routine to derive from $M : T^\diamond$ and $N : T$ that $M * N : T^\diamond$, $|M * N| = \mathsf{succ}\,|M|$ and $(M * N)\,\wr |M| \wr = N$. Finally, we define infinite extension of a list with a fixed element:

$$M \mathbin{@} N \triangleq \lambda x.\mathsf{if}_<\,x\,|M|\,(M\,\wr x \wr)\,N$$

We can derive from $M : T^\diamond$ and $N : T$ that $M \mathbin{@} N : \iota \to T$ and:

$$(M_0 * M_1 * ... * M_{n-1} \mathbin{@} N)\,\wr \overline{\mathsf{m}} \wr = \begin{cases} M_m \text{ if } m < n \\ N \text{ otherwise} \end{cases}$$

The bar recursion operator

We have now all necessary material to define formally the bar recursion operator:

$$\mathsf{barrec} \triangleq \lambda uv.\mathsf{Y}\,(\lambda zy.\mu\alpha.\,[\alpha]\;v\,(y \mathbin{@} (\mu\beta.\,[\alpha]\;u\,y\,(\lambda x.z\,(y * x)))))$$

Bar recursor can be typed as expected:

$$\mathsf{barrec} : (T^\diamond \to (T \to U) \to U) \to ((\iota \to T) \to U) \to T^\diamond \to U$$

And it verifies indeed the equation:

$$\mathsf{barrec}\,M\,N\,P = \mu\alpha.\,[\alpha]\;N\,(P \mathbin{@} \mu\beta.\,[\alpha]\;M\,P\,(\lambda x.\mathsf{barrec}\,M\,N\,(P * x)))$$

Non-totality of bar recursion

We now explain how bar recursion can diverge when not restricted to the case of U being a base type. Fix $T = \iota$ and $U = \iota \to \iota$ so we have:

$$\mathsf{barrec} : (\iota^\diamond \to (\iota \to \iota \to \iota) \to \iota \to \iota) \to ((\iota \to \iota) \to \iota \to \iota) \to \iota^\diamond \to \iota \to \iota$$

Take also $M \triangleq \lambda xyz.y\,\overline{0}\,(\mathsf{succ}\;|x|)$ and $N = \lambda x.x$. We have for any $P : \iota^\diamond$:

$$\begin{aligned} \mathsf{barrec}\,M\,N\,P &= \mu\alpha.\,[\alpha]\;N\,(P \mathbin{@} \mu\beta.\,[\alpha]\;M\,P\,(\lambda x.\mathsf{barrec}\,M\,N\,(P * x))) \\ &= \mu\alpha.\,[\alpha]\;P \mathbin{@} \mu\beta.\,[\alpha]\;M\,P\,(\lambda x.\mathsf{barrec}\,M\,N\,(P * x)) \\ &= \mu\alpha.\,[\alpha]\;P \mathbin{@} \mu\beta.\,[\alpha]\,\left(\lambda z.\mathsf{barrec}\,M\,N\;\left(P * \overline{0}\right)\,(\mathsf{succ}\;|P|)\right) \end{aligned}$$

Then we have the following:

$$\mathsf{barrec}\,M\,N\,\epsilon\,\overline{0} = \left(\mu\alpha.\,[\alpha]\;\epsilon \mathbin{@} \mu\beta.\,[\alpha]\,\left(\lambda z.\mathsf{barrec}\,M\,N\;\left(\epsilon * \overline{0}\right)\,(\mathsf{succ}\;|\epsilon|)\right)\right)\,\overline{0}$$

$$
\begin{aligned}
&= \mu\alpha.\,[\alpha]\left(\epsilon \mathbin{@} \mu\beta.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0}\right)\left(\mathsf{succ}\ |\epsilon|\right)\right)\overline{0}\\
&= \mu\alpha.\,[\alpha]\,\mu\beta.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0}\right)\left(\mathsf{succ}\ |\epsilon|\right)\\
&= \mu\alpha.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0}\right)\left(\mathsf{succ}\,\overline{0}\right)\\
&= \mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0}\right)\overline{1}\\
&= \left(\mu\alpha.\,[\alpha]\ \epsilon * \overline{0} \mathbin{@} \mu\beta.\,[\alpha]\left(\lambda z.\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0}\right)\left(\mathsf{succ}\ \left|\epsilon * \overline{0}\right|\right)\right)\right)\overline{1}\\
&= \mu\alpha.\,[\alpha]\left(\epsilon * \overline{0} \mathbin{@} \mu\beta.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0}\right)\left(\mathsf{succ}\ \left|\epsilon * \overline{0}\right|\right)\right)\overline{1}\\
&= \mu\alpha.\,[\alpha]\,\mu\beta.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0}\right)\left(\mathsf{succ}\ \left|\epsilon * \overline{0}\right|\right)\\
&= \mu\alpha.\,[\alpha]\,\mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0}\right)\left(\mathsf{succ}\,\overline{1}\right)\\
&= \mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0}\right)\overline{2}\\
&\quad\vdots\\
&= \mathsf{barrec}\,M\,N\,\left(\epsilon * \overline{0} * \overline{0} * \overline{0}\right)\overline{3}\\
&\quad\vdots
\end{aligned}
$$

This non-total behavior comes from the fact that the identity function on sequences of natural numbers is obviously non continuous when we take the product topology on the domain and the discrete topology on the codomain (each sequence, as a singleton is an open set). Since the identity function on sequences is definable in μPCF, this example is reproducible in any model of μPCF. However, if we restrict to the case where U is built only from ι, 0 and product, then in many models every element of type $(\iota \to T) \to U$ is continuous when taking on $\iota \to T$ the product topology of the discrete topology of T, and on U the discrete topology. The model of game semantics that we describe in the next chapter gives such an example since it has a CPO structure. An obvious example of a model where this continuity requirement does not hold is the full set-theoretic model (take for example the function from sequences of natural numbers to natural numbers which maps the constant zero sequence to 0 and all other sequences to 1).

The definition of the realizer of $(\boldsymbol{DC}^r)$ based on barrec will be given in section 5.2.3.

Chapter 4

Game semantics

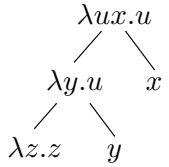
We first give in this chapter an intuitive view on the model of Hyland-Ong-Nickau games. We then define it formally and prove that it is a category of continuations. Finally, we describe how μPCF (and therefore Peano arithmetic and the axiom of choice) can be interpreted in this model.

4.1 Introduction

In this section, we give an informal presentation of game semantics, introducing it through its correspondence with certain normal forms of simply-typed λ-calculus. Another presentation of this correspondence can be found in [DHR96], together with the connection with linear head reduction. At first, we will only consider simply-typed lambda-calculus with one base type, i.e. no primitive datatypes and no fixed-point operator. The games model relies heavily on the notion of Böhm tree of a program, a Böhm tree being a representation of a normal form of a program. Let us see an example. Take the following λ-term in normal form:

$$M = \lambda ux.u\,(\lambda y.u\,(\lambda z.z)\,y)\,x \tag{4.1}$$

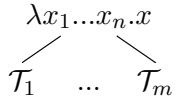
This term will be the running example throughout this section. The Böhm tree of M is the following:



A simple way to look at Böhm trees is to see them as syntactic trees corresponding to the following grammar:

$$N \quad ::= \quad \lambda y_1 ... y_n.x\,N_1 ... N_m$$

which is a grammar generating the β-normal forms of λ-calculus. The Böhm tree corresponding to some $N = \lambda x_1 ... x_n.x\,N_1 ... N_m$ is then:

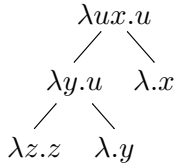


where $\mathcal{T}_i$ is the Böhm tree of t_i. Since Böhm trees are very tightly related to this syntax, we will write a λ even for an empty λ-abstraction to make explicit that it is indeed generated by

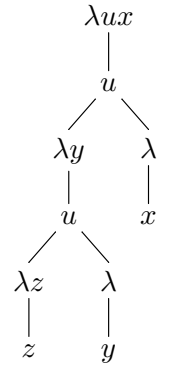
the grammar. For example, M will be written:

$$\lambda ux.u\,(\lambda y.u\,(\lambda z.z)\,(\lambda.y))\,(\lambda.x)$$

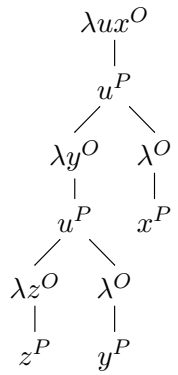
and the corresponding Böm tree:



This cumbersome notation makes explicit the fact that in a Böhm tree, nodes can be decomposed in two parts: the λ part consisting of a (possibly empty) list of variables, and the head part, consisting of a single variable. For example, the root $\lambda ux.u$ can be decomposed as the list u, x and the variable u, and the rightmost leaf can be decomposed as the empty list and the variable x. Using this observation, we can consider a slight modification of Böhm trees where these two parts are distinguished, the single variable part becoming the unique son of the corresponding λ part. We obtain the following tree for M:

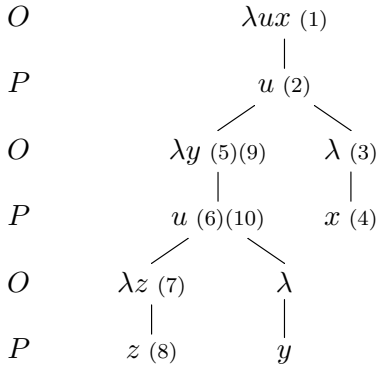


The nodes of the tree are now of two distinct types that alternate along each branch. The λ nodes will be called O-moves (for opponent), while the variable nodes will be called P-moves. Because of the definition of our Böhm trees, these polarities alternate along each branch. Here is the Böhm tree of M where the moves are annotated by their polarities:



We now describe what is a walk in that tree. As stated informally in the previous section, a walk is an exploration of the tree by an opponent, player describing it step-by-step. In that sense it will be a sequence of moves such that the move following any O-move is its son (which is a P-move), and conversely the move preceding any P-move is its father (which is an O-move). Moreover, opponent must start the exploration with the root, and it can play an O-move only

if all the moves higher in the branch have already been played. Here is an example of a walk through the tree of M, together with the polarities of moves:



O	P	O	P	O	P	O	P	O	P
λux	u	λ	x	λy	u	λz	z	λy	u

The next thing is: the player represents the program, so he knows the tree, but how can the opponent play moves in the tree without knowing it? This is where arenas come into play. Since we work in a simply-typed setting with one base type, we can be more precise. Let ι be the base type. The types (ranged over by T, U) are either ι, or some $T \to U$. For the sake of conciseness of notations, we will simply write $T\,U$ for $T \to U$. A possible grammar to describe the types is then the following:

$$T \quad ::= \quad T_1 ... T_n\, \iota$$

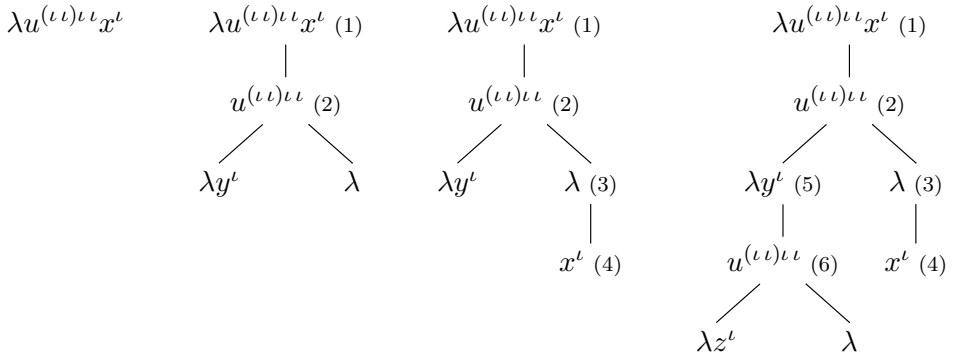
On the side of terms, the following grammar generates what are called the long $\beta\eta$-normal forms:

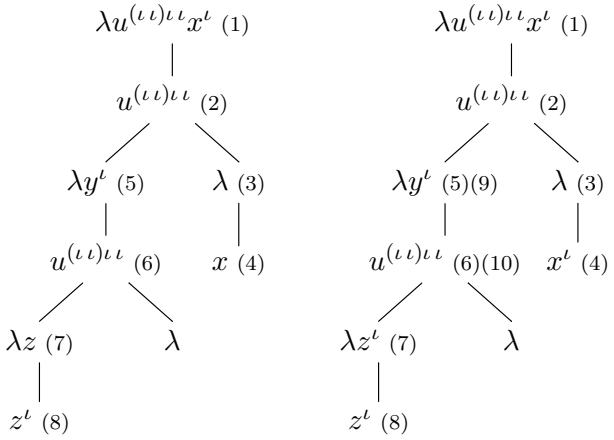
$$N^{T_1 ... T_n\, \iota} \quad ::= \quad \lambda x_1^{T_1} ... x_n^{T_n}.x^{U_1 ... U_m\, \iota} N_1^{U_1} ... N_m^{U_m}$$

Remark that the grammar of types has not been chosen randomly, but fits quite precisely that of long $\beta\eta$-normal forms. Let us write M again, but this time annotated with types:

$$M^{((\iota\,\iota)\iota)\iota\,\iota} = \lambda u^{(\iota\,\iota)\iota\,\iota} x^{\iota}.u^{(\iota\,\iota)\iota\,\iota} \left(\lambda y^{\iota}.u^{(\iota\,\iota)\iota\,\iota}\,(\lambda z^{\iota}.z^{\iota})\,y^{\iota}\right) x^{\iota}$$

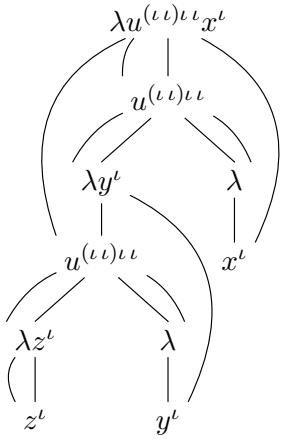
Now we explain how O, who does not know the tree, is able to play moves in it. Since the type of the program is known to both O and P, O knows that if the type of the program is $T_1 ... T_n\, \iota$, then the root has to be $\lambda x_1^{T_1} ... x_n^{T_n}$. Later on, when P plays some variable x of type $(U_{1,1} ... U_{1,k_1}\, \iota)\, ...\, (U_{m,1} ... U_{m,k_m}\, \iota)\, \iota$ then this variable must be part of a λ-move played earlier by O, who therefore knows its type, and knows that its sons are the $\lambda y_{i,1}^{U_{i,1}} ... y_{i,k_i}^{U_{i,k_i}}$ for $1 \leq i \leq m$. The Böhm tree is constructed iteratively starting from an empty tree, and at each point O knows the sons of each P-move which has already been played. We illustrate this with the example walk given above: the part of the tree that O consecutively "sees" is:



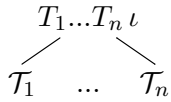


For example, when P plays the 6th move $u^{(\iota\,\iota)\,\iota\,\iota}$, O knows that u is of type $(\iota\,\iota)\,\iota\,\iota$ because he provided it in the 1st move. Therefore, O knows that the two arguments of u are of type $\iota\,\iota$ and ι, so the two sons of u are λz^{ι} and λ.

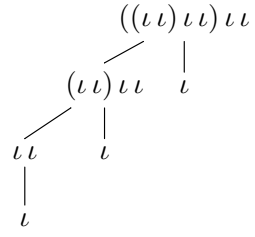
These considerations make appear a new type-directed structure on Böhm trees. Indeed, each O-move $\lambda x_1^{T_1}...x_n^{T_n}$ enables the P-moves $x_i^{T_i}$ for $1 \leq i \leq n$, and conversely each P-move $x^{(U_{1,1}...U_{1,k_1}\,\iota)...(U_{m,1}...U_{m,k_m}\,\iota)\,\iota}$ enables the O-moves $\lambda y_{i,1}^{U_{i,1}}...y_{i,k_i}^{U_{i,k_i}}$ for $1 \leq i \leq m$. Let make this enabling relation explicit in the Böhm tree of M:



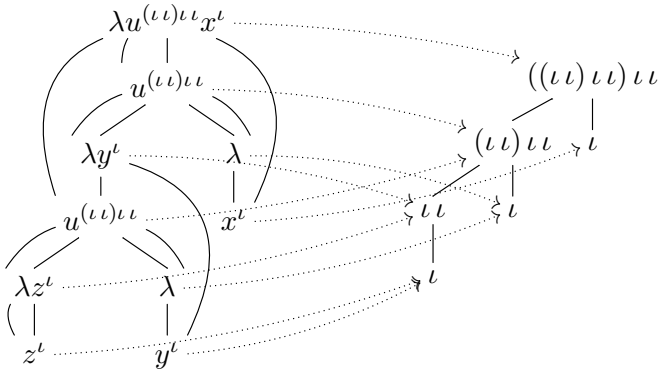
We see that because of the structure of the tree, the P-moves may be enabled by O-moves higher in the branch, but the O-moves must be enabled by their direct father. As stated above, this enabling relation is type-directed. Just like Böhm trees are the syntactic trees of programs, arenas are the syntactic trees of types, for the grammar given above: if $T = T_1...T_n\,\iota$, then the associated arena is:



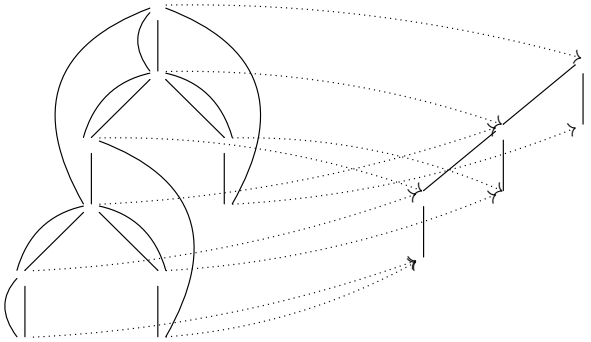
where $\mathcal{T}_i$ is the syntactic tree of T_i. For example, the arena for the type $((\iota\,\iota)\,\iota\,\iota)\,\iota\,\iota$ of M is the following:



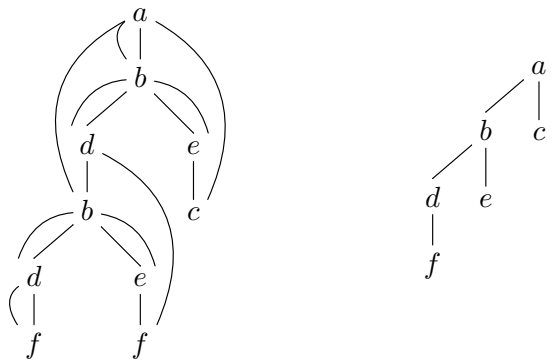
Let put side-by-side the Böhm tree of M and the arena of its type, and draw the correspondence:



The curved full lines on the Böhm tree are mapped through the dashed arrows to the straight full lines on the arena. This makes clear that the enabling relation is type-directed. In fact we revealed enough structure now to forget completely about the labels:



We can retrieve the full Böhm tree (modulo α-conversion) from the above structure. In order to clarify the picture, we define arbitrary labels for the nodes of the arena, and use them to describe the dashed arrows:

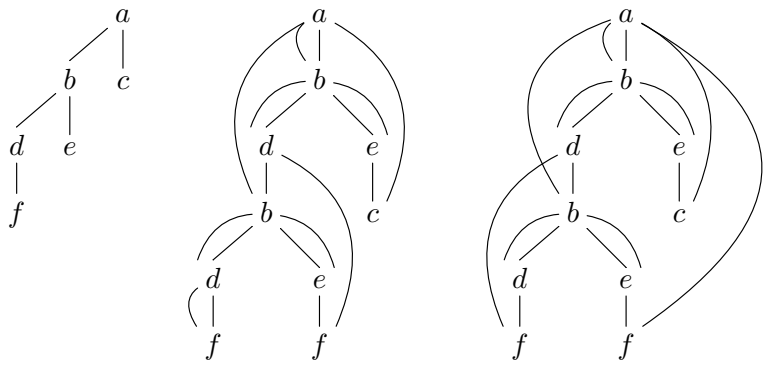


With this representation we get a model of simply typed λ-calculus which is sound with respect to $\beta\eta$ equivalence.

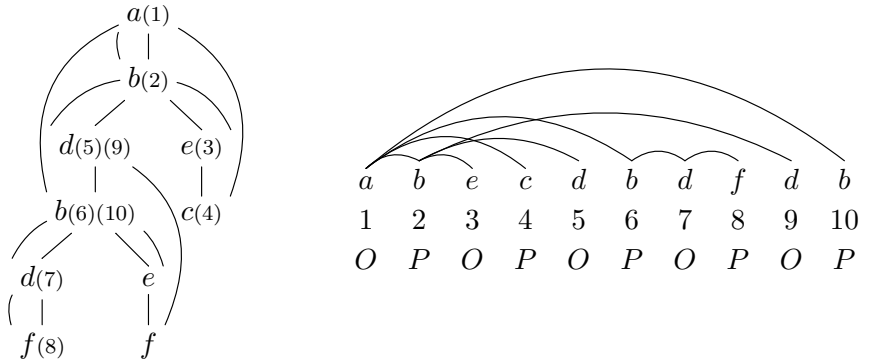
But what about composition? Indeed, given two Böhm trees together with their arenas with compatible types, how do we compose them? Let us go back to the notion of walk presented earlier. In order to compose Böhm trees, it is easier to manipulate walks in these trees rather than the trees directly. A walk in a Böhm tree is called a play. It can be seen as an exploration of the tree by an opponent, the tree being given step-by-step by the player. The idea is to switch

from the concept of Böhm tree to the concept of strategy, a strategy being the set of plays on this Böhm tree.

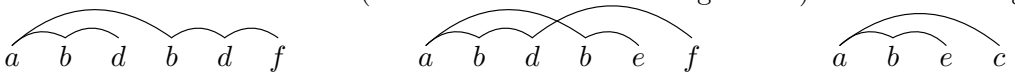
Let first look at how we can recover the tree from the set of plays on that tree. The idea is that a branch of the tree is a particular play, and if we know every branch of the tree, then we can recover the full tree. The enabling structure of the tree is essential to recover the full tree. Take for example the terms $M = \lambda ux.u\,(\lambda y.u\,(\lambda z.z)\,y)\,x$ and $\lambda ux.u\,(\lambda y.u\,(\lambda z.y)\,x)\,x$ of same type $((\iota\,\iota)\,\iota\,\iota)\,\iota\,\iota$, which are variants of the Kierstead terms (the only reason for not choosing exactly Kierstead terms is that it allows us to keep the same running example). Here are their common arena and their Böhm trees:



As we can see, the trees have the same shape, but a different enabling structure. Therefore, plays must not be only words of labels of the arena, but words of labels of the arena with pointers. The example play that was given earlier is then:



Now we can recover the full tree from the plays on it, by looking at the branches. For example, the branches of the Böhm tree of M (from the leftmost to the rightmost) are the following plays:



We now look at the other direction: given an arena, is any set of plays the description of a Böhm tree? Obviously the answer is no. Let fix a set σ of plays on an arena and let formulate some necessary conditions for it to be the set of plays on a Böhm tree $\mathcal{T}$ on an arena $\mathcal{A}$.

First, since the empty play (denoted ϵ) is a play on any Böhm tree, we must have $\epsilon \in \sigma$.

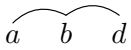
Next, since plays of σ are explorations of $\mathcal{T}$, a prefix of a play of σ is itself an exploration of $\mathcal{T}$, and therefore it must be in σ. This condition is called prefix closure of strategies. Using this condition, we have $\epsilon \in \sigma \Leftrightarrow \sigma \neq \emptyset$.

The next restriction that comes to mind is the fact that for any O-move being played, there is a unique P-move that follows it. Indeed, in $\mathcal{T}$, each O-move has a unique son (which is a

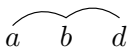
P-move). To make it more formal, let w be a play of σ, that is, w is a word of nodes of $\mathcal{A}$ together with pointers. If w ends with an O-move, then there exists a unique P-move a and a unique O-move b of w such that $wa \in \sigma$ (where in wa the pointer of a goes to b). The existence corresponds to the notion of totality, and the uniqueness to that of determinism.

In the example play, *abecd* (with the pointers defined above) is a play on the Böhm tree (by prefix closure, since it is a prefix of the example play), and in the Böhm tree, d has b (pointing to a) as unique son. Therefore, *abecdb* (with the pointers) is a play on the Böhm tree.

Still, we did not put enough restrictions on the sets of plays for them to describe Böhm trees. Indeed, the moves of P must depend only on the current branch of the tree that is being explored, while a play may go into one branch, and then to another (as it is the case with the example play, after the 4th move, O chooses to go to the left branch by playing the 5th move). In order to make this idea more formal, we describe the notion of view of a play, which is a way to compute the current branch. Take the example play. The current branch after the 10th move has been played is:



as can be seen on the Böhm tree. Now we give an algorithm to compute it. Remember that a P move that comes after an O-move must be its child in the Böhm tree, and an O-move is always the child (in the Böhm tree) of the P-move it points to. Therefore, to compute the current branch, we do the following. Take the last move, here it is b. It is a P-move, so the O-move that comes just before is its father in the Böhm tree, we take it, we have $d_9 b_{10}$. Now we are at the 9th move d. It is an O-move, so it points to its father in the Böhm tree. Therefore, we follow the pointer from d and get to the 2nd move b. We take it to obtain $b_2 d_9 b_{10}$. Now, b is a P-move, so the O-move that comes just before is its father in the Böhm tree, we take it, we have $a_1 b_2 d_9 b_{10}$. Finally, a is a root so we stop here. By taking back the pointers coming from the play we get the branch:



What we just described is the computation of a view from a play. If w is a play, then its view will be denoted by $\ulcorner w \urcorner$. Then the condition of P seeing only the current branch is the following: if w is a play of σ ending with an O-move and if a is the unique P-move such that $\ulcorner w \urcorner a \in \sigma$, then a is also the unique P-move such that $wa \in \sigma$. The pointers must also be dealt with, but it is straightforward (even though technical). This condition is what game semanticists call innocence.

The last condition corresponds to the fact that Böhm trees, which are finitely branching because of the simply-typed structure, are also of finite depth. We saw in the preceding paragraph that branches of $\mathcal{T}$ correspond to views of σ. Therefore, we ask for the views of σ to be finite, or equivalently (since $\mathcal{T}$ is finitely branching) that σ contains only a finite number of views. This condition is called finiteness.

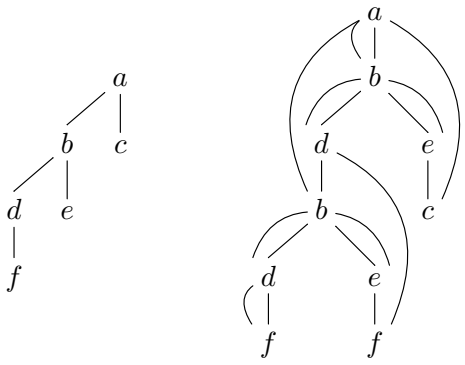
Given an arena, a set of plays which is non empty, prefix-closed, total, deterministic, innocent and finite is called a strategy. Finally we put all the necessary conditions to get the following result:

Theorem. *The set of strategies on finite arenas is in one-to-one bijection with the set of simply typed λ-terms with one base type quotiented by $\beta\eta$ equivalence.*

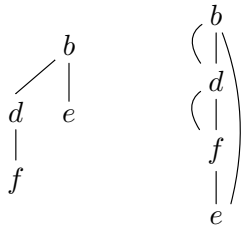
Still, we do not know how to compose strategies. But the definition of strategies as sets of plays makes the definition of composition very easy. If σ is a strategy on the arena associated to $T\,U$ and if τ is a strategy on the arena associated to T, then we can look at the plays $w \in \sigma$ such that when we erase every move of w which is in the arena of U we get a play of τ. Then for each such w, we erase all the moves that are in the arena of T. We obtain a set of plays on

U that we call $\sigma\,(\tau)$. It is very technical to prove that $\sigma\,(\tau)$ is a strategy on U but it is true, and the good thing is that it corresponds to usual λ-calculus composition.

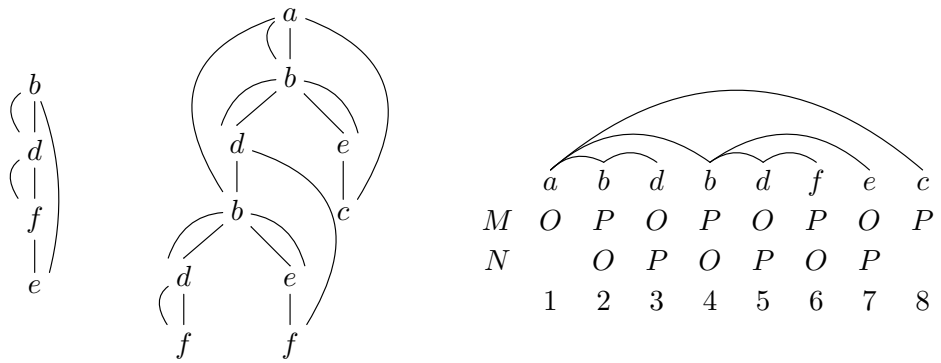
Let now illustrate this. $M = \lambda ux.u\,(\lambda y.u\,(\lambda z.z)\,y)\,x$ is a term of type $((\iota\,\iota)\,\iota\,\iota)\,\iota\,\iota$. Its arena $\mathcal{A}$ and Böhm tree $\mathcal{T}$ are:



$N = \lambda vx.v\,x$ is a term of type $(\iota\,\iota)\,\iota\,\iota$. Its arena $\mathcal{B}$ and Böhm tree $\mathcal{U}$ are:



Of course, we chose the label on $\mathcal{B}$ to match the ones on $\mathcal{A}$ during composition. We describe here a play (in fact the only maximal one) of $\mathcal{T}$ such that by keeping only the moves of $\mathcal{B}$ we have a play of $\mathcal{U}$:

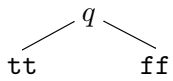


If we keep only the moves of this play which are in $\mathcal{A}$ and not in $\mathcal{B}$, we get the play $\overset{\frown}{a\ \ c}$, and we obtain the following Böhm tree (with its arena):

which is the Böhm tree of $\lambda x.x$ of type $\iota\,\iota$, and we have indeed $M\,N =_{\beta\eta} \lambda x.x$. We describe now how we computed this play, step-by-step. We refer to the moves of this play as m_i, where m is the label and i the index. We will not recall the pointers, since they are given above. The strategy interpreting M will be called σ, and the one for N τ. First, $\mathcal{A}$ has only one root which must be the first move of the play. Therefore, the play starts with a_1. The second move is given by M: b is the only P-move such that $a_1 b_2 \in \sigma$. Then if we keep only the moves that are in

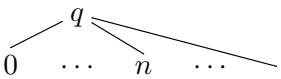
$\mathcal{B}$, we have the one-move play b_2. The next move is then given by N: d is the only P-move such that $b_2d_3 \in \tau$. Now, the play $a_1b_2d_3$ ends with an O-move for σ, and the only solution to complete it with a P-move while being still in σ is to add b_4, so we get $a_1b_2d_3b_4$. At that point, it is N's turn. If we keep only the moves which are in $\mathcal{B}$ we obtain $b_2d_3b_4$. Computing the view we have $\ulcorner b_2d_3b_4 \urcorner = b_4$, which can only be completed by τ with d_5, so by innocence of τ we get $b_2d_3b_4d_5 \in \tau$. The current play is then $a_1b_2d_3b_4d_5$ and it is σ's turn, and it completes the play with f_6 to get $a_1b_2d_3b_4d_5f_6$ which is the complete leftmost branch of $\mathcal{T}$. It is now τ's turn. Keeping only the moves in $\mathcal{B}$ we are at $b_2d_3b_4d_5f_6$. We compute again the view to get $\ulcorner b_2d_3b_4d_5f_6 \urcorner = b_4d_5f_6$ so the play is completed with e_7 into the (unique) full branch of $\mathcal{U}$. The current play is then $a_1b_2d_3b_4d_5f_6e_7$ and it is σ's turn. We have $\ulcorner a_1b_2d_3b_4d_5f_6e_7 \urcorner = a_1b_2d_3b_4e_7$, which is completed with c_8. We then get the full play $a_1b_2d_3b_4d_5f_6e_7c_8$, and we cannot go further since c is a leaf of the result arena. The play we obtain is in σ and if we keep only the moves in $\mathcal{B}$ we get $b_2d_3b_4d_5f_6e_7 \in \tau$. Finally, keeping only the moves in $\mathcal{A}$ and not in $\mathcal{B}$ we get the play a_1c_8 which is a play of the composition. Since the result arena is very simple, we obtain the only strategy on this arena, and this strategy has only one maximal play (its Böhm tree has only one branch of length 2), but the principle lifts to higher order very well.

We described the case of simply-typed λ-calculus with one base type, that base type being interpreted as the one-move arena. If we want for example to have a primitive datatype for booleans, we define the associated arena to be:



and we define question and answers among moves. In the arena for booleans q is a question and `tt` and `ff` are answers. We must then restrict the strategies to be well-bracketed. This means that a player can answer a question only if all other questions after it have already been answered and the question it answers has not been answered already.

Next, if we want a primitive datatype for natural numbers, then we have to switch to infinite arenas. The arena of natural numbers is then:



The finiteness condition is no more equivalent to having a finite number of views: the Böhm trees of λ-terms may now be infinitely branching. However, every branch is still finite. One would be tempted then to change the finiteness condition for a condition saying that every "maximal" view is finite. However, a non-computable function on natural numbers also satisfies this condition, so we would not get the definability result. The solution is then to distinguish the set of finite strategies (the ones which have a finite number of views) as a subset of all strategies. Then any finite well-bracketed strategy is the interpretation of some term (modulo an extension of the language with a definition-by-case construct), and every term can be interpreted as a (possibly non finite) well-bracketed strategy.

Finally, when we add a fixpoint operator (and therefore model the language PCF), we have to first consider call-by-name operational semantics, and we have to drop the finiteness and totality conditions. Indeed, since fixpoint operators induce non-termination, their Böhm trees may be of infinite depth (indeed, the Böhm tree of the Y fixpoint operator of PCF has only one branch, and it is infinite). The totality disappears also because it is not preserved under composition when Böhm trees are allowed to have infinite branches. For example, the Böhm tree of $Y\,(\lambda x.x)$ consists of only one root without any son, and so the corresponding strategy cannot be total.

As stated before, we are interested here into getting a model for classical proofs, so in the light of the Curry-Howard isomorphism we want to have control operators in the model. Because of that we drop the well-bracketing condition. We also want to be able to express imperative behaviors, so we drop the innocence constraint. However, since we still want a model of λ-calculus, we need to have products. Therefore, we drop innocence but still require single-threadedness of the strategies.

4.2 Arenas and strategies

4.2.1 Arenas and justified sequences

Here we define arenas, which are forests of moves. Arenas are the objects of the category of HON games.

A forest is a partial order $(E, \leq)$ such that $\forall x \in E, \{y \in E \,|\, y < x\}$ is well-ordered. In particular, if $x \leq z$ and $y \leq z$ then $x \leq y$ or $y \leq x$: compatible elements are comparable. The binary relation $<_1$ on E is then defined as:

$$\forall x, y \in E \qquad x <_1 y \quad \iff \quad x < y \ \wedge \ \forall z \, (x < z < y \Rightarrow z = x \vee z = y)$$

$x <_1 y$ means that y is a direct child of x. The roots of a forest correspond to the minimal elements for $\leq$.

Definition 4.1 (Arena)**.** *An* arena *is a countably branching, finite-depth forest. Its nodes are called* moves*. Each move is given a polarity O (for Opponent) or P (for Player or Proponent):*

- *The roots are of polarity O*
- *If $a <_1 b$ then a and b are of opposite polarities*

A root of an arena is also called an initial move. We will often identify an arena with its set of moves. Here is an example of arena, the polarities of the moves being given on the left:

$$\begin{array}{l} O \\ \\ P \\ \\ O \end{array} \qquad \begin{array}{ccccc} & & a & & f \\ & & | & & | \\ & b & & g & \\ & | & c & & h \\ d & & & & \\ | & & & & \\ e & & & & \end{array} \tag{4.2}$$

If $\mathcal{A}$ is an arena and X is a subset of $\mathcal{A}$, then X inherits the ordered structure of $\mathcal{A}$ and it is still an arena (but the polarities of moves may be different than the ones in $\mathcal{A}$). For example, $X = \{a; c; d; e; g; h\}$ is a set of moves of the arena (4.2) and the associated arena is:

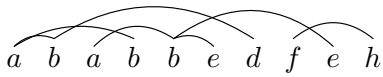
$$\begin{array}{l} O \\ \\ P \end{array} \qquad \begin{array}{ccccc} & & a & g & h \\ & & | & & \\ d & e & c & & \end{array} \tag{4.3}$$

Definition 4.2 (Justified sequence)**.** *Given an arena $\mathcal{A}$, we define a* justified sequence *on $\mathcal{A}$ to be a word s (finite or infinite) of $\mathcal{A}$ together with a partial* justifying function *$f : |s| \rightharpoonup |s|$ such that:*

- *If $f(i)$ is undefined, then s_i is an initial move*
- *If $f(i)$ is defined, then $f(i) < i$ and $s_{f(i)} <_1 s_i$*

We denote the set of justified sequences on $\mathcal{A}$ by $\mathcal{S}_{\mathcal{A}}$.

We denote the empty justified sequence by ϵ. Remark here that by definition of the polarity, if $f(i)$ is undefined (s_i is initial), then s_i is of polarity O, and if $f(i)$ is defined, then s_i and $s_{f(i)}$ are of opposite polarities. Also, $f(0)$ is never defined, and so s_0 is always an initial O-move. A justified sequence on the arena (4.2) is represented for example as:



(4.4)

Definition 4.3 (Restriction of a justified sequence)**.** *If $\mathcal{A}$ is an arena, X is a subset of $\mathcal{A}$ (and therefore an arena) and s is a justified sequence on $\mathcal{A}$ with justifying function f, then we define $s_{|X}$ to be the subword $s_{i_0}...s_{i_m}$ of s consisting of the moves of s which are in X, together with the justifying function g defined by:*

$$g(j) = \begin{cases} \text{the } k \text{ s.t. } i_k = f^p(i_j) \text{ where } p = \min\left\{q > 0 \,\middle|\, s_{f^q(i_j)} \in X\right\} \text{ if } \exists q > 0 \; s_{f^q(i_j)} \in X \\ \text{undefined otherwise} \end{cases}$$

For example taking X as in (4.3) and s to be the sequence (4.4), $s_{|X}$ is the sequence:

Since $b \notin X$ the pointers from the moves d and e have been updated to point to the corresponding a, and since $f \notin X$ the pointer from h has been erased. We have to ensure that a restricted justified sequence is still a justified sequence:

Lemma 4.1. *Let $\mathcal{A}$ be an arena. For any justified sequence s on $\mathcal{A}$ and any subset X of $\mathcal{A}$, $s_{|X}$ is a justified sequence on the arena X.*

Proof. Let us write $s = s_0...s_n$ with justifying function f and $s_{|X} = s_{i_0}...s_{i_m}$ with justifying function g. First we have to check that if $g(j)$ is undefined, then s_{i_j} is a root of X. If $g(j)$ is undefined then by definition of a restriction we have:

$$\left\{s_{f^q(i_j)} \,\middle|\, q > 0\right\} \cap X = \emptyset$$

but on the other hand since for any i, $s_{f(i)} <_1 s_i$ and $<$ restricts to a well-order on the set of $a \in \mathcal{A}$ such that $a < s_{i_j}$ we get:

$$\left\{s_{f^q(i_j)} \,\middle|\, q > 0\right\} = \left\{a \in \mathcal{A} \,\middle|\, a < s_{i_j}\right\}$$

Putting this together we get:

$$\left\{a \in X \,\middle|\, a < s_{i_j}\right\} = \emptyset$$

and therefore s_{i_j} is a root of X. Next we have to check that if $g(j)$ is defined, then $s_{i_{g(j)}} <_1 s_{i_j}$ on X. If $g(j)$ is defined then by definition of a restriction we have $i_{g(j)} = f^p(i_j)$ where $p > 0$ is the least such that $s_{f^p(i_j)} \in X$, so:

$$\left\{s_{f(i_j)}; ...; s_{f^{p-1}(i_j)}\right\} \cap X = \emptyset$$

but on the other hand since for any i, $s_{f(i)} <_1 s_i$ and $<$ restricts to a total order on the set of $a \in \mathcal{A}$ such that $s_{f^p(i_j)} < a < s_{i_j}$ we get:

$$\left\{s_{f(i_j)}; ...; s_{f^{p-1}(i_j)}\right\} = \left\{a \in \mathcal{A} \,\middle|\, s_{f^p(i_j)} < a < s_{i_j}\right\}$$

Putting this together we get:

$$\left\{a \in X \,\middle|\, s_{f^p(i_j)} < a < s_{i_j}\right\} = \emptyset$$

and therefore $s_{i_{g(j)}} = s_{f^p(i_j)} <_1 s_{i_j}$ on X. □

In a sequence s with justifying function f, a move s_j is hereditarily justified by a move s_i if s_i is initial and for some n, $f^n(j) = i$.

Definition 4.4 (Thread)**.** *If s is a justified sequence on $\mathcal{A}$ and if s_i is initial, then the* thread associated *to s_i is the sequence consisting of the moves of s hereditarily justified by s_i, the justifying function being defined accordingly. The* set of threads *of s,* $\mathrm{Threads}(s)$*, is the set of threads associated to the initial moves of s. By extension a justified sequence s will be called a thread if it contains exactly one thread (i.e.* $\mathrm{Threads}(s) = \{s\}$*). In particular,* $\mathrm{Threads}(\epsilon) = \emptyset$ *and so ϵ is not a thread.*

The justifying function of a thread can indeed be defined since if a points to b in s, then a and b are hereditarily justified by the same initial move. For example we have:

$$\mathrm{Threads}\Big(\overbrace{a\ b\ c\ d\ e\ f\ g\ h\ i\ j}\Big) = \Big\{\overbrace{a\ b\ d\ g}\,;\,\overbrace{c\ e\ f\ i}\,;\,\overbrace{h\ j}\Big\}$$

A P-sequence (resp. O-sequence) is a sequence ending with a P-move (resp. an O-move). Write $t \sqsubseteq s$ if t is a prefix of s, i.e. t is a prefix of s as a word and their justifying functions coincide (this is a particular case of subsequence). Write $t \sqsubseteq_P s$ (resp. $t \sqsubseteq_O s$) if t is a P-prefix (resp. O-prefix) of s, i.e. $t \sqsubseteq s$ and t is a P-sequence (resp. O-sequence).

4.2.2 Plays and strategies

Definition 4.5 (Play)**.** *A* play *s on $\mathcal{A}$ is an alternating justified sequence of $\mathcal{A}$, i.e., for any i, s_{2i} is an O-move and s_{2i+1} is a P-move. We denote the set of plays on $\mathcal{A}$ by $\mathcal{P}_{\mathcal{A}}$.*

A play on an arena is the trace of an interaction between a program and a context, each one performing an action alternatively. A P-play (resp. O-play) is a play which is a P-sequence (resp. O-sequence). If $s \in \mathcal{P}_{\mathcal{A}}$ such that $\mathrm{Threads}(s) \subseteq \mathcal{P}_{\mathcal{A}}$, then it is easy to see that in s, any sequence of an O-move followed by a P-move must be in the same thread.

Definition 4.6 (Strategy)**.** *A* strategy *σ on $\mathcal{A}$ is a P-prefix-closed set of finite P-plays on $\mathcal{A}$ such that:*

- *σ is deterministic: if sa and sb are in σ, then $a = b$.*
- *σ is single-threaded: for any finite P-play s, $s \in \sigma \Leftrightarrow \mathrm{Threads}(s) \subseteq \sigma$.*

Our notion of single-threadedness matches the usual notion of thread-independence (see e.g. [AHM98]). Remark also that a strategy always contains the empty play ϵ since $\mathrm{Threads}(\epsilon) = \emptyset$. Finally, using the single-threadedness of strategies, it suffices to give the threads of a strategy in order to fully determine it.

4.3 The category $\mathcal{G}$ of arenas and strategies

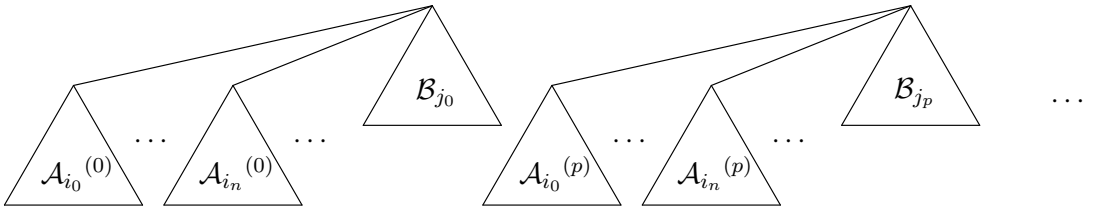
4.3.1 The category $\mathcal{G}$

Objects and morphisms

The constructions we use will sometimes contain multiple copies of the same arena (for example $\mathcal{A} \to \mathcal{A}$), so we distinguish the instances with superscripts (for example $\mathcal{A}^{(1)} \to \mathcal{A}^{(2)}$).

The objects of $\mathcal{G}$ are arenas. In order to define morphisms between objects $\mathcal{A}$ and $\mathcal{B}$, we first need to define the arena $\mathcal{A} \to \mathcal{B}$. If $\mathcal{A}$ and $\mathcal{B}$ are arenas consisting of the trees $(\mathcal{A}_i)_{i \in I}$ and

$(\mathcal{B}_j)_{j\in J}$ and if $i_0, ..., i_n, ...$ is an enumeration of I and $j_0, ..., j_p, ...$ of J, then the arena $\mathcal{A} \to \mathcal{B}$ is the following forest:



This arena contains J copies of $\mathcal{A}$, and each root of the j_pth copy becomes a direct son of the root of $\mathcal{B}_{j_p}$. This definition is correct since the arena we obtain is still countably branching. We define morphisms from $\mathcal{A}$ to $\mathcal{B}$ to be the strategies on the arena $\mathcal{A} \to \mathcal{B}$. Formally, if $s \in \mathcal{S}_{\mathcal{A}\to\mathcal{B}}$, then $s_{|\mathcal{A}}$ is a justified sequence on the juxtaposition of J copies of $\mathcal{A}$, however, we will consider that it is a justified sequence on $\mathcal{A}$ by forgetting the particular copy of $\mathcal{A}$ each move belongs to.

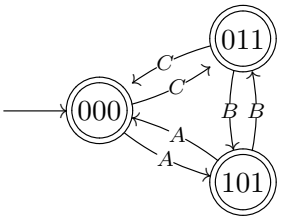
Composition

Let $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ be three arenas. We define the set of interactions on $\mathcal{A}, \mathcal{B}, \mathcal{C}$ by:

$$\mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C}) = \left\{ s \in \mathcal{S}_{(\mathcal{A}\to\mathcal{B})\to\mathcal{C}} \;\middle|\; s_{|\mathcal{A}\to\mathcal{B}} \in \mathcal{P}_{\mathcal{A}\to\mathcal{B}} \wedge s_{|\mathcal{B}\to\mathcal{C}} \in \mathcal{P}_{\mathcal{B}\to\mathcal{C}} \wedge s_{|\mathcal{A}\to\mathcal{C}} \in \mathcal{P}_{\mathcal{A}\to\mathcal{C}} \right\}$$

where similarly to the projection on the left of an arrow, $s_{|\mathcal{A}\to\mathcal{C}}$ can be considered as a justified sequence on $\mathcal{A} \to \mathcal{C}$ by forgetting for each move of s in $\mathcal{A}$ which root of $\mathcal{B}$ it was attached to.

We consider the following automaton on the alphabet $\{A; B; C\}$:



(4.5)

We can run this automaton on a sequence $s \in \mathcal{S}_{(\mathcal{A}\to\mathcal{B})\to\mathcal{C}}$ by mapping moves to the letters A, B, C, depending on the arena they belong to. The following lemma will be useful in order to prove properties of the composition of strategies:

Lemma 4.2. *If $s \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$ then s is accepted by the automaton (4.5). Moreover, the final state of the automaton describes the parity of respectively $s_{|\mathcal{A}\to\mathcal{B}}$, $s_{|\mathcal{B}\to\mathcal{C}}$ and $s_{|\mathcal{A}\to\mathcal{C}}$ (0 for even, 1 for odd)*

Proof. It is important to notice that the polarities of moves depend on which arena we consider. The following table gives the polarities of each kind of move in $\mathcal{A} \to \mathcal{B}$, $\mathcal{B} \to \mathcal{C}$ and $\mathcal{A} \to \mathcal{C}$:

	$\mathcal{A} \to \mathcal{B}$	$\mathcal{B} \to \mathcal{C}$	$\mathcal{A} \to \mathcal{C}$
O-move in $\mathcal{A}$	P		P
P-move in $\mathcal{A}$	O		O
O-move in $\mathcal{B}$	O	P	
P-move in $\mathcal{B}$	P	O	
O-move in $\mathcal{C}$		O	O
P-move in $\mathcal{C}$		P	P

We prove the result by induction on s. If $s = \epsilon$, the result is immediate. If $s = s'a$, we distinguish on the state of s':

- 000: if it appears in the corresponding arena, a must be:
  - an O-move in $\mathcal{A} \to \mathcal{B}$
  - an O-move in $\mathcal{B} \to \mathcal{C}$
  - an O-move in $\mathcal{A} \to \mathcal{C}$

  Using the table, the only possibilities are a P-move in $\mathcal{A}$ (in which case s is in state 101) or an O-move in $\mathcal{C}$ (in which case s is in state 011)
- 011: if it appears in the corresponding arena, a must be:
  - an O-move in $\mathcal{A} \to \mathcal{B}$
  - a P-move in $\mathcal{B} \to \mathcal{C}$
  - a P-move in $\mathcal{A} \to \mathcal{C}$

  Using the table, the only possibilities are an O-move in $\mathcal{B}$ (in which case s is in state 101) or a P-move in $\mathcal{C}$ (in which case s is in state 011)
- 101: if it appears in the corresponding arena, a must be:
  - a P-move in $\mathcal{A} \to \mathcal{B}$
  - an O-move in $\mathcal{B} \to \mathcal{C}$
  - a P-move in $\mathcal{A} \to \mathcal{C}$

  Using the table, the only possibilities are an O-move in $\mathcal{A}$ (in which case s is in state 000) or a P-move in $\mathcal{B}$ (in which case s is in state 011)

□

Let now $\sigma : \mathcal{A} \to \mathcal{B}$ and $\tau : \mathcal{B} \to \mathcal{C}$ be morphisms. We define the interactions between σ and τ as:

$$\sigma \parallel \tau = \left\{ s \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C}) \,\middle|\, s_{|\mathcal{A} \to \mathcal{B}} \in \sigma \wedge s_{|\mathcal{B} \to \mathcal{C}} \in \tau \right\}$$

Finally we define the composition $\sigma ; \tau : \mathcal{A} \to \mathcal{C}$ as:

$$\sigma ; \tau = \left\{ s_{|\mathcal{A} \to \mathcal{C}} \,\middle|\, s \in \sigma \parallel \tau \right\}$$

We now prove that it indeed defines a strategy. First, $\sigma ; \tau \subseteq \mathcal{P}_{\mathcal{A} \to \mathcal{C}}$ by definition of $\mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$. Then if $s \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$ is such that $s_{|\mathcal{A} \to \mathcal{B}} \in \sigma$ and $s_{|\mathcal{B} \to \mathcal{C}} \in \tau$, $|s_{|\mathcal{A} \to \mathcal{B}}|$ and $|s_{|\mathcal{B} \to \mathcal{C}}|$ are even so $|s_{|\mathcal{A} \to \mathcal{C}}| = 2|s| - |s_{|\mathcal{A} \to \mathcal{B}}| - |s_{|\mathcal{B} \to \mathcal{C}}|$ is also even so $\sigma ; \tau$ is a set of P-plays.

Lemma 4.3. *$\sigma ; \tau$ is P-prefix closed*

Proof. Let $s \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$ be such that $s_{|\mathcal{A} \to \mathcal{B}} \in \sigma$ and $s_{|\mathcal{B} \to \mathcal{C}} \in \tau$, and let t be a P-prefix of $s_{|\mathcal{A} \to \mathcal{C}}$. Let s' be a prefix of s such that $s'_{|\mathcal{A} \to \mathcal{C}} = t$. By definition of $\mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$, $s' \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C})$, so by lemma 4.2. s' is accepted by the automaton. Moreover, since $|s'_{|\mathcal{A} \to \mathcal{C}}| = |t|$ is even, the final state of the automaton is of the form $ij0$, so it is 000. This means that $|s'_{|\mathcal{A} \to \mathcal{B}}|$ and $|s'_{|\mathcal{B} \to \mathcal{C}}|$ are even, so by P-prefix closedness of σ and τ, $s'_{|\mathcal{A} \to \mathcal{B}} \in \sigma$ and $s'_{|\mathcal{B} \to \mathcal{C}} \in \tau$. Finally, $t = s'_{|\mathcal{A} \to \mathcal{C}} \in \sigma ; \tau$. □

Lemma 4.4 (Zipping)**.** *If $s, t \in \sigma \parallel \tau$ are such that $s_{|\mathcal{A} \to \mathcal{C}}$ and $t_{|\mathcal{A} \to \mathcal{C}}$ have the same sequence of O-moves, then $s = t$*

Proof. Write $s = s_0...s_m$ and $t = t_0...t_n$ and suppose without loss of generality that $m \leq n$. If $s_0...s_{i-1} = t_0...t_{i-1} = u$, we distinguish on the state of the automaton (4.5) after reading u:

- 000: $u_{|\mathcal{A}\to\mathcal{C}}$ is of even length and the next moves s_i and t_i are in $\mathcal{A}$ or $\mathcal{C}$ (by lemma 4.2), so they are O-moves in $\mathcal{A} \to \mathcal{C}$ and therefore equal by assumption
- 011: $u_{|\mathcal{B}\to\mathcal{C}}$ is of odd length and the next moves s_i and t_i are in $\mathcal{B}$ or $\mathcal{C}$ (by lemma 4.2), so $(us_i)_{|\mathcal{B}\to\mathcal{C}} = u_{|\mathcal{B}\to\mathcal{C}}s_i$ is a P-prefix of $s_{|\mathcal{B}\to\mathcal{C}}$ and $(ut_i)_{|\mathcal{B}\to\mathcal{C}} = u_{|\mathcal{B}\to\mathcal{C}}t_i$ is a P-prefix of $s_{|\mathcal{B}\to\mathcal{C}}$. Therefore $u_{|\mathcal{B}\to\mathcal{C}}s_i \in \tau$ and $u_{|\mathcal{B}\to\mathcal{C}}t_i \in \tau$ and $s_i = t_i$ by determinism of τ.
- 101: $u_{|\mathcal{A}\to\mathcal{B}}$ is of odd length and the next moves s_i and t_i are in $\mathcal{A}$ or $\mathcal{B}$ (by lemma 4.2), so $(us_i)_{|\mathcal{A}\to\mathcal{B}} = u_{|\mathcal{A}\to\mathcal{B}}s_i$ is a P-prefix of $s_{|\mathcal{A}\to\mathcal{B}}$ and $(ut_i)_{|\mathcal{A}\to\mathcal{B}} = u_{|\mathcal{A}\to\mathcal{B}}t_i$ is a P-prefix of $s_{|\mathcal{A}\to\mathcal{B}}$. Therefore $u_{|\mathcal{A}\to\mathcal{B}}s_i \in \sigma$ and $u_{|\mathcal{A}\to\mathcal{B}}t_i \in \sigma$ and $s_i = t_i$ by determinism of σ.

We have then $s = s_0...s_m = t_0...t_m$. Since $s \in \sigma \parallel \tau$, $s_{|\mathcal{A}\to\mathcal{C}}$ is of even length and the automaton 4.5 ends up in state 000 after $t_0...t_m$, so if $n > m$, t_{m+1} is an O-move in $\mathcal{A} \to \mathcal{C}$, contradicting the fact that $s_{|\mathcal{A}\to\mathcal{C}}$ and $t_{|\mathcal{A}\to\mathcal{C}}$ have the same sequence of O-moves. Therefore, $m = n$ and $s = t$. □

Lemma 4.5. *$\sigma;\tau$ is deterministic*

Proof. This is an immediate consequence of the previous lemma. □

Lemma 4.6. *$\sigma;\tau$ is single-threaded*

Proof. We have to prove that $s \in \sigma;\tau$ if and only if $\text{Threads}(s) \subseteq \sigma;\tau$. We prove the two implications:

- From left to right: let $s \in \sigma \parallel \tau$ and let $t \in \text{Threads}\left(s_{|\mathcal{A}\to\mathcal{C}}\right)$.

  $$\text{Threads}\left(s_{|\mathcal{A}\to\mathcal{C}}\right) = \text{Threads}(s)_{|\mathcal{A}\to\mathcal{C}} \text{ so } t = u_{|\mathcal{A}\to\mathcal{C}} \text{ for some } u \in \text{Threads}(s)$$

  $$s_{|\mathcal{A}\to\mathcal{B}} \in \sigma \text{ and } \text{Threads}\left(u_{|\mathcal{A}\to\mathcal{B}}\right) \subseteq \text{Threads}\left(s_{|\mathcal{A}\to\mathcal{B}}\right) \text{ so } \text{Threads}\left(u_{|\mathcal{A}\to\mathcal{B}}\right) \subseteq \sigma$$

  and we get $u_{|\mathcal{A}\to\mathcal{B}} \in \sigma$. On the other hand:

  $$s_{|\mathcal{B}\to\mathcal{C}} \in \tau \text{ and } u_{|\mathcal{B}\to\mathcal{C}} \in \text{Threads}\left(s_{|\mathcal{B}\to\mathcal{C}}\right) \text{ so } u_{|\mathcal{B}\to\mathcal{C}} \in \tau$$

  Therefore $u \in \sigma \parallel \tau$ and $t = u_{|\mathcal{A}\to\mathcal{C}} \in \sigma;\tau$.

- From right to left: let $s \in \mathcal{P}_{\mathcal{A}\to\mathcal{C}}$ be such that $\text{Threads}(s) \subseteq \sigma;\tau$. Let us write:

  $$\text{Threads}(s) = \{t_1;...;t_n\}$$

  Then there are $v_1,...v_n \in \sigma \parallel \tau$ such that $t_i = v_{i|\mathcal{A}\to\mathcal{C}}$. We can construct an interleaving u of the v_i by starting with the first move of v_1, picking each time the next move of the current v_i, and changing the current v_i each time we must play an O-move in $\mathcal{A} \to \mathcal{C}$, by looking at which t_i the corresponding move of s is in. This interleaving is such that:

  $$u \in \text{Int}(\mathcal{A},\mathcal{B},\mathcal{C}),\ u_{|\mathcal{A}\to\mathcal{C}} = s \text{ and } \text{Threads}(u) = \{v_1;...;v_n\}$$

  On one hand we have:

  $$\text{Threads}\left(u_{|\mathcal{A}\to\mathcal{B}}\right) = \cup_i \text{Threads}\left(v_{i|\mathcal{A}\to\mathcal{B}}\right) \subseteq \sigma \text{ since } v_{i|\mathcal{A}\to\mathcal{B}} \in \sigma, \text{ so } u_{|\mathcal{A}\to\mathcal{B}} \in \sigma$$

  On the other hand we have:

  $$\text{Threads}\left(u_{|\mathcal{B}\to\mathcal{C}}\right) = \left\{v_{i|\mathcal{B}\to\mathcal{C}}\right\}_{1\leq i\leq n} \subseteq \tau \text{ since } v_{i|\mathcal{B}\to\mathcal{C}} \in \tau, \text{ so } u_{|\mathcal{B}\to\mathcal{C}} \in \tau$$

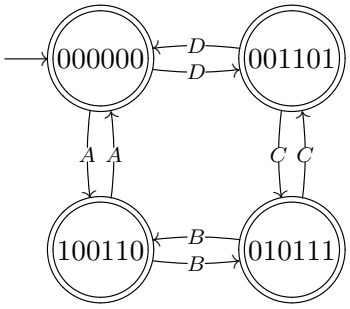
  Putting this together we get $s = u_{|\mathcal{A}\to\mathcal{C}} \in \sigma;\tau$.

□

We could also have defined composition by describing only the threads of the composite, so single-threadedness would have been immediate. However, we have here a more symmetric definition, which is more suited when it comes to associativity of composition. Indeed, we proved that composition of strategies is well-defined. We must still prove that it is associative. Let $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$ be four arenas. We define the set of interactions on $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}$ by:

$$\mathrm{Int}\,(\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D}) = \left\{ s \in \mathcal{S}_{((\mathcal{A}\to\mathcal{B})\to\mathcal{C})\to\mathcal{D}} \;\middle|\; \begin{array}{c} s_{|\mathcal{A}\to\mathcal{B}} \in \mathcal{P}_{\mathcal{A}\to\mathcal{B}} \wedge s_{|\mathcal{B}\to\mathcal{C}} \in \mathcal{P}_{\mathcal{B}\to\mathcal{C}} \\ s_{|\mathcal{C}\to\mathcal{D}} \in \mathcal{P}_{\mathcal{C}\to\mathcal{D}} \wedge s_{|\mathcal{A}\to\mathcal{D}} \in \mathcal{P}_{\mathcal{A}\to\mathcal{D}} \end{array} \right\}$$

We consider the following automaton on the alphabet $\{A; B; C; D\}$:



(4.6)

We can run this automaton on a sequence $s \in \mathcal{S}_{((\mathcal{A}\to\mathcal{B})\to\mathcal{C})\to\mathcal{D}}$ by mapping moves to the letters A, B, C, D, depending on the arena they belong to. The following lemma will be useful in order to prove associativity of the composition of strategies:

Lemma 4.7. *If $s \in \mathrm{Int}\,(\mathcal{A},\mathcal{B},\mathcal{C},\mathcal{D})$ then s is accepted by the automaton (4.6). The final state of the automaton describes the parity of respectively $s_{|\mathcal{A}\to\mathcal{B}}$, $s_{|\mathcal{B}\to\mathcal{C}}$, $s_{|\mathcal{C}\to\mathcal{D}}$, $s_{|\mathcal{A}\to\mathcal{D}}$, $s_{|\mathcal{A}\to\mathcal{C}}$ and $s_{|\mathcal{B}\to\mathcal{D}}$ (0 for even, 1 for odd). Moreover $s_{|\mathcal{A}\to\mathcal{C}} \in \mathcal{P}_{\mathcal{A}\to\mathcal{C}}$ and $s_{|\mathcal{B}\to\mathcal{D}} \in \mathcal{P}_{\mathcal{B}\to\mathcal{D}}$.*

Proof. It important to notice that the polarities of moves depend on which arena we consider. The following table gives the polarities of each kind of move in $\mathcal{A}\to\mathcal{B}$, $\mathcal{B}\to\mathcal{C}$, $\mathcal{C}\to\mathcal{D}$, $\mathcal{A}\to\mathcal{D}$, $\mathcal{A}\to\mathcal{C}$ and $\mathcal{B}\to\mathcal{D}$:

	$\mathcal{A}\to\mathcal{B}$	$\mathcal{B}\to\mathcal{C}$	$\mathcal{C}\to\mathcal{D}$	$\mathcal{A}\to\mathcal{D}$	$\mathcal{A}\to\mathcal{C}$	$\mathcal{B}\to\mathcal{D}$
O-move in $\mathcal{A}$	P			P	P	
P-move in $\mathcal{A}$	O			O	O	
O-move in $\mathcal{B}$	O	P				P
P-move in $\mathcal{B}$	P	O				O
O-move in $\mathcal{C}$		O	P		O	
P-move in $\mathcal{C}$		P	O		P	
O-move in $\mathcal{D}$			O	O		O
P-move in $\mathcal{D}$			P	P		P

We prove the result by induction on s. If $s = \epsilon$, the result is immediate. If $s = s'a$, we distinguish on the state of s':

- 000000: if it appears in the corresponding arena, a must be either:
  - an O-move in $\mathcal{A}\to\mathcal{B}$
  - an O-move in $\mathcal{B}\to\mathcal{C}$

– an O-move in $\mathcal{C} \to \mathcal{D}$
– an O-move in $\mathcal{A} \to \mathcal{D}$

Using the table, the only possibilities are a P-move in $\mathcal{A}$ (in which case s is in state 100110, and a is an O-move in $\mathcal{A} \to \mathcal{C}$, which preserves alternation on $\mathcal{A} \to \mathcal{C}$) or an O-move in $\mathcal{D}$ (in which case s is in state 001101, and a is an O-move in $\mathcal{B} \to \mathcal{D}$, which preserves alternation on $\mathcal{B} \to \mathcal{D}$).

- 001101: if it appears in the corresponding arena, a must be either:
  – an O-move in $\mathcal{A} \to \mathcal{B}$
  – an O-move in $\mathcal{B} \to \mathcal{C}$
  – a P-move in $\mathcal{C} \to \mathcal{D}$
  – a P-move in $\mathcal{A} \to \mathcal{D}$

  Using the table, the only possibilities are an O-move in $\mathcal{C}$ (in which case s is in state 010111, and a is an O-move in $\mathcal{A} \to \mathcal{C}$, which preserves alternation on $\mathcal{A} \to \mathcal{C}$) or a P-move in $\mathcal{D}$ (in which case s is in state 000000, and a is a P-move in $\mathcal{B} \to \mathcal{D}$, which preserves alternation on $\mathcal{B} \to \mathcal{D}$).

- 010111: if it appears in the corresponding arena, a must be either:
  – an O-move in $\mathcal{A} \to \mathcal{B}$
  – a P-move in $\mathcal{B} \to \mathcal{C}$
  – an O-move in $\mathcal{C} \to \mathcal{D}$
  – a P-move in $\mathcal{A} \to \mathcal{D}$

  Using the table, the only possibilities are an O-move in $\mathcal{B}$ (in which case s is in state 100110, and a is a P-move in $\mathcal{B} \to \mathcal{D}$, which preserves alternation on $\mathcal{B} \to \mathcal{D}$) or a P-move in $\mathcal{C}$ (in which case s is in state 001101, and a is a P-move in $\mathcal{A} \to \mathcal{C}$, which preserves alternation on $\mathcal{A} \to \mathcal{C}$).

- 100110: if it appears in the corresponding arena, a must be either:
  – a P-move in $\mathcal{A} \to \mathcal{B}$
  – an O-move in $\mathcal{B} \to \mathcal{C}$
  – an O-move in $\mathcal{C} \to \mathcal{D}$
  – a P-move in $\mathcal{A} \to \mathcal{D}$

  Using the table, the only possibilities are an O-move in $\mathcal{A}$ (in which case s is in state 000000, and a is a P-move in $\mathcal{A} \to \mathcal{C}$, which preserves alternation on $\mathcal{A} \to \mathcal{C}$) or a P-move in $\mathcal{B}$ (in which case s is in state 010111, and a is an O-move in $\mathcal{B} \to \mathcal{D}$, which preserves alternation on $\mathcal{B} \to \mathcal{D}$).

□

In particular, with this lemma we get that if $s \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$, then:

$$s_{|(\mathcal{A}\to\mathcal{B})\to\mathcal{C}} \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{C}) \qquad s_{|(\mathcal{A}\to\mathcal{B})\to\mathcal{D}} \in \mathrm{Int}\,(\mathcal{A}, \mathcal{B}, \mathcal{D})$$

$$s_{|(\mathcal{A}\to\mathcal{C})\to\mathcal{D}} \in \mathrm{Int}\,(\mathcal{A}, \mathcal{C}, \mathcal{D}) \qquad s_{|(\mathcal{B}\to\mathcal{C})\to\mathcal{D}} \in \mathrm{Int}\,(\mathcal{B}, \mathcal{C}, \mathcal{D})$$

The following lemmas are the essential steps towards the associativity of composition. The first lemma will be used because if $\sigma : \mathcal{A} \to \mathcal{B}$, $\tau : \mathcal{B} \to \mathcal{C}$ and $\nu : \mathcal{C} \to \mathcal{D}$, then $\sigma \parallel \tau \subseteq \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C})$ and $(\sigma; \tau) \parallel \nu \subseteq \mathrm{Int}(\mathcal{A}, \mathcal{C}, \mathcal{D})$.

Lemma 4.8 (Double zipping 1)**.** *Let $s \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C})$ and $t \in \mathrm{Int}(\mathcal{A}, \mathcal{C}, \mathcal{D})$ such that $s_{\mid \mathcal{A} \to \mathcal{C}} = t_{\mid \mathcal{A} \to \mathcal{C}}$. There exists $u \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ such that $u_{\mid (\mathcal{A} \to \mathcal{B}) \to \mathcal{C}} = s$ and $u_{\mid (\mathcal{A} \to \mathcal{C}) \to \mathcal{D}} = t$.*

Proof. The proof goes by induction on the length of $s_{\mid \mathcal{A} \to \mathcal{C}}$. If $s_{\mid \mathcal{A} \to \mathcal{C}} = \epsilon$ then since $s \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C})$, we have $s = \epsilon$ by lemma 4.2, and so we take $u = t$. Otherwise we can write $s = s'av$ with v containing only moves in $\mathcal{B}$ and a a move in $\mathcal{A} \to \mathcal{C}$. We have $s_{\mid \mathcal{A} \to \mathcal{C}} = s'_{\mid \mathcal{A} \to \mathcal{C}} a$. Since $s_{\mid \mathcal{A} \to \mathcal{C}} = t_{\mid \mathcal{A} \to \mathcal{C}}$, we can write $t = t'aw$ with w containing only moves in $\mathcal{D}$, and we have $s'_{\mid \mathcal{A} \to \mathcal{C}} = t'_{\mid \mathcal{A} \to \mathcal{C}}$. By induction hypothesis, there is some $u' \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ such that $u'_{\mid (\mathcal{A} \to \mathcal{B}) \to \mathcal{C}} = s'$ and $u'_{\mid (\mathcal{A} \to \mathcal{C}) \to \mathcal{D}} = t'$. We prove now that $v = \epsilon$ or $w = \epsilon$. Suppose $v \neq \epsilon$. Since $s = s'av \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C})$ and v contains at least one move in $\mathcal{B}$, using lemma 4.2 we get that s' must bring the automaton (4.5) in state 000 and a is an O-move in $\mathcal{A} \to \mathcal{C}$. But then $t'a_{\mid \mathcal{A} \to \mathcal{C}}$ is of odd-length so $t'a$ must bring the automaton (4.5) in state 101 (beware that here the automaton is on A, C, D). Therefore, the first move of w (if it existed) would be a move in $\mathcal{A}$ or $\mathcal{C}$, which is false since w contains only moves in $\mathcal{D}$. If $v = \epsilon$, then we take $u = u'aw$, and if $w = \epsilon$ we take $u = u'av$. □

This second lemma will be used because if $\sigma : \mathcal{A} \to \mathcal{B}$, $\tau : \mathcal{B} \to \mathcal{C}$ and $\nu : \mathcal{C} \to \mathcal{D}$, then $\sigma \parallel (\tau; \nu) \subseteq \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{D})$ and $\tau \parallel \nu \subseteq \mathrm{Int}(\mathcal{B}, \mathcal{C}, \mathcal{D})$. Its proof is very similar to that of the first lemma.

Lemma 4.9 (Double zipping 2)**.** *Let $s \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{D})$ and $t \in \mathrm{Int}(\mathcal{B}, \mathcal{C}, \mathcal{D})$ such that $s_{\mid \mathcal{B} \to \mathcal{D}} = t_{\mid \mathcal{B} \to \mathcal{D}}$. There exists $u \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ such that $u_{\mid (\mathcal{A} \to \mathcal{B}) \to \mathcal{D}} = s$ and $u_{\mid (\mathcal{B} \to \mathcal{C}) \to \mathcal{D}} = t$.*

Now we can prove associativity of composition:

Lemma 4.10. *Let $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ and $\mathcal{D}$ be four arenas, and let $\sigma : \mathcal{A} \to \mathcal{B}$, $\tau : \mathcal{B} \to \mathcal{C}$ and $\nu : \mathcal{C} \to \mathcal{D}$. We have $(\sigma; \tau); \nu = \sigma; (\tau; \nu)$.*

Proof. We prove the left-to-right inclusion, the other one being similar (using lemma 4.9). Suppose $v \in (\sigma; \tau); \nu$. By definition, there exists $t \in \mathrm{Int}(\mathcal{A}, \mathcal{C}, \mathcal{D})$ such that $t_{\mid \mathcal{A} \to \mathcal{D}} = v$, $t_{\mid \mathcal{A} \to \mathcal{C}} \in \sigma; \tau$ and $t_{\mid \mathcal{C} \to \mathcal{D}} \in \nu$. By definition of $\sigma; \tau$, there is some $s \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C})$ such that $s_{\mid \mathcal{A} \to \mathcal{C}} = t_{\mid \mathcal{A} \to \mathcal{C}}$, $s_{\mid \mathcal{A} \to \mathcal{B}} \in \sigma$ and $s_{\mid \mathcal{B} \to \mathcal{C}} \in \tau$. By lemma 4.8 we get some $u \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D})$ such that $u_{\mid (\mathcal{A} \to \mathcal{B}) \to \mathcal{C}} = s$ and $u_{\mid (\mathcal{A} \to \mathcal{C}) \to \mathcal{D}} = t$. Therefore, $u_{\mid \mathcal{B} \to \mathcal{C}} = s_{\mid \mathcal{B} \to \mathcal{C}} \in \tau$ and $u_{\mid \mathcal{C} \to \mathcal{D}} = t_{\mid \mathcal{C} \to \mathcal{D}} \in \nu$, so since $u_{\mid (\mathcal{B} \to \mathcal{C}) \to \mathcal{D}} \in \mathrm{Int}(\mathcal{B}, \mathcal{C}, \mathcal{D})$ we get $u_{\mid \mathcal{B} \to \mathcal{D}} \in \tau; \nu$. Finally, $u_{\mid \mathcal{A} \to \mathcal{B}} = s_{\mid \mathcal{A} \to \mathcal{B}} \in \sigma$ and $u_{\mid \mathcal{A} \to \mathcal{D}} = t_{\mid \mathcal{A} \to \mathcal{D}} = v$, so since $u_{\mid (\mathcal{A} \to \mathcal{B}) \to \mathcal{D}} \in \mathrm{Int}(\mathcal{A}, \mathcal{B}, \mathcal{D})$ we get $v \in \sigma; (\tau; \nu)$. □

Identity

If $\mathcal{A}$ is an arena and if $s \in \mathcal{S}_{\mathcal{A}}$ is a justified sequence with pointer f, we define the copycat play s^* of s in $\mathcal{P}_{\mathcal{A} \to \mathcal{A}}$ with pointer f^*. We distinguish the two occurrences of $\mathcal{A}$ in $\mathcal{A} \to \mathcal{A}$ by writing $\mathcal{A}^{(1)} \to \mathcal{A}^{(2)}$ and the moves a of $\mathcal{A} \to \mathcal{A}$ by writing $a^{(1)}$ or $a^{(2)}$, depending on which occurrence of $\mathcal{A}$ they are in. Given s as above, let s^* be the following sequence of moves:

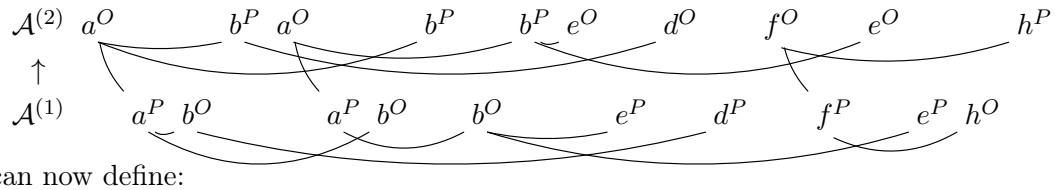
$$s^*_{2i} s^*_{2i+1} = \begin{cases} {s_i}^{(2)} {s_i}^{(1)} \text{ if } s_i \text{ is an } O\text{-move in } \mathcal{A} \\ {s_i}^{(1)} {s_i}^{(2)} \text{ if } s_i \text{ is a } P\text{-move in } \mathcal{A} \end{cases}$$

and f^* be the following pointer function:

- if $f(i)$ is undefined then $f^*(2i)$ is undefined and $f^*(2i+1) = 2i$

- if $f(i)$ is defined then $f^*(2i) = 2f(i) + 1$ and $f^*(2i+1) = 2f(i)$

For example, if s is the sequence (4.4) on the arena (4.2), then s^* is:



We can now define:

$$\mathbf{Id}_{\mathcal{A}} = \{s^* \mid s \in \mathcal{S}_{\mathcal{A}}\}$$

Lemma 4.11. *s^* is an element of $\mathcal{P}_{\mathcal{A}\to\mathcal{A}}$ such that $|s^*| = 2|s|$ and $s^*_{|\mathcal{A}^{(1)}} = s^*_{|\mathcal{A}^{(2)}} = s$.*

Proof. $s^* \in \mathcal{P}_{\mathcal{A}\to\mathcal{A}}$ and $|s^*| = 2|s|$ are immediate. For $s^*_{|\mathcal{A}^{(1)}} = s^*_{|\mathcal{A}^{(2)}} = s$, one has to remark that a move of s^* in $\mathcal{A}^{(1)}$ will get an inverted polarity in $s^*_{|\mathcal{A}^{(1)}}$. □

Lemma 4.12. *$\mathbf{Id}_{\mathcal{A}}$ is a strategy: it is P-prefix-closed, deterministic and single-threaded.*

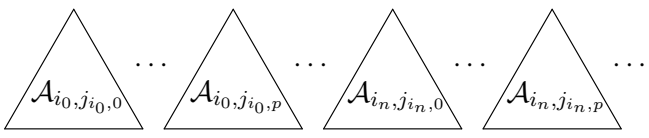
Proof. It is easy to see that the set of P-prefixes of s^* is the set of all t^* for t prefix of s. Therefore $\mathbf{Id}_{\mathcal{A}}$ is P-prefix-closed. If s^*ab and s^*ac are in $\mathbf{Id}_{\mathcal{A}}$, then a is either some $a^{(1)}$, in which case $b = c = a^{(2)}$, or some $a^{(2)}$, in which case $b = c = a^{(1)}$. Therefore $\mathbf{Id}_{\mathcal{A}}$ is deterministic. If $s \in \mathcal{S}_{\mathcal{A}}$, then $\mathrm{Threads}(s^*) = \mathrm{Threads}(s)^* \subseteq \mathbf{Id}_{\mathcal{A}}$. Conversely, let $s \in \mathcal{P}_{\mathcal{A}}$ be a P-play such that $\mathrm{Threads}(s) \subseteq \mathbf{Id}_{\mathcal{A}}$. Consider a sequence in s of an O-move followed by a P-move. Since these moves are in the same thread (see the remark before the definition of a strategy) and this thread is in $\mathbf{Id}_{\mathcal{A}}$, the sequence is of the form $a^{(1)}a^{(2)}$ or $a^{(2)}a^{(1)}$. Therefore we get $s \in \mathbf{Id}_{\mathcal{A}}$. □

Lemma 4.13. *If $\sigma : \mathcal{A} \to \mathcal{B}$, then $\mathbf{Id}_{\mathcal{A}}; \sigma = \sigma = \sigma; \mathbf{Id}_{\mathcal{B}}$.*

Proof. Let $t \in \mathbf{Id}_{\mathcal{A}}; \sigma$. By definition of composition, there exists some $s \in \mathrm{Int}\left(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \mathcal{B}\right)$ such that $s_{|\mathcal{A}^{(1)}\to\mathcal{A}^{(2)}} \in \mathbf{Id}_{\mathcal{A}}$, $s_{|\mathcal{A}^{(2)}\to\mathcal{B}} \in \sigma$ and $s_{|\mathcal{A}^{(1)}\to\mathcal{B}} = t$. By property of $\mathbf{Id}_{\mathcal{A}}$, $s_{|\mathcal{A}^{(1)}} = s_{|\mathcal{A}^{(2)}}$, so $t = s_{|\mathcal{A}^{(1)}\to\mathcal{B}} = s_{|\mathcal{A}^{(2)}\to\mathcal{B}} \in \sigma$. Let now $t \in \sigma$. We build $s \in \mathcal{S}_{\left(\mathcal{A}^{(1)}\to\mathcal{A}^{(2)}\right)\to\mathcal{B}}$ by replacing every move a of t which is in $\mathcal{A}$ by $a^{(2)}a^{(1)}$ if it is an O-move in $\mathcal{A}$ or by $a^{(1)}a^{(2)}$ if it is a P-move in $\mathcal{A}$. We check that by construction, $s \in \mathrm{Int}\left(\mathcal{A}^{(1)}, \mathcal{A}^{(2)}, \mathcal{B}\right)$, $s_{|\mathcal{A}^{(1)}} = s_{|\mathcal{A}^{(1)}}$ so $s_{|\mathcal{A}^{(1)}\to\mathcal{A}^{(2)}} \in \mathbf{Id}_{\mathcal{A}}$ and $s_{|\mathcal{A}^{(1)}\to\mathcal{B}} = s_{|\mathcal{A}^{(2)}\to\mathcal{B}} = t \in \sigma$. The second equality $\sigma = \sigma; \mathbf{Id}_{\mathcal{B}}$ is proved similarly. □

4.3.2 Countable products

If $(\mathcal{A}_i)_{i\in I}$ is a countable family of arenas consisting of the trees $(\mathcal{A}_{i,j})_{j\in J_i}$, if $i_0, ..., i_n, ...$ is an enumeration of I, and if $j_{i,0}, ..., j_{i,p}, ...$ are enumerations of the J_i, then the arena $\prod_{i\in I}\mathcal{A}_i$ is defined as the disjoint sum:



This is correct since this arena is still countably-branching. We define the projection $\mathbf{proj}_{i_0} : \prod_{i\in I}\mathcal{A}_i \to \mathcal{A}_{i_0} = \mathbf{Id}_{\mathcal{A}_{i_0}}$. This is correct since $\mathcal{A}_{i_0} \to \mathcal{A}_{i_0} \subseteq \prod_{i\in I}\mathcal{A}_i \to \mathcal{A}_{i_0}$, so a P-play on $\mathcal{A}_{i_0} \to \mathcal{A}_{i_0}$ is a P-play on $\prod_{i\in I}\mathcal{A}_i \to \mathcal{A}_{i_0}$. If $\sigma_i : \mathcal{C} \to \mathcal{A}_i$, we define the product $(\sigma_i)_{i\in I} : \mathcal{C} \to \prod_{i\in I}\mathcal{A}_i$ by its set of threads:

$$\mathrm{Threads}\left((\sigma_i)_{i\in I}\right) = \bigcup_{i\in I}\mathrm{Threads}(\sigma_i)$$

This definition is correct because since $\mathcal{C} \to \mathcal{A}_i \subseteq \mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$ and the roots of $\mathcal{C} \to \mathcal{A}_i$ are roots of $\mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$, a thread on $\mathcal{C} \to \mathcal{A}_i$ which is a P-play is a thread on $\mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$ which is a P-play. Moreover, since the sets of roots of the $\mathcal{C} \to \mathcal{A}_i$ are disjoint, determinism of $(\sigma)_{i\in I}$ is a consequence of the determinism of the σ_i.

Lemma 4.14. *The constructions above define a product on $\mathcal{G}$.*

Proof. First, we prove $(\sigma_i)_{i\in I}\,;\mathbf{proj}_{i_0} = \sigma_{i_0}$ for $\sigma_i : \mathcal{C} \to \mathcal{A}_i$:

$$
\begin{aligned}
(\sigma_i)_{i\in I}\,;\mathbf{proj}_{i_0} &= \left\{ s_{|\mathcal{C}\to\mathcal{A}_i} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, \prod_{i\in I}\mathcal{A}_i, \mathcal{A}_{i_0}\right) \\ s_{|\mathcal{C}\to\prod_{i\in I}\mathcal{A}_i} \in (\sigma_i)_{i\in I} \\ s_{|\prod_{i\in I}\mathcal{A}_i\to\mathcal{A}_{i_0}} \in \mathbf{proj}_{i_0} \end{array} \right\} \\
&= \left\{ s_{|\mathcal{C}\to\mathcal{A}_i} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, \prod_{i\in I}\mathcal{A}_i, \mathcal{A}_{i_0}\right) \\ \mathrm{Threads}\left(s_{|\mathcal{C}\to\prod_{i\in I}\mathcal{A}_i}\right) \subseteq (\sigma_i)_{i\in I} \\ \forall i \neq i_0, s_{|\mathcal{A}_i} = \epsilon \wedge s_{|\mathcal{A}_{i_0}\to\mathcal{A}_{i_0}} \in \mathbf{Id}_{\mathcal{A}_{i_0}} \end{array} \right\} \\
&= \left\{ s_{|\mathcal{C}\to{\mathcal{A}_{i_0}}^{(2)}} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, {\mathcal{A}_{i_0}}^{(1)}, {\mathcal{A}_{i_0}}^{(2)}\right) \\ \mathrm{Threads}\left(s_{|\mathcal{C}\to{\mathcal{A}_{i_0}}^{(1)}}\right) \subseteq \sigma_{i_0} \\ s_{|{\mathcal{A}_{i_0}}^{(1)}\to{\mathcal{A}_{i_0}}^{(2)}} \in \mathbf{Id}_{\mathcal{A}_{i_0}} \end{array} \right\} \\
&= \left\{ s_{|\mathcal{C}\to{\mathcal{A}_{i_0}}^{(2)}} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, {\mathcal{A}_{i_0}}^{(1)}, {\mathcal{A}_{i_0}}^{(2)}\right) \\ s_{|\mathcal{C}\to{\mathcal{A}_{i_0}}^{(1)}} \in \sigma_{i_0} \\ s_{|{\mathcal{A}_{i_0}}^{(1)}\to{\mathcal{A}_{i_0}}^{(2)}} \in \mathbf{Id}_{\mathcal{A}_{i_0}} \end{array} \right\} \\
&= \sigma_{i_0}; \mathbf{Id}_{\mathcal{A}_{i_0}} \\
&= \sigma_{i_0}
\end{aligned}
$$

Now, we prove $(\sigma; \mathbf{proj}_i)_{i\in I} = \sigma$ for $\sigma : \mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$:

$$
\mathrm{Threads}\left((\sigma; \mathbf{proj}_i)_{i\in I}\right) = \bigcup_{i\in I} \mathrm{Threads}\left(\sigma; \mathbf{proj}_i\right)
$$

$$
\begin{aligned}
\sigma; \mathbf{proj}_{i_0} &= \left\{ s_{|\mathcal{C}\to\mathcal{A}_{i_0}} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, \prod_{i\in I}\mathcal{A}_i, \mathcal{A}_{i_0}\right) \\ s_{|\mathcal{C}\to\prod_{i\in I}\mathcal{A}_i} \in \sigma \\ s_{|\prod_{i\in I}\mathcal{A}_i\to\mathcal{A}_{i_0}} \in \mathbf{proj}_{i_0} \end{array} \right\} \\
&= \left\{ s_{|\mathcal{C}\to\mathcal{A}_{i_0}} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, \prod_{i\in I}\mathcal{A}_i, \mathcal{A}_{i_0}\right) \\ s_{|\mathcal{C}\to\prod_{i\in I}\mathcal{A}_i} \in \sigma \\ \forall i \neq i_0, s_{|\mathcal{A}_i} = \epsilon \wedge s_{|\mathcal{A}_{i_0}\to\mathcal{A}_{i_0}} \in \mathbf{Id}_{\mathcal{A}_{i_0}} \end{array} \right\}
\end{aligned}
$$

$$= \left\{ s_{|\mathcal{C}\to\mathcal{A}_{i_0}{}^{(2)}} \;\middle|\; \begin{array}{l} s \in \mathrm{Int}\left(\mathcal{C}, \mathcal{A}_{i_0}{}^{(1)}, \mathcal{A}_{i_0}{}^{(2)}\right) \\ s_{|\mathcal{C}\to\mathcal{A}_{i_0}{}^{(1)}} \in \sigma \cap \mathcal{S}_{\mathcal{C}\to\mathcal{A}_{i_0}} \\ s_{|\mathcal{A}_{i_0}{}^{(1)}\to\mathcal{A}_{i_0}{}^{(2)}} \in \mathbf{Id}_{\mathcal{A}_{i_0}} \end{array} \right\}$$
$$= \left(\sigma \cap \mathcal{S}_{\mathcal{C}\to\mathcal{A}_{i_0}}\right); \mathbf{Id}_{\mathcal{A}_{i_0}}$$
$$= \sigma \cap \mathcal{S}_{\mathcal{C}\to\mathcal{A}_{i_0}}$$

therefore:

$$\begin{aligned} \mathrm{Threads}\left((\sigma; \mathbf{proj}_i)_{i\in I}\right) &= \bigcup_{i\in I} \mathrm{Threads}\left(\sigma \cap \mathcal{S}_{\mathcal{C}\to\mathcal{A}_i}\right) \\ &= \bigcup_{i\in I} \left(\mathrm{Threads}\left(\sigma\right) \cap \mathcal{S}_{\mathcal{C}\to\mathcal{A}_i}\right) \\ &= \mathrm{Threads}\left(\sigma\right) \cap \bigcup_{i\in I} \mathcal{S}_{\mathcal{C}\to\mathcal{A}_i} \end{aligned}$$

Now, since the set of roots of $\mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$ is the disjoint union of the roots of $\mathcal{C} \to \mathcal{A}_i$, every thread on $\mathcal{C} \to \prod_{i\in I} \mathcal{A}_i$ is in one of the $\mathcal{S}_{\mathcal{C}\to\mathcal{A}_i}$. Finally we get $\mathrm{Threads}\left((\sigma; \mathbf{proj}_i)_{i\in I}\right) = \mathrm{Threads}\left(\sigma\right)$ and so $(\sigma; \mathbf{proj}_i)_{i\in I} = \sigma$. □

It is clear that the empty product and terminal object $\mathbf{1}$ of $\mathcal{G}$ is the empty arena $\mathcal{U}$.

4.3.3 Closed structure

If $\mathcal{A}$ and $\mathcal{B}$ are two arenas, then the exponential $\mathcal{B}^{\mathcal{A}}$ is just the arena $\mathcal{A} \to \mathcal{B}$. In $\mathcal{G}$, the arenas $(\mathcal{A} \to \mathcal{B}) \times \mathcal{A} \to \mathcal{B}$ and $(\mathcal{A} \to \mathcal{B}) \to (\mathcal{A} \to \mathcal{B})$ are identical, so the strategy $\mathbf{Id}_{\mathcal{A}\to\mathcal{B}}$ is a strategy on the arena $(\mathcal{A} \to \mathcal{B}) \times \mathcal{A} \to \mathcal{B}$. We define then the evaluation morphism $\mathbf{ev} = \mathbf{Id}_{\mathcal{A}\to\mathcal{B}}$. Let now $\sigma : \mathcal{C} \times \mathcal{A} \to \mathcal{B}$. Again, in $\mathcal{G}$ the arenas $\mathcal{C} \times \mathcal{A} \to \mathcal{B}$ and $\mathcal{C} \to \mathcal{A} \to \mathcal{B}$ are identical, so $\sigma : \mathcal{C} \to \mathcal{B}^{\mathcal{A}}$ and we can define $\mathbf{\Lambda}\left(\sigma\right) = \sigma$. Therefore, the commutativity of the diagram is just a consequence of $\sigma; \mathbf{Id}_{\mathcal{A}\to\mathcal{B}} = \sigma$.

4.4 Interpreting μPCF in game semantics

4.4.1 $\mathcal{G}$ as a category of continuations

In this section we describe how the category $\mathcal{G}$ of unbracketed single-threaded HON games can be seen as a category of continuations. Recall from section 3.2.2 that since $\mathcal{G}$ is a cartesian closed category with countable products, for any arena $\mathcal{A} \in Ob\left(\mathcal{G}\right)$ the full subcategory $\{\mathcal{A}\}^{\mathrm{Fam}(\mathcal{G})}$ of $\mathrm{Fam}\left(\mathcal{G}\right)$ is a category of continuations. Here we write:

$$\mathcal{C} = \mathrm{Fam}\left(\mathcal{G}\right)$$

we write $\mathcal{V}$ for the one-move arena, and we fix the object R of $\mathcal{C}$ to be the singleton family $\{\mathcal{V}\}$. The full subcategory $R^{\mathcal{C}} = \{\mathcal{V}\}^{\mathrm{Fam}(\mathcal{G})}$ of $\mathcal{C} = \mathrm{Fam}\left(\mathcal{G}\right)$ is then a category of continuations.

To define the interpretation of μPCF in $R^{\mathcal{C}}$ we need to define an object of continuations $[\![T]\!] \in Ob\left(\mathcal{C}\right)$ for each sort T, the object of computations being then defined as $[T] = R^{[\![T]\!]} \in Ob\left(R^{\mathcal{C}}\right)$. Then we also need to define for each constant $\mathsf{c} : T$ a morphism in $R^{\mathcal{C}}$ from $\mathbf{1} = R^{\mathbf{0}}$ to $[T]$. The interpretation of $\lambda\mu$-calculus is then such that a $\lambda\mu$-term of type σ with free λ-variables of type $U_1, ..., U_n$ and free μ-variables of type $V_1, ..., V_m$ is interpreted as a morphism in $R^{\mathcal{C}}$ from $[U_1] \times ... \times [U_n]$ to $[T] \parr [V_1] \parr ... \parr [V_m]$. However, we would like the realizers to be strategies, that is, morphisms in $\mathcal{G}$. The following lemma fills the gap.

Lemma 4.15. *The category $R^{\mathcal{C}} = R^{\mathrm{Fam}(\mathcal{G})}$ is isomorphic to the category $\mathcal{G}$*

Proof. The only non-trivial part is that any singleton family $\{\mathcal{A}\}$ in $\mathrm{Fam}(\mathcal{G})$ can be written as $R^{\{A_i \mid i\in I\}}$. Let $(a_i)_{i\in I}$ be the set of roots of $\mathcal{A}$ and let $\mathcal{A}_i$ be the forest obtained by removing the move a_i from the tree rooted in a_i. Then by definition of the arrow and product in $\mathcal{G}$ and since $\mathcal{V}$ is the one-move arena, the arena $\prod_{i\in I}\mathcal{V}^{\mathcal{A}_i}$ is equal to $\mathcal{A}$. But by definition of the exponential in $\mathrm{Fam}(\mathcal{G})$, we have:

$$\left\{\prod_{i\in I}\mathcal{V}^{\mathcal{A}_i}\right\} = \{\mathcal{V}\}^{\{\mathcal{A}_i \mid i\in I\}} = R^{\{\mathcal{A}_i \mid i\in I\}}$$

Therefore any singleton family in $\mathcal{C}$ is in fact an object of $R^{\mathcal{C}}$. We define now a functor F from $\mathcal{G}$ to $R^{\mathrm{Fam}(\mathcal{G})}$. It is defined on objects as $F(\mathcal{A}) = \{\mathcal{A}\}$ (singleton family, so in $R^{\mathcal{C}}$). If $\sigma : \mathcal{A}\to\mathcal{B}$ is a morphism (strategy) in $\mathcal{G}$ then $F(\sigma)$ is defined as:

$$F(\sigma) = \left(\mathbf{Id}_{\{*\}} : \{*\}\to\{*\}, \{\sigma : \mathcal{A}\to\mathcal{B}\}\right) : \{\mathcal{A}\}\to\{\mathcal{B}\}$$

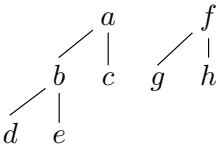
which is a morphism in $R^{\mathcal{C}}$ since $\{\mathcal{A}\}$ and $\{\mathcal{B}\}$ are objects of $R^{\mathcal{C}}$ as singleton families, and $R^{\mathcal{C}}$ is a full sub-category. Then it is straightforward to show that F is indeed a functor, and that it is bijective on objects and on hom-sets. □

Under this isomorphism, the countable product in $R^{\mathrm{Fam}(\mathcal{G})}$ is as follows:

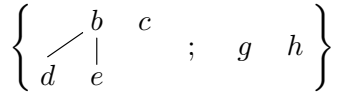
$$\prod_{i\in I} R^{\{\mathcal{A}_{i,j} \mid j\in J_i\}} = R^{\left\{\mathcal{A}_{i,j} \,\middle|\, (i,j)\in\sum_{i\in I} J_i\right\}}$$

which is a correct definition since $\sum_{i\in I} J_i$ is countable if I and the J_i are.

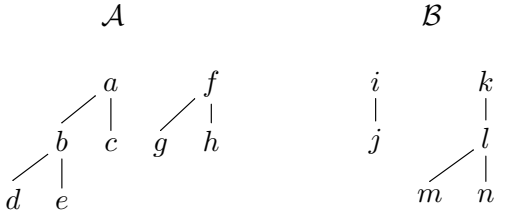
This isomorphism between $R^{\mathrm{Fam}(\mathcal{G})}$ and $\mathcal{G}$ allows to see $\mathcal{G}$ as a category of continuations. As an example, the arena (4.2):



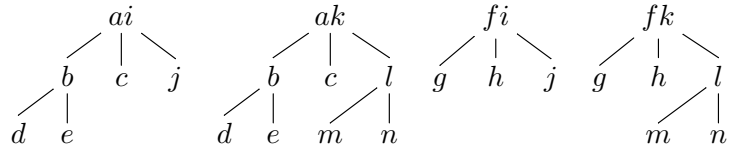
is the exponential of $R = \{\mathcal{V}\}$ by the following family:



Another way to look at it is that the arena (4.2) can be written as $(\mathcal{A}_1\to\mathcal{V})\times(\mathcal{A}_2\to\mathcal{V})$, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are the two arenas above. In the following we will not distinguish between $R^{\mathcal{C}}$ and $\mathcal{G}$, and we will for example consider that an object R^A of $R^{\mathcal{C}}$ is an object of $\mathcal{G}$. In particular, if $\mathcal{A}$ and $\mathcal{B}$ are arenas, then $\mathcal{A}\parr\mathcal{B}$ is also an arena. For example, if $\mathcal{A}$ and $\mathcal{B}$ are the following arenas:



then the arena $\mathcal{A} \parr \mathcal{B}$ is:



Another way to see it is that if:

$$\mathcal{A} = R^{\{\mathcal{A}_i \mid i \in I\}} = \prod_{i \in I} (\mathcal{A}_i \to \mathcal{V}) \qquad \text{and} \qquad \mathcal{B} = R^{\{\mathcal{B}_j \mid j \in J\}} = \prod_{j \in J} (\mathcal{B}_j \to \mathcal{V})$$

then:

$$\mathcal{A} \parr \mathcal{B} = R^{\{\mathcal{A}_i \times \mathcal{B}_j \mid (i,j) \in I \times J\}} = \prod_{(i,j) \in I \times J} (\mathcal{A}_i \times \mathcal{B}_j \to \mathcal{V})$$

Therefore $\prod_{(i,j) \in I \times J} (\mathcal{A}_i \to \mathcal{V}) = \prod_{j \in J} \mathcal{A}$ is a subarena of $\mathcal{A} \parr \mathcal{B}$, but if $s \in \mathcal{S}_{\mathcal{A} \parr \mathcal{B}}$, we will write $s_{|\mathcal{A}} \in \mathcal{S}_{\mathcal{A}}$ by forgetting for each move of s in $\mathcal{A}$ its J-index. We give now the example of an interpretation of a $\lambda\mu$-term:

Lemma 4.16. *The strategy* $[\vdash \lambda x.\, [\alpha]\ x : T \to 0 \mid \alpha : T]$ *is the identity strategy on* $[T]$.

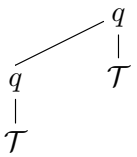
Proof. For simplicity, we suppose that $[T]$ has a unique root:

$$[T] = \begin{array}{c} q \\ | \\ \mathcal{T} \end{array}$$

First, observe that the arenas $[T \to 0]$ and $[T]$ are:

$$\begin{array}{cc} [T] \to [0] & [T] \\ \begin{array}{c} q \\ | \\ q \\ | \\ \mathcal{T} \end{array} & \begin{array}{c} q \\ | \\ \mathcal{T} \end{array} \end{array}$$

so the arena $([T] \to [0]) \parr [T]$, obtained by merging the roots of $[T] \to [0]$ and $[T]$, is actually equal to $[T] \to [T]$:



then $[x : T \vdash [\alpha]\ x : 0 \mid \alpha : T]$ is obtained by composing:

$$[T] \xrightarrow{w_1} [T] \parr [T] \xrightarrow{\nabla} [T] \xrightarrow{r} [0] \parr [T]$$

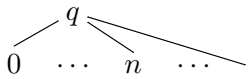
where w_1 is the copycat strategy between $[T]$ on the left and the left copy of $[T]$ in $[T] \parr [T]$ on the right, ∇ merges the plays on $[T] \parr [T]$ into $[T]$, and r is just the identity strategy since $[T] = [0] \parr [T]$ in $\mathcal{G}$. Therefore the composite $[x : T \vdash [\alpha]\ x : 0 \mid \alpha : T] = w_1; \nabla; r$ is the identity strategy on $[T]$ and since the currying in $\mathcal{G}$ is transparent, $[\vdash \lambda x.\, [\alpha]\ x : T \to 0 \mid \alpha : T]$ is the identity strategy on $[T]$. □

4.4.2 Natural numbers, successor and predecessor

In order to interpret μPCF in $\mathcal{G}$, we have to provide an object of continuations in $\mathcal{C}$ for each base type, a morphism in $R^{\mathcal{C}}$ from $\mathbf{1} = R^{\mathbf{0}}$ to $[T]$ for each constant $\mathsf{c} : T$, and we have to verify that all the equations of μPCF are verified in $\mathcal{G}$. We define the family $\mathbf{N} = \{\mathcal{U}_n \mid n \in \mathbb{N}\}$ where $\mathcal{U}$ is the empty arena. The continuation object for the unique base type ι is then:

$$[\![\iota]\!] = R^{\mathbf{N}}$$

By writing out the definition of the exponential in $\mathcal{C}$ and using the isomorphism between $\mathcal{G}$ and $R^{\mathcal{C}}$, we get that $[\iota]$ is the usual flat arena of natural numbers:



We now move to the definition of the interpretations of the constants. In order to define the interpretations of the constants manipulating natural numbers we use the continuation monad on $\mathcal{C}$, mapping an object A to its double exponentiation by R: $\mathbb{T}A = R^{R^A}$. We write $\eta_A : A \to \mathbb{T}A$ and $\mu_A : \mathbb{T}\mathbb{T}A \to \mathbb{T}A$ for the unit and the multiplication. We have:

$$[\iota] = R^{[\![\iota]\!]} = R^{R^{\mathbf{N}}} = \mathbb{T}\mathbf{N}$$

The set of morphisms in $\mathcal{C}$ from $\mathbf{1}$ to $\mathbf{N}$ is isomorphic to the set $\mathbb{N}$ of natural numbers since there is only one morphism in $\mathbf{1} \to \mathbf{1} = \mathbf{1}$ and the index set of $\mathbf{N}$ is $\mathbb{N}$. We write $\varphi_n : \mathbf{1} \to \mathbf{N}$ for the inverse image of $n \in \mathbb{N}$ through this isomorphism. We write $\mathbf{n} = \varphi_n ; \eta_{\mathbf{N}} : \mathbf{1} \to [\iota]$, which is a morphism in $\mathcal{G} \approx R^{\mathcal{C}}$ since $\mathbf{1} = R^{\mathbf{0}} \in Ob\left(R^{\mathcal{C}}\right)$ and $[\iota] = R^{[\![\iota]\!]} \in Ob\left(R^{\mathcal{C}}\right)$. $\mathbf{n}$ is the usual strategy for natural number $n \in \mathbb{N}$:

Lemma 4.17. $\mathbf{n}$ *is the usual strategy for the natural number* n *on* $[\iota]$*: it answers* n *to the unique question of opponent.*

Proof. $\mathbf{n} = \varphi_n ; \eta_{\mathbf{N}}$ is obtained by currying:

$$\begin{array}{ccc}
R^{\mathbf{N}} \times \mathbf{1} & = & \left\{\left(\prod_{i\in\mathbb{N}} \mathcal{V}^{\mathcal{U}_i}\right) \times \mathcal{U}\right\} \\
\downarrow{\scriptstyle \mathbf{Id}_{R^{\mathbf{N}}} \times \varphi_n} & & \downarrow{\scriptstyle \left(*\mapsto n, \mathbf{Id}_{\prod_{i\in\mathbb{N}} \mathcal{V}^{\mathcal{U}_i}} \times \mathbf{Id}_{\mathcal{U}}\right)} \\
R^{\mathbf{N}} \times \mathbf{N} & = & \left\{\left(\prod_{i\in\mathbb{N}} \mathcal{V}^{\mathcal{U}_i}\right) \times \mathcal{U}_j \;\middle|\; j \in \mathbb{N}\right\} \\
 & & \downarrow{\scriptstyle \left(j\mapsto j, \left\{\mathbf{proj}_j \times \mathbf{Id}_{\mathcal{U}_j} \;\middle|\; j\in\mathbb{N}\right\}\right)} \\
\downarrow{\scriptstyle \mathbf{ev}} & & \left\{\mathcal{V}^{\mathcal{U}_j} \times \mathcal{U}_j \;\middle|\; j \in \mathbb{N}\right\} \\
 & & \downarrow{\scriptstyle (j\mapsto *, \{\mathbf{ev} \mid j\in\mathbb{N}\})} \\
R & = & \{\mathcal{V}\}
\end{array}$$

If we follow the path on the right, we get the simple composition:

$$\left(\prod_{i\in\mathbb{N}} \mathcal{V}^{\mathcal{U}_i}\right) \times \mathcal{U} \xrightarrow{\mathbf{proj}_n \times \mathbf{Id}_{\mathcal{U}}} \mathcal{V}^{\mathcal{U}} \times \mathcal{U} \xrightarrow{\quad\mathbf{ev}\quad} \mathcal{V}$$

In $\mathcal{G}$, $\left(\prod_{i\in\mathbb{N}} \mathcal{V}^{\mathcal{U}_i}\right) \times \mathcal{U} = \prod_{i\in\mathbb{N}} \mathcal{V}$, $\mathcal{V}^{\mathcal{U}} \times \mathcal{U} = \mathcal{V}$ and the above composition is just the n-th projection. As a strategy, it therefore answers in the n-th component of $\prod_{i\in\mathbb{N}} \mathcal{V}$ to the question in $\mathcal{V}$, which is the behavior of the usual strategy for the n-th natural number. □

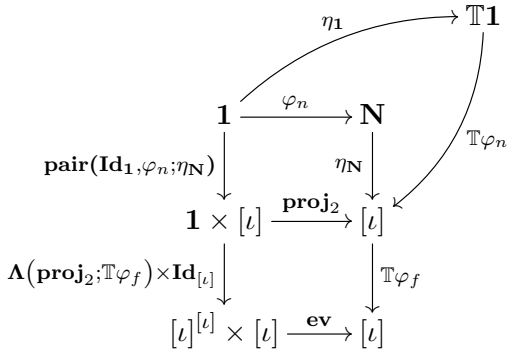
We define:

$$[\overline{\mathsf{n}}] = \mathbf{n}$$

Similarly, the set of morphisms in $\mathcal{C}$ from $\mathbf{N}$ to $\mathbf{N}$ is isomorphic to the set $\mathbb{N}^{\mathbb{N}}$ of functions from natural numbers to natural numbers. We write $\varphi_f : \mathbf{N} \to \mathbf{N}$ for the inverse image of a function $f : \mathbb{N} \to \mathbb{N}$ through this isomorphism, so in particular $\varphi_n; \varphi_f = \varphi_{f(n)}$ for any $n \in \mathbb{N}$. We have $\mathbb{T}\varphi_f : [\iota] \to [\iota]$, which is a morphism in $\mathcal{G} \approx R^{\mathcal{C}}$ since $[\iota] = R^{[\![\iota]\!]} \in Ob\left(R^{\mathcal{C}}\right)$ and we write $\mathbf{f} = \mathbf{\Lambda}\left(\mathbf{proj}_2; \mathbb{T}\varphi_f\right)$ which is a morphism in $\mathcal{G}$ from $\mathbf{1}$ to $[\iota]^{[\iota]}$. Since $\mathbb{T}$ is a monad, we have the following lemma:

Lemma 4.18. *Let $f : \mathbb{N} \to \mathbb{N}$, $n \in \mathbb{N}$ and $m = f(n)$. Then we have $\mathbf{pair}\,(\mathbf{f}, \mathbf{n}); \mathbf{ev} = \mathbf{m}$ in $\mathcal{G} \approx R^{\mathcal{C}}$*

Proof. The following diagram in $\mathcal{C}$ commutes, from the facts that η is a natural transformation and the universal properties of product and exponentiation ($[\iota]^{[\iota]}$ exists since $[\iota]$ is a singleton family):



Since $\varphi_n; \varphi_f = \varphi_{f(n)} = \varphi_m$ we get:

$$\begin{aligned}
\mathbf{pair}\,(\mathbf{f}, \mathbf{n}); \mathbf{ev} &= \mathbf{pair}\left(\mathbf{\Lambda}\left(\mathbf{proj}_2; \mathbb{T}\varphi_f\right), \varphi_n; \eta_{\mathbf{N}}\right); \mathbf{ev} \\
&= \mathbf{pair}\left(\mathbf{Id_1}, \varphi_n; \eta_{\mathbf{N}}\right); \left(\mathbf{\Lambda}\left(\mathbf{proj}_2; \mathbb{T}\varphi_f\right) \times \mathbf{Id}_{[\iota]}\right); \mathbf{ev} \\
&= \eta_{\mathbf{1}}; \mathbb{T}\varphi_n; \mathbb{T}\varphi_f \\
&= \eta_{\mathbf{1}}; \mathbb{T}\left(\varphi_n; \varphi_f\right) \\
&= \eta_{\mathbf{1}}; \mathbb{T}\varphi_{f(n)} \\
&= \eta_{\mathbf{1}}; \mathbb{T}\varphi_m \\
&= \varphi_m; \eta_{\mathbf{N}}
\end{aligned}$$

the last equality coming from the naturality of η. By definition $\mathbf{m} = \varphi_m; \eta_{\mathbf{N}}$ so we have $\mathbf{pair}\,(\mathbf{f}, \mathbf{n}); \mathbf{ev} = \mathbf{m}$ in $\mathcal{C}$, and therefore in $\mathcal{G} \approx R^{\mathcal{C}}$ since $\mathbf{1} = R^{\mathbf{0}} \in Ob\left(R^{\mathcal{C}}\right)$ and $[\iota] = R^{[\![\iota]\!]} \in Ob\left(R^{\mathcal{C}}\right)$. □

We define the interpretations of the predecessor and successor of μPCF as:

$$[\mathsf{pred}] = \mathbf{pred} \qquad\qquad [\mathsf{succ}] = \mathbf{succ}$$

where *pred* and *succ* are the usual predecessor and successor functions on $\mathbb{N}$. The soundness of the defining equations of pred and succ are a consequence of the above lemma, since $[M\,N] = \mathbf{pair}\,([M], [N]); \mathbf{ev}$.

4.4.3 Conditionals

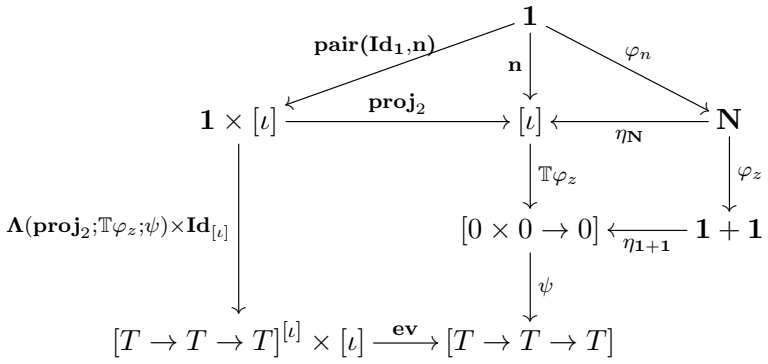
In order to define the interpretation of if_0, we first define a morphism in $\mathcal{G} \approx R^{\mathcal{C}}$ from $[\iota]$ to $R^{R\times R}$ using the continuation monad $\mathbb{T}$, and then we use the fact that $\mathcal{G} \approx R^{\mathcal{C}}$ is a category of continuations to define a morphism from $[0 \times 0 \to 0] = R^{R\times R}$ to $[T \to T \to T]$ by a $\lambda\mu$-term of type $T \to T \to T$ with one free λ-variable of type $0 \times 0 \to 0$. Combining these two morphisms we obtain a morphism from $\mathbf{1}$ to $[\iota \to T \to T \to T]$. First, the set of morphism in $\mathcal{C}$ from $\mathbf{N}$ to $\mathbf{1}+\mathbf{1}$ is isomorphic to the set of set-theoretic functions from $\mathbb{N}$ to the two-element set $\{\blacktriangledown;\blacktriangle\}$. Let z be the set-theoretic function that maps 0 to $\blacktriangledown$ and every $n \neq 0$ to $\blacktriangle$ and φ_z be the corresponding morphism in $\mathcal{C}$ from $\mathbf{N}$ to $\mathbf{1}+\mathbf{1}$. Then $\mathbb{T}\varphi_z$ is a morphism in $\mathcal{G} \approx R^{\mathcal{C}}$ from $[\iota]$ to $R^{\left(R^{\mathbf{1}+\mathbf{1}}\right)} = R^{R\times R} = [0 \times 0 \to 0]$. On the other hand we define:

$$\psi = [x : 0 \times 0 \to 0 \vdash \lambda yz.\mu\alpha.x\,\langle [\alpha]\,y, [\alpha]\,z\rangle : T \to T \to T] : [0 \times 0 \to 0] \to [T \to T \to T]$$

Finally we define:

$$[\mathsf{if}_0] = \mathbf{\Lambda}\left(\mathbf{proj}_2; \mathbb{T}\varphi_z; \psi\right) : \mathbf{1} \to [\iota \to T \to T \to T]$$

In order to prove the soundness of the equations defining if_0, we first observe that the following diagram commutes:



Therefore we have $[\mathsf{if}_0\,\overline{\mathsf{n}}] = \mathbf{pair}\left(\mathbf{\Lambda}\left(\mathbf{proj}_2; \mathbb{T}\varphi_z; \psi\right), \mathbf{n}\right); \mathbf{ev} = \varphi_n; \varphi_z; \eta_{\mathbf{1}+\mathbf{1}}; \psi$. The morphism $\eta_{\mathbf{1}+\mathbf{1}}$ from $\mathbf{1}+\mathbf{1}$ to $R^{\left(R^{\mathbf{1}+\mathbf{1}}\right)} = R^{R\times R} = [0 \times 0 \to 0]$ can be described as:

$$\eta_{\mathbf{1}+\mathbf{1}} = \left(\begin{array}{c}\{\blacktriangledown;\blacktriangle\} \longrightarrow \{*\} \\ \_ \longmapsto *\end{array}, \left\{\begin{array}{l}\sigma_{\blacktriangledown} = [\lambda x.\mathfrak{p}_1\,x] : \mathbf{1} \to [0 \times 0 \to 0] \\ \sigma_{\blacktriangle} = [\lambda x.\mathfrak{p}_2\,x] : \mathbf{1} \to [0 \times 0 \to 0]\end{array}\right\}\right) : \mathbf{1}+\mathbf{1} \to R^{\left(R^{\mathbf{1}+\mathbf{1}}\right)}$$

The morphisms $\varphi_0; \varphi_z$ and $\varphi_{n+1}; \varphi_z$ are by definition of z:

$$\varphi_0; \varphi_z = \left(\begin{array}{c}\{*\} \longrightarrow \{\blacktriangledown;\blacktriangle\} \\ * \longmapsto \blacktriangledown\end{array}, \{\{\epsilon\} : \mathbf{1} \to \mathbf{1}\}\right) : \mathbf{1} \to \mathbf{1}+\mathbf{1}$$

$$\varphi_{n+1}; \varphi_z = \left(\begin{array}{c}\{*\} \longrightarrow \{\blacktriangledown;\blacktriangle\} \\ * \longmapsto \blacktriangle\end{array}, \{\{\epsilon\} : \mathbf{1} \to \mathbf{1}\}\right) : \mathbf{1} \to \mathbf{1}+\mathbf{1}$$

Therefore we have:

$$\varphi_0; \varphi_z; \eta_{\mathbf{1}+\mathbf{1}} = [\lambda x.\mathfrak{p}_1\,x] : \mathbf{1} \to R^{R\times R} \qquad \text{and} \qquad \varphi_{n+1}; \varphi_z; \eta_{\mathbf{1}+\mathbf{1}} = [\lambda x.\mathfrak{p}_2\,x] : \mathbf{1} \to R^{R\times R}$$

so:

$$\left[\mathsf{if}_0\,\overline{0}\right] = \varphi_0; \varphi_z; \eta_{\mathbf{1}+\mathbf{1}}; \psi$$

$$
\begin{aligned}
&= [\lambda x.\mathfrak{p}_1\, x]\,; [x : 0 \times 0 \to 0 \vdash \lambda yz.\mu\alpha.x\,\langle [\alpha]\ y, [\alpha]\ z\rangle : T \to T \to T] \\
&= [\lambda yz.\mu\alpha.\,(\lambda x.\mathfrak{p}_1\, x)\,\langle [\alpha]\ y, [\alpha]\ z\rangle] \\
&= [\lambda yz.\mu\alpha.\mathfrak{p}_1\,\langle [\alpha]\ y, [\alpha]\ z\rangle] \\
&= [\lambda yz.\mu\alpha.\,[\alpha]\ y] \\
&= [\lambda yz.y]
\end{aligned}
$$

and similarly $\left[\mathsf{if}_0\,\overline{\mathsf{n}+\mathsf{1}}\right] = [\lambda yz.z]$. Therefore we get $\left[\mathsf{if}_0\,\overline{\mathsf{0}}\,M\,N\right] = [M]$ and $\left[\mathsf{if}_0\,\overline{\mathsf{n}+\mathsf{1}}\,M\,N\right] = [N]$.

4.4.4 Fixed point operators

In order to define the interpretation of the fixed point operators of μPCF, we first prove that for any strategy σ on the arena $\mathcal{A} \to \mathcal{A}$ there exists a strategy $\overline{\sigma}$ on $\mathcal{A}$ such that $\overline{\sigma}; \sigma = \overline{\sigma}$. Then we perform this construction on a well-chosen σ to get a general fixed point operator. Let σ be a strategy on the arena $\mathcal{A} \to \mathcal{A}$. We define a sequence $(\overline{\sigma}_n)_{n\in\mathbb{N}}$ by induction:

$$\overline{\sigma}_0 = \{\epsilon\} \qquad\qquad \overline{\sigma}_{n+1} = \overline{\sigma}_n; \sigma$$

First, we prove that this sequence is increasing:

Lemma 4.19. *For any $n \in \mathbb{N}$, $\overline{\sigma}_n \subseteq \overline{\sigma}_{n+1}$*

Proof. We write $\mathcal{A}^{(1)} \to \mathcal{A}^{(1)}$ for the arena of σ. We prove the result by induction on n:

- $n = 0$: $\overline{\sigma}_0 = \{\epsilon\}$ and the empty play ϵ is in any strategy.
- $n + 1$: Let $s \in \overline{\sigma}_{n+1} = \overline{\sigma}_n; \sigma$. By definition of composition of strategies, there exists some $t \in \sigma$ such that $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}_n$ and $t_{|\mathcal{A}^{(2)}} = s$. By induction hypothesis we then have $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}_{n+1}$, and therefore $s = t_{|\mathcal{A}^{(2)}} \in \overline{\sigma}_{n+2} = \overline{\sigma}_{n+1}$.

□

We then define:

$$\overline{\sigma} \stackrel{\Delta}{=} \bigcup_{n\in\mathbb{N}} \overline{\sigma}_n$$

Lemma 4.20. *$\overline{\sigma}$ is a strategy on $\mathcal{A}$*

Proof. Since the $\overline{\sigma}_n$ are strategies, it is easy to check that $\overline{\sigma}$ is a P-prefix-closed set of finite P-plays. If $sa, sb \in \overline{\sigma}$ there are $m, n \in \mathbb{N}$ such that $sa \in \overline{\sigma}_m$ and $sb \in \overline{\sigma}_n$, therefore $sa, sb \in \overline{\sigma}_{\max(m,n)}$ and $a = b$. If $s \in \overline{\sigma}$, then $s \in \overline{\sigma}_n$ for some $n \in \mathbb{N}$, so $\mathrm{Threads}\,(s) \subseteq \overline{\sigma}_n \subseteq \overline{\sigma}$. Conversely, if s is a finite P-play on $\mathcal{A}$ and if $\mathrm{Threads}\,(s) \subseteq \overline{\sigma}$, then since s is finite, $\mathrm{Threads}\,(s)$ is finite and since $(\overline{\sigma}_n)_{n\in\mathbb{N}}$ is increasing, there is some $n \in \mathbb{N}$ such that $\mathrm{Threads}\,(s) \subseteq \overline{\sigma}_n$ and therefore $s \in \overline{\sigma}_n \subseteq \overline{\sigma}$. □

We prove now that it defines a fixed point of σ.

Lemma 4.21. *$\overline{\sigma}; \sigma = \overline{\sigma}$*

Proof. We prove the double inclusion. Let $s \in \overline{\sigma}; \sigma$. By definition of composition of strategies, there exists $t \in \sigma$ such that $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}$ and $t_{|\mathcal{A}^{(2)}} = s$. By definition of $\overline{\sigma}$, there is some $n \in \mathbb{N}$ such that $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}_n$. Then we get $s \in \overline{\sigma}_n; \sigma = \overline{\sigma}_{n+1} \subseteq \overline{\sigma}$. For the other direction, let now $s \in \overline{\sigma}$. By definition of $\overline{\sigma}$, there is some $n \in \mathbb{N}$ such that $s \in \overline{\sigma}_n$. By monotonicity of $(\overline{\sigma}_n)_{n\in\mathbb{N}}$, we get $s \in \overline{\sigma}_{n+1} = \overline{\sigma}_n; \sigma$, so by definition of composition there is some $t \in \sigma$ such that $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}_n$ and $t_{|\mathcal{A}^{(2)}} = s$. But then $t_{|\mathcal{A}^{(1)}} \in \overline{\sigma}$, and so $s = t_{|\mathcal{A}^{(2)}} \in \overline{\sigma}; \sigma$. □

We now define the general fixed point operator:

$$[\mathsf{Y}] = \overline{[\lambda xy.y\,(x\,y)]}$$

By the above lemma, we have:

$$[\mathsf{Y}] = [\lambda xy.y\,(x\,y)]\,[\mathsf{Y}] = [\lambda y.y\,(\mathsf{Y}\,y)]$$

And therefore the defining equation of Y is verified in $\mathcal{G}$:

$$[\mathsf{Y}\,M] = [\lambda y.y\,(\mathsf{Y}\,y)\,M] = [M\,(\mathsf{Y}\,M)]$$

Chapter 5

An orthogonality-based realizability model for classical analysis

In this chapter, we define a realizability model which is based on the duality between realizers and counter-realizers. After defining the realizability relation, we prove its adequacy for first-order logic, Peano arithmetic and the axiom of choice. Finally, we use our orthogonality-based model to extract computational content from classical proofs.

5.1 The realizability relation

5.1.1 Negative translation and orthogonality

The first realizability models for classical logic were obtained by combining Gödel's negative translation with intuitionistic realizability, see e.g. [Koh08]. Gödel's negative translation from Peano arithmetic $PA^\omega_=$ (equivalent to PA^ω) to $HA^\omega_=$ (see section 2.4.2) maps a formula A to $A^\neg$ by prefixing inductively all the positive connectives and atomic predicates of A with a double negation. We have the following result: if $PA^\omega_= \vdash\!\!\sim A$, then $HA^\omega_= \vdash\!\!\sim A^\neg$, relying on the fact that for every axiom A of $PA^\omega_=$, $HA^\omega_= \vdash\!\!\sim A \Rightarrow A^\neg$. Therefore, a realizability model for $PA^\omega_=$ can be obtained from a realizability model for $HA^\omega_=$ using Gödel's negative translation. Concerning the extraction of witnesses, if $PA^\omega_= \vdash\!\!\sim \exists x^\iota\,(t = \mathsf{0})$, then $HA^\omega_= \vdash\!\!\sim \neg\neg\exists x^\iota\,\neg\neg\,(t = \mathsf{0})$ from which we easily get $HA^\omega_= \vdash\!\!\sim \neg\neg\exists x^\iota\,(t = \mathsf{0})$. While in usual intuitionistic realizability the formula $\bot$ has no realizer so the model is sound, Friedman's trick is to allow $\bot$ to have realizers. If we then take the realizers of $\bot$ to be the same as those of $\exists x^\iota\,(t = \mathsf{0})$, then combining the proof of $\neg\neg\exists x^\iota\,(t = \mathsf{0})$ with the identity gives a realizer of $\exists x^\iota\,(t = \mathsf{0})$, and therefore the witness.

In Krivine's [Kri09] classical realizability models this double step is avoided through the use of orthogonality in system F. In these models there is a set of terms Λ and a set of stacks Π. Each formula has a set of realizers (the truth value, subset of Λ) and a set of counter-realizers (the falsity value, subset of Π). The falsity values are primitive, an orthogonality relation is defined between Λ and Π, and the truth values are defined as the orthogonals of the falsity values, so they are orthogonally closed.

Here we work in a typed setting so we must choose the types of realizers and counter-realizers so they can interact. Recall from section 4.4.1 that:

$$\mathcal{C} = \mathrm{Fam}\,(\mathcal{G}) \qquad R = \{\mathcal{V}\} \qquad \mathcal{G} \simeq R^{\mathcal{C}} = R^{\mathrm{Fam}(\mathcal{G})}$$

Given a formula A, the set of realizers of A would normally be a set of morphisms in $\mathcal{G}$ from the terminal object $\mathbf{1}$ to the interpretation of A: $[A^*] = R^{[\![A^*]\!]}$, and under the duality between

terms and contexts of $\lambda\mu$-calculus, a natural choice for the counter-realizers of A is a set of morphisms in $\mathcal{C}$ from $\mathbf{1}$ to $[\![A^*]\!]$. Then we can combine a potential realizer of A with a potential counter-realizer using the evaluation morphism $\mathbf{ev} : R^{[\![A^*]\!]} \times [\![A^*]\!] \to R$ so we obtain a morphism from $\mathbf{1}$ to R. However, since in $\mathcal{G}$ there is only one such morphism (the empty strategy), it is not possible to check whether the potential realizer and counter-realizer are orthogonal to each other by observing this composite. The solution adopted here is to rely on Friedman's trick directly in the definition of the realizability relation, and to define an orthogonality relation relying on an artificially added output channel.

Formally we add a μ-variable in the process of interpreting logic in $\lambda\mu$-calculus. A proof:

$$\frac{\pi}{A_1, ..., A_n \vdash A \mid B_1, ..., B_m}$$

is translated to a $\lambda\mu$-term:

$$x_1 : {A_1}^*, ..., x_n : {A_n}^* \vdash \pi^* : A^* \mid \alpha_1 : {B_1}^*, ..., \alpha_n : {B_m}^*, \kappa : \varsigma$$

where ς is some fixed base sort, by applying the (admissible) rule of right weakening of $\lambda\mu$-calculus. The μ-variable κ is, intuitively, a continuation variable which can be used by a realizer to stop computation and give an answer. Apart from its use in the definition of the realizability relation, this feature will also be used in the proof of the extraction result. After translating the proof π to a $\lambda\mu$-term π^*, we interpret it in $\mathcal{G}$ as a morphism:

$$[\pi^*] \in \mathcal{G}\left([{A_1}^*] \times ... \times [{A_n}^*], [A^*] \parr [{B_1}^*] \parr ... \parr [{B_m}^*] \parr [\varsigma]\right)$$

In the particular case of empty contexts, $[\pi^*]$ is a morphism in $\mathcal{G}\left(\mathbf{1}, [A^*] \parr [\varsigma]\right)$, and we therefore choose the potential realizers of a closed formula A to be such morphisms. As explained in section 3.2.3 and by taking the convention that these morphisms have a free μ-variable κ of type ς, we use the syntax of $\lambda\mu$-calculus (and possibly $[\kappa]$ and $\mu\kappa$) to manipulate these. We also substitute morphisms of $\mathcal{C}$ for μ-variables, as in section 3.2.5, so if $\sigma \in \mathcal{G}\left(\mathbf{1}, [A^*] \parr [\varsigma]\right)$ is a potential realizer, and if $\phi \in \mathcal{C}\left([\![\varsigma]\!], [\![A^*]\!]\right)$, then $[\phi]\ \sigma \in \mathcal{G}\left(\mathbf{1}, [\bot^*] \parr [\varsigma]\right)$ and $\mu\kappa.\,[\phi]\ \sigma \in \mathcal{G}\left(\mathbf{1}, [\varsigma]\right)$, that is a strategy on the flat arena for the set $\varsigma^{\mathcal{M}}$. Since the arena $[\varsigma]$ is (in the general case) larger than the one-move arena $[0] = R$, we can now define when ϕ is orthogonal to σ by looking at $\mu\kappa.\,[\phi]\ \sigma$. Potential counter-realizers of A are therefore chosen to be morphisms in $\mathcal{C}\left([\![\varsigma]\!], [\![A^*]\!]\right)$.

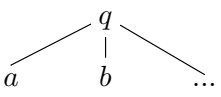
In [BR13] the goals of orthogonally-defined realizability and extraction were achieved by taking realizers of a formula A to be interpretations of closed terms of type $[\![\varsigma]\!] \to [A^*]$. However, this choice implied that realizers were passing a continuation k to each other, leading to an unnecessary complex interpretation.

5.1.2 Truth values, falsity values

We fix a first-order signature Σ and a Σ-structure $\mathcal{M}$ such that $\iota^{\mathcal{M}}$ is countable for each base sort ι. We also fix a corresponding $\lambda\mu$ signature and a type P^* in this signature for each predicate P of Σ, as in section 3.1.3. The interpretation of $\lambda\mu$-calculus in the category of continuations $\mathcal{G}$ (see section 4.4.1) is as in section 3.2.3, the base types ι of $\lambda\mu$-calculus being interpreted in $\mathcal{G}$ as the arenas:

$$[\![\iota]\!] \triangleq R^{\{\mathcal{U}_a \mid a \in \iota^{\mathcal{M}}\}}$$

where $\mathcal{U}$ is the empty arena, so $[\iota] = R^{[\![\iota]\!]}$ is the flat arena corresponding to the set $\iota^{\mathcal{M}}$:



and we suppose given for each constant $\mathsf{c} : T \in \mathcal{C}st$ a strategy interpreting it:

$$\mathsf{c} : \mathbf{1} \to [T]$$

Since as stated in section 3.2.3 we use the syntax of $\lambda\mu$-calculus to describe strategies of $\mathcal{G}$, we will omit the interpretation brackets. All the $\lambda\mu$-terms that we write from now on are to be understood as morphisms in $\mathcal{G}$.

In order to build our realizability relation by orthogonality, and later on to perform extraction on Π^0_2 formulas using Friedman's trick, our model is parameterized with a set

$$\perp\!\!\!\perp \subseteq \varsigma^{\mathcal{M}}$$

which is, intuitively, the set of "correct" values that can be output through the variable κ. We define the corresponding strategies:

$$|\perp\!\!\!\perp| = \{\mathbf{a} \mid a \in \perp\!\!\!\perp\} \subseteq \mathcal{G}(\mathbf{1}, [\varsigma])$$

where $\mathbf{a}$ is the strategy which answers a to the unique question of opponent.

We now define the set of realizers of a formula, that we call its truth value. We fix for each predicate P of Σ and each $a_1, ..., a_n \in {T_1}^{\mathcal{M}} \times ... \times {T_n}^{\mathcal{M}}$ a set of morphisms $|P(a_1, ..., a_n)| \subseteq \mathcal{G}(\mathbf{1}, [P^*] \parr [\varsigma])$. We extend this to every closed formula A on Σ with parameters in $\mathcal{M}$ by:

$$|A \wedge B| = \{\sigma \mid \mathfrak{p}_1\, \sigma \in |A| \wedge \mathfrak{p}_2\, \sigma \in |B|\} \qquad \left|\forall x^T\, A\right| = \bigcap_{a \in T^{\mathcal{M}}} |A\{a/x\}|$$

$$|A \Rightarrow B| = \{\sigma \mid \forall \tau \in |A|, \sigma\, \tau \in |B|\} \qquad |\bot| = \{\sigma \mid \mu\kappa.\sigma \in |\perp\!\!\!\perp|\}$$

so $|A| \subseteq \mathcal{G}(\mathbf{1}, [A^*] \parr [\varsigma])$. Remark that contrary to [Kri09], we do not define the truth values as the orthogonals of the falsity values. Since our logic is first-order and our basic logical connectives are negative, the truth value of every formula is already orthogonally closed. However, in the adequacy lemma we give a realizability interpretation for sequents $A_1, ..., A_n \vdash A \mid B_1, ..., B_m$, in which the $\mid$ and the "," between the B_i are to be interpreted as disjunctions. Since in categories of continuations, disjunction between A and B is interpreted as $A \parr B = R^{A \times B}$ that is a negated object, we define its realizability interpretation by orthogonality. That is why we define for every formula A with parameters the falsity value of A: $\|A\| \subseteq \mathcal{C}(\llbracket\varsigma\rrbracket, \llbracket A^*\rrbracket)$ (where $\mathcal{C} = \mathrm{Fam}(\mathcal{G})$).

We suppose given for each predicate P of Σ and each $a_1, ..., a_n \in {T_1}^{\mathcal{M}} \times ... \times {T_n}^{\mathcal{M}}$ a set of morphisms $\|P(a_1, ..., a_n)\| \subseteq \mathcal{C}(\llbracket\varsigma\rrbracket, \llbracket P^*\rrbracket)$ and we extend it to every formula A with parameters in $\mathcal{M}$ by:

$$\begin{aligned}
\|A \wedge B\| &= \{\mathsf{in}_1\, \phi \mid \phi \in \|A\|\} \cup \{\mathsf{in}_2\, \phi \mid \phi \in \|B\|\} \\
\|A \Rightarrow B\| &= \{\langle \widetilde{\sigma}, \phi \rangle \mid \sigma \in |A| \wedge \phi \in \|B\|\} \\
\left\|\forall x^T\, A\right\| &= \bigcup_{a \in T^{\mathcal{M}}} \|A\{a/x\}\| \\
\|\bot\| &= \{*\}
\end{aligned}$$

so $\|A\| \subseteq \mathcal{C}(\llbracket\varsigma\rrbracket, \llbracket A^*\rrbracket)$. As explained in section 3.2.4, we use here the syntax of $\lambda^{R\times+}$ to manipulate morphisms in $\mathcal{C}$. We define now an orthogonality relation between $\mathcal{G}(\mathbf{1}, [A^*] \parr [\varsigma])$ and $\mathcal{C}(\llbracket\varsigma\rrbracket, \llbracket A^*\rrbracket)$: if $\sigma \in \mathcal{G}(\mathbf{1}, [A^*] \parr [\varsigma])$ and $\phi \in \mathcal{C}(\llbracket\varsigma\rrbracket, \llbracket A^*\rrbracket)$, then $\mu\kappa.[\phi]\, \sigma \in \mathcal{G}(\mathbf{1}, [\varsigma])$, so we define:

$$\sigma \perp \phi \quad \stackrel{\Delta}{=} \quad \mu\kappa.[\phi]\, \sigma \in |\perp\!\!\!\perp|$$

The following lemma states that the truth values are indeed the orthogonals of the falsity values defined above:

Lemma 5.1. *Suppose that for every predicate P of Σ and every $a_1, ..., a_n \in {T_1}^{\mathcal{M}} \times ... \times {T_n}^{\mathcal{M}}$, $|P(a_1, ..., a_n)| = \{\sigma \,|\, \forall \phi \in \|P(a_1, ..., a_n)\|,\ \sigma \perp \phi\}$. Then for every closed formula A with parameters in $\mathcal{M}$:*

$$|A| = \{\sigma \,|\, \forall \phi \in \|A\|,\ \sigma \perp \phi\}$$

Proof. We prove this result by induction on the structure of the formula:

- $P(a_1, ..., a_n)$: this is an assumption of the lemma
- $\bot$: if $\sigma \in \mathcal{G}(\mathbf{1}, [\bot^*] \parr [\varsigma])$ then:

  $$\sigma \in |\bot| \Leftrightarrow \mu\kappa.\sigma \in |\perp\!\!\!\perp| \Leftrightarrow \mu\kappa.\,[*]\ \sigma \in |\perp\!\!\!\perp| \Leftrightarrow \sigma \perp *$$

  from which we conclude since $\|\bot\| = \{*\}$.
- $A \wedge B$: if $\sigma \in \mathcal{G}(\mathbf{1}, ([A^*] \times [B^*]) \parr [\varsigma])$, $\phi_1 \in \mathcal{C}([\![\varsigma]\!], [\![A^*]\!])$ and $\phi_2 \in \mathcal{C}([\![\varsigma]\!], [\![B^*]\!])$, then $\mu\kappa.\,[\mathsf{in}_i\, \phi_i]\ \sigma = \mu\kappa.\,[\phi_i]\ \mathfrak{p}_i\, \sigma$, therefore:

  $$\mathfrak{p}_i\, \sigma \perp \phi_i \Leftrightarrow \mu\kappa.\,[\phi_i]\ \mathfrak{p}_i\, \sigma \in |\perp\!\!\!\perp| \Leftrightarrow \mu\kappa.\,[\mathsf{in}_i\, \phi_i]\ \sigma \in |\perp\!\!\!\perp| \Leftrightarrow \sigma \perp \mathsf{in}_i\, \phi_i$$

  and finally:

  $$\begin{aligned}
  \sigma \in |A \wedge B| &\Leftrightarrow \begin{cases} \mathfrak{p}_1\, \sigma \in |A| \\ \mathfrak{p}_2\, \sigma \in |B| \end{cases} \\
  &\Leftrightarrow \begin{cases} \forall \phi_1 \in \|A\|, \mathfrak{p}_1\, \sigma \perp \phi_1 \\ \forall \phi_2 \in \|B\|, \mathfrak{p}_2\, \sigma \perp \phi_2 \end{cases} \text{ by induction hypothesis} \\
  &\Leftrightarrow \begin{cases} \forall \phi_1 \in \|A\|, \sigma \perp \mathsf{in}_1\, \phi_1 \\ \forall \phi_2 \in \|B\|, \sigma \perp \mathsf{in}_2\, \phi_2 \end{cases} \\
  &\Leftrightarrow \forall \phi \in \|A \wedge B\|, \sigma \perp \phi
  \end{aligned}$$

- $A \Rightarrow B$: if $\sigma \in \mathcal{G}\left(\mathbf{1}, [B^*]^{[A^*]} \parr [\varsigma]\right)$, $\tau \in \mathcal{G}(\mathbf{1}, [A^*] \parr [\varsigma])$ and $\phi \in \mathcal{C}([\![\varsigma]\!], [\![B^*]\!])$, then $\mu\kappa.\,[\langle \widetilde{\tau}, \phi \rangle]\ \sigma = [\phi]\ \sigma\, \tau$, therefore:

  $$\sigma\, \tau \perp \phi \Leftrightarrow \mu\kappa.\,[\phi]\ \sigma\, \tau \in |\perp\!\!\!\perp| \Leftrightarrow \mu\kappa.\,[\langle \widetilde{\tau}, \phi \rangle]\ \sigma \in |\perp\!\!\!\perp| \Leftrightarrow \sigma \perp \langle \widetilde{\tau}, \phi \rangle$$

  and finally:

  $$\begin{aligned}
  \sigma \in |A \Rightarrow B| &\Leftrightarrow \forall \tau \in |A|, \sigma\, \tau \in |B| \\
  &\Leftrightarrow \forall \tau \in |A|, \forall \phi \in \|B\|, \sigma\, \tau \perp \phi \qquad \text{by induction hypothesis} \\
  &\Leftrightarrow \forall \tau \in |A|, \forall a \in \|B\|, \sigma \perp \langle \widetilde{\tau}, a \rangle \\
  &\Leftrightarrow \forall \phi' \in \|A \Rightarrow B\|, \tau \perp \phi'
  \end{aligned}$$

- $\forall x^T\, A$: if $\sigma \in \mathcal{G}(\mathbf{1}, [A^*] \parr [\varsigma])$, then:

  $$\begin{aligned}
  \sigma \in \left|\forall x^T\, A\right| &\Leftrightarrow \forall a \in T^{\mathcal{M}}, \sigma \in |A\{a/x\}| \\
  &\Leftrightarrow \forall a \in T^{\mathcal{M}}, \forall \phi \in \|A\{a/x\}\|, \sigma \perp \phi \\
  &\Leftrightarrow \forall \phi \in \mathcal{C}([\![\varsigma]\!], [\![A^*]\!]), \left(\exists a \in T^{\mathcal{M}}, \phi \in \|A\{a/x\}\|\right) \Rightarrow \sigma \perp \phi \\
  &\Leftrightarrow \forall \phi \in \left\|\forall x^T\, A\right\|, \sigma \perp \phi
  \end{aligned}$$

□

5.2 Adequacy

5.2.1 Adequacy for first-order logic

We now have all the necessary material to give the adequacy lemma. For that we suppose that the interpretations of the terms associated to the axioms are realizers of these axioms: for every $A \in \mathcal{A}x$, $M_A \in |A|$. The adequacy lemma is then:

Lemma 5.2. *Suppose π is a proof of $\Gamma \vdash A \mid \Delta$ with $FV(\Gamma, A, \Delta) \subseteq \vec{y}^{\vec{T}}$, so:*

$$\vec{x} : \Gamma^* \vdash \pi^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

then for any $\vec{a} \in \vec{T}^{\mathcal{M}}, \vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|, \vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$, we have:

$$\pi^* \left\{ \vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha} \right\} \in |A\{\vec{a}/\vec{y}\}|$$

In particular if A is a closed formula, $\pi^* \in |A|$

Proof. By induction on the proof tree:

- $\pi = \dfrac{}{\Gamma, A \vdash A \mid \Delta}$:

  $$\pi^* = \vec{x} : \Gamma^*, z : A^* \vdash z : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

  Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$, $\tau \in |A\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. Then we have:

  $$\pi^* \left\{ \vec{\sigma}/\vec{x}, \tau/z, \vec{\phi}/\vec{\alpha} \right\} = \tau \in |A\{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{}{\Gamma \vdash A \mid \Delta}\ {\scriptstyle (A \in \mathcal{A}x)}$:

  $$\pi^* = \vec{x} : \Gamma^* \vdash M_A : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

  Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. Then we have:

  $$\pi^* \left\{ \vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha} \right\} = M_A \in |A| = |A\{\vec{a}/\vec{y}\}|$$

  since A is closed and $M_A \in |A|$ by assumption.

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma, A \vdash B \mid \Delta}}{\Gamma \vdash A \Rightarrow B \mid \Delta}$:

  $$\pi^* = \vec{x} : \Gamma^* \vdash \lambda x.{\pi_0}^* : A^* \to B^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

  Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. Then for any $\tau \in |A\{\vec{a}/\vec{y}\}|$ we have:

  $$\left(\pi^* \left\{ \vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha} \right\} \right) \tau = {\pi_0}^* \left\{ \vec{\sigma}/\vec{x}, \tau/y, \vec{\phi}/\vec{\alpha} \right\} \in |B\{\vec{a}/\vec{y}\}|$$

  by induction hypothesis, since $\tau \in |A\{\vec{a}/\vec{y}\}|$. Therefore:

  $$\pi^* \left\{ \vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha} \right\} \in |(A \Rightarrow B)\{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_1}{\Gamma \vdash A \Rightarrow B \mid \Delta} \quad \dfrac{\pi_2}{\Gamma \vdash A \mid \Delta}}{\Gamma \vdash B \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash {\pi_1}^* \, {\pi_2}^* : B^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma \{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta \{\vec{a}/\vec{y}\}\|$. Then we have:

$$\pi^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} = \left({\pi_1}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right) \left({\pi_2}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right) \in |B \{\vec{a}/\vec{y}\}|$$

since by induction hypothesis:

$${\pi_1}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |(A \Rightarrow B) \{\vec{a}/\vec{y}\}| \text{ and } {\pi_2}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |A \{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_1}{\Gamma \vdash A \mid \Delta} \quad \dfrac{\pi_2}{\Gamma \vdash B \mid \Delta}}{\Gamma \vdash A \wedge B \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash \langle {\pi_1}^*, {\pi_2}^* \rangle : A^* \times B^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma \{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta \{\vec{a}/\vec{y}\}\|$. Then we have:

$$\pi^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} = \left\langle {\pi_1}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}, {\pi_2}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \right\rangle \in |(A \wedge B) \{\vec{a}/\vec{y}\}|$$

since by induction hypothesis:

$${\pi_1}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |A \{\vec{a}/\vec{y}\}| \text{ and } {\pi_2}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |B \{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash A \wedge B \mid \Delta}}{\Gamma \vdash A \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash \mathfrak{p}_1 \, {\pi_0}^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma \{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta \{\vec{a}/\vec{y}\}\|$. Then we have:

$$\pi^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} = \mathfrak{p}_1 \left({\pi_0}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right) \in |A \{\vec{a}/\vec{y}\}|$$

since by induction hypothesis:

$${\pi_0}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |(A \wedge B) \{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash A \wedge B \mid \Delta}}{\Gamma \vdash B \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash \mathfrak{p}_2 \, {\pi_0}^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma \{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta \{\vec{a}/\vec{y}\}\|$. Then we have:

$$\pi^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} = \mathfrak{p}_2 \left({\pi_0}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right) \in |B \{\vec{a}/\vec{y}\}|$$

since by induction hypothesis:

$${\pi_0}^* \left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |(A \wedge B) \{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash A \mid \Delta}}{\Gamma \vdash \forall z^U A \mid \Delta}\left(z^U \notin \mathrm{FV}(\Gamma, \Delta)\right)$:

$$\pi^* = \vec{x} : \Gamma^* \vdash {\pi_0}^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. Then for any $b \in U^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}, b/z\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}, b/z\}\|$ (since $z^U \notin \mathrm{FV}(\Gamma, \Delta)$), so by induction hypothesis:

$${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |A\{\vec{a}/\vec{y}, b/z\}|$$

Therefore $\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in \left|\left(\forall z^U A\right)\{\vec{a}/\vec{y}\}\right|$.

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash \forall z^U A \mid \Delta}}{\Gamma \vdash A\left\{t^U/z^U\right\} \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash {\pi_0}^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. By induction hypothesis:

$${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in \bigcap_{b \in U^{\mathcal{M}}} |A\{\vec{a}/\vec{y}, b/z\}|$$

so taking $b = (t\{\vec{a}/\vec{y}\})^{\mathcal{M}} \in U^{\mathcal{M}}$ we get:

$${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in \left|A\left\{\vec{a}/\vec{y}, (t\{\vec{a}/\vec{y}\})^{\mathcal{M}}/z\right\}\right|$$

and since $\pi^* = {\pi_0}^*$ and $A\left\{\vec{a}/\vec{y}, (t\{\vec{a}/\vec{y}\})^{\mathcal{M}}/z\right\} = A\{t/z\}\{\vec{a}/\vec{y}\}$ we get:

$$\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |A\{t/z\}\{\vec{a}/\vec{y}\}|$$

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash A \mid \Delta, A}}{\Gamma \vdash \bot \mid \Delta, A}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash [\beta]\ {\pi_0}^* : 0 \mid \vec{\alpha} : \Delta^*, \beta : A^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$, $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$ and $\psi \in \|A\{\vec{a}/\vec{y}\}\|$. Then we have:

$$\begin{aligned}\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\} &= ([\beta]\ {\pi_0}^*)\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\} \\ &= [\psi]\left({\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\}\right) \in |\bot| = |\bot\{\vec{a}/\vec{y}\}|\end{aligned}$$

since by induction hypothesis:

$${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\} \in |A\{\vec{a}/\vec{y}\}|$$

so ${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\}\ \bot\ \psi$ by lemma 5.1, and $\mu\kappa.\,[\psi]\left({\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\}\right) \in |\perp\!\!\!\perp|$.

- $\pi = \dfrac{\dfrac{\pi_0}{\Gamma \vdash \bot \mid \Delta, A}}{\Gamma \vdash A \mid \Delta}$:

$$\pi^* = \vec{x} : \Gamma^* \vdash \mu\beta.{\pi_0}^* : A^* \mid \vec{\alpha} : \Delta^*, \kappa : \varsigma$$

Let $\vec{a} \in \vec{T}^{\mathcal{M}}$, $\vec{\sigma} \in |\Gamma\{\vec{a}/\vec{y}\}|$ and $\vec{\phi} \in \|\Delta\{\vec{a}/\vec{y}\}\|$. Then for any $\psi \in \|A\{\vec{a}/\vec{y}\}\|$ we have:

$$\mu\kappa.[\psi]\left(\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right) = \mu\kappa.\left([\psi]\ \mu\beta.{\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\right)$$
$$= \mu\kappa.\left({\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\}\right) \in |\perp\!\!\!\perp|$$

since by induction hypothesis, ${\pi_0}^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}, \psi/\beta\right\} \in |\bot\{\vec{a}/\vec{y}\}| = |\bot|$. Therefore:

$$\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\}\ \bot\ \psi$$

and so by lemma 5.1:

$$\pi^*\left\{\vec{\sigma}/\vec{x}, \vec{\phi}/\vec{\alpha}\right\} \in |A\{\vec{a}/\vec{y}\}|$$

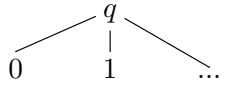
□

5.2.2 Adequacy for Peano arithmetic

We fix now the theory to be $PA^{\omega \mathrm{r}}$ on $\Sigma_{PA^{\omega \mathrm{r}}}$, and the structure $\mathcal{M}$ to be a model of $PA^{\omega \mathrm{r}}$. The predicates are interpreted in $\mathcal{M}$ as in section 2.6, $\mathcal{M}$ is required to verify $\iota^{\mathcal{M}} \simeq \mathbb{N}$ and the individual constants $\mathsf{0}$ and S are supposed to be interpreted in $\iota^{\mathcal{M}}$ accordingly. The $\lambda\mu$ signature is that of μPCF, we interpret the two predicates as in section 3.1.3:

$$\neq_T{}^* \ \triangleq\ 0 \qquad (\!|.^{\iota}|\!)^* \ \triangleq\ \iota$$

and the axioms as in section 3.4.1. μPCF is interpreted in $\mathcal{G}$ as in section 4.4, so $[\iota]$ is, as in section 4.4.2, the usual flat arena of natural numbers:



Since $PA^{\omega \mathrm{r}}$ has only one base sort ι, we fix $\varsigma = \iota$.

The realizability values for the predicates are:

$$|a \neq_T b| \triangleq \begin{cases} |\bot| & \text{if } a = b \\ \mathcal{G}\,(\mathbf{1}, [0] \parr [\iota]) & \text{otherwise} \end{cases} \qquad |(\!|n^{\iota}|\!)| \triangleq \{\overline{\mathsf{n}}\}$$

since n is a natural number. The falsity value of $\neq$ is:

$$\|a \neq_T b\| \triangleq \begin{cases} \{*_{[\![\iota]\!]}\} & \text{if } a = b \\ \emptyset & \text{otherwise} \end{cases}$$

and it is easy to verify that the hypotheses of lemma 5.1 are verified for the inequality predicate. The falsity value of $(\!|.|\!)$ cannot be defined since it is a positive predicate, however since $\|A \Rightarrow B\|$ depends on $|A|$ but not on $\|A\|$, we can still define the falsity values of formulas in which the relativization predicate does not appears at the rightmost position. We observe that in the translations of proofs from PA^{ω} to $PA^{\omega \mathrm{r}}$, the relativization predicate never appears at the rightmost position of a formula in Δ, and that in the adequacy lemma the falsity values are necessary only for the μ-context. Therefore, the falsity value of $(\!|.|\!)$ is superfluous when interpreting PA^{ω}.

First, all equalities which are true in the model are trivially realized:

Lemma 5.3. *Let t^T and u^T be first-order terms with $FV(t,u) = \vec{x}^{\vec{U}}$.*

$$\textit{If } \mathcal{M} \vDash \forall \vec{x}^{\vec{U}}\,(t = u) \;\; \textit{then } \lambda x.x \in \left|\forall \vec{x}^{\vec{U}}\,(t = u)\right|$$

Proof. Let $\vec{a} \in \vec{U}^{\mathcal{M}}$. Since $\mathcal{M} \vDash \forall \vec{x}^{\vec{U}}\,(t = u)$, we have $(t\,\{\vec{a}/\vec{x}\})^{\mathcal{M}} = (u\,\{\vec{a}/\vec{x}\})^{\mathcal{M}}$, so:

$$|t\,\{\vec{a}/\vec{x}\} \neq u\,\{\vec{a}/\vec{x}\}| = |\bot|$$

Therefore, for any $\sigma \in |t\,\{\vec{a}/\vec{x}\} \neq u\,\{\vec{a}/\vec{x}\}|$:

$$\lambda x.x\sigma = \sigma \in |t\,\{\vec{a}/\vec{x}\} \neq u\,\{\vec{a}/\vec{x}\}| = |\bot|$$

and so:

$$\lambda x.x \in \left|\forall \vec{x}^{\vec{U}}\,(t \neq u \Rightarrow \bot)\right| = \left|\forall \vec{x}^{\vec{U}}\,(t = u)\right|$$

□

Therefore, since $\mathcal{M}$ is a model of $PA^{\omega \mathrm{r}}$ we have immediately the following results:

$$\begin{aligned}
M_{(\boldsymbol{refl})} &= \lambda x.x \in \left|\forall x^T\,(x =_T x)\right| \\
M_{(\boldsymbol{\Delta s})} &= \lambda x.x \in \left|\forall x^{T\to U\to V}\forall y^{T\to U}\forall z^T\,(\mathsf{s}\,x\,y\,z =_V x\,z\,(y\,z))\right| \\
M_{(\boldsymbol{\Delta k})} &= \lambda x.x \in \left|\forall x^T\forall y^U\,(\mathsf{k}\,x\,y =_T x)\right| \\
M_{(\boldsymbol{\Delta rec0})} &= \lambda x.x \in \left|\forall x^T\forall y^{\iota\to T\to T}\,(\mathsf{rec}\,x\,y\,\mathsf{0} =_T x)\right| \\
M_{(\boldsymbol{\Delta recS})} &= \lambda x.x \in \left|\forall x^T\forall y^{\iota\to T\to T}\forall z^\iota\,(\mathsf{rec}\,x\,y\,(\mathsf{S}\,z) =_T y\,z\,(\mathsf{rec}\,x\,y\,z))\right|
\end{aligned}$$

The non-confusion axiom and Leibniz scheme are easy:

Lemma 5.4.

$$\begin{aligned}
M_{(\boldsymbol{Snz})} &= \Omega \in |\forall x^\iota\,(\mathsf{S}\,x \neq_\iota \mathsf{0})| \\
M_{(\boldsymbol{Leib})} &= \lambda x.x \in \left|\forall \vec{z}^{\vec{U}}\,\forall x^T\,\forall y^T\,(\neg A \Rightarrow A\,\{y/x\} \Rightarrow x \neq_T y)\right|
\end{aligned}$$

Proof. Let $n \in \iota^{\mathcal{M}}$, we have $(\mathsf{S}\,n)^{\mathcal{M}} = n + 1 \neq 0 = \mathsf{0}^{\mathcal{M}}$, so $|\mathsf{S}\,n \neq \mathsf{0}| = \mathcal{G}\,(\mathbf{1}, [0] \parr [\iota])$, so any strategy in $\mathcal{G}\,(\mathbf{1}, [0] \parr [\iota])$ realizes $\forall x^\iota\,(\mathsf{S}\,x \neq_\iota \mathsf{0})$. Let now $\vec{a} \in \vec{U}^{\mathcal{M}}$, $b, c \in T^{\mathcal{M}}$, $\sigma \in |\neg A\,\{b/x, \vec{a}/\vec{z}\}|$ and $\tau \in |A\,\{c/x, \vec{a}/\vec{z}\}|$. If $b \neq c$, then $|b \neq c| = \mathcal{G}\,(\mathbf{1}, [0] \parr [\iota])$ and therefore $\lambda x.x\,\sigma\,\tau \in |b \neq c|$. Otherwise, $b = c$ so $|A\,\{b/x, \vec{a}/\vec{z}\}| = |A\,\{c/x, \vec{a}/\vec{z}\}|$ and by definition of $|A \Rightarrow B|$ we get $\sigma\,\tau \in |\bot|$. Moreover, since $b = c$, $|b \neq c| = |\bot|$ and $\sigma\,\tau \in |b \neq c|$. Finally we get $M_{(\boldsymbol{Leib})} \in \left|\forall \vec{z}^{\vec{U}}\,\forall x^T\,\forall y^T\,(\neg A \Rightarrow A\,\{y/x\} \Rightarrow x \neq_T y)\right|$. □

The adequacy for the induction axiom scheme is as follows:

Lemma 5.5.

$$M_{(\boldsymbol{ind^r})} = \mathsf{rec} \in \left|\forall \vec{y}^{\vec{U}}\,(A\,\{\mathsf{0}/x\} \Rightarrow \forall^{\mathrm{r}} x^\iota\,(A \Rightarrow A\,\{\mathsf{S}\,x/x\}) \Rightarrow \forall^{\mathrm{r}} x^\iota A)\right|$$

Proof. Let $\vec{a} \in \vec{U}^{\mathcal{M}}$, $\sigma \in |A\,\{\mathsf{0}/x, \vec{a}/\vec{y}\}|$, $\tau \in |\forall^{\mathrm{r}} x^\iota\,(A\,\{\vec{a}/\vec{y}\} \Rightarrow A\,\{\mathsf{S}\,x/x, \vec{a}/\vec{y}\})|$. Since $\iota^{\mathcal{M}} \simeq \mathbb{N}$, we can use induction to prove that for any $n \in \iota^{\mathcal{M}}$, $\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}} = \mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}} \in |A\,\{n/x, \vec{a}/\vec{y}\}|$:

- $n = 0$: $\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{0}} = \sigma \in |A\,\{\mathsf{0}/x, \vec{a}/\vec{y}\}|$

- $n+1$: $\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}+\mathsf{1}} = \mathsf{rec}\,\sigma\,\tau\,\mathsf{succ}\,\overline{\mathsf{n}} = \tau\,\overline{\mathsf{n}}\,(\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}})$. Since $\overline{\mathsf{n}} \in |(\!|n|\!)|$, we get $\tau\,\overline{\mathsf{n}} \in |A\{n/x, \vec{a}/\vec{y}\} \Rightarrow A\{\mathsf{S}\,n/x, \vec{a}/\vec{y}\}|$, and since by induction hypothesis we have $\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}} \in |A\{n/x, \vec{a}/\vec{y}\}|$ we get $\tau\,\overline{\mathsf{n}}\,(\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}}) \in |A\{\mathsf{S}\,n/x, \vec{a}/\vec{y}\}|$. Finally, since $(\mathsf{S}\,n)^{\mathcal{M}} = n+1$, $\mathsf{rec}\,\sigma\,\tau\,\overline{\mathsf{n}+\mathsf{1}} \in |A\{n+1/x, \vec{a}/\vec{y}\}|$.

□

The last axioms are the relativization ones:

Lemma 5.6.

$$M_{(\!|\mathsf{0}|\!)} = \overline{\mathsf{0}} \Vdash (\!|\mathsf{0}|\!) \qquad M_{(\!|\mathsf{S}|\!)} = \mathsf{succ} \Vdash (\!|\mathsf{S}|\!)$$

Proof. The first one is immediate, since $|(\!|\mathsf{0}|\!)| = \{\overline{\mathsf{0}}\}$. For the second one, remember that:

$$(\!|\mathsf{S}|\!) \equiv \forall^{\mathrm{r}} x^{\iota} (\!|\mathsf{S}\,x|\!) \equiv \forall x^{\iota}\,((\!|x|\!) \Rightarrow (\!|\mathsf{S}\,x|\!))$$

let $n \in \iota^{\mathcal{M}}$, we have to prove $\mathsf{succ}\,\overline{\mathsf{n}} \in |(\!|\mathsf{S}\,n|\!)|$. Since $(\mathsf{S}\,n)^{\mathcal{M}} = n+1$, this amounts to $\mathsf{succ}\,\overline{\mathsf{n}} \in |(\!|n+1|\!)|$, which is true since $\mathsf{succ}\,\overline{\mathsf{n}} = \overline{\mathsf{n}+\mathsf{1}}$. □

5.2.3 Adequacy for the axiom of choice

When it comes to the axiom of countable choice AC, the usual route of negative translation followed by intuitionistic realizability becomes much more difficult. Indeed, $\mathrm{AC} \Rightarrow \mathrm{AC}^{\neg}$ is not provable in intuitionistic logic, therefore the path described in section 5.1.1 cannot be followed as-is and a realizer of $\mathrm{AC}^{\neg}$ must be provided. In [BBC98], a variant of bar recursion was used to realize $\mathrm{AC}^{\neg}$, while in [BO05] the principle of double negation shift $\forall x\,\neg\neg A \Rightarrow \neg\neg\forall x\,A$ was realized using bar recursion (see 3.5). With this principle, it becomes possible to derive $\mathrm{AC} \Rightarrow \mathrm{AC}^{\neg}$ in intuitionistic logic, and since AC is realized in intuitionistic models by the identity, one obtains a realizer of $\mathrm{AC}^{\neg}$.

We follow here a different approach, since our realizability model is for classical logic, and we prove that bar recursion realizes the axiom of dependent choice ($\boldsymbol{DC}^{r}$) in our classical model. The exact connection with the negative translation + intuitionistic realizability technique has not been investigated. The negative translation of proofs corresponds to the continuation-passing-style translation on terms and the semantics of $\lambda\mu$-calculus in a category of continuations $R^{\mathcal{C}}$ corresponds to the semantics of its CPS-translation into λ-calculus in the cartesian "R-closed" category $\mathcal{C}$ as stated in section 3.2.4, so in order to compare more closely our model with the usual negative translation + intuitionistic realizability technique, one would need to define a realizability relation for intuitionistic logic in the category $\mathcal{C} = \mathrm{Fam}(\mathcal{G})$. Therefore, it seems more natural to work directly in the category of continuations $\mathcal{G}$ which has built-in control features, and in which the usual arena of natural numbers can be seen as a negated object $R^{(R^{\mathbf{N}})}$ (see section 4.4.2).

Interpreting dependent choice using bar recursion

We use here the bar-recursion operator defined in section 3.5.4 to provide the term $M_{(\boldsymbol{DC}^{r})}$ interpreting the axiom of dependent choice. First, remark that the derivation of $(\boldsymbol{DC})^{\mathrm{r}}$ from ($\boldsymbol{DC}^{r}$) given in section 2.5.2 doesn't involve relativization predicates at the rightmost position in formulas of Δ, so it is still correct to have no falsity value for $(\!|.|\!)$. For technical reasons, we actually use a slight modification of this bar recursor and define:

$$\mathsf{barrec}' \stackrel{\Delta}{=} \lambda wuv.\mathsf{Y}\,(\lambda zy.\mu\alpha.\,[\alpha]\ v\,(y \mathbin{@} \langle w, \mu\beta.\,[\alpha]\ u\,y\,(\lambda x.z\,(y * x))\rangle))$$

which can be typed with:

$$\mathsf{barrec}' : T \to ((T \times U)^{\diamond} \to (T \times U \to V) \to V) \to ((\iota \to T \times U) \to V) \to (T \times U)^{\diamond} \to V$$

and which satisfies the equation:

$$\mathsf{barrec}'\, Q\, M\, N\, P = \mu\alpha.\, [\alpha]\ N\left(P \mathbin{@} \left\langle Q, \mu\beta.\, [\alpha]\ M\, P\left(\lambda x.\mathsf{barrec}'\, Q\, M\, N\, (P * x)\right)\right\rangle\right)$$

Remember that $(\boldsymbol{DC}^{r})$ is:

$$\forall \vec{u}^{\vec{U}} \forall^{\mathrm{r}} v^{T}\left(\forall^{\mathrm{r}} x^{\iota} \forall^{\mathrm{r}} y^{T} \neg \forall z^{T} \neg\left(\llparenthesis z \rrparenthesis \wedge A^{\mathrm{r}}\right) \Rightarrow \neg \forall y^{\iota \to T} \neg\left(\llparenthesis y\, 0 \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota}\left(\llparenthesis y\,(\mathsf{S}\, x) \rrparenthesis \wedge A^{\mathrm{r}}\left\{y\, x/y, y\,(\mathsf{S}\, x)\,/z\right\}\right)\right)\right)$$

so if we write $B \equiv \llparenthesis z \rrparenthesis \wedge A^{\mathrm{r}}$ it becomes:

$$\forall \vec{u}^{\vec{U}} \forall^{\mathrm{r}} v^{T}\left(\forall^{\mathrm{r}} x^{\iota}\, \forall^{\mathrm{r}} y^{T}\, \neg \forall z^{T}\, \neg B \Rightarrow \neg \forall y^{\iota \to T}\, \neg\left(\llparenthesis y\, 0 \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota} B\left\{y\, x/y, y\,(\mathsf{S}\, x)\,/z\right\}\right)\right)$$

and $(\boldsymbol{DC}^{r})^{*}$ is:

$$M_{(\boldsymbol{DC}^{r})} : T \to (\iota \to T \to (B^{*} \to 0) \to 0) \to (T \times (\iota \to B^{*}) \to 0) \to 0$$

In order to define $M_{(\boldsymbol{DC}^{r})}$ we make an informal reasoning. Suppose $Q : T$ is a witness of $\llparenthesis v \rrparenthesis$, $M : \iota \to T \to (B^{*} \to 0) \to 0$ is a witness of:

$$\forall^{\mathrm{r}} x^{\iota} \forall^{\mathrm{r}} y^{T}\left(\forall z^{T}\,(B \Rightarrow \bot) \Rightarrow \bot\right)$$

and $N : T \times (\iota \to B^{*}) \to 0$ is a witness of:

$$\forall y^{\iota \to T}\left(\llparenthesis y\, 0 \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota} B\left\{y\, x/y, y\,(\mathsf{S}\, x)\,/z\right\} \Rightarrow \bot\right)$$

We want to build from this, using barrec', a witness of $\bot$. We will use the following instance of barrec':

$$\mathsf{barrec}' : T \to (B^{*\diamond} \to (B^{*} \to 0) \to 0) \to ((\iota \to B^{*}) \to 0) \to (T \times B^{*})^{\diamond} \to 0$$

As an aside remark, since in this instance the μ-variable α is of type 0, the following equation holds:

$$\mathsf{barrec}' = \lambda w u v y.v\left(y \mathbin{@} \left\langle w, \mu\beta.u\, y\left(\lambda x.\mathsf{barrec}'\, w\, u\, v\,(y * x)\right)\right\rangle\right)$$

The first argument is without surprise Q. The idea now is that barrec' will build a sequence of witnesses of $B\left\{y\, x/y, y\,(\mathsf{S}\, x)\,/z\right\}$. The second argument represents the recursive step. If we have an element $y : B^{\diamond}$ which represents the sequence of witnesses already computed, then M computes the next element of y, given its last element. Here we have two cases, either $|y| = \overline{0}$, in which case we must initialize the sequence with the first element Q, or the last element of y is $y\,(\mathsf{pred}\ |y|)$. Therefore, we provide M with $\mathsf{if}_0\ |y|\ Q\,(\mathfrak{p}_1\, y\,(\mathsf{pred}\ |y|))$ (we keep it informal and do not explain here the $\mathfrak{p}_1$). We also give to M the current step which is $|y|$ so the second argument is:

$$\lambda y.M\ |y|\ \mathsf{if}_0\ |y|\ Q\,(\mathfrak{p}_1\, y\,(\mathsf{pred}\ |y|))$$

The third argument represents the behavior if we have an infinite sequence of witnesses $y : \iota \to B$. We provide N with the first witness, that is Q, and the infinite sequence y, so the third argument is:

$$\lambda y.N\,\langle Q, y\rangle$$

Finally, the last argument of barrec' is the initial sequence of witnesses, that is the empty sequence ϵ. We have then:

$$\mathsf{barrec}'\, Q\,(\lambda y.M\ |y|\ \mathsf{if}_0\ |y|\ w\,(\mathfrak{p}_1\, y\,(\mathsf{pred}\ |y|))\,(\lambda y.N\,\langle Q, y\rangle)\,\epsilon : 0$$

The interpretation of $(\boldsymbol{DC}^{r})$ is therefore defined as:

$$M_{(\boldsymbol{DC}^{r})} \stackrel{\Delta}{=} \lambda w u v.\mathsf{barrec}'\, w\,(\lambda y.u\ |y|\ \mathsf{if}_0\ |y|\ w\,(\mathfrak{p}_1\, y\,(\mathsf{pred}\ |y|))\,(\lambda y.v\,\langle w, y\rangle)\,\epsilon$$

Adequacy

We suppose now that $\mathcal{M}$ satisfies $(\boldsymbol{DC})$, and we use the variant of the bar recursion operator defined above to realize the axiom of dependent choice $(\boldsymbol{DC}^{\mathrm{r}})$.

First, we prove that the continuity requirement of section 3.5.3 holds in $\mathcal{G}$:

Lemma 5.7. *Let $\sigma \in \mathcal{G}\,(\mathbf{1}, [(\iota \to T) \to 0] \parr [\iota])$ and $\tau \in \mathcal{G}\,(\mathbf{1}, [\iota \to T] \parr [\iota])$ such that $\mu\kappa.\sigma\,\tau = \overline{\mathsf{n}}$. There exists $m \in \mathbb{N}$ such that for any $\tau' \in \mathcal{G}\,(\mathbf{1}, [\iota \to T] \parr [\iota])$:*

$$\left(\forall m' < m,\, \tau'\,\overline{\mathsf{m}'} = \tau\,\overline{\mathsf{m}'}\right) \Rightarrow \mu\kappa.\sigma\,\tau' = \mu\kappa.\sigma\,\tau = \overline{\mathsf{n}}$$

Proof. $\mu\kappa.\sigma\,\tau = \overline{\mathsf{n}}$ means that there is a play $s \in \sigma$ in $\left([\iota]^{(1)} \to [T]\right) \to [\iota]^{(2)}$ such that $s_{|\left([\iota]^{(1)} \to [T]\right) \parr [\iota]^{(2)}} \in \tau$ and $s_{|[\iota]^{(2)}} = qn$. Since s is a finite sequence, there is some $m \in \mathbb{N}$ such that all the moves m' in $s_{|[\iota]^{(1)}}$ verify $m' < m$. Now, if $\tau' \in \mathcal{G}\,(\mathbf{1}, [\iota \to T] \parr [\iota])$ is such that $\forall m' < m,\, \tau'\,\overline{\mathsf{m}'} = \tau\,\overline{\mathsf{m}'}$, then $s_{|[\iota]^{(1)} \to [T] \parr [\iota]^{(2)}}$ is also a play of τ', and therefore $qn \in \mu\kappa.\sigma\,\tau'$ and $\mu\kappa.\sigma\,\tau' = \overline{\mathsf{n}}$. □

We also prove that in $\mathcal{G}$ we can build a strategy from an infinite sequence of strategies:

Lemma 5.8. *Let $(\sigma_n)_{n\in\mathbb{N}}$ be a sequence of strategies in an arena $[T]$. There exists a strategy σ in the arena $[\iota] \to [T]$ such that for any $n \in \mathbb{N}$, $\sigma\,\overline{\mathsf{n}} = \sigma_n$.*

Proof. The view s of the strategy σ are such that $s_{|[\iota]} = qn$ and $s_{|[T]} \in \sigma_n$ for $n \in \mathbb{N}$. It is easy to see then that for any $n \in \mathbb{N}$, $\sigma\,\overline{\mathsf{n}} = \sigma_n$. □

Using these two results on $\mathcal{G}$, we now prove that the interpretation of $(\boldsymbol{DC}^{\mathrm{r}})$ realizes $(\boldsymbol{DC}^{\mathrm{r}})$:

Lemma 5.9. *Let $\perp\!\!\!\perp \subseteq \iota^{\mathcal{M}}$.*

$$\begin{aligned} M_{(\boldsymbol{DC}^{\mathrm{r}})} &= \lambda wuv.\mathsf{barrec}'\,w\,(\lambda y.u\ |y|\ \mathsf{if}_0\ |y|\ w\,(\mathfrak{p}_1\,y\,\wr\mathsf{pred}\ |y|\wr))\,(\lambda y.v\,\langle w, y\rangle)\,\epsilon \\ &\in \left|\forall \vec{u}^{\vec{U}} \forall^{\mathrm{r}} v^{T}\,\left(\forall^{\mathrm{r}} x^{\iota}\,\forall^{\mathrm{r}} y^{T}\,\neg\forall z^{T}\,\neg B \Rightarrow \neg\forall y^{\iota\to T}\,\neg\left(\langle\!|y\,0|\!\rangle \wedge \forall^{\mathrm{r}} x^{\iota} B\,\{y\,x/y, y\,(\mathsf{S}\,x)\,/z\}\right)\right)\right| \end{aligned}$$

where $B \equiv \langle\!|z|\!\rangle \wedge A^{\mathrm{r}}$

Proof. Recall that we use the bar recursor:

$$\mathsf{barrec}' : T \to (B^{*\diamond} \to (B^* \to 0) \to 0) \to ((\iota \to B^*) \to 0) \to (T \times B^*)^{\diamond} \to 0$$

and $M_{(\boldsymbol{DC}^{\mathrm{r}})}$ has type $(\boldsymbol{DC}^{\mathrm{r}})^*$:

$$M_{(\boldsymbol{DC}^{\mathrm{r}})} : T \to (\iota \to T \to (B^* \to 0) \to 0) \to (T \times (\iota \to B^*) \to 0) \to 0$$

Let now $\vec{a} \in \vec{U}^{\mathcal{M}}$ and $b \in T^{\mathcal{M}}$ and write $A' \equiv A^{\mathrm{r}}\,\{\vec{a}/\vec{u}, b/v\}$ and $B' \equiv B\,\{\vec{a}/\vec{u}, b/v\}$. Let:

$$\sigma \in \mathcal{G}\,(\mathbf{1}, [T] \parr [\iota]) \qquad \tau \in \mathcal{G}\,(\mathbf{1}, [\iota \to T \to (B^* \to 0) \to 0] \parr [\iota])$$
$$\nu \in \mathcal{G}\,(\mathbf{1}, [T \times (\iota \to B^*) \to 0] \parr [\iota])$$

be such that:

$$\sigma \in |\langle\!|b|\!\rangle| \qquad \tau \in \left|\forall^{\mathrm{r}} x^{\iota}\forall^{\mathrm{r}} y^{T}\,\neg\forall z^{T}\,\neg B'\right| \qquad \nu \in \left|\forall y^{\iota\to T}\,\neg\left(\langle\!|y\,0|\!\rangle \wedge \forall^{\mathrm{r}} x^{\iota}\,B'\,\{y\,x/y, y\,(\mathsf{S}\,x)\,/z\}\right)\right|$$

We then have to prove:

$$\mathsf{barrec}'\,\nu\,(\lambda y.\tau\ |y|\ \mathsf{if}_0\ |y|\ \nu\,(\mathfrak{p}_1\,y\,\wr\mathsf{pred}\ |y|\wr))\,(\lambda y.\sigma\,\langle w, y\rangle)\,\epsilon \in |\bot|$$

We introduce more notations:

$$\tau' = \lambda y.\tau\ |y|\,(\mathsf{if}_0\,|y|\ \sigma\,(\mathfrak{p}_1\,y\,\wr\mathsf{pred}\ |y|\wr)) \in \mathcal{G}\,(\mathbf{1}, [B^{*\diamond} \to (B^* \to 0) \to 0] \parr [\iota])$$
$$\nu' = \lambda y.\nu\ \langle \sigma, y\rangle \in \mathcal{G}\,(\mathbf{1}, [(\iota \to B^*) \to 0] \parr \iota)$$
$$\psi = \mathsf{barrec}'\,\sigma\,\tau'\,\nu' \in \mathcal{G}\,(\mathbf{1}, [B^{*\diamond} \to 0] \parr [\iota])$$

so we must prove that $\psi\,\epsilon \in |\bot|$. With our notations, for any $\rho \in \mathcal{G}\,(\mathbf{1}, [B^{*\diamond}] \parr [\iota])$ we have:

$$\begin{aligned}\psi\,\rho &= \mathsf{barrec}'\,\sigma\,\tau'\,\nu'\,\rho\\ &= \nu'\left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau'\,\rho\left(\lambda x.\mathsf{barrec}'\,\sigma\,\tau'\,\nu'\,(\rho * x)\right)\right\rangle\right)\\ &= \nu'\left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau'\,\rho\,(\lambda x.\psi\,(\rho * x))\right\rangle\right)\end{aligned}$$

The following iteration lemma (the proof of which is deferred to the end of the section) is the heart of the adequacy:

Lemma 5.10. *Let $c_0, ..., c_n \in T^{\mathcal{M}}$ and $\theta_0, ..., \theta_{n-1} \in \mathcal{G}\,(\mathbf{1}, [B^*] \parr [\iota])$ be such that:*

$$c_0 = b \qquad \forall 0 \leq i < n,\ \theta_i \in \left|B'\,\{i/x, c_i/y, c_{i+1}/z\}\right| \qquad \psi\,(\theta_0 * ... * \theta_{n-1}) \notin |\bot|$$

then there exists $c_{n+1} \in T^{\mathcal{M}}$ and $\theta_n \in \mathcal{G}\,(\mathbf{1}, [B^*] \parr [\iota])$ such that:

$$\theta_n \in \left|B'\,\{n/x, c_n/y, c_{n+1}/z\}\right| \qquad \psi\,(\theta_0 * ... * \theta_{n-1} * \theta_n) \notin |\bot|$$

In order to prove $\psi\,\epsilon \in |\bot|$, we use a reductio-ad-absurdum and suppose $\psi\,\epsilon \notin |\bot|$. By iterating the lemma, we build sequences $(c_n)_{n\in\mathbb{N}}$ in $T^{\mathcal{M}}$ and $(\theta_n)_{n\in\mathbb{N}}$ in $\mathcal{G}\,(\mathbf{1}, [B^*] \parr [\iota])$ such that:

$$c_0 = b \qquad \forall n \in \mathbb{N},\ \theta_n \in \left|B'\,\{n/x, c_n/y, c_{n+1}/z\}\right| \text{ and } \psi\,(\theta_0 * ... * \theta_{n-1}) \notin |\bot|$$

Since $\mathcal{M}$ satisfies (***DC***), we can build $c \in (\iota \to T)^{\mathcal{M}}$ such that for every $n \in \iota^{\mathcal{M}} \simeq \mathbb{N}$, $c\,n = c_n$, and using lemma 5.8 we also build $\theta \in \mathcal{G}\,(\mathbf{1}, [\iota \to B^*] \parr [\iota])$ such that for every $n \in \mathbb{N}$, $\theta\,\overline{\mathsf{n}} = \theta_n$. We now prove that:

$$\theta \in \left|\forall^{\mathrm{r}} x^{\iota}\ B'\,\{c\,x/y, c\,(\mathsf{S}\,x)\,/z\}\right|$$

Indeed, let $n \in \mathbb{N} \simeq \iota^{\mathcal{M}}$, since $|(\!|n|\!)| = \{\overline{\mathsf{n}}\}$, we must prove $\theta\,\overline{\mathsf{n}} \in |B'\,\{n/x, c\,n/y, c\,(\mathsf{S}\,n)\,/z\}|$, but since $\theta\,\overline{\mathsf{n}} = \theta_n$, $c\,n = c_n$ and $c\,(\mathsf{S}\,n) = c_{n+1}$, this is immediate. Now, since:

$$\nu \in \left|\forall y^{\iota\to T} \neg\left((\!|y\,\mathsf{0}|\!) \wedge \forall^{\mathrm{r}} x^{\iota}\ B'\,\{y\,x/y, y\,(\mathsf{S}\,x)\,/z\}\right)\right| \subseteq \left|\neg\left((\!|c\,\mathsf{0}|\!) \wedge \forall^{\mathrm{r}} x^{\iota}\ B'\,\{c\,x/y, c\,(\mathsf{S}\,x)\,/z\}\right)\right|$$

and $\sigma \in |(\!|b|\!)| = |(\!|c\,\mathsf{0}|\!)|$, we get easily:

$$\nu' = \lambda y.\nu\,\langle \sigma, y\rangle \in \left|\neg\forall^{\mathrm{r}} x^{\iota}\ B'\,\{c\,x/y, c\,(\mathsf{S}\,x)\,/z\}\right|$$

and therefore $\nu'\,\theta \in |\bot|$, so $\mu\kappa.\nu'\,\theta \in |\perp\!\!\!\perp|$. By lemma 5.7, there is some $m \in \mathbb{N}$ such that any strategy $\theta' \in \mathcal{G}\,(\mathbf{1}, [\iota \to B^*] \parr [\iota])$ which satisfies:

$$\forall m' < m, \theta'\,\overline{\mathsf{m'}} = \theta\,\overline{\mathsf{m'}}$$

is such that $\mu\kappa.\nu'\,\theta' = \mu\kappa.\nu'\,\theta \in |\perp\!\!\!\perp|$. If we write $\rho = \theta_0 * ... * \theta_{m-1}$, this is verified in particular for:

$$\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau'\,\rho\,(\lambda x.\psi\,(\rho * x))\right\rangle$$

so we get:

$$\mu\kappa.\nu'\left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau'\,\rho\,(\lambda x.\psi\,(\rho * x))\right\rangle\right) \in |\perp\!\!\!\perp|$$

and finally:

$$\psi\,(\theta_0 * ... * \theta_{m-1}) = \nu'\left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau'\,\rho\,(\lambda x.\psi\,(\rho * x))\right\rangle\right) \in |\bot|$$

from which we get our contradiction. □

Here is the proof of the iteration lemma:

Proof. Write $\rho = \theta_0 * ... * \theta_{n-1}$. We have:

$$\begin{aligned}\nu \left\langle \sigma, \rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau' \, \rho \lambda x.\psi \left(\rho * x\right)\right\rangle\right\rangle &= \nu' \left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right)\right\rangle\right) \\ &= \psi \, \rho \\ &= \psi \left(\theta_0 * ... * \theta_{n-1}\right) \notin |\bot|\end{aligned}$$

Using $\mathsf{k}^{\mathcal{M}}$, $\mathsf{s}^{\mathcal{M}}$ and $\mathsf{rec}^{\mathcal{M}}$ we can build some $c \in (\iota \to T)^{\mathcal{M}}$ such that for any $0 \leq i \leq n$, $c \, i = c_i$, and for any $i > n$, $c \, i = b$. Since $\nu \in \left|\forall y^{\iota \to T} \neg \left(\llparenthesis y \, \mathsf{0} \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota} \, B' \left\{y \, x / y, y \left(\mathsf{S} \, x\right) / z\right\}\right)\right|$, we have in particular:

$$\nu \in \left|\neg \left(\llparenthesis c \, \mathsf{0} \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota} \, B' \left\{c \, x / y, c \left(\mathsf{S} \, x\right) / z\right\}\right)\right|$$

therefore:

$$\left\langle \sigma, \rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right)\right\rangle\right\rangle \notin \left|\llparenthesis c \, \mathsf{0} \rrparenthesis \wedge \forall^{\mathrm{r}} x^{\iota} \, B' \left\{c \, x / y, c \left(\mathsf{S} \, x\right) / z\right\}\right|$$

and since $\sigma \in |\llparenthesis b \rrparenthesis| = |\llparenthesis c_0 \rrparenthesis| = |\llparenthesis c \, \mathsf{0} \rrparenthesis|$, we get:

$$\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right)\right\rangle \notin \left|\forall^{\mathrm{r}} x^{\iota} \, B' \left\{c \, x / y, c \left(\mathsf{S} \, x\right) / z\right\}\right|$$

so there exists $i \in \mathbb{N} \simeq \iota^{\mathcal{M}}$ such that:

$$\left(\rho \mathbin{@} \left\langle \sigma, \mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right)\right\rangle\right) \, \overline{\mathsf{i}} \notin \left|B' \left\{i / x, c \, i / y, c \left(\mathsf{S} \, i\right) / z\right\}\right|$$

If $i < n$ then we get:

$$\rho \, \wr \overline{\mathsf{i}} \wr = \theta_i \notin \left|B' \left\{i / x, c_i / y, c_{i+1} / z\right\}\right|$$

which contradicts the hypothesis of the lemma. Therefore, $i \geq n$ and:

$$\left\langle \sigma, \mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right)\right\rangle \notin \left|B' \left\{i / x, c \, i / y, c \left(\mathsf{S} \, i\right) / z\right\}\right| = \left|\llparenthesis c \left(\mathsf{S} \, i\right) \rrparenthesis \wedge A' \left\{i / x, c \, i / y, c \left(\mathsf{S} \, i\right) / z\right\}\right|$$

Since again $\sigma \in |\llparenthesis b \rrparenthesis| = |\llparenthesis c \left(\mathsf{S} \, i\right) \rrparenthesis|$, we get:

$$\mu\beta.\tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right) \notin \left|A' \left\{i / x, b \, i / y, b \left(\mathsf{S} \, i\right) / z\right\}\right|$$

Since $\lambda x.\mu\beta.x$ is the interpretation in $\lambda\mu$-calculus of a proof of $\bot \Rightarrow A^{\mathrm{r}}$, we get by the adequacy lemma:

$$\tau \ |\rho| \left(\mathsf{if}_{\mathsf{0}} \, |\rho| \ \sigma \left(\mathfrak{p}_1 \, \rho \, \wr \mathsf{pred} \ |\rho| \wr\right)\right) \left(\lambda x.\psi \left(\rho * x\right)\right) = \tau' \, \rho \left(\lambda x.\psi \left(\rho * x\right)\right) \notin |\bot|$$

first, since $\rho = \theta_0 * ... * \theta_{n-1}$, $|\rho| = \overline{\mathsf{n}} \in |\llparenthesis n \rrparenthesis|$, so $\tau \ |\rho| \in \left|\forall^{\mathrm{r}} y^{T} \neg \forall z^{T} \neg B' \left\{n / x\right\}\right|$. We prove now that $\mathsf{if}_{\mathsf{0}} \, |\rho| \ \sigma \left(\mathfrak{p}_1 \, \rho \, \wr \mathsf{pred} \ |\rho| \wr\right) \in |\llparenthesis c_n \rrparenthesis|$ by distinguishing cases:

- $n = 0$: $\mathsf{if}_{\mathsf{0}} \, |\rho| \ \sigma \left(\mathfrak{p}_1 \, \rho \, \wr \mathsf{pred} \ |\rho| \wr\right) = \sigma \in |\llparenthesis b \rrparenthesis| = |\llparenthesis c_0 \rrparenthesis|$
- $n \neq 0$: $\mathsf{if}_{\mathsf{0}} \, |\rho| \ \sigma \left(\mathfrak{p}_1 \, \rho \, \wr \mathsf{pred} \ |\rho| \wr\right) = \mathfrak{p}_1 \, \theta_{n-1} \in |\llparenthesis c_n \rrparenthesis|$ since:

$$\theta_{n-1} \in \left|B' \left\{n - 1 / x, c_{n-1} / y, c_n / z\right\}\right| = \left|\llparenthesis c_n \rrparenthesis \wedge A' \left\{n - 1 / x, c_{n-1} / y, c_n / z\right\}\right|$$

Therefore we have:

$$\tau \ |\rho| \left(\mathsf{if}_{\mathsf{0}} \, |\rho| \ \sigma \left(\mathfrak{p}_1 \, \rho \, \wr \mathsf{pred} \ |\rho| \wr\right)\right) \in \left|\neg \forall z^{T} \neg B' \left\{n / x, c_n / y\right\}\right|$$

and so:

$$\lambda x.\psi \left(\rho * x\right) \notin \left|\forall z^{T} \neg B' \left\{n / x, c_n / y\right\}\right|$$

which means that there exists some $c_{n+1} \in T^{\mathcal{M}}$ such that:

$$\lambda x.\psi \left(\rho * x\right) \notin \left|\neg B' \left\{n / x, c_n / y, c_{n+1} / z\right\}\right|$$

and so there exists $\theta_n \in \mathcal{G} \left(\mathbf{1}, \left[B^*\right] \parr \left[\iota\right]\right)$ such that $\theta_n \in |B' \left\{n / x, c_n / y, c_{n+1} / z\right\}|$ and:

$$\psi \left(\theta_0 * ... * \theta_{n-1} * \theta_n\right) = \psi \left(\rho * \theta_n\right) = \lambda x.\psi \left(\rho * x\right) \theta_n \notin |\bot|$$

□

5.3 Extraction

Since the Friedman translation is directly built in the realizability, the extraction result is an easy consequence of the definitions. We show that from any Π^0_2-formula provable in CA^ω we can extract a computable witnessing function. Note that in the extraction lemma, the equality $t =_U u$ is at any type:

Lemma 5.11. *From a proof of* $\vdash \forall x^T \exists y^\iota \, (t =_U u)$ *in* CA^ω*, one can extract a* $\lambda\mu$*-term* $M : T \to \iota$ *such that for any* $a \in T^{\mathcal{M}}$ *and* $\sigma \Vdash (\!|a|\!)$*, there is some* $n \in \mathbb{N}$ *such that:*

$$(t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}} \text{ and } M\,\sigma = \overline{\mathsf{n}}$$

Proof. First, we obtain by double-negation elimination a proof of $\vdash \forall x^T \neg\forall y^\iota \, (t \neq_U u)$, and by relativization a proof π in $CA^{\omega \mathrm{r}}$ of $\vdash \forall^{\mathrm{r}} x^T \neg\forall^{\mathrm{r}} y^\iota \, (t \neq_U u)$. The adequacy lemma then tells us:

$$\pi^* \in \left|\forall^{\mathrm{r}} x^T \neg\forall^{\mathrm{r}} y^\iota \, (t \neq_U u)\right|$$

Then, if $a \in T^{\mathcal{M}}$ and $\sigma \in |(\!|a|\!)|$, we get $\pi^*\,\sigma \in |\neg\forall^{\mathrm{r}} y^\iota \, (t\,\{a/x\} \neq_U u\,\{a/x\})|$. Let now fix:

$$\perp\!\!\!\perp = \left\{ n \in \iota^{\mathcal{M}} \,\middle|\, (t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}} \right\}$$

We prove now that $\lambda x.\,[\kappa]\ x \in |\forall^{\mathrm{r}} y^\iota \, (t\,\{a/x\} \neq_U u\,\{a/x\})|$. For that let $n \in \iota^{\mathcal{M}}$, and let prove that $(\lambda x.\,[\kappa]\ x)\,\overline{\mathsf{n}} = [\kappa]\ \overline{\mathsf{n}} \in |t\,\{a/x, n/y\} \neq u\,\{a/x, n/y\}|$. There are two cases:

- $n \in \perp\!\!\!\perp$: in that case, $(t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}}$, so:

  $$|t\,\{a/x, n/y\} \neq u\,\{a/x, n/y\}| = |\bot|$$

  and $\mu\kappa.\,[\kappa]\ \overline{\mathsf{n}} = \overline{\mathsf{n}} \in |\perp\!\!\!\perp|$ so $[\kappa]\ \overline{\mathsf{n}} \in |\bot|$

- $n \notin \perp\!\!\!\perp$: in that case, $(t\,\{a/x, n/y\})^{\mathcal{M}} \neq (u\,\{a/x, n/y\})^{\mathcal{M}}$, so:

  $$|t\,\{a/x, n/y\} \neq u\,\{a/x, n/y\}| = \mathcal{G}\,(\mathbf{1}, [0] \parr [\iota])$$

  and therefore $[\kappa]\ \overline{\mathsf{n}} \in |t\,\{a/x, n/y\} \neq u\,\{a/x, n/y\}|$

Therefore we get $\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x) \in |\bot|$, and so $\mu\kappa.\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x) \in |\perp\!\!\!\perp|$. This means that $\mu\kappa.\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x)$ is some $\overline{\mathsf{n}}$ such that $n \in \perp\!\!\!\perp$ (so $(t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}}$). Finally, we have the claimed result with $M \triangleq \lambda u.\mu\kappa.\pi^*\,u\ (\lambda x.\,[\kappa]\ x)$. □

In the particular case of $T = \iota$, we get that for any $m \in \mathbb{N}$, $M\,\overline{\mathsf{m}}$ is some $\overline{\mathsf{n}}$ such that $(t\,\{m/x, n/y\})^{\mathcal{M}} = (u\,\{m/x, n/y\})^{\mathcal{M}}$.

Chapter 6

A realizability model of winning strategies for classical arithmetic

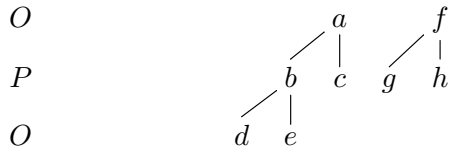
The realizability model of the previous chapter makes heavy use of orthogonality in categories of continuations, and does not rely heavily on the underlying games model. In this chapter, we devise another realizability model based on winning conditions, which is very dependent on the particular model of HON games. In particular, the realizability semantics of implication is not anymore the usual Kleene arrow, but requires to look directly at the set of plays constituting a strategy. Our winning conditions can be seen as an extension to first-order logic of [Cla09] and [Hyl97]. This dependency upon the plays constituting a strategy, rather than the sole result of putting the strategy in a context, allows a much more fine-grained control. This particularity has already been exploited for different purposes, for example decidability results for fragments of programming languages [GM03], type isomorphisms [Lau05, Cla12], or bounds on reduction in abstract machines [Cla11]. In the context of realizability, one hope is to obtain and use an operator similar to Krivine's clock [Kri03]. The clock operator implements an enumeration of terms and it is used to realize the axiom of dependent choice in a second-order setting. On the other side, when realizing the axiom of choice through bar recursion, one needs to turn any sequence of elements of the model into a single element. This implies that the model is uncountable, and therefore there is no enumeration of its elements. However, in the context of winning conditions, we do not look at the uncountable set of all strategies, but only at the countable set of plays. Therefore, an enumeration of all plays may suffice to obtain a behavior similar to Krivine's clock, leading to a single realizability model in which the two realizations of choice co-exist.

We first follow ideas from [CH09] and highlight the causality structure of justified sequences by reformulating these as thick subtrees (defined in [Bou09]). Then we define the notion of winning conditions on arenas, which are sets of desequentialized plays. We define a realizability relation based on these winning conditions and prove the adequacy of this interpretation for first-order logic and Peano arithmetic and use it to extract computational content from classical proofs through a variant of Friedman's translation for classical logic. Finally, we give ideas about the realization of the axiom of choice using the higher-order references of the model of Hyland-Ong-Nickau games. The work presented in this chapter is an improved version of [Blo13].

6.1 Justified sequences via thick subtrees

In usual game semantics [HO00], justified sequences are defined as in the previous chapter, as words of moves with pointers between them. However, it was shown in [CH09] that justified

sequences could also be represented by trees with a sequential ordering of nodes (which they call "multiplexed positions"), and that this representation was compatible with the dynamic aspects of composition. Here we use the thick subtrees of [Bou09] to define positions on arenas, and the interaction sequences are then positions together with a sequentiality information. This choice is justified by the positional nature of our winning conditions. In this section we will use the following arena for the examples:



(6.1)

Remember from the previous chapter that an arena is a partial order, and that we use $<, \leq$ and $<_1$ as relations between the nodes. For example, in the arena (6.1) we have $b <_1 e$.

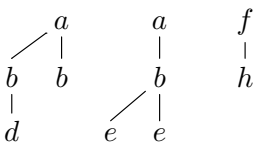
6.1.1 Justified sequences as thick subtrees

We choose to define positions on arenas using the thick subtrees of [Bou09], extended to handle the case of forests. This formalism is a nice way to deal with non-affine programs: programs that may use their arguments several times. A thick subtree of a given tree is a subtree which can be extended in width, meaning that branches of the initial tree can be duplicated. These duplications correspond to the distinct computations of arguments during the execution of a non-affine program.

Definition 6.1 (State)**.** *Given an arena $\mathcal{A}$, a* state *on $\mathcal{A}$ is a thick subforest of $\mathcal{A}$, that is a forest s together with a labeling function $l : s \to \mathcal{A}$ such that:*

$$\forall x \in s, \{l(y) \mid y < x\} = \{a \in \mathcal{A} \mid a < l(x)\}$$

This condition ensures that the roots are mapped to roots, and that the relation $<_1$ is preserved. Here is an example of a state on the arena (6.1):

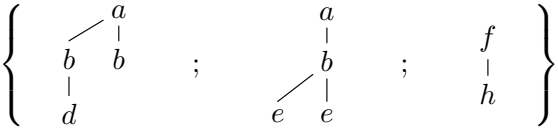


(6.2)

The nodes are considered distinct, even if they have the same label. By definition of the polarity, the roots of a state are labeled with O-moves, and if $x <_1 y$, then the labels of x and y are of opposite polarities. We denote the empty state by ϵ.

Definition 6.2 (Position)**.** *A position on an arena is a state which is a tree (i.e. it has exactly one root). If s is a state on an arena $\mathcal{A}$, the* set of positions *of s, $\mathrm{Pos}(s)$, is the set of trees composing the state. In particular, a state s is a position if and only if $\mathrm{Pos}(s) = \{s\}$, and since $\mathrm{Pos}(\epsilon) = \emptyset$, ϵ is not a position.*

For example, the set of positions of the state (6.2) is:



Our notion of position can be seen as an attempt to give a positional account of game semantics, as in [Mel06], but in a quite different way.

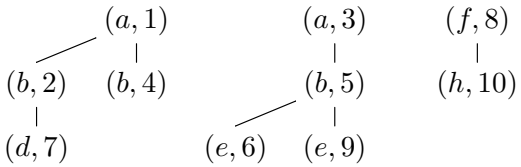
The sequential states are then states on arenas together with a sequentiality information:

Definition 6.3 (Sequential state)**.** *Given an arena $\mathcal{A}$, we define a* sequential state *on $\mathcal{A}$ to be a state (finite of infinite) s on $\mathcal{A}$ equipped with a total order $\prec$ of type at most ω*[1]*. Moreover, a sequential state must verify:*

$$\forall x, y \in s \qquad x < y \quad \Rightarrow \quad x \prec y$$

The partial order $<$ of a sequential state corresponds to the pointers of usual game semantics settings: $x <_1 y$ corresponds to y pointing to x. The total order $\prec$ corresponds to the sequentiality of moves. The coherence condition of sequential states means that if there is a pointer from a move y to a move x in the sequence, then x must have been played before y. The fact that $l(x)$ must be the father of $l(y)$ in the arena comes from the definition of a state.

Here is an example of a sequential state on the arena (6.1):

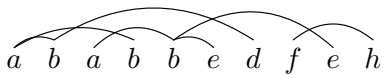


(6.3)

Where the labeling is given by the first components and the injection to natural numbers by the second components. The empty sequential state will be denoted ϵ, like the empty state. By the coherence condition of sequential states, the minimal element for $\prec$ is always a minimal element for $<$, and therefore labeled with a root of the arena. We can map any sequential state to a state by forgetting the sequentiality information, so in the following we will implicitly consider that a sequential state is a state. For example the state corresponding to the sequential state (6.3) is the state (6.2).

6.1.2 Correspondence with the usual setting

In the previous chapter, justified sequences were defined as usual in game semantics (see for example [HO00]), as words of moves with pointers between them. For example, the justified sequence (6.3) was represented as:



We prove here the correspondence between the notion of a sequential state and that of a justified sequence which was defined in the previous chapter.

From justified sequences to sequential states

If s is a justified sequence with justifying function f as in Definition 4.2, we define the corresponding sequential state $\mathrm{SeqSt}(s)$ as the thick subforest of $\mathcal{A}$ whith nodes (s_i, i). The order $<$ of the forest is defined as:

$$(s_i, i) < (s_j, j) \Longleftrightarrow \exists n > 0, f^n(j) = i$$

the labeling function is given by the first components and the order $\prec$ is given by the second components.

Lemma 6.1. *If s is a justified sequence with justifying function f, then* $\mathrm{SeqSt}(s)$ *is a state.*

[1] This means that $\prec$ can be described by an injection to natural numbers.

Proof. Let (s_j, j) be a node. We have to prove:

$$\{s_i \mid (s_i, i) < (s_j, j)\} = \{a \in \mathcal{A} \mid a < s_j\}$$

For the left-to-right inclusion, if $(s_i, i) < (s_j, j)$ then by definition there is some $n > 0$ such that $f^n(j) = i$, so by the definition of a justified sequence, in $\mathcal{A}$ we have $s_i = s_{f^n(j)} <_1 s_{f^{n-1}(j)} <_1 ... <_1 s_{f(j)} <_1 s_j$, and therefore $s_i < s_j$. For the right-to-left inclusion, since an arena has finite depth, $\{a \in \mathcal{A} \mid a < s_j\}$ is finite, and it is also a well-order (by definition of a forest), so there are $a_n <_1 ... <_1 a_1 \in \mathcal{A}$ such that $\{a \in \mathcal{A} \mid a < s_j\} = \{a_n; ...; a_1\}$. If $a \in \mathcal{A}$ is such that $a < s_j$, then $a = a_k$ for some $1 \leq k \leq n$, and then $a = s_{f^k(j)}$. Moreover, by definition $\left(s_{f^k(j)}, f^k(j)\right) < (s_j, j)$ so we conclude. □

Lemma 6.2. *If s is a justified sequence with justifying function f, then* $\mathrm{SeqSt}(s)$ *is a sequential state.*

Proof. We only have to prove that if $(s_i, i) < (s_j, j)$, then $i < j$. Indeed, if $(s_i, i) < (s_j, j)$, then by definition $f^n(j) = i$ for some $n > 0$, but since $f(k) < k$ as soon as $f(k)$ is defined, we get immediately $i < j$. □

From sequential states to justified sequences

If s is a sequential state with labelling l we define here the corresponding justified sequence $\mathrm{Just}(s)$. We can order the elements of s using the total ordering $\prec$, which is of type at most ω, and obtain a (possibly infinite) sequence $l(x_0)...l(x_n)...$ such that $x_i \prec x_j \Leftrightarrow i < j$, which is the word of moves of $\mathrm{Just}(s)$. We now define the justifying function f of $\mathrm{Just}(s)$. If $x_i \in s$, there are two cases. Either $\{x_j \in s \mid x_j < x_i\} = \emptyset$, in which case $f(i)$ is left undefined, or there is a unique $x_j \in s$ such that $x_j <_1 x_i$ (in s), in which case we define $f(i) = j$.

Lemma 6.3. *If s is a sequential state with labelling l, then* $\mathrm{Just}(s)$ *is a justified sequence.*

Proof. Let f be the justifying function of $\mathrm{Just}(s)$. If $f(i)$ is undefined, then $\{x_j \in s \mid x_j < x_i\} = \emptyset$, but then:

$$\{a \in \mathcal{A} \mid a < l(x_i)\} = \{l(x_j) \mid x_j < x_i\} = \emptyset$$

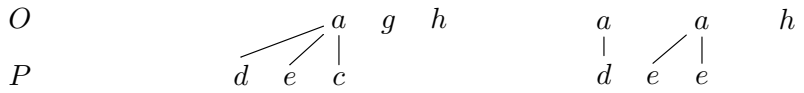
so $l(x_i)$ is a root of $\mathcal{A}$. If $f(i) = j$, then $x_j <_1 x_i$ so $x_j < x_i$ and $x_j \prec x_i$ by definition of a sequential state. By definition of the ordering that we chose, this means that $j < i$. Moreover, by definition of a state, since $x_j <_1 x_i$ we get also $l(x_j) <_1 l(x_i)$. □

Equivalence of the two notions

Sequential states are in one-to-one correspondence with usual justified sequences. The two transformations above, from justified sequences to sequential states, and from sequential states to justified sequences, can be composed, and we can prove that for any justified sequence s, $\mathrm{Just}(\mathrm{SeqSt}(s)) = s$, and for any sequential state s, $\mathrm{SeqSt}(\mathrm{Just}(s)) = s$. In the following we will sometimes use "sequence" instead of "sequential state".

Another interesting thing about sequential states is that the restriction of a sequential state is much easier to define than that of a justified sequence: if $\mathcal{A}$ is an arena, X is a subset of $\mathcal{A}$ (and therefore an arena, see the previous chapter) and s is a state on $\mathcal{A}$, then $s_{|X}$ is the state on X with nodes $l^{-1}(X)$ (where $l : s \to \mathcal{A}$ is the labeling of the state), with the restricted ordering. It is immediate to check that it is a state on X. Moreover, this is consistent with the definition of restriction of a justified sequence: if s is a justified sequence then $\mathrm{SeqSt}\left(s_{|X}\right) = \mathrm{SeqSt}(s)_{|X}$

and if s is a sequential state, then $\mathrm{Just}\left(s_{|X}\right) = \mathrm{Just}\left(s\right)_{|X}$. Here is an example of a subset X of the arena (6.1), together with the state (6.2) restricted to X:



(6.4)

Recall from section 4.2.1 that the set of threads of a justified sequence is obtained by taking for each initial move of the sequence the set of moves hereditarily justified by it. In the case of sequential states, this amounts to take the set of trees of the sequential state together with the restricted ordering $\succ$. Then, $\mathrm{Threads}\left(s\right)$ is a set of sequential states, which become positions by forgetting about sequentiality, and these positions are exactly the positions of $\mathrm{Pos}\left(s\right)$ defined above, obtained by considering s as a state without sequentiality. Finally the notions of O/P-sequence, prefix, O/P-prefix, play and O/P-play also translate to sequential states through the isomorphism between justified sequences and sequential states.

6.2 Winning conditions on arenas

It is well-known that preservation of totality by composition of strategies is problematic in game semantics. In the context of realizability, however, we are not interested in totality, but only in winningness. We thus do not impose any totality condition on strategies, but when it turns to the definition of winning positions, we have to take into account all maximal positions, including both infinite and odd-length ones. This leads to the notion of winning strategy proposed in definition 6.8.

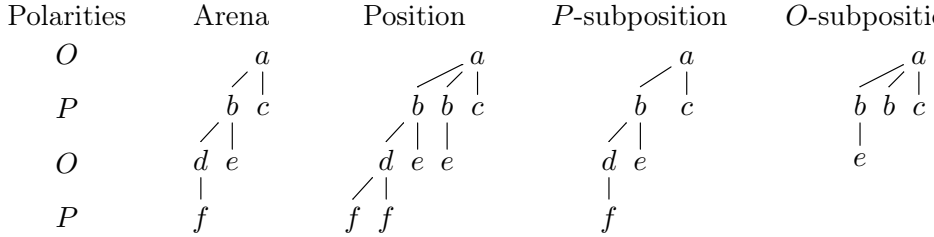
6.2.1 P-subpositions, O-subpositions

In order to define the notion of winning condition on an arena we introduce the notion of P-subposition and O-subposition:

Definition 6.4 (P-subposition, O-subposition)**.** *If s is a position and t is a downward-closed subset of s (therefore it is a position), then t is a:*

- *P-subposition of s if when $a^P <_1 b^O$ in s and $a^P \in t$, then $b^O \in t$*
- *O-subposition of s if when $a^P <_1 b^O$ in s and $a^O \in t$, then $b^P \in t$*

The intuitive meaning is that a P-subposition is a weakening of player (but any answer of opponent to a player move must be kept), and an O-subposition is a weakening of opponent (but any answer of player to an opponent move must be kept). A P-subposition (resp. an O-subposition) is obtained from a given position by cutting some subtrees having roots labeled with P-moves (resp. O-moves). Here is an example of a position together with a P-subposition and an O-subposition:



6.2.2 Winning conditions

Now we can define the notion of winning condition on an arena:

Definition 6.5 (Winning condition)**.** *A* winning condition *on $\mathcal{A}$ is a set $\mathcal{W}_\mathcal{A}$ of positions on $\mathcal{A}$ such that:*

- *If s is a position on $\mathcal{A}$ and if some P-subposition of s is in $\mathcal{W}_\mathcal{A}$, then $s \in \mathcal{W}_\mathcal{A}$.*
- *If $s \in \mathcal{W}_\mathcal{A}$ then all the O-subpositions of s are in $\mathcal{W}_\mathcal{A}$.*

A state s (and by extension a sequence or a play) on the arena $\mathcal{A}$ equipped with the winning condition $\mathcal{W}_\mathcal{A}$ is said to be winning *if* $\mathrm{Pos}(s) \subseteq \mathcal{W}_\mathcal{A}$.

A winning position must be thought as a position that is winning for player P. Under this interpretation, the two requirements read as: if some player-weakening of a position is already winning, then the full position is also winning, and if a position is winning, then any opponent-weakening of that position is all the more winning.

Our notion of winning state can be seen as a generalization of the winning conditions defined in [Cla09] and [Hyl97]. In these works, every even-length position is winning, every odd-length position is losing, and the non-trivial part lies in the winningness of infinite positions. In order to obtain a realizability model of first-order logic, the notion of winning finite state is here non-trivial and there can be odd-length plays which are winning and even-length plays which are losing. Winning conditions were also defined in [Mel05a] using payoffs on positions, but with different purposes, and in the framework of asynchronous games.

We now define constructions on winning conditions, corresponding to the arena constructors $\to$, $\times$, $\mathcal{V}$ and $⅋$ (the binoidal functor of categories of continuations, acting on arenas as described in section 4.4.1), but first we give a few general remarks about positions in the arenas $\mathcal{A} \to \mathcal{B}$, $\mathcal{A} \times \mathcal{B}$ and $\mathcal{A} ⅋ \mathcal{B}$. If s is a position on $\mathcal{A} \to \mathcal{B}$, then $s_{|\mathcal{B}}$ is a position on $\mathcal{B}$, so $s_{|\mathcal{B}}$ is winning iff $s_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B}$, if s is a position on $\mathcal{A} \times \mathcal{B}$, then s is either a position on $\mathcal{A}$, or a position on $\mathcal{B}$, and if s is a position on $\mathcal{A} ⅋ \mathcal{B}$, then $s_{|\mathcal{A}}$ is a position on $\mathcal{A}$ and $s_{|\mathcal{B}}$ is a position on $\mathcal{B}$, so $s_{|\mathcal{A}}$ (respectively $s_{|\mathcal{B}}$) is winning iff $s_{|\mathcal{A}} \in \mathcal{W}_\mathcal{A}$ (respectively $s_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B}$).

Definition 6.6 (Arrow, product and par of winning conditions)**.** *If $\mathcal{W}_\mathcal{A}$ and $\mathcal{W}_\mathcal{B}$ are sets of positions on the arenas $\mathcal{A}$ and $\mathcal{B}$, then we define:*

$$\mathcal{W}_{\mathcal{A}\to\mathcal{B}} = \left\{ s \text{ position on } \mathcal{A} \to \mathcal{B} \;\middle|\; \mathrm{Pos}\left(s_{|\mathcal{A}}\right) \subseteq \mathcal{W}_\mathcal{A} \;\Rightarrow\; s_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B} \right\}$$

$$\mathcal{W}_{\mathcal{A}\times\mathcal{B}} = \left\{ s \text{ position on } \mathcal{A} \times \mathcal{B} \;\middle|\; \begin{array}{l} s \text{ position on } \mathcal{A} \;\Rightarrow\; s \in \mathcal{W}_\mathcal{A} \\ s \text{ position on } \mathcal{B} \;\Rightarrow\; s \in \mathcal{W}_\mathcal{B} \end{array} \right\}$$

$$\mathcal{W}_\mathcal{V} = \emptyset \qquad \mathcal{W}_{\mathcal{A}⅋\mathcal{B}} = \left\{ s \text{ position on } \mathcal{A} ⅋ \mathcal{B} \;\middle|\; s_{|\mathcal{A}} \in \mathcal{W}_\mathcal{A} \;\vee\; s_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B} \right\}$$

Remark that the winning conditions on these connectives have indeed the correct intuitive interpretation: a position is winning on $\mathcal{A} \to \mathcal{B}$ if whenever its projection on $\mathcal{A}$ is winning, its projection on $\mathcal{B}$ is also winning, a position is winning on the conjunction $\mathcal{A} \times \mathcal{B}$ if its projections on $\mathcal{A}$ and $\mathcal{B}$ are both winning (one of the two being the empty play), no position is winning on the empty disjunction $\mathcal{V}$, and a position in winning on the binary disjunction $\mathcal{A} ⅋ \mathcal{B}$ if one of its projections on $\mathcal{A}$ or $\mathcal{B}$ is winning.

The following lemma states that these are indeed winning conditions:

Lemma 6.4. *If $\mathcal{W}_\mathcal{A}$ and $\mathcal{W}_\mathcal{B}$ are winning conditions on $\mathcal{A}$ and $\mathcal{B}$, then $\mathcal{W}_{\mathcal{A}\to\mathcal{B}}$ is a winning condition on $\mathcal{A} \to \mathcal{B}$, $\mathcal{W}_{\mathcal{A}\times\mathcal{B}}$ is a winning condition on $\mathcal{A} \times \mathcal{B}$, $\mathcal{W}_\mathcal{V}$ is a winning condition on $\mathcal{V}$ and $\mathcal{W}_{\mathcal{A}⅋\mathcal{B}}$ is a winning condition on $\mathcal{A} ⅋ \mathcal{B}$.*

Proof.

- Let s be a position on $\mathcal{A} \to \mathcal{B}$ and let t be a P-subposition of s such that $t \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$. Suppose that $\mathrm{Pos}\left(s_{|\mathcal{A}}\right) \subseteq \mathcal{W}_{\mathcal{A}}$. If $u \in \mathrm{Pos}\left(t_{|\mathcal{A}}\right)$ then u is an O-subposition of some $v \in \mathrm{Pos}\left(s_{|\mathcal{A}}\right) \subseteq \mathcal{W}_{\mathcal{A}}$, so $u \in \mathcal{W}_{\mathcal{A}}$, hence $\mathrm{Pos}\left(t_{|\mathcal{A}}\right) \subseteq \mathcal{W}_{\mathcal{A}}$. Then since $t \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$, $t_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$ and since $t_{|\mathcal{B}}$ is a P-subposition of $s_{|\mathcal{B}}$ we conclude that $s_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$. Finally $s \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$.

- Let s be a position on $\mathcal{A} \to \mathcal{B}$ such that $s \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$ and let t be an O-subposition of s. Let suppose that $\mathrm{Pos}\left(t_{|\mathcal{A}}\right) \subseteq \mathcal{W}_{\mathcal{A}}$. If $u \in \mathrm{Pos}\left(s_{|\mathcal{A}}\right)$ and if a is the root of u, then the father of a is the root of s which is an O-move, and since t is an O-subposition of s, we get $a \in t$. Now the position of $t_{|\mathcal{A}}$ with root a is in $\mathcal{W}_{\mathcal{A}}$ and it is a P-subposition of u, so $u \in \mathcal{W}_{\mathcal{A}}$. Therefore $\mathrm{Pos}\left(s_{|\mathcal{A}}\right) \subseteq \mathcal{W}_{\mathcal{A}}$, and since $s \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$ we have $s_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$. Since $t_{|\mathcal{B}}$ is an O-subposition of $s_{|\mathcal{B}}$, we get $t_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$. Finally $t \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$.

- Let s be a position on $\mathcal{A} \times \mathcal{B}$. s is either a position on $\mathcal{A}$, or a position on $\mathcal{B}$, so if t is a winning P-subposition of s, then either $t \in \mathcal{W}_{\mathcal{A}}$, or $t \in \mathcal{W}_{\mathcal{B}}$. Therefore $s \in \mathcal{W}_{\mathcal{A}}$ or $s \in \mathcal{W}_{\mathcal{B}}$, and so $s \in \mathcal{W}_{\mathcal{A}\times\mathcal{B}}$.

- Let s be a position on $\mathcal{A} \times \mathcal{B}$ such that $s \in \mathcal{W}_{\mathcal{A}\times\mathcal{B}}$. Either $s \in \mathcal{W}_{\mathcal{A}}$, or $s \in \mathcal{W}_{\mathcal{B}}$, so any O-subposition of s is in $\mathcal{W}_{\mathcal{A}}$ or $\mathcal{W}_{\mathcal{B}}$, so in $\mathcal{W}_{\mathcal{A}\times\mathcal{B}}$.

- The empty set is trivially a winning condition on any arena, so in particular on $\mathcal{V}$.

- Let s be a position on $\mathcal{A} \parr \mathcal{B}$ and let t be a P-subposition of s such that $t \in \mathcal{W}_{\mathcal{A}\parr\mathcal{B}}$. Then either $t_{|\mathcal{A}} \in \mathcal{W}_{\mathcal{A}}$, or $t_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$. But $t_{|\mathcal{A}}$ is a P-subposition of $s_{|\mathcal{A}}$ and $t_{|\mathcal{B}}$ is a P-subposition of $s_{|\mathcal{B}}$, so in both cases we get $s \in \mathcal{W}_{\mathcal{A}\parr\mathcal{B}}$.

- Let s be a position on $\mathcal{A} \parr \mathcal{B}$ such that $s \in \mathcal{W}_{\mathcal{A}\parr\mathcal{B}}$ and let t be an O-subposition of s. Then $t_{|\mathcal{A}}$ is an O-subposition of $s_{|\mathcal{A}}$ and $t_{|\mathcal{B}}$ is an O-subposition of $s_{|\mathcal{B}}$. Therefore, if $s_{|\mathcal{A}} \in \mathcal{W}_{\mathcal{A}}$ then $t_{|\mathcal{A}} \in \mathcal{W}_{\mathcal{A}}$ and if $s_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$ then $t_{|\mathcal{B}} \in \mathcal{W}_{\mathcal{B}}$. In both cases $t \in \mathcal{W}_{\mathcal{A}\parr\mathcal{B}}$.

□

Finally, in order to have a realizability interpretation for first-order logic, the following lemma (the proof of which is straightforward) is fundamental:

Lemma 6.5. *The intersection of winning conditions on a fixed arena is itself a winning condition.*

6.2.3 Winning strategies

In order to define what a winning strategy is, we use a notion of augmented plays of a strategy inspired from [Mel05b]:

Definition 6.7 (Augmented play)**.** *If σ is a strategy on $\mathcal{A}$ and s is a play on $\mathcal{A}$, then s is an* augmented play *of σ if one of the following holds:*

- $s \in \sigma$, *or*
- s *is such that* $\forall t \sqsubset_P s,\ t \in \sigma$ *and* $\forall t \in \sigma, s \not\sqsubseteq t$.

In particular, in the second case of the above definition, s is either an O-play, or an infinite play (in which case $s \sqsubseteq t \Leftrightarrow s = t$ and so the second condition, equivalent to $s \notin \sigma$, is always true since strategies contain only finite plays). Unlike [Mel05b], we consider not only odd-length extensions (with an O-move), but also infinite ones. The following lemma states that the definition of composition of strategies can be extended to augmented plays:

Lemma 6.6. *Let $\sigma : \mathcal{A} \to \mathcal{B}$ and $\tau : \mathcal{B} \to \mathcal{C}$. If s is an augmented play of $\sigma;\tau$, then there exists $t \in \mathrm{Int}(\mathcal{A},\mathcal{B},\mathcal{C})$ such that $t_{|\mathcal{A}\to\mathcal{B}}$ is an augmented play of σ, $t_{|\mathcal{B}\to\mathcal{C}}$ is an augmented play of τ and $t_{|\mathcal{A}\to\mathcal{C}} = s$.*

Proof. In this proof, we rely on the results of section 4.3.1 about composition of strategies. There are three possibilities: either $s \in \sigma;\tau$, or $s = s'a$ with $s' \in \sigma;\tau$ and no extension of s is in $\sigma;\tau$, or s is infinite and every P-prefix of s is in $\sigma;\tau$.

- In the first case, the result comes from the definition of composition.
- In the second case, by definition of composition there is some $t' \in \mathrm{Int}(\mathcal{A},\mathcal{B},\mathcal{C})$ such that $t'_{|\mathcal{A}\to\mathcal{B}} \in \sigma$, $t'_{|\mathcal{B}\to\mathcal{C}} \in \tau$ and $t'_{|\mathcal{A}\to\mathcal{C}} = s'$. By lemma 4.2, the automaton 4.5 is in state 000 after t'. Since a is an O-move in $\mathcal{A} \to \mathcal{C}$, it is an O-move in $\mathcal{A} \to \mathcal{C}$ or an O-move in $\mathcal{B} \to \mathcal{C}$, and therefore $t'a \in \mathrm{Int}(\mathcal{A},\mathcal{B},\mathcal{C})$. Then we extend $t'a$ deterministically with moves in $\mathcal{B}$ by taking at each step the unique move given by σ (if we are in state 101) or by τ (if we are in state 011) until it is not possible anymore. The move b provided by σ and τ will always be in $\mathcal{B}$, since otherwise $s'ab = sb$ would be an extension of s which is in $\sigma;\tau$ and s would not be an augmented play of $\sigma;\tau$. We call this extension u, and we have $u \in \mathrm{Int}(\mathcal{A},\mathcal{B},\mathcal{C})$ and $u_{|\mathcal{A}\to\mathcal{C}} = s$. Either u infinite, or u brings the automaton in state 101 or 011. If u is infinite, then both $u_{|\mathcal{A}\to\mathcal{B}}$ and $u_{|\mathcal{B}\to\mathcal{C}}$ are infinite and therefore augmented plays of σ and τ. If u brings the automaton in state 101, then σ fails to extend $u_{|\mathcal{A}\to\mathcal{B}}$ so $u_{|\mathcal{A}\to\mathcal{B}}$ is an augmented play of σ, and $u_{|\mathcal{B}\to\mathcal{C}}$ is even, so $u_{|\mathcal{B}\to\mathcal{C}} \in \tau$ is an augmented play of τ. Finally, if u brings the automaton in state 011, then τ fails to extend $u_{|\mathcal{B}\to\mathcal{C}}$ so $u_{|\mathcal{B}\to\mathcal{C}}$ is an augmented play of τ, and $u_{|\mathcal{A}\to\mathcal{B}}$ is even, so $u_{|\mathcal{A}\to\mathcal{B}} \in \sigma$ is an augmented play of σ.
- In the third case, or $u_{|\mathcal{A}\to\mathcal{B}}$ and $u_{|\mathcal{B}\to\mathcal{C}}$ are both infinite, in which case they are extended plays of σ and τ, or $u_{|\mathcal{A}\to\mathcal{B}}$ is finite and $u_{|\mathcal{B}\to\mathcal{C}}$ is infinite, in which case at some point all the moves of u are in $\mathcal{C}$ so the automaton oscillates between states 000 and 101, so $u_{|\mathcal{A}\to\mathcal{B}}$ is even and $u_{|\mathcal{A}\to\mathcal{B}} \in \sigma$ is an augmented play of σ, and $u_{|\mathcal{B}\to\mathcal{C}}$ being infinite it is an augmented play of τ, or $u_{|\mathcal{B}\to\mathcal{C}}$ is finite and $u_{|\mathcal{A}\to\mathcal{B}}$ is infinite, in which case at some point all the moves of u are in $\mathcal{A}$ so the automaton oscillates between states 000 and 011, so $u_{|\mathcal{B}\to\mathcal{C}}$ is even and $u_{|\mathcal{B}\to\mathcal{C}} \in \tau$ is an augmented play of τ, and $u_{|\mathcal{A}\to\mathcal{B}}$ being infinite it is an augmented play of σ.

□

The converse implication of this lemma is also true, however we will not use it. We define now the concept of winning strategy:

Definition 6.8 (Winning strategy)**.** *If σ is a strategy on $\mathcal{A}$ equipped with the winning condition $\mathcal{W}_\mathcal{A}$, then σ is said to be* winning *if all its augmented plays are winning.*

The following lemma will be useful to prove that a strategy σ is winning on $(\mathcal{A},\mathcal{W}_\mathcal{A})$.

Lemma 6.7. *If σ is a strategy on $\mathcal{A}$ and if s is an augmented play of σ, then every $t \in \mathrm{Threads}(s)$ is an augmented play of σ*

Proof.
- If $s \in \sigma$, then by single-threadedness of σ, $\mathrm{Threads}(s) \subseteq \sigma$.
- If s is an O-play, then we write $s = s'a$ with $s' \in \sigma$. Let $t \in \mathrm{Threads}(s)$. If a is not a move in t, then $t \in \mathrm{Threads}(s') \subseteq \sigma$. If a is a move in t, then we write $t = t'a$, so $t' \in \mathrm{Threads}(s') \subseteq \sigma$. If there is some b such that $tb = t'ab \in \sigma$, then $\mathrm{Threads}(s'ab) = (\mathrm{Threads}(s') \setminus \{t'\}) \cup \{t'ab\} \subseteq \sigma$, so by single-threadedness of σ, $sb = s'ab \in \sigma$, contradicting the fact that s is an augmented play of σ.

- If s is infinite, let $t \in \text{Threads}(s)$. If t is finite, then there is some $s' \sqsubseteq_P s$ such that $t \in \text{Threads}(s')$, but $s' \in \sigma$, so by single-threadedness of σ, $t \in \sigma$. If t is infinite, then for all $t' \sqsubseteq_P t$ there is some $s' \sqsubseteq_P s$ such that $t' \in \text{Threads}(s')$, but $s' \in \sigma$, so by single-threadedness of σ, $t' \in \sigma$.

□

The above result is the exact analogue of the left-to-right direction of the single-threadedness condition in the definition of a strategy, but for the set of augmented plays of a strategy. The other direction is also true, but will not be used. An augmented play of a strategy which is a thread will be called an augmented thread of the strategy. Using this lemma, if any augmented thread of σ is in $\mathcal{W}_\mathcal{A}$ and if s is an augmented play of σ, then for any $t \in \text{Threads}(s)$, t is an extended thread of σ, so $t \in \mathcal{W}_\mathcal{A}$. Then $\text{Threads}(s) \subseteq \mathcal{W}_\mathcal{A}$ so s is winning. Therefore, it is sufficient to prove that every augmented thread of σ is in $\mathcal{W}_\mathcal{A}$ in order to prove that σ is winning on $(\mathcal{A}, \mathcal{W}_\mathcal{A})$.

We now prove that the winning conditions on the arrow, product and par ($\parr$) are compatible with identity, application, pairing, projection and codiagonal ∇ of strategies. This will imply that the realizability semantics is preserved through modus ponens, and more generally through all the propositional rules of the logic.

Lemma 6.8. *For any winning condition $\mathcal{W}_\mathcal{A}$ on $\mathcal{A}$, $\mathbf{Id}_\mathcal{A}$ is winning on $(\mathcal{A} \to \mathcal{A}, \mathcal{W}_{\mathcal{A}\to\mathcal{A}})$.*

Proof. Let $\mathcal{A}^{(1)} \to \mathcal{A}^{(2)}$ denote the arena $\mathcal{A} \to \mathcal{A}$. Let s be an augmented thread of $\mathbf{Id}_\mathcal{A}$ such that $\text{Threads}\left(s_{|\mathcal{A}^{(1)}}\right) \subseteq \mathcal{W}_\mathcal{A}$. Since s is a thread, $s_{|\mathcal{A}^{(2)}}$ is a thread and $s_{|\mathcal{A}^{(1)}} = s_{|\mathcal{A}^{(2)}}$ is also a thread. Then $s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_\mathcal{A}$, so $s_{|\mathcal{A}^{(2)}} = s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_\mathcal{A}$. □

Lemma 6.9. *If σ is winning on $(\mathcal{A} \to \mathcal{B}, \mathcal{W}_{\mathcal{A}\to\mathcal{B}})$ and τ is winning on $(\mathcal{A}, \mathcal{W}_\mathcal{A})$, then $\sigma(\tau)$ is winning on $(\mathcal{B}, \mathcal{W}_\mathcal{B})$.*

Proof. Let s be an augmented thread of $\sigma(\tau)$. By lemma 6.6, there is some augmented play t of σ such that $t_{|\mathcal{A}}$ is an augmented play of τ and $t_{|\mathcal{B}} = s$. Since s is a thread, t is also a thread, so since σ is winning on $\mathcal{A} \to \mathcal{B}$, $t \in \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$. $t_{|\mathcal{A}}$ is an augmented play of τ which is winning on $\mathcal{A}$, so $\text{Threads}\left(t_{|\mathcal{A}}\right) \subseteq \mathcal{W}_\mathcal{A}$, and therefore $t_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B}$. Finally, $s = t_{|\mathcal{B}} \in \mathcal{W}_\mathcal{B}$. Therefore $\sigma(\tau)$ is winning. □

Lemma 6.10. *If σ is winning on $(\mathcal{A}, \mathcal{W}_\mathcal{A})$ and τ is winning on $(\mathcal{B}, \mathcal{W}_\mathcal{B})$, then $\mathbf{pair}(\sigma, \tau)$ is winning on $(\mathcal{A} \times \mathcal{B}, \mathcal{W}_{\mathcal{A}\times\mathcal{B}})$.*

Proof. Let s be an augmented thread of $\mathbf{pair}(\sigma, \tau)$. Then s is either an augmented thread of σ on $\mathcal{A}$, in which case $s \in \mathcal{W}_\mathcal{A}$, or an augmented thread of τ on $\mathcal{B}$, in which case $s \in \mathcal{W}_\mathcal{B}$. Therefore we get $s \in \mathcal{W}_{\mathcal{A}\times\mathcal{B}}$, and so $\mathbf{pair}(\sigma, \tau)$ is winning. □

Lemma 6.11. *Strategies $\mathbf{proj}_1$ and $\mathbf{proj}_2$ are winning respectively on $(\mathcal{A} \times \mathcal{B} \to \mathcal{A}, \mathcal{W}_{\mathcal{A}\times\mathcal{B}\to\mathcal{A}})$ and $(\mathcal{A} \times \mathcal{B} \to \mathcal{B}, \mathcal{W}_{\mathcal{A}\times\mathcal{B}\to\mathcal{B}})$.*

Proof. We prove the case of $\mathbf{proj}_1$, the other case being similar. Let $\mathcal{A}^{(1)} \times \mathcal{B} \to \mathcal{A}^{(2)}$ denote the arena $\mathcal{A} \times \mathcal{B} \to \mathcal{A}$. Let s be an augmented thread of $\mathbf{proj}_1$ such that $\text{Threads}\left(s_{|\mathcal{A}^{(1)}\times\mathcal{B}}\right) \subseteq \mathcal{W}_{\mathcal{A}\times\mathcal{B}}$. By definition of $\mathbf{proj}_1$, $s_{|\mathcal{B}} = \epsilon$ and $s_{|\mathcal{A}^{(1)}} = s_{|\mathcal{A}^{(2)}}$. Since s is a thread, $s_{|\mathcal{A}^{(2)}}$ is a thread and $s_{|\mathcal{A}^{(1)}} = s_{|\mathcal{A}^{(2)}}$ is also a thread. Since $s_{|\mathcal{B}} = \epsilon$, $s_{|\mathcal{A}^{(1)}\times\mathcal{B}} = s_{|\mathcal{A}^{(1)}}$ is a thread on $\mathcal{A}^{(1)}$ so by definition of $\mathcal{W}_{\mathcal{A}\times\mathcal{B}}$, $s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_\mathcal{A}$. Finally, $s_{|\mathcal{A}^{(2)}} = s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_\mathcal{A}$. □

Lemma 6.12. *The strategy i is winning on $(\mathcal{V} \to \mathcal{A}, \mathcal{W}_{\mathcal{V}\to\mathcal{A}})$.*

Proof. i is the strategy that answers the unique move b of $\mathcal{V}$ to any root move a of $\mathcal{A}$. Therefore, if s is an augmented thread of i, then $s = ab$ for some root a of $\mathcal{A}$, so $s_{|\mathcal{V}} = b$ is losing on $(\mathcal{V}, \mathcal{W}_{\mathcal{V}})$ and therefore $s \in \mathcal{W}_{\mathcal{V} \to \mathcal{A}}$. □

Lemma 6.13. *The strategy ∇ is winning on $(\mathcal{A} \parr \mathcal{A} \to \mathcal{A}, \mathcal{W}_{\mathcal{A} \parr \mathcal{A} \to \mathcal{A}})$.*

Proof. Let $\mathcal{A}^{(1)} \parr \mathcal{A}^{(2)} \to \mathcal{A}^{(3)}$ denote the arena $\mathcal{A} \parr \mathcal{A} \to \mathcal{A}$. Let s be an augmented thread of ∇ such that $\text{Threads}\left(s_{|\mathcal{A}^{(1)} \parr \mathcal{A}^{(2)}}\right) \subseteq \mathcal{W}_{\mathcal{A} \parr \mathcal{A}}$. Since the strategy ∇ merges the plays on the left into a play on the right, $s_{|\mathcal{A}^{(3)}}$ is the interleaving of $s_{|\mathcal{A}^{(1)}}$ and $s_{|\mathcal{A}^{(2)}}$. Since s is a thread, $s_{|\mathcal{A}^{(3)}}$ is a thread and $s_{|\mathcal{A}^{(1)}}$ and $s_{|\mathcal{A}^{(2)}}$ are also threads. Therefore $s_{|\mathcal{A}^{(1)} \parr \mathcal{A}^{(2)}} \in \mathcal{W}_{\mathcal{A} \parr \mathcal{A}}$. By definition of $\mathcal{W}_{\mathcal{A} \parr \mathcal{A}}$, either $s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_{\mathcal{A}}$, or $s_{|\mathcal{A}^{(2)}} \in \mathcal{W}_{\mathcal{A}}$. Since $s_{|\mathcal{A}^{(1)}} \in \mathcal{W}_{\mathcal{A}}$ and $s_{|\mathcal{A}^{(2)}} \in \mathcal{W}_{\mathcal{A}}$ are P-subpositions of $s_{|\mathcal{A}^{(3)}}$, we get in both cases $s_{|\mathcal{A}^{(3)}} \in \mathcal{W}_{\mathcal{A}}$. □

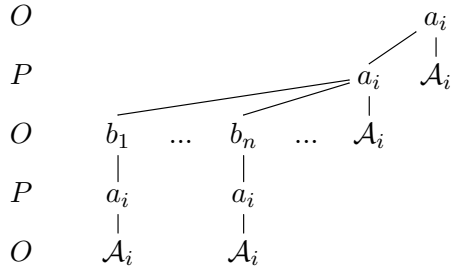
In [Sel01], Selinger presents control categories as algebras, providing object and morphism constructors. Using lemmas 6.8, 6.9, 6.10, 6.11, 6.12 and 6.13, we can prove that winning conditions are preserved under all the morphism constructors of control categories, most of these morphisms being the identity in the case of HO games. This result will be helpful to prove the adequacy lemma in the propositional case.

The following lemma on the interpretation of cc (see section 3.1.1) illustrates the use of winning conditions and justifies the notions of O- and P-subpositions in order to interpret classical logic through the law of Peirce. Note however that it is also a consequence of the lemmas above.

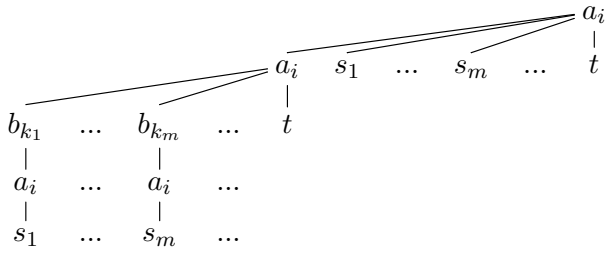
Lemma 6.14. *For any winning conditions $\mathcal{W}_{\mathcal{A}}$ and $\mathcal{W}_{\mathcal{B}}$, cc is winning on the arena*

$$\left(((\mathcal{A} \to \mathcal{B}) \to \mathcal{A}) \to \mathcal{A}, \mathcal{W}_{((\mathcal{A} \to \mathcal{B}) \to \mathcal{A}) \to \mathcal{A}}\right)$$

Proof. First, write $\left(\left(\mathcal{A}^{(1)} \to \mathcal{B}\right) \to \mathcal{A}^{(2)}\right) \to \mathcal{A}^{(3)}$ for the arena $((\mathcal{A} \to \mathcal{B}) \to \mathcal{A}) \to \mathcal{A}$. Let now s be an augmented thread of cc, which is even or infinite since cc is a total strategy. Since s is a thread and player never plays in $\mathcal{B}$, s is a position on the subarena:



where $\mathcal{A}_i$ is a forest such that $\begin{matrix} a_i \\ | \\ \mathcal{A}_i \end{matrix}$ is one of the trees of the arena $\mathcal{A}$, and where $b_1, ..., b_n$ are the roots of the arena $\mathcal{B}$. The position s is then of the following form:



where the s_j and t are positions on the arena $\mathcal{A}_i$. With these notations we have:

$$\mathrm{Pos}\left(s_{|\mathcal{A}^{(1)}}\right) = \left\{ \begin{array}{ccc} a_i & & a_i \\ | & ;...; & | \\ s_1 & & s_m \end{array} ;... \right\} \text{ and } \mathrm{Pos}\left(s_{|\mathcal{A}^{(2)}}\right) = \left\{ \begin{array}{c} a_i \\ | \\ t \end{array} \right\}$$

which are all P-subpositions of:

$$s_{|\mathcal{A}^{(3)}} = \quad \begin{array}{ccccccc} & & & & & & a_i \\ & & & & & & | \\ s_1 & & ... & & s_m & ... & t \end{array}$$

Suppose now that $s_{|(\mathcal{A}^{(1)}\to\mathcal{B})\to\mathcal{A}^{(2)}}$ is winning. Since:

$$s_{|(\mathcal{A}^{(1)}\to\mathcal{B})\to\mathcal{A}^{(2)}} = \quad \begin{array}{ccccccc} & & & & & & a_i \\ & & & & & & | \\ b_{k_1} & & ... & & b_{k_m} & ... & t \\ | & & & & | & & \\ a_i & & ... & & a_i & ... & \\ | & & & & | & & \\ s_1 & & ... & & s_m & ... & \end{array}$$

is a position, this means that $s_{|(\mathcal{A}^{(1)}\to\mathcal{B})\to\mathcal{A}^{(2)}} \in \mathcal{W}_{(\mathcal{A}\to\mathcal{B})\to\mathcal{A}}$, so:

$$\begin{array}{c} a_i \\ | \\ t \end{array} \in \mathcal{W}_{\mathcal{A}} \text{ or for some } 1 \leq j \leq m, \begin{array}{c} b_{k_j} \\ | \\ a_i \\ | \\ s_j \end{array} \notin \mathcal{W}_{\mathcal{A}\to\mathcal{B}}$$

In the first case, since $a_i t$ is a P-subposition of $s_{|\mathcal{A}^{(3)}}$ we get $s_{|\mathcal{A}^{(3)}} \in \mathcal{W}_{\mathcal{A}}$, and in the second case we get $a_i s_j \in \mathcal{W}_{\mathcal{A}}$, and since $a_i s_j$ is a P-subposition of $s_{|\mathcal{A}^{(3)}}$ we get also $s_{|\mathcal{A}^{(3)}} \in \mathcal{W}_{\mathcal{A}}$. □

6.2.4 $\mathcal{W}_{\mathcal{A}\to\mathcal{B}}$ versus Kleene arrow

Let $\mathcal{A}$, $\mathcal{B}$ be arenas equipped with winning conditions $\mathcal{W}_{\mathcal{A}}$, $\mathcal{W}_{\mathcal{B}}$. We define here a strategy σ on $\mathcal{A}\to\mathcal{B}$ such that for any winning strategy τ on $\mathcal{A}$, $\sigma(\tau)$ is winning on $\mathcal{B}$, but σ is not winning on $\mathcal{A}\to\mathcal{B}$. Hence the arrow on winning conditions differs from the usual Kleene realizability arrow (see [Tro98]).
We choose $\mathcal{A}$ and $\mathcal{B}$ to be the same arena $\mathcal{A}$ consisting of one root with three children $\sharp$, $\flat$ and $\natural$, equipped with the winning condition

$$\mathcal{W}_{\mathcal{A}} = \left\{ \begin{array}{ccccc} & & q^O & & \\ & \swarrow & | & \searrow & \\ a_1^P & & a_2^P & & ... \end{array} \;\middle|\; \exists i, a_i \in \{\sharp, \natural\} \right\}$$

where the positions may be finite or infinite. We define a strategy σ on $\mathcal{A}\to\mathcal{A}$ such that for any τ winning on $(\mathcal{A}, \mathcal{W}_{\mathcal{A}})$, $\sigma(\tau)$ is winning on $(\mathcal{A}, \mathcal{W}_{\mathcal{A}})$, but σ is not winning on $(\mathcal{A}\to\mathcal{A}, \mathcal{W}_{\mathcal{A}\to\mathcal{A}})$. σ is the innocent strategy defined by the views:

$$\begin{array}{ccccc} \mathcal{A} & & q^O & & \sharp^P \\ \uparrow & & & \nearrow & \\ \mathcal{A} & & q^P{}^\frown a^O & & q^P{}^\frown a^O \end{array} \qquad \begin{array}{ccccc} \mathcal{A} & & q^O & & \flat^P \\ \uparrow & & & \nearrow & \\ \mathcal{A} & & q^P{}^\frown a^O & & q^P{}^\frown b^O \end{array}$$

where a and b are distinct moves. The interaction with any single threaded strategy produces the left view, and so the projection $q^O\sharp^P$ is winning, but the right view (which will never happen in an interaction with a single-threaded strategy) with $a = \sharp$ and $b = \natural$ is losing, so σ is losing.

6.3 Realizability

We fix a first-order signature Σ, a Σ-structure $\mathcal{M}$, a $\lambda\mu$ signature and interpretations of logic in $\lambda\mu$-calculus and $\lambda\mu$-calculus in $\mathcal{G}$ with the same requirements as in section 5.1.2.

Suppose we have for each predicate P of Σ and each $a_1, ..., a_n \in {T_1}^{\mathcal{M}} \times ... \times {T_n}^{\mathcal{M}}$ a winning condition $\mathcal{W}_{P(a_1,...,a_n)}$ on the arena $[P^*]$. We extend this to give a winning condition $\mathcal{W}_A$ on $[A^*]$ for every closed formula A on Σ with parameters in $\mathcal{M}$:

$$\mathcal{W}_\bot \triangleq \mathcal{W}_\mathcal{V} \qquad \mathcal{W}_{A\Rightarrow B} \triangleq \mathcal{W}_{A\to B} \qquad \mathcal{W}_{A\wedge B} \triangleq \mathcal{W}_{A\times B} \qquad \mathcal{W}_{\forall x^T\, A} \triangleq \bigcap_{a\in T^\mathcal{M}} \mathcal{W}_{A\{a/x\}}$$

$\mathcal{W}_\bot$, $\mathcal{W}_{A\to B}$ and $\mathcal{W}_{A\times B}$ are winning conditions by lemma 6.4, and $\mathcal{W}_{\forall x^T\, A}$ is a winning condition by lemma 6.5.

We finally define the realizability relation $\Vdash$ between a strategy σ on the arena $[A^*]$ and the closed formula A on Σ with parameters in $\mathcal{M}$:

$$\sigma \Vdash A \quad \triangleq \quad \sigma \text{ is winning on } [A^*] \text{ equipped with } \mathcal{W}_A$$

6.3.1 Adequacy for first-order logic

We now fix a first-order theory $\mathcal{A}x$ on Σ and a term M_A for each $A \in \mathcal{A}x$, as in section 3.1.3. The adequacy lemma is as follows:

Lemma 6.15. *Suppose that for each $A \in \mathcal{A}x$ (which is a closed formula) we have:*

$$M_A \Vdash A$$

Then for any proof π of a sequent $A_1, ..., A_n \vdash A \mid B_1, ..., B_m$ in the theory $\mathcal{A}x$ and for any substitution θ of an element $a \in T^\mathcal{M}$ for each individual variable x^T in $FV(A_1, ..., A_n, A, B_1, ..., B_m)$ we have:

$$\pi^* \text{ is winning on } [{A_1}^*] \times ... \times [{A_n}^*] \to [A^*] \parr [{B_1}^*] \parr ... \parr [{B_m}^*]$$
$$\text{equipped with } \mathcal{W}_{A_1\{\theta\}\times...\times A_n\{\theta\}\to A\{\theta\}\parr B_1\{\theta\}\parr...\parr B_m\{\theta\}}$$

Proof. For the propositional part it is a consequence of the remark following lemmas 6.8, 6.9, 6.10, 6.11, 6.12 and 6.13: every morphism constructor of control categories preserves winningness. The remaining rules are those of quantifications:

- For the introduction of $\forall$, let π be a proof of $\Gamma \vdash A \mid \Delta$ and $x^T \notin \mathrm{FV}(\Gamma, \Delta)$. Let s be an augmented thread of π^* such that:

  $$\mathrm{Threads}\left(s_{|\Gamma^*}\right) \subseteq \mathcal{W}_{\Gamma\{\theta\}}$$

  We have to prove that $s_{|A^*\parr\Delta^*} \in \mathcal{W}_{(\forall x^T\, A)\{\theta\}\parr\Delta\{\theta\}}$, that is:

  $$s_{|A^*} \in \mathcal{W}_{(\forall x^T\, A)\{\theta\}} \qquad \text{or} \qquad s_{|\Delta^*} \in \mathcal{W}_{\Delta\{\theta\}}$$

  Suppose $s_{|\Delta^*} \notin \mathcal{W}_{\Delta\{\theta\}}$, so we have to prove $s_{|A^*} \in \mathcal{W}_{(\forall x^T\, A)\{\theta\}}$. Let $a \in T^\mathcal{M}$ and let $\theta' = \theta \cup \{a/x\}$. Since $x \notin \mathrm{FV}(\Gamma)$, $\Gamma\{\theta'\} = \Gamma\{\theta\}$, therefore $\mathrm{Threads}\left(s_{|\Gamma^*}\right) \subseteq \mathcal{W}_{\Gamma\{\theta'\}}$ and the induction hypothesis gives $s_{|A^*\parr\Delta^*} \in \mathcal{W}_{A\{\theta'\}\parr\Delta\{\theta'\}}$. Since $x \notin \mathrm{FV}(\Delta)$, $\Delta\{\theta'\} = \Delta\{\theta\}$ so we get $s_{|A^*\parr\Delta^*} \in \mathcal{W}_{A\{\theta'\}\parr\Delta\{\theta\}}$, and since we supposed that $s_{|\Delta^*} \notin \mathcal{W}_{\Delta\{\theta\}}$ we obtain $s_{|A^*} \in \mathcal{W}_{A\{\theta'\}}$. Finally, this is true for every $a \in T^\mathcal{M}$, so we get:

  $$s_{|A^*} \in \mathcal{W}_{(\forall x^T\, A)\{\theta\}}$$

- For the elimination of $\forall$, let π be a proof of $\Gamma \vdash \forall x^T A \mid \Delta$, let t^T be a first-order term, and let s be an augmented thread of π^* such that:

$$\mathrm{Threads}\left(s_{|\Gamma^*}\right) \subseteq \mathcal{W}_{\Gamma\{\theta\}}$$

Since (modulo α-conversion) $x \notin \mathrm{FV}(\Gamma)$, we have $\Gamma\{\theta\} = \Gamma\left\{t^T\{\theta\}/x^T\right\}\{\theta\}$ and so by induction hypothesis we get:

$$s_{|A^* \parr \Delta^*} \in \mathcal{W}_{A\{t^T\{\theta\}/x^T\}\{\theta\} \parr \Delta\{t^T\{\theta\}/x^T\}\{\theta\}}$$

Since $A\left\{t^T\{\theta\}/x^T\right\}\{\theta\} = \left(A\left\{t^T/x^T\right\}\right)\{\theta\}$ and (again modulo α-conversion) $x \notin \mathrm{FV}(\Gamma)$ so $\Gamma\{\theta\} = \Gamma\left\{t^T\{\theta\}/x^T\right\}\{\theta\}$, we get finally:

$$s_{|A^* \parr \Delta^*} \in \mathcal{W}_{(A\{t^T/x^T\})\{\theta\} \parr \Delta\{\theta\}}$$

□

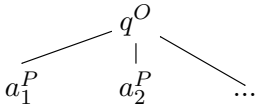
6.3.2 Adequacy for Peano arithmetic

We fix now the theory to be $PA^{\omega \mathrm{r}}$ on $\Sigma_{PA^{\omega \mathrm{r}}}$, $\mathcal{M}$ to be a model of $PA^{\omega \mathrm{r}}$, the signature of $\lambda\mu$-calculus to be that of μPCF, the interpretation of $PA^{\omega \mathrm{r}}$ in μPCF to be that of section 3.1.3 for the predicates and of section 3.4.1 for the axioms, and the interpretation of μPCF in $\mathcal{G}$ to be that of section 4.4. We also make the same requirements on $\mathcal{M}$ as in section 5.2.2.

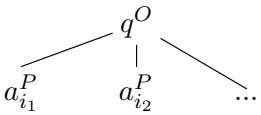
The winning conditions for the two predicates are:

$$\mathcal{W}_{a \neq_T b} \triangleq \begin{cases} \emptyset & \text{if } a = b \\ \{q\} & \text{otherwise} \end{cases} \qquad \mathcal{W}_{(\!|a^\iota|\!)} \triangleq \left\{ \begin{array}{c} q^O \\ | \\ a_1^P \quad a_2^P \quad \ldots \end{array} \;\middle|\; \exists i, a_i = a \right\}$$

Since $(\neq_T)^* = 0$ and $[0] = \mathcal{V}$ is the one move arena, on which there is only one thread, q, $\{q\}$ is the set of all threads on $[\neq_T{}^*]$ and is therefore trivially a winning condition. $\mathcal{W}_{(\!|a^\iota|\!)}$ is a winning condition since the position:



that may be finite or infinite has only itself as O-subposition and



for $1 \leq i_1 < i_2 < \ldots$ as P-subpositions, and if it is winning then some a_{i_k} is equal to a.

First, as in the previous model, the identity realizes the equalities which are true in the model:

Lemma 6.16. *Let t^T and u^T be first-order terms with $FV\left(t^T, u^T\right) = \vec{x}^{\vec{U}}$.*

$$\textit{If } \mathcal{M} \vDash \forall \vec{x}^{\vec{U}}\, t^T = u^T \textit{ then } \lambda x.x \Vdash \forall \vec{x}^{\vec{U}}\, t^T = u^T$$

Proof. Let $\vec{a} \in \vec{U}^{\mathcal{M}}$. Since $\mathcal{M} \vDash \forall \vec{x}^{\vec{U}}\, t^T = u^T$, we have $\left(t^T\{\vec{a}/\vec{x}\}\right)^{\mathcal{M}} = \left(u^T\{\vec{a}/\vec{x}\}\right)^{\mathcal{M}}$, so:

$$\mathcal{W}_{t^T\{\vec{a}/\vec{x}\} \neq u^T\{\vec{a}/\vec{x}\}} = \emptyset = \mathcal{W}_\perp$$

Therefore by lemma 6.8, $\lambda x.x = \mathbf{Id}_{\mathcal{V}}$ is winning on:

$$\mathcal{W}_{t^T\{\vec{a}/\vec{x}\} = u^T\{\vec{a}/\vec{x}\}} = \mathcal{W}_{t^T\{\vec{a}/\vec{x}\} \neq u^T\{\vec{a}/\vec{x}\} \Rightarrow \perp} = \mathcal{W}_{\perp \Rightarrow \perp}$$

and so $\lambda x.x \Vdash \forall \vec{x}^{\vec{U}}\, t^T = u^T$. □

Therefore, since $\mathcal{M}$ is a model of $PA^{\omega \mathrm{r}}$, we have immediately the following results:

$$\begin{aligned}
M_{(\boldsymbol{refl})} &= \lambda x.x \Vdash \forall x^T\,(x =_T x)\\
M_{(\boldsymbol{\Delta s})} &= \lambda x.x \Vdash \forall x^{T\to U\to V} \forall y^{T\to U} \forall z^T\,(\mathsf{s}\,x\,y\,z =_V x\,z\,(y\,z))\\
M_{(\boldsymbol{\Delta k})} &= \lambda x.x \Vdash \forall x^T \forall y^U\,(\mathsf{k}\,x\,y =_T x)\\
M_{(\boldsymbol{\Delta rec0})} &= \lambda x.x \Vdash \forall x^T \forall y^{\iota\to T\to T}\,(\mathsf{rec}\,x\,y\,\mathsf{0} =_T x)\\
M_{(\boldsymbol{\Delta recS})} &= \lambda x.x \Vdash \forall x^T \forall y^{\iota\to T\to T} \forall z^\iota\,(\mathsf{rec}\,x\,y\,(\mathsf{S}\,z) =_T y\,z\,(\mathsf{rec}\,x\,y\,z))
\end{aligned}$$

The non-confusion axiom and Leibniz scheme are easy:

Lemma 6.17.

$$\begin{aligned}
M_{(\boldsymbol{Snz})} &= \Omega \Vdash \forall x^\iota\,(\mathsf{S}\,x \neq_\iota \mathsf{0})\\
M_{(\boldsymbol{Leib})} &= \lambda x.x \Vdash \forall \vec{z}^{\vec{U}}\, \forall x^T\, \forall y^T\,(\neg A \Rightarrow A\{y/x\} \Rightarrow x \neq_T y)
\end{aligned}$$

Proof. Let $n \in \iota^{\mathcal{M}}$, we have $(\mathsf{S}\,n)^{\mathcal{M}} = n+1 \neq 0 = \mathsf{0}^{\mathcal{M}}$, so $\mathcal{W}_{\mathsf{S}\,n \neq \mathsf{0}} = \{q\}$ is the set of all threads on $[0] = \mathcal{V}$, so any strategy realizes $\forall x^\iota\,(\mathsf{S}\,x \neq_\iota \mathsf{0})$.

Let now $\vec{a} \in \vec{U}^{\mathcal{M}}$ and $b, c \in T^{\mathcal{M}}$. If $b \neq c$, then $\mathcal{W}_{b \neq c} = \{q\}$, which is the set of all threads on $[0]$, so for any thread s on $[(A^* \to 0) \to A^* \to 0]$, the projection of s on the rightmost $\mathcal{V}$ is in $\mathcal{W}_{b \neq c}$ and therefore s is in $\mathcal{W}_{\neg A\{b/x,\vec{a}/\vec{z}\} \Rightarrow A\{c/x,\vec{a}/\vec{z}\} \Rightarrow b \neq c}$. Otherwise, if $b = c$, then $\mathcal{W}_{b \neq c} = \mathcal{W}_\perp$ and $\mathcal{W}_{A\{b/x,\vec{a}/\vec{z}\}} = \mathcal{W}_{A\{c/x,\vec{a}/\vec{z}\}}$, so:

$$\mathcal{W}_{\neg A\{b/x,\vec{a}/\vec{z}\}} = \mathcal{W}_{A\{c/x,\vec{a}/\vec{z}\} \Rightarrow b \neq c}$$

and by lemma 6.8 $M_{(\boldsymbol{Leib})} = \lambda x.x = \mathbf{Id}_{[A^* \to 0]} \Vdash \neg A\,\{b/x, \vec{a}/\vec{z}\} \Rightarrow A\,\{c/x, \vec{a}/\vec{z}\} \Rightarrow b \neq c$. □

The induction axiom scheme is a little bit more complicated:

Lemma 6.18.

$$M_{(\boldsymbol{ind^r})} = \mathsf{rec} \Vdash \forall \vec{y}^{\vec{U}}\,(A\,\{\mathsf{0}/x\} \Rightarrow \forall^{\mathrm{r}} x^\iota\,(A \Rightarrow A\,\{\mathsf{S}\,x/x\}) \Rightarrow \forall^{\mathrm{r}} x^\iota A)$$

Proof. It will be useful in the proof to write:

$$\begin{aligned}
M_{(\boldsymbol{ind^r})} = \mathsf{rec} &= \lambda xy.\mathsf{Y}\,(\lambda uz.\mathsf{if}_0\,z\,x\,(y\,(\mathsf{pred}\,z)\,(u\,(\mathsf{pred}\,z))))\\
&= \lambda xy.\lambda z.\mathsf{if}_0\,z\,x\,(y\,(\mathsf{pred}\,z)\,(\mathsf{rec}\,x\,y\,(\mathsf{pred}\,z)))\\
&= (\lambda uxy.\lambda z.\mathsf{if}_0\,z\,x\,(y\,(\mathsf{pred}\,z)\,(u\,x\,y\,(\mathsf{pred}\,z))))\,\mathsf{rec}\\
&= \mathsf{rec};(\lambda uxyz.\mathsf{if}_0\,z\,x\,(y\,(\mathsf{pred}\,z)\,(u\,x\,y\,(\mathsf{pred}\,z))))
\end{aligned}$$

where ";" is the categorical composition and:

$\lambda uxyz.\mathsf{if}_0\, z\, x\, (y\, (\mathsf{pred}\, z)\, (u\, x\, y\, (\mathsf{pred}\, z)))$

$$: (A^* \to (\iota \to A^* \to A^*) \to \iota \to A^*) \to A^* \to (\iota \to A^* \to A^*) \to \iota \to A^*$$

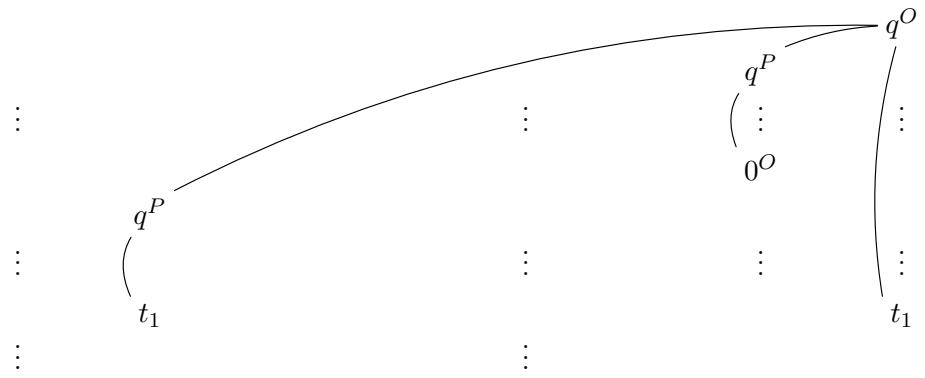
We will also write:

$$\mathcal{B} = [A^*]^{(1)} \to \left([\iota]^{(1)} \to [A^*]^{(2)} \to [A^*]^{(3)}\right) \to [\iota]^{(2)} \to [A^*]^{(4)}$$

so $\lambda uxyz.\mathsf{if}_0\, z\, x\, (y\, (\mathsf{pred}\, z)\, (u\, x\, y\, (\mathsf{pred}\, z)))$ is a strategy on $\mathcal{B} \to \mathcal{B}$ (that we write $\mathcal{B}^{(1)} \to \mathcal{B}^{(2)}$, and we write therefore $[A^*]^{(14)}$, $[\iota]^{(21)}$, ...). Let now s be an augmented thread of $M_{(\boldsymbol{ind^r})} = \mathsf{rec}$. There exists an augmented thread t of $\lambda uxyz.\mathsf{if}_0\, z\, x\, (y\, (\mathsf{pred}\, z)\, (u\, x\, y\, (\mathsf{pred}\, z)))$ such that $t_{|\mathcal{B}^{(1)}}$ is an augmented play of rec and $t_{|\mathcal{B}^{(2)}} = s$. Suppose $s_{|A^{*(1)}}$ is winning on $A\{\mathsf{0}/x\}$ and suppose $s_{|\iota^{(1)} \to A^{*(2)} \to A^{*(3)}}$ is winning on $\forall^{\mathrm{r}} x^{\iota}\, (A \Rightarrow A\{\mathsf{S}\, x/x\})$. We prove by induction on $n \in \mathbb{N}$ that if $s_{|\iota^{(2)}}$ is winning on $(\!|n|\!)$, then $s_{|A^{*(4)}} \in \mathcal{W}_{A\{n/x\}}$:

- $n = 0$: $s_{|\iota^{(2)}}$ being winning on $(\!|\mathsf{0}|\!)$ means that every q^P in $s_{|[\iota]^{(2)}} = t_{|[\iota]^{(22)}}$ has some 0^O pointing to it, t has then the following shape:

$$\mathcal{B}^{(1)} \to [A^*]^{(21)} \to \left([\iota]^{(21)} \to [A^*]^{(22)} \to [A^*]^{(23)}\right) \to [\iota]^{(22)} \to [A^*]^{(24)}$$



  Moreover, keeping only the moves written in the figure gives a P-subthread of t. We have $q^P t_1 \in \mathrm{Threads}\left(s_{|[A^*]^{(1)}}\right)$ so $q^P t_1 \in \mathcal{W}_{A\{\mathsf{0}/x\}}$, therefore $q^O t_1 \in \mathcal{W}_{A\{\mathsf{0}/x\}}$, but $q^O t_1$ is a P-subthread of $t_{|A^{*(4)}} = s_{|A^{*(4)}}$, therefore $s_{|A^{*(4)}} \in \mathcal{W}_{A\{\mathsf{0}/x\}}$.

- $n + 1$: $s_{|\iota^{(2)}}$ being winning on $(\!|n+1|\!)$ means that every q^P in $s_{|[\iota]^{(2)}} = t_{|[\iota]^{(22)}}$ has some $(n+1)^O$ pointing to it, t has then the shape given in figure 6.1. The pairs of threads $q^O t_i / q^P t_i$ $(i = 2, 3, 4)$ and $q^O n^P / q^P\, (n+1)^P$ may appear several times, and keeping only the moves written in the figure gives a P-subthread of t. $q^O t_3 = q^P t_3 \in \mathcal{W}_{A\{\mathsf{0}/x\}}$ and $q^O t_4 = q^P t_4 \in \mathcal{W}_{\forall^{\mathrm{r}} x^{\iota}(A \Rightarrow A\{\mathsf{S}\, x/x\})}$ by hypothesis. Since $s_{|[\iota]^{(2)}} = t_{|[\iota]^{(22)}}$ is winning on $(\!|n+1|\!)$, $t_{|[\iota]^{(12)}}$ is winning on $(\!|n|\!)$, and since $t_{|\mathcal{B}^{(1)}}$ is an extended play of rec, we have by induction hypothesis $q^O t_2 = q^P t_2 \in \mathcal{W}_{A\{n/x\}}$. Again, $s_{|[\iota]^{(2)}} = t_{|[\iota]^{(22)}}$ is winning on $(\!|n+1|\!)$ so $t_{|[\iota]^{(21)}} = s_{|[\iota]^{(1)}}$ is winning on $(\!|n|\!)$. Therefore, since $s_{|[\iota]^{(1)} \to [A^*]^{(2)} \to [A^*]^{(3)}} \in \mathcal{W}_{\forall^{\mathrm{r}} x^{\iota}(A \Rightarrow A\{\mathsf{S}\, x/x\})}$ we get $q^O t_1 = q^P t_1 \in \mathcal{W}_{A\{n+1/x\}}$.

□

The last axioms are the relativization ones:

Lemma 6.19.

$$M_{(\!|\mathsf{0}|\!)} = \overline{\mathsf{0}} \Vdash (\!|\mathsf{0}|\!) \qquad M_{(\!|\mathsf{S}|\!)} = \mathsf{succ} \Vdash (\!|\mathsf{S}|\!)$$

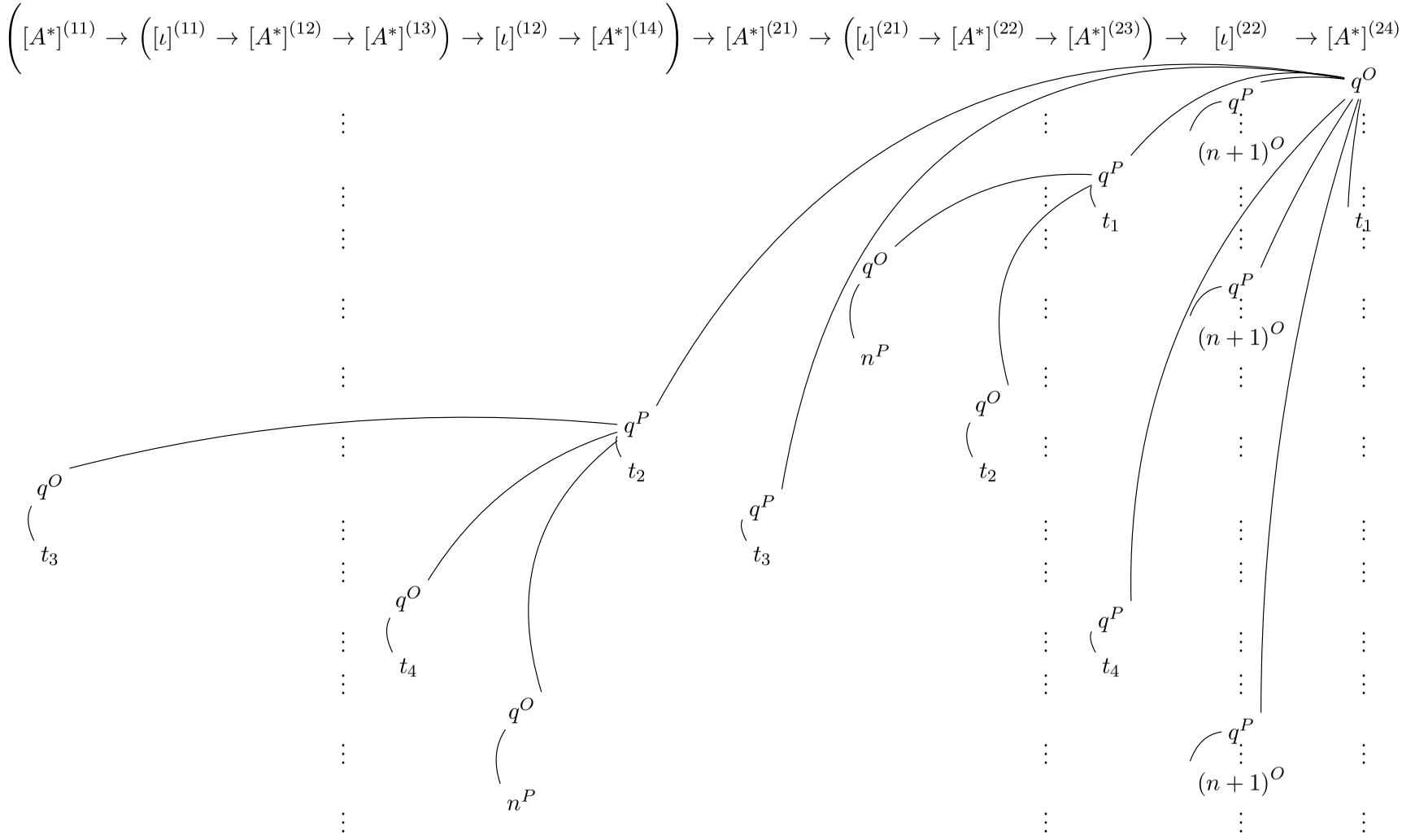
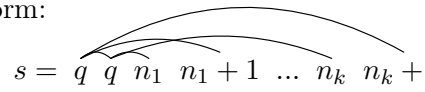


Figure 6.1: an extended interaction between rec and $\lambda uxyz.\mathsf{if}_0\, z\, x\, (y\, (\mathsf{pred}\, z)\, (u\, x\, y\, (\mathsf{pred}\, z)))$

Proof. The only augmented thread of $\overline{0}$ is $\overset{\frown}{q\ 0}$, which is obviously in $\mathcal{W}_{(\!|0|\!)}$. The augmented threads of succ are of the form:



$$s = q\ q\ n_1\ n_1+1\ \ldots\ n_k\ n_k+1\ \ldots$$

Let $n \in \iota^{\mathcal{M}}$. First, $s_{|[\iota]^{(1)}}$ is a thread. If $s_{|[\iota]^{(1)}} \in \mathcal{W}_{(\!|n|\!)}$, then $n_i = n$ for some i, and so $n_i + 1$ appears in $s_{|[\iota]^{(2)}}$. Therefore $s_{|[\iota]^{(2)}} \in \mathcal{W}_{(\!|n+1|\!)} = \mathcal{W}_{(\!|(\mathsf{S}\,n)^{\mathcal{M}}|\!)}$, so $s \in \mathcal{W}_{(\!|n^{\mathcal{M}}|\!)\Rightarrow(\!|(\mathsf{S}\,n)^{\mathcal{M}}|\!)}$. It follows that:

$$s \in \mathcal{W}_{\forall x^{\iota}((\!|x|\!)\Rightarrow(\!|\mathsf{S}\,x|\!))} = \mathcal{W}_{(\!|\mathsf{S}|\!)}$$

□

6.3.3 Extraction through Friedman translation

In order to extract algorithms from Π^0_2-formulas provable in Peano arithmetic, we perform a variant of Friedman translation [Fri78] on classical proofs. Contrary to the previous model, the Friedman translation does not appear here before this section. Indeed, since the model in this chapter is not orthogonality-based, we did not need the notion of realizers and counter-realizers interacting, so the set $\perp\!\!\!\perp$ was unnecessary through the proofs of adequacy. In the original work of Friedman, the translation of an intuitionistic formula A is obtained by replacing each atomic predicate $P\,(t_1, ..., t_n)$ or $\bot$ with the disjunction $P\,(t_1, ..., t_n) \vee Q$ or $\bot \vee Q$, where Q is a fixed predicate. Here, we use our classical system with multi-conclusioned sequents to simply add Q in the conclusion of every sequent. Indeed, in our system the right weakening rule is admissible, while in intuitionistic systems (with only one formula on the right of a sequent) the right weakening does not exist and Friedman translation must be defined inductively on the structure of formulas. Therefore, our translation is a version of Friedman's original one in a classical setting. Moreover, since we want to use it to extract computational content from proofs, we also provide a realizability interpretation for this predicate.

Formally, Friedman translation in our setting amounts to adding a particular, fixed nullary predicate Q in the signature Σ of $PA^{\omega \mathrm{r}}$, then adding this predicate to every sequent of a relativized proof in $PA^{\omega \mathrm{r}}$ the following way:

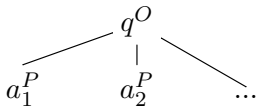
$$\Gamma \vdash A \mid \Delta \qquad \rightsquigarrow \qquad \Gamma \vdash A \mid \Delta, Q$$

and then interpreting these proofs in μPCF by adding a fixed free μ-variable κ to every term of μPCF. This transformation is adequate with respect to realizability, since the right weakening rule is admissible. Because we extract from proofs in PA^{ω} algorithms that compute natural numbers, we choose the type of κ to be $Q^* \triangleq \iota$ (which is the type of natural numbers in PA^{ω}).

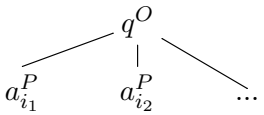
In order to have a realizability interpretation for the new predicate Q, we fix a set $\perp\!\!\!\perp \subseteq \iota^{\mathcal{M}}$. As in section 5.1.2, $\perp\!\!\!\perp$ is intuitively the set of correct values that can be output through the μ-variable κ, so we will fix it wisely in the proofs of correctness of extracted algorithms. The winning condition on ι associated to the predicate Q is then:

$$\mathcal{W}_Q \triangleq \left\{ \begin{array}{c} q^O \\ \diagup \quad | \quad \diagdown \\ a_1^P \quad a_2^P \quad \ldots \end{array} \;\middle|\; \exists i, a_i \in \perp\!\!\!\perp \right\}$$

It is a winning condition since the position:



that may be finite or infinite has only itself as O-subposition and



for $1 \leq i_1 < i_2 < ...$ as P-subpositions, and if it is winning then some a_{i_k} is in $\perp\!\!\!\perp$.

The adequacy lemma for this extension of $PA^{\omega \mathrm{r}}$ with a new predicate Q is immediate since we do not change the axioms and the admissible rule of right weakening preserves realizability (as a consequence of lemma 6.12). Since we added the new predicate Q, a proof π of a closed formula A is now translated to a term $\vdash \pi^* : A^* \mid \kappa : Q^*$ with a free μ-variable κ of type $Q^* = \iota$, that is a strategy in $\mathcal{G}$ from $\mathbf{1}$ to $[A^*] \parr [\iota]$. In order to manipulate these strategies using the syntax of $\lambda\mu$-calculus, we use the convention that such strategies have a free μ-variable κ, so we can write for example $[\kappa]\ \pi^* : \mathbf{1} \to [0] \parr [\iota]$ if $A^* = \iota$, or $\mu\kappa.\pi^* : \mathbf{1} \to [\iota]$ if $A^* = 0$. We now prove the extraction result, in which the equality $t =_U u$ is, as in the previous model, at any type:

Lemma 6.20. *From a proof of* $\vdash \forall x^T \exists y^\iota\ (t =_U u)$ *in* PA^ω*, one can extract a* $\lambda\mu$*-term* $M : T \to \iota$ *such that for any* $a \in T^{\mathcal{M}}$ *and* $\sigma \Vdash (\!|a|\!)$*, there is some* $n \in \mathbb{N}$ *such that:*

$$(t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}} \text{ and } M\,\sigma = \overline{\mathsf{n}}$$

Proof. First, we obtain by double-negation elimination a proof of $\vdash\ \forall x^T \neg\forall y^\iota\ (t \neq_U u)$, and by relativization a proof in $PA^{\omega \mathrm{r}}$ of $\vdash\ \forall^{\mathrm{r}} x^T \neg\forall^{\mathrm{r}} y^\iota\ (t \neq_U u)$ which becomes through Friedman's translation a proof π of $\vdash \forall^{\mathrm{r}} x^T \neg\forall^{\mathrm{r}} y^\iota\ (t \neq_U u) \mid Q$. The adequacy lemma then tells us:

$$\pi^* \text{ is winning on } \forall^{\mathrm{r}} x^T\ (\neg\forall^{\mathrm{r}} y^\iota\ (t \neq_U u)) \parr Q$$

Then, if $a \in T^{\mathcal{M}}$ and $\sigma \Vdash (\!|a|\!)$, we get that $\pi^*\,\sigma$ is winning on $(\neg\forall^{\mathrm{r}} y^\iota\ (t\,\{a/x\} \neq_U u\,\{a/x\})) \parr Q$. Let now fix:

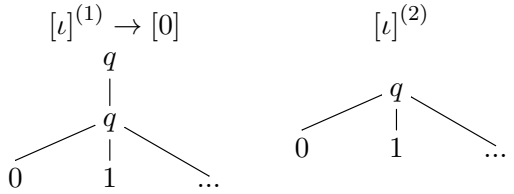
$$\perp\!\!\!\perp = \left\{ n \in \iota^{\mathcal{M}} \;\middle|\; (t\,\{a/x, n/y\})^{\mathcal{M}} = (u\,\{a/x, n/y\})^{\mathcal{M}} \right\}$$

This choice of $\perp\!\!\!\perp$ allows use to prove the following intermediary lemma:

Lemma 6.21. *The strategy* $\vdash \lambda x.\,[\kappa]\ x : \iota \to 0 \mid \kappa : \iota$ *is winning on:*

$$\left(\left([\iota]^{(1)} \to [0] \right) \parr [\iota]^{(2)}, \mathcal{W}_{(\forall^{\mathrm{r}} y^\iota (t\{a/x\} \neq u\{a/x\})) \parr Q} \right)$$

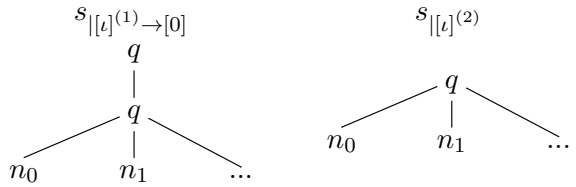
Proof. First the arenas $[\iota]^{(1)} \to [0]$ and $[\iota]^{(2)}$ are:



so the arena $\left([\iota]^{(1)} \to [0] \right) \parr [\iota]^{(2)}$, obtained by merging the roots of $[\iota]^{(1)} \to [0]$ and $[\iota]^{(2)}$, is actually equal to $[\iota]^{(1)} \to [\iota]^{(2)}$, and by lemma 4.16 $\lambda x.\,[\kappa]\ x$ is the identity strategy on $[\iota]$. Let now s be an extended thread of $\lambda x.\,[\kappa]\ x$. We must prove that:

$$s_{|[\iota]^{(1)} \to [0]} \text{ is winning on } \left([\iota]^{(1)} \to [0], \mathcal{W}_{\forall^{\mathrm{r}} y^\iota (t\{a/x\} \neq u\{a/x\})} \right) \text{ or } s_{|[\iota]^{(2)}} \text{ is winning on } \left([\iota]^{(2)}, \mathcal{W}_Q \right)$$

These two projections have the following shape:



There are now two possibilities: if some n_i is in $\perp\!\!\!\perp$, then $s_{|[\iota]^{(2)}}$ is winning on $\left([\iota]^{(2)}, \mathcal{W}_Q\right)$, otherwise we must prove that $s_{|[\iota]^{(1)}\to[0]}$ is winning on:

$$\left([\iota]^{(1)} \to [0], \mathcal{W}_{\forall^{\mathrm{r}} y^\iota (t\{a/x\}\neq u\{a/x\})}\right)$$

so let $n \in \mathbb{N}$ and suppose that $s_{|[\iota]^{(1)}}$ is winning on $\left([\iota]^{(1)}, \mathcal{W}_{(\!|n|\!)}\right)$. By definition of $\mathcal{W}_{(\!|n|\!)}$, n is some n_i, and therefore $n \notin \perp\!\!\!\perp$ so by definition of $\perp\!\!\!\perp$, $(t\{a/x, n/y\})^{\mathcal{M}} \neq (u\{a/x, n/y\})^{\mathcal{M}}$. In that case, $\mathcal{W}_{(t\{a/x,n/y\})\neq(u\{a/x,n/y\})} = \{q\}$ and $s_{|[0]} \in \mathcal{W}_{(t\{a/x,n/y\})\neq(u\{a/x,n/y\})}$. This achieves the proof that $\lambda x.\,[\kappa]\ x$ is winning on:

$$\left(\left([\iota]^{(1)} \to [0]\right) \parr [\iota]^{(2)}, \mathcal{W}_{(\forall^{\mathrm{r}} y^\iota (t\{a/x\}\neq u\{a/x\}))\parr Q}\right)$$

□

Now, using Lemma 6.9 we get that $\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x)$ is winning on $\bot \parr Q$, and therefore $\mu\kappa.\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x)$ is winning on $([\iota], \mathcal{W}_Q)$. This means that $\mu\kappa.\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x)$ is some $\overline{\mathsf{n}}$ such that $n \in \perp\!\!\!\perp$ (so $(t\{a/x, n/y\})^{\mathcal{M}} = (u\{a/x, n/y\})^{\mathcal{M}}$). Indeed, if $\mu\kappa.\pi^*\,\sigma\ (\lambda x.\,[\kappa]\ x)$ is the empty strategy then its only augmented play is q, which is losing on Q. Finally, we have the claimed result with $M \stackrel{\Delta}{=} \lambda u.\mu\kappa.\pi^*\,u\ (\lambda x.\,[\kappa]\ x)$. □

In the particular case of $T = \iota$, we get that for any $m \in \mathbb{N}$, $M\,\overline{\mathsf{m}}$ is some $\overline{\mathsf{n}}$ such that $(t\{m/x, n/y\})^{\mathcal{M}} = (u\{m/x, n/y\})^{\mathcal{M}}$.

6.3.4 About the axiom of choice

In this section, we describe the strategy implementing bar recursion and explain why it does not realize the axiom of choice as-is. We also give ideas for a modified version of bar recursion which would use higher-order references.

We consider here only the simpler axiom of countable choice:

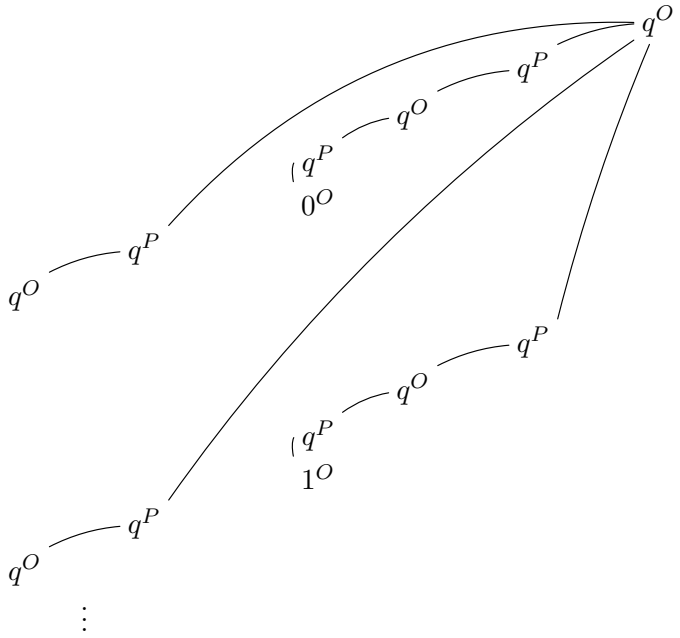
$$\forall^{\mathrm{r}} x^\iota \neg\forall y^T \neg A \Rightarrow \neg\forall z^{\iota\to T} \neg\forall^{\mathrm{r}} x^\iota A\{z\,x/y\}$$

and the potential realizer:

$$\lambda uv.\mathsf{barrec}\,(\lambda y.u\,|y|)\,v\,\epsilon : (\iota \to (A^* \to 0) \to 0) \to ((\iota \to A^*) \to 0) \to 0$$

Here is a particular infinite thread of $\lambda uv.\mathsf{barrec}\,(\lambda y.u\,|y|)\,v\,\epsilon$:

$$\Big(\iota \to \big(\; A^* \to 0 \;\big) \to 0 \Big) \to \Big(\big(\; \iota \to A^* \;\big) \to 0 \Big) \to 0$$



The projections on $(\iota \to (A^* \to 0) \to 0) \to 0$ and $(\iota \to A^*) \to 0$ are both winning since the one-move position is losing on $\bot$ and (in the general case) on $A\,\{n/x, a/y\}$, and therefore the whole thread is losing. The issue is that the projection on $(\iota \to A^*) \to 0$ asks successively for the values of the first, second... elements of the sequence that we are building, and each time this element is outside the part of the sequence that has already been asked, so the bar recursion operator opens a new thread on the left. This behavior is similar to the one described in section 3.5.4. The thread defined here does not correspond to any interaction, since we are in a single-threaded setting, and any realizer of $\forall z^{\iota\to T}\neg\forall^{\mathrm{r}} x^{\iota} A\,\{z\,x/y\}$ asks for the same sequence of xs each time it is started. However, one of our model's peculiarity is that we do not look at interactions between realizers, but rather to their set of traces, which may not correspond to interactions. A particular example of this was given in the remark of the end of section 6.2. In the model defined in the previous chapter, we only looked at interactions happening between strategies, by using the Kleene arrow, therefore the problem described here did not arise and we proved that the bar recursion operator realizes the axiom of choice.

In the context of winning conditions, we can imagine a different bar recursor which would call the realizer of $\forall z^{\iota\to T}\neg\forall^{\mathrm{r}} x^{\iota} A\,\{z\,x/y\}$ only once, but then update the sequence of elements of A^* that it gives to this realizer. This could be achieved by considering a higher-order reference of type $\iota \to A^*$, since our model contains non-innocent strategies, and therefore has the power of higher-order references. A typical thread of such a strategy would look like figure 6.2. We can observe that this thread necessarily comes from a non-innocent strategy, because the q^P move 27 points the q^O move 12 which is not in the current view. Indeed, using references, this bar recursor "remembers" that the value for n_0 has already been asked.

The thread can be interpreted as follows. When the bar recursor is queried at step 1, it gets this value from its second argument at step 2. When this second argument asks for the function at step 3, the bar recursor gets the particular index that is asked at steps $4-5$, and then asks its first argument to provide this value at steps $6-7-8$. When the first argument provides the

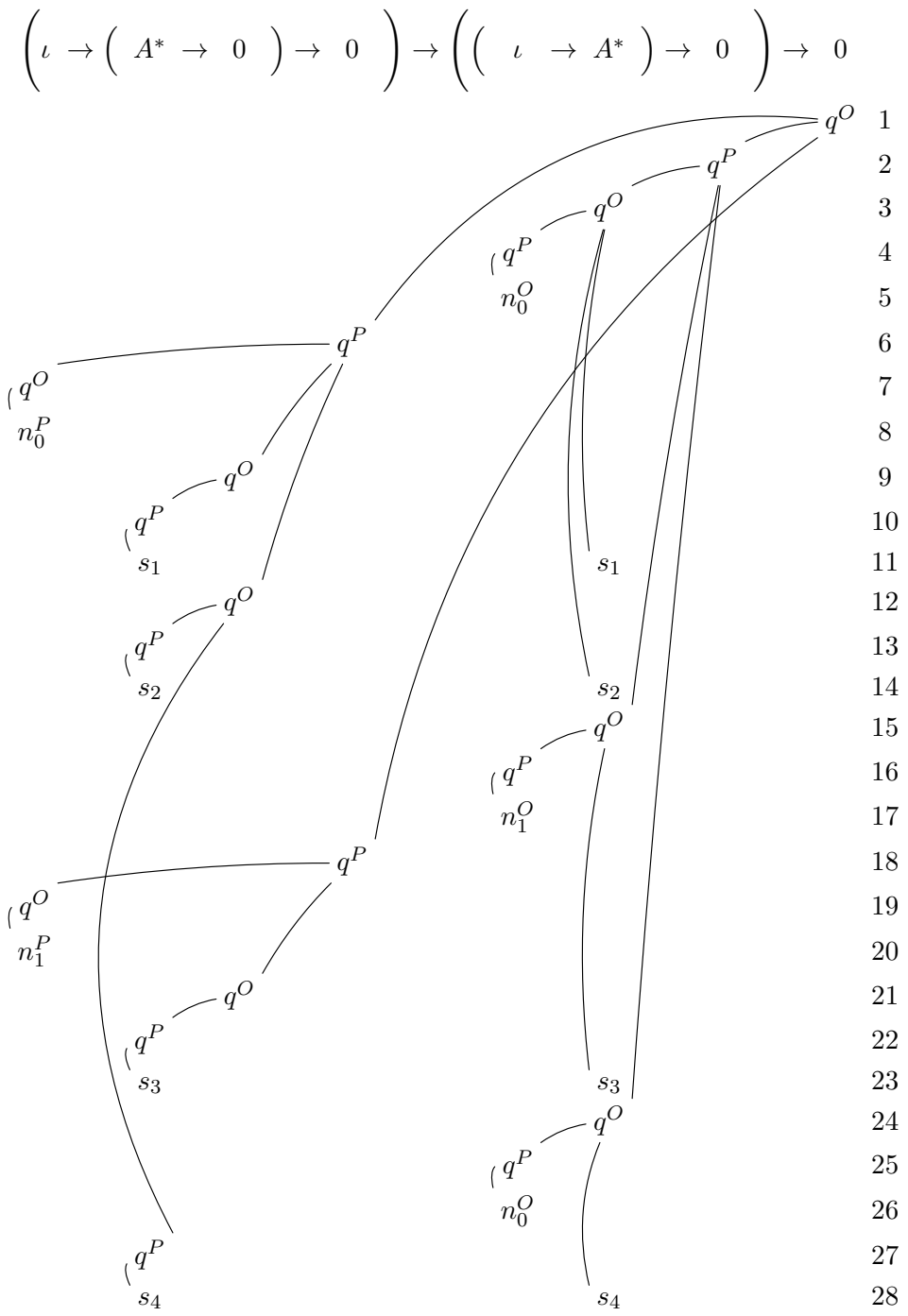


Figure 6.2: a thread of bar recursion with references

(higher-order) value at step 9, the bar recursor performs double-negation-elimination to copy it back to the second argument at steps $10 - 11$. If the first arguments provides a new value at step 12, then this value is also copied back to the second argument at steps $13 - 14$, behaving like the cc operator. If the second argument of bar recursor asks for another index at steps $15 - 16 - 17$, then the bar recursor recognizes that this value has not been asked before and opens a new thread in the first argument at steps $18 - 19 - 20 - 21 - 22 - 23$. However, if the second argument asks again for the element of index n_0 at steps $24 - 25 - 26$, then the bar recursor remembers that it has already been asked and answers with the last value that has been given by the first argument at steps $27 - 28$.

In addition to be relying on references to provide values on the fly, this bar recursor does not build a sequence linearly like usual bar recursion, but instead builds the sequence pointwise. Under this aspect, it seems intuitively much more related to the functional of [BBC98] than to the modified bar recursion of [BO05] that was used in the previous chapter.

Chapter 7

Conclusion

After Kleene defined his untyped realizability [Kle45] for Heyting arithmetic, Kreisel defined his modified realizability [Kre59] by adapting Kleene's ideas to a typed setting. Kreisel's realizability proved to be quite different from Kleene's and opened interesting perspectives. In this thesis we devised a notion of typed realizability for classical logic, while currently the contributions to classical realizability are mainly in the lines of Krivine's untyped models [Kri09]. Also, instead of working in second-order logic, where the comprehension axiom scheme is a consequence of the rules of the logic, we work in first-order logic and realize the axiom of countable choice which gives then another interpretation of the comprehension axiom scheme, computationally different.

The framework of game semantics is very interesting when pursuing this goal. First, by considering non-well-bracketed strategies we get the control features (see [Lai97]) which are common to every direct computational interpretation of classical proofs. This is enlightened by the fact that the category of unbracketed games can be seen as a category of continuations (defined in [Sel01]), and therefore as a model of $\lambda\mu$-calculus. Another feature is that without the innocence requirement, we are potentially able to define realizers with imperative features through the modelling of higher-order references, as shown in [AHM98].

The model based on orthogonality validates the fact that the duality between realizers and counter-realizers works very well in the context of realizability for first-order logic, and proves that the bar recursion operator which was until now used in intuitionistic realizability [BO05] fits also well in a classical setting with orthogonality. On the other side, the structure of the games model where strategies are sets of interactions was heavily exploited when building our model with winning conditions, and the non-sequential nature of these winning conditions was made clear using thick subtrees of [Bou09].

In the model based on winning conditions, the usual bar recursion operator does not realize the axiom of choice, but using ideas of the symmetric recursor of Berardi, Bezem and Coquand [BBC98], we define a strategy which updates the choice sequence on-the-fly and could hopefully give a new realizer of the axiom of choice, introducing at the same time imperative features in the computational interpretation of proofs. In this direction it would be interesting to compare our winning conditions with the ones of [Coq95], in which the author uses infinitary sequent calculus for first order quantifications.

Another question is the relationship with the clock operator of Krivine [Kri03], which is used to realize the second-order axiom of choice. In Krivine's setting, this operator needs to live in a countable model of realizers, while the continuity argument of bar recursion relies on the existence of all (even non-computable) sequences in the model, which must therefore be uncountable. For these reasons, the clock operator and the bar recursion operator seem incompatible. However, the games model has also the interesting property that while the set

of all strategies is uncountable, the set of all interactions is countable. Therefore, since the realizability relation in the model with winning conditions is defined by looking at interactions, an enumeration of these may suffice to mimick the behavior of Krivine's clock, leading to a single model in which the two realizations of choice co-exist.

Bibliography

[AHM98] Samson Abramsky, Kohei Honda, and Guy McCusker. A Fully Abstract Game Semantics for General References. In *13th Annual IEEE Symposium on Logic in Computer Science*, pages 334–344. IEEE Computer Society, 1998.

[AHS07] Zena Ariola, Hugo Herbelin, and Amr Sabry. A proof-theoretic foundation of abortive continuations. *Higher-Order and Symbolic Computation*, 20(4):403–429, 2007.

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

[AM96] Samson Abramsky and Guy McCusker. Linearity, Sharing and State: a fully abstract game semantics for Idealized Algol with active expressions. *Electronic Notes in Theoretical Computer Science*, 3:2–14, 1996.

[AM97] Samson Abramsky and Guy McCusker. Call-by-Value Games. In *6th EACSL Annual Conference on Computer Science Logic*, Lecture Notes in Computer Science, pages 1–17. Springer, 1997.

[BBC98] Stefano Berardi, Marc Bezem, and Thierry Coquand. On the Computational Content of the Axiom of Choice. *Journal of Symbolic Logic*, 63(2):600–622, 1998.

[BC82] Gérard Berry and Pierre-Louis Curien. Sequential Algorithms on Concrete Data Structures. *Theoretical Computer Science*, 20(3):265–321, 1982.

[Blo13] Valentin Blot. Realizability for Peano Arithmetic with Winning Conditions in HON Games. In *11th International Conference on Typed Lambda Calculi and Applications*, volume 7941 of *Lecture Notes in Computer Science*, pages 77–92. Springer, 2013.

[BO05] Ulrich Berger and Paulo Oliva. Modified bar recursion and classical dependent choice. In *Logic Colloquium '01, Proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic*, volume 20 of *Lecture Notes in Logic*, pages 89–107. A K Peters, Ltd., 2005.

[BO06] Ulrich Berger and Paulo Oliva. Modified bar recursion. *Mathematical Structures in Computer Science*, 16(2):163–183, 2006.

[Bou09] Pierre Boudes. Thick Subtrees, Games and Experiments. In *9th International Conference on Typed Lambda Calculi and Applications*, volume 5608 of *Lecture Notes in Computer Science*, pages 65–79. Springer, 2009.

[BR13] Valentin Blot and Colin Riba. On Bar Recursion and Choice in a Classical Setting. In *11th Asian Symposium on Programming Languages and Systems*, volume 8301 of *Lecture Notes in Computer Science*, pages 349–364. Springer, 2013.

[Can74] Georg Cantor. Ueber eine Eigenschaft des Inbegriffes aller reellen algebraischen Zahlen. *Journal für die reine und angewandte Mathematik*, 77:258–262, 1874.

[Can92] Georg Cantor. Ueber eine elementare Frage der Mannigfaltigketislehre. *Jahresbericht der Deutschen Mathematiker-Vereinigung*, 1:75–78, 1892.

[CF58] Haskell Curry and Robert Feys. *Combinatory Logic*, volume 22 of *Studies in Logic and The Foundations of Mathematics*. Elsevier, 1958.

[CH00] Pierre-Louis Curien and Hugo Herbelin. The duality of computation. In *5th International Conference on Functional Programming*, pages 233–243. ACM Press, 2000.

[CH09] Pierre-Louis Curien and Hugo Herbelin. Abstract Machines for Dialogue Games. *Panoramas et synthèses*, 27:231–275, 2009.

[Chu36] Alonzo Church. An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, 58(2):345–363, 1936.

[Cla09] Pierre Clairambault. Least and Greatest Fixpoints in Game Semantics. In *12th International Conference on Foundations Of Software Science And Computational Structures*, Lecture Notes in Computer Science, pages 16–31. Springer, 2009.

[Cla11] Pierre Clairambault. Estimation of the Length of Interactions in Arena Game Semantics. In *14th International Conference on Foundations of Software Science and Computational Structures*, pages 335–349. Springer, 2011.

[Cla12] Pierre Clairambault. Isomorphisms of types in the presence of higher-order references (extended version). *Logical Methods in Computer Science*, 8(3), 2012.

[Coq95] Thierry Coquand. A Semantics of Evidence for Classical Arithmetic. *Journal of Symbolic Logic*, 60(1):325–337, 1995.

[Cur07] Pierre-Louis Curien. Definability and Full Abstraction. *Electronic Notes in Theoretical Computer Science*, 172:301–310, 2007.

[dG94] Philippe de Groote. On the Relation between the Lambda-Mu-Calculus and the Syntactic Theory of Sequential Control. In *5th International Conference on Logic Programming and Automated Reasoning*, volume 822 of *Lecture Notes in Computer Science*, pages 31–43. Springer, 1994.

[DHR96] Vincent Danos, Hugo Herbelin, and Laurent Regnier. Game Semantics & Abstract Machines. In *11th IEEE Symposium on Logic in Computer Science*, pages 394–405. IEEE Computer Society, 1996.

[DP01] René David and Walter Py. $\lambda\mu$-calculus and Böhm's theorem. *Journal of Symbolic Logic*, 66(1):407–413, 2001.

[Fre79] Gottlob Frege. *Begriffsschrift, eine der mathematischen nachgebildete Formelsprache des reinen Denkens*. Nebert, Louis, 1879.

[Fri78] Harvey Friedman. Classically and intuitionistically provably recursive functions. In Gert Müller and Dana Scott, editors, *Higher Set Theory*, volume 669 of *Lecture Notes in Mathematics*, pages 21–27. Springer, 1978.

[Gen35] Gerhard Gentzen. Untersuchungen über das logische Schließen. I, II. *Mathematische zeitschrift*, 39(1):176–210, 405–431, 1935.

[GM03] Dan Ghica and Guy McCusker. The regular-language semantics of second-order idealized $\mathrm{A}_{\mathrm{LGOL}^{\cdot}}$. *Theoretical Computer Science*, 309(1-3):469–502, 2003.

[Göd29] Kurt Gödel. *Über die Vollständigkeit des Logikkalküls.* PhD thesis, University of Vienna, 1929.

[Göd31] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für mathematik und physik*, 38(1):173–198, 1931.

[Göd40] Kurt Gödel. *The consistency of the continuum hypothesis.* Annals of Mathematics Studies. Princeton University Press, 1940.

[Göd58] Kurt Gödel. Über eine bisher noch nicht benützte erweiterung des finiten standpunktes. *Dialectica*, 12(3-4):280–287, 1958.

[Gri90] Timothy Griffin. A Formulae-as-Types Notion of Control. In *17th Symposium on Principles of Programming Languages*, pages 47–58. ACM Press, 1990.

[Her95] Hugo Herbelin. *Séquents qu'on calcule: de l'interprétation du calcul des séquents comme calcul de λ-termes et comme calcul de stratégies gagnantes.* PhD thesis, Université Paris 7, 1995.

[HO00] Martin Hyland and Luke Ong. On Full Abstraction for PCF: I, II, and III. *Information and Computation*, 163(2):285–408, 2000.

[How80] William Alvin Howard. The formulae-as-types notion of construction. *To HB Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, pages 479–490, 1980.

[HS02] Martin Hofmann and Thomas Streicher. Completeness of Continuation Models for $\lambda\mu$-Calculus. *Information and Computation*, 179(2):332–355, 2002.

[HS09] Hugo Herbelin and Alexis Saurin. $\lambda\mu$-calculus and $\Lambda\mu$-calculus: a Capital Difference. http://hal.inria.fr/inria-00524942, 2009.

[Hyl97] Martin Hyland. Game semantics. In Andrew Pitts and Peter Dybjer, editors, *Semantics and logics of computation*, Publications of the Newton Institute, chapter 4, pages 131–184. Cambridge University Press, 1997.

[Kle45] Stephen Cole Kleene. On the Interpretation of Intuitionistic Number Theory. *Journal of Symbolic Logic*, 10(4):109–124, 1945.

[Koh90] Ulrich Kohlenbach. *Theory of majorizable and continuous functionals and their use for the extraction of bounds from non-constructive proofs: effective moduli of uniqueness for best approximations from ineffective proofs of uniqueness.* PhD thesis, Goethe Universität Frankfurt, 1990.

[Koh08] Ulrich Kohlenbach. *Applied Proof Theory: Proof Interpretations and their Use in Mathematics.* Springer Monographs in Mathematics. Springer, 2008.

[Kre59] Georg Kreisel. Interpretation of analysis by means of constructive functionals of finite types. In *Constructivity in mathematics: Proceedings of the colloquium held at Amsterdam, 1957*, Studies in Logic and the Foundations of Mathematics, pages 101–128. North-Holland Publishing Company, 1959.

[Kri94] Jean-Louis Krivine. A General Storage Theorem for Integers in Call-by-Name lambda-Calculus. *Theoretical Computer Science*, 129(1):79–94, 1994.

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

[Kri03] Jean-Louis Krivine. Dependent choice, 'quote' and the clock. *Theoretical Computer Science*, 308(1–3):259–276, 2003.

[Kri09] Jean-Louis Krivine. Realizability in classical logic. *Panoramas et synthèses*, 27:197–229, 2009.

[Lai97] James Laird. Full Abstraction for Functional Languages with Control. In *12th Annual IEEE Symposium on Logic in Computer Science*, pages 58–67. IEEE Computer Society, 1997.

[Lai99] James Laird. *A semantic analysis of control*. PhD thesis, University of Edinburgh, 1999.

[Lau05] Olivier Laurent. Classical isomorphisms of types. *Mathematical Structures in Computer Science*, 15(5):969–1004, 2005.

[Loa01] Ralph Loader. Finitary PCF is not decidable. *Theoretical Computer Science*, 266(1-2):341–364, 2001.

[LRS93] Yves Lafont, Bernhard Reus, and Thomas Streicher. Continuations Semantics or Expressing Implication by Negation. Technical Report 93-21, Ludwig-Maximilians-Universität, München, 1993.

[Mel05a] Paul-André Melliès. Asynchronous Games 3 An Innocent Model of Linear Logic. *Electronic Notes in Theoretical Computer Science*, 122:171–192, 2005.

[Mel05b] Paul-André Melliès. Sequential algorithms and strongly stable functions. *Theoretical Computer Science*, 343(1–2):237–281, 2005.

[Mel06] Paul-André Melliès. Asynchronous games 2: The true concurrency of innocence. *Theoretical Computer Science*, 358(2-3):200–228, 2006.

[Mil77] Robin Milner. Fully abstract models of typed λ-calculi. *Theoretical Computer Science*, 4(1):1–22, 1977.

[Miq11] Alexandre Miquel. Existential witness extraction in classical realizability and via a negative translation. *Logical Methods in Computer Science*, 7(2), 2011.

[Nic94] Hanno Nickau. Hereditarily Sequential Functionals. In *Third International Symposium on Logical Foundations of Computer Science*, Lecture Notes in Computer Science, pages 253–264. Springer, 1994.

[Oli08] Paulo Oliva. An analysis of Gödel's Dialectica interpretation via linear logic. *Dialectica*, 62(2):269–290, 2008.

[Ong96] Luke Ong. A Semantic View of Classical Proofs: Type-Theoretic, Categorical, and Denotational Characterizations (Preliminary Extended Abstract). In *11th Annual IEEE Symposium on Logic in Computer Science*, pages 230–241. IEEE Computer Society, 1996.

[OS97] Luke Ong and Charles Stewart. A Curry-Howard Foundation for Functional Computation with Control. In *24th Symposium on Principles of Programming Languages*, pages 215–227. ACM Press, 1997.

[Par92] Michel Parigot. $\lambda\mu$-Calculus: An Algorithmic Interpretation of Classical Natural Deduction. In *3rd International Conference on Logic Programming and Automated Reasoning*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[Plo77] Gordon Plotkin. LCF Considered as a Programming Language. *Theoretical Computer Science*, 5(3):223–255, 1977.

[Pow13] Thomas Powell. *On bar recursive interpretations of analysis.* PhD thesis, Queen Mary University of London, 2013.

[Pow14] Thomas Powell. The equivalence of bar recursion and open recursion. *Annals of Pure and Applied Logic*, 165(11):1727–1754, 2014.

[RT99] Jon Riecke and Hayo Thielecke. Typed Exeptions and Continuations Cannot Macro-Express Each Other. In *26th International Colloquium on Automata, Languages and Programming*, pages 635–644. Springer, 1999.

[RW13] Bertrand Russell and Alfred North Whitehead. *Principia Mathematica*, volume 1, 2, 3. Cambridge University Press, 1910, 1912, 1913.

[Sau05] Alexis Saurin. Separation with Streams in the $\Lambda\mu$-calculus. In *20th IEEE Symposium on Logic in Computer Science*, pages 356–365. IEEE Computer Society, 2005.

[Sco93] Dana Scott. A Type-Theoretical Alternative to ISWIM, CUCH, OWHY. *Theoretical Computer Science*, 121(1-2):411–440, 1993.

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

[Spe62] Clifford Spector. Provably recursive functionals of analysis: a consistency proof of analysis by an extension of principles in current intuitionistic mathematics. In *Recursive Function Theory: Proceedings of Symposia in Pure Mathematics*, volume 5, pages 1–27. American Mathematical Society, 1962.

[Str06] Thomas Streicher. *Domain-theoretic foundations of functional programming.* World Scientific, 2006.

[Tro98] Anne Sjerp Troelstra. Realizability. In Samuel Buss, editor, *Handbook of Proof Theory*, volume 137 of *Studies in Logic and the Foundations of Mathematics*, chapter 6, pages 407–473. Elsevier, 1998.

[Tur37] Alan Turing. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265, 1937.

[TVD88] Anne Sjerp Troelstra and Dirk Van Dalen. *Constructivism in mathematics: an introduction*, volume 121, 123 of *Studies in logic and the foundations of mathematics.* Elsevier, 1988.

[Zer08] Ernst Zermelo. Untersuchungen über die Grundlagen der Mengenlehre. I. *Mathematische Annalen*, 65(2):261–281, 1908.