The Partial Lambda-Calculus

Eugenio Moggi

Ph. D.
University of Edinburgh
1988

Abstract

This thesis investigates various formal systems for reasoning about partial functions or partial elements, with particular emphasis on lambda calculi for partial functions. Beeson's (intuitionistic) logic of partial terms (LPT) is taken as the basic formal system and some of its metamathematical properties are established (for later application). Three different *flavours* of Scott's logic of partial elements (LPE) are considered and it is shown that they are conservative extensions of LPT. This result, we argue, corroborates the choice of LPT as the basic formal system.

Variants of LPT are introduced for reasoning about partial terms with a restriction operator (LPT + $\upharpoonright$), monotonic partial functions (monLPT), λ-terms (λ_p-calculus) and λY-terms ($\lambda_\mathrm{p}\mu$Y-calculus). The expressive powers of some (in)equational fragments are compared in LPT and its variants. Two equational formal systems are related to some of the logics above: Obtulowicz's p-equational logic is related to LPT + $\upharpoonright$ and Plotkin's λ_v-calculus is related to one flavour of LPE.

The deductive powers of LPT and its variants are compared, using various techniques (among them logical relations). The main conclusion drawn from this comparison is that there are four different lambda calculi for partial functions: intuitionistic or classical, partial or monotonic partial functions.

An (in)equational presentation of the intuitionistic lambda calculus for (monotonic) partial functions is given as an extension of p-equational logic. We conjecture that there is no equational presentation of the classical λ_p-calculus. Via a special kind of diamond property, the (in)equational formal system is characterized in terms of β-reduction for partial functions and some decidability problems are solved.

Acknowledgements

The help of my supervisor, Gordon Plotkin, has been essential for providing the early motivation and discussion with him has always been extremely useful. Correspondence with Pino Rosolini, especially in the early stages of my research, has greatly influenced the work in this thesis.

I would like to thanks AT&T Bell Labs, in particular Dave MacQueen and John Mitchell, for making it possible to visit the USA. When I was there I had very fruitful discussions with Val Breazu-Tannen, who convinced me of the importance of conservative extension results, and Rick Statman, who emphasized the importance of Jacopini's lemma. Conversations with Andrea Asperti, Pierre-Louis Curien, Carl Gunter, Martin Hyland, Giuseppe Longo, John Mitchell, Axel Poigné, Luke Ong, Edmund Robinson, Pino Rosolini, Andrzej Tarlecki, were also helpful. A special thanks to Furio Honsell for critically reading a draft of this thesis.

Duncan Baillie enlarged LaTeX when this thesis was growing, Paul Taylor sent me his macros for diagrams (which I would never been able to define by myself) and Julian Bradfield helped me for the english.

During these last four years the Department and the LFCS (Laboratory for the Foundations of Computer Science) have provided a very stimulating environment and wonderful computer facilities.

I was financially supported by a University of Edinburgh Studentship and a Research Assistantship in the LFCS.

I am especially grateful to Giuseppe Longo, my former supervisor in Italy, who advised me to go to Edinburgh in the first place, and to my parents Carlo and Anna, my relatives Marisa and Fosco and my sister Paola for supporting and encouraging me all the way.

Declaration

This report contains the revised version of my Ph.D. thesis, according to the recommandations of the examiners: J.M.E. Hyland, R. Milner and G. Plotkin. Comments of any nature are greatly appreciated and can be sent to:

- Dr. E. Moggi
  LFCS
  University of Edinburgh
  the King's Buildings
  EH9 3JZ Edinburgh
  Scotland

- or to em@ed.lfcs via e-mail

Contents

Chapter 0

Introduction

Partial functions arise naturally in various mathematical theories relevant to computer science, such as algebra and domain theory. In **algebra** partial functions occur in relation to partial algebras, and (in **algebraic specifications**) are used to model operations that may cause an *abortive exception*, or in connection with *behavioural equivalence* (see [Bur82, BBTW81, Rei87, Sch87]). In **domain theory** partial functions are *disguised* as total continuous functions between *spaces of partial elements* (see [Sco70, SS71, Sco76, Plo81]), but a recent approach, based on continuous partial functions, makes their role explicit (see [Plo85]).

This thesis studies formal systems for reasoning about partial functions in the setting of first order logic and the lambda calculus. Models play a background, but nevertheless important, role either by justifying the study of a formal system, when it is sound and complete w.r.t. an *interesting* class of models, or as a technical tool for studying the relations among formal systems. There is a (confusing) wealth of formal systems, and a major achievement of this thesis is to classify them in a common framework according to a few *parameters* and to investigate (exhaustively) how they relate to each other.

There is no *best* formal system: instead each of them is better than the others for certain applications. For instance, only the formal systems based on intuitionistic free logic are sound w.r.t. interpretation in categories of partial maps or topoi, the formal systems based on λ-terms have a natural interpretation in *partial cartesian closed categories*, the formal systems based on posets and with (least) fixed-point operators are more appropriate for domain theory.

Our main objective is to provide a basis for proving equivalence of programs, therefore particular attention is given to the formal systems based on

λ-terms (like the polymorphic predicate λ-calculus $PP\lambda$ of [GMW79]), since the λ-calculus is a *paradigmatic* programming language, and to *equational* formal systems, since they provide the minimum required to express equivalence. Another major achievement is an (in)equational presentation for an intuitionistic λ-calculus of (monotonic) partial functions and the study of β-reduction.

The following Section reviews the work that has directly influenced this thesis and mentions some (more recent) related work. Section 0.2 explains the main achievements of this thesis and gives a summary of each chapter.

0.1 Review of related work

The works related to this thesis can be roughly divided among four subject areas:

- first order logic and partial functions,
- categories of partial maps,
- λ-calculus and programming languages,
- logics for computable functions and domain theory.

0.1.1 First order logic and partial functions

There are two ways of representing partial functions in first order logic: as *strict* total functions on *partial elements* or as *single-valued* relations. The first approach corresponds to making a theory of partial functions using total functions, while the second approach is more direct and avoids the introduction of a *new concept* (i.e. partial element).

The logic of partial elements

The logic of partial elements in [Sco79, Fou77] is inspired by early work on *descriptions* in formal logic (see [Sco67]). The *description* $\mathbf{I}x.A(x)$ is a term that may fail to denote, since it denotes an element a iff a is the unique x s.t. $A(x)$.

From our point of view, the main achievement of the logic is the *intuitionistically correct* treatment of partial elements, in fact ([Sco79]): "in classical logic ... it is always possible to split the definition (or theorem) into cases according as the partial element in question does or does not exists. In intuitionistic logic this way is not open to us". The idea is to permit a wider interpretation of *free* variables, by introducing for each set X a *superset* $\tilde{X}$ of *partial elements*, and to interpret a term of sort X with a partial element, by turning partial functions on X into *strict* total functions on $\tilde{X}$. In intuitionistic set theory, the partial elements of sort X can be defined to be the subsets of X with **at most** one element[1] and X can be identified with the extension of the *existence predicate* $\mathrm{E}(Y) \overset{\Delta}{\Longleftrightarrow} \exists x \in X.x \in Y$ (over partial elements). Consequently, the partial function $f: X \rightharpoonup X$ is *turned* into the *strict* total function $\tilde{f}: \tilde{X} \to \tilde{X}$ s.t. $\tilde{f}: Y \mapsto \{x \in X | \exists y \in Y.x = f(y)\}$.

In [Fou77] the higher order logic of partial elements was proposed as a logic of topoi, which avoids the restrictions on the rule of modus ponens and substitution due to *empty* sorts; in fact in a topos there are *objects* without *global sections*. This approach, however, seems to *confuse* the two unrelated problems of empty types and nondenoting terms. Moreover, in (intuitionistic) higher order logic the notion of partial element is actually redundant, since the superset of partial elements can be *defined* at higher types according to the set-theoretic intuition.

From a computational perspective a logic with descriptions (and the set-theoretic notion of partial element itself) is not particularly interesting, as it is *inconsistent* with the view of *partial elements as (denotations of) programs*. This *computational view* is investigated by Rosolini in the setting of higher order intuitionistic logic and topoi (see [Ros86]), where he proposes the notion of **dominance**, as classifier for the *computable partial elements*.

In [Sco79] Dana Scott considers, among others, a first order logic of partial elements without descriptions (LPE), which is quite *liberal* and it is consistent with the view of partial elements as (denotation of) programs and even with *all partial elements exists*. We consider an even more liberal logic of partial elements (iLPE) and relate it to the λ_v-calculus of [Plo75].

From a *model-theoretic* perspective LPE is not *appropriate* for reasoning about partial algebras. In fact, given a partial algebra $\mathcal{A}$ (with carrier X) there is in general more than one choice for the set of partial elements, so that $\mathcal{A}$ is turned into a model of LPE. Therefore, the validity of an open formula in $\mathcal{A}$ is ambiguously defined, as the definition involves a quantification over the set of partial elements. For instance, if $\mathcal{A}$ is a total algebra, then the set of partial

[1] to simply add a new element $\perp$ for *undefined* would not be intuitionistically correct

elements can be *anything* between X and $\tilde{X}$ (intuitionistically there is a whole range of possibilities).

Since our general notion of model is based on partial algebras, we do not take LPE as framework for studying other formal systems; nevertheless we will investigate LPE and its relations with other formal systems. This investigation shows that LPE has other (more technical) inconveniences, owing to free variables ranging over partial elements.

A logic for partial algebras

Standard first order logic is for reasoning about mathematical structures where (all sorts are inhabited and) functions symbols are interpreted by **total** functions, therefore it is not sound for partial algebras. Partial algebras seem a very simple generalization of total algebras, but people working in this area have come up with all sorts of (bizarre) ideas about the meaning of an equation to the point of suggesting three valued logic.

Burmeister's survey paper (see [Bur82]) presents a nice and simple model theory of partial algebras, based on a unified view of partial and total functions as single-valued relations, and develops a sound and complete (two-valued) first order logic of partial algebras. In Set Theory functions are **single-valued** binary relations ($\forall x, y, z.R(x, y) \land R(x, z) \rightarrow y = z$), in other words a function f is identified with its graph R. According to this view, a partial algebra (A, f^A), where f^A is a unary partial function, is actually a relational structure (A, R^A), where R^A is the graph of f^A.

The interpretation of (terms and) atomic formulae differs from the standard one, because terms may not denote (e.g. $f(x) = f(y)$ is true iff "$f(x)$ and $f(y)$ are both defined and equal"), but from the relational point of view it is the natural extension of the interpretation in total algebras (see Remark 2.18 of [Bur82]). In fact, to define when (A, R^A) satisfies the formula F (involving f) under the evaluation ρ, one has to get rid of f (which is just a notational convention) by replacing F with a formula F' involving R only, so that F is satisfied $\stackrel{\Delta}{\iff}$ F' is satisfied. F' is defined by induction on F, but the only interesting case is when F is atomic. For instance, the equation $f(x) = f(y)$ is replaced by a clumsy formula $\exists u, v.R(x, u) \land R(y, v) \land u = v$, which is equivalent to the familiar meaning of $f(x) = f(y)$ when (A, f^A) is a total algebra.

Burmeister reviews three (main) concepts of *varieties*, corresponding to three kind of *quasi-equations*: we consider two of them (equivalent to equations in the case of total algebras) when comparing the *expressive power* of equational fragments.

The logic of partial terms

In [Bee85] the logic of partial terms (LPT) is introduced as a preliminary to the *theories of rules*, e.g. the theory of partial combinatory algebras, where it is handy to allow the formation of terms that may fail to denote, like $\{e\}(x)$. LPT is a *notational variant* modification of the logic presented in [Bur82]. More precisely, LPT has an existence predicate $t\downarrow$, while Burmeister has to write $t = t$, and it distinguishes between constants and nullary partial functions, namely the former *exist* and the latter may not exist. The existence predicate is very convenient for formulating the inference rules of substitution, strictness and quantification so that they no longer depend on the equality predicate. **We take a sequent calculus variant of** LPT **as our framework for studying other formal systems.**

0.1.2 Categories of partial maps

There have been various attempts to answer the question of "what extra structure do we need on a category for it to be a *well-behaved* category of partial maps". In [RR86] (many of) these attempts are reviewed and compared. From our perspective, the most relevant approach is the one in [Ros86], which is the first to make explicit connections with intuitionistic logic, the theory of computation and the lambda calculus. For completeness we recall how this approach originated and developed.

The original idea of *dominical category* was presented by Heller at a conference in Siena. His objective was to introduce a basic categorical setting for abstract recursion theory, by describing axiomatically some properties of the monoid of partial recursive functions (see [dPH86]).

In [LM84] Heller's ideas are taken up and developed into *concrete* notions of *partial morphism* and *partial cartesian closed category* (with enough points) adequate for recursion theory at higher types.

Dominical categories are intended to capture axiomatically the idea of a *category of partial maps* $P(C, \mathcal{M})$, i.e. the category of *partial maps in* C *with domain in* $\mathcal{M}$, where C is a category and $\mathcal{M}$ is a collection of monos in C (called *dominion* in [Ros86] and *domain structure* in [Mog86]). This consideration and a topos-theoretic approach (whose usefulness is demonstrated in [Hyl82]) led Rosolini to the more general notion of *p-category* and to make the connection with topoi (and higher order intuitionistic logic) by proving various embedding theorems of an arbitrary p-category into a category of partial maps in a topos defined by means of a *dominance* (see [Ros86]).

The categories $P(C, \mathcal{M})$ are already present in [Obt86] (they are called *categories of* $\mathcal{M}$*-functional relations in* C) together with *categories with ordered strict pre-cartesian structure* that are essentially equivalent to p-categories, but the notions of partial cartesian closed category and dominance are missing.

There is an important background difference between Rosolini's and Obtulowicz's approach, which is made explicit in the introduction of [Obt86]: "In the branch of category theory applied to logic there are two concepts of a logic of a category. The first concept, due to Lawvere, is meant in such a way that if the logical connectives and quantifiers have an interpretation in a category C, then this interpretation constitutes a logic of a category C .... The second concept is due to Lambek .... A logic in Lambek's sense of a category C is an additional equational structure on C, i.e. a system of partial operations defined on arrows of C together with a set of equations describing properties of the operations ..." [Ros86] uses Lawvere's concept of logic (of a topos), while the aim of [Obt86] (and [CO87]) is to show that certain categories of partial maps have a logic in Lambek's sense and to develop *equational presentations*.

Obtulowicz also considers some applications to universal algebra and establishes an equivalence between p-categories and theories in a new logic, called **p-equational logic** (see Chapter 7 of [Obt86]). **Our (in)equational presentation of the intuitionistic lambda calculus for (monotonic) partial functions** *extends* **p-equational logic**, and it differs from the *equational presentations* of partial cartesian closed categories developed in [CO87] in the same way as Lambek's logic for cartesian closed categories differs from the typed lambda calculus (see [Lam80]).

0.1.3 λ-calculus and programming languages

The *direct connections* between the pure λ-calculus (i.e. the subject of [Bar81]) and functional programming languages sometimes do not go beyond the binding mechanism of λ-abstraction (see [McC62, Lan64]). This is not surprising if we look at the history of the type free lambda calculus (Preface of [Bar81]): "Around 1930 the type free lambda calculus was introduced as a foundation for logic and mathematics. Due to the appearance of paradoxes, this aim was not fulfilled, however. Nevertheless a consistent part of the theory turned out to be quite successful as a theory of computations. It gave an important momentum to early recursion theory and more recently to computer science. ...As a result of these developments. the lambda calculus has grown into **a theory worth studying for its own sake**. People interested in applications may also find the pure λ-calculus useful, since these **applications are usually heuristic rather than direct**. ..."

Types arise in various applications, for instance they are a common *feature* of many programming languages, in logic they can be *identified* with formulae, in (cartesian closed) categories they *correspond* to objects. As shown in [Sco80], "nothing is lost in considering type free theories just as *special parts* of typed theories". For this reason, we give more emphasis to the typed λ-calculus (and its extensions), as it provides a more flexible and structured basis for applications than the pure λ-calculus.

Call-by-value, call-by-name and the λ-calculus

In [Plo75] call-by-value and call-by-name are studied in the setting of the lambda calculus: this study exemplifies very clearly the *mismatch* between the pure λ-calculus and (some) programming languages.

From an operational point of view, a programming language is completely specified by giving the set Prog of programs and the **evaluation mechanism** Eval: Prog $\rightharpoonup$ Prog, i.e. a partial function mapping every program to its resulting **value** (if any).

> In the setting of the lambda calculus, programs are identified with the (closed) λ-terms (possibly with extra constants, corresponding to some *features* of the programming language). The evaluation mecha-

nism *induces* a congruence relation $\approx$ on λ-terms, called **operational equivalence**, and a *calculus* is said to be **correct** w.r.t. $\approx$ when $M = N$ (derivable in the calculus) implies $M \approx N$.

Plotkin's intention is to study programming languages, therefore he accepts the evaluation mechanism and looks for *the corresponding calculus*.

The λ-calculus is not correct w.r.t. call-by-value operational equivalence, therefore it cannot be used to prove equivalence of programs. Starting from this observation, Plotkin introduces the λ_{v}-calculus and shows that it is correct w.r.t. call-by-value operational equivalence. However, to prove the Church-Rosser and Standardization theorems Plotkin proceeds by *analogy* and reuses some techniques already applied to the λ-calculus. In the case of call-by-name the situation is much simpler, since only the η axiom is not *correct* w.r.t. call-by-name operational equivalence.

Plotkin *discovers* the λ_{v}-calculus by operational considerations, and his only criterion for *judging* a calculus is its correctness w.r.t. the operational semantics, but in general there are plenty of correct calculi. On the other hand, we *discover* the λ_{p}-calculus by model-theoretic considerations, namely as the **unique** calculus which is sound and complete w.r.t. a certain class of models.

The λ_{p}-calculus

The λ_{p}-calculus is a formal system presented in [Mog86] and (with minor differences) in Chapter 4 of [Ros86], which is **sound and complete** w.r.t. interpretation in *partial cartesian closed categories*. The terms of the λ_{p}-calculus are typed λ-terms, and the formal system is for deriving one-sided sequents $\eta.A_1, \ldots, A_n \Longrightarrow A_0$, where η is a *type environment* and the A_i are either existence statements ($\mathrm{E}(t)$) or *equivalences* ($t_1 \equiv t_2$). The (type free) λ_{p}-calculus proves more equivalences than the λ_{v}-calculus (e.g. $(\lambda x.x)(yz) \equiv yz$), nevertheless it is still correct w.r.t. call-by-value operational equivalence. We will *explain* the difference between the two calculus after presenting them in a common framework.

In [Mog86] it is pointed out that the λ_{p}-calculus is not complete w.r.t. classical type structures of partial functionals. We consider modifications of the λ_{p}-calculus that are sound and complete w.r.t. other classes of models, e.g. classical type structures of (monotonic) partial functionals, and develop an equational presentation of the λ_p-calculus (rather than one based on one-sided sequents), which allows an analysis of β-reduction for partial functions.

In Chapter 4 of [Ros86] the λ_p-calculus is extended with fixed-point operators. We consider a different λ_p-calculus with fixed-points, which is more *appropriate* for the category of cpos and continuous partial functions (see [Plo85]).

Lazy λ-calculus

In [Ong88] a theory of *lazy* (equivalently call-by-name) functional programming languages is developed in the framework of the pure λ-calculus. Sensible theories (in [Bar81]) are incorrect w.r.t. call-by-name operational equivalence, therefore they are replaced with *lazy λ-theories*.

Luke Ong introduces also the λ_L-calculus, a formal system sound (but not complete) w.r.t. partial cartesian closed categories, and compares it to the λ_p-calculus (of [Mog86]). A *natural* extension of the λ_L-calculus, the untyped $\lambda_{L,C}$-calculus, obtained by adding a constant C for *convergence testing* (first introduced in [Abr87]), is also considered and proved sound and complete w.r.t. partial cartesian closed categories with a *reflexive object* $(D_\perp \rightharpoonup D) \lhd D$. He describes also a (conservative) interpretation of the untyped $\lambda_{L,C}$-calculus in the typed λ_p-calculus, which is the *straightforward* way of translating call-by-name into call-by-value (using closures).

0.1.4 Logics for computable functions and domain theory

The use of logics for proving correctness of programs was pioneered in [Flo67, Hoa69]. The programming languages considered in these early approaches are very simple (while programs) and the formal languages of expressing properties of programs are a mixture of programming language and predicate calculus (Hoare's triples). The main feature of these logics, e.g. **Hoare's logic**, is that they are **programming language dependent** and support a **structured methodology** for proving correctness, i.e. properties are proved by induction on the structure of programs.

We are interested in a different kind of approach, based on logics for reasoning about mathematical structures that provide models for (functional) programming languages. Such a logic is **programming language independent**, but potentially it can be used to prove properties of programs (like equivalence or correctness). To exploit this potentiality the logic has to be **flexible**, so that one can describe how a particular programming language is interpreted in a certain mathematical structure. More precisely, its language should be extendable (to include the features of a programming language), it should be possible to add (non-logical) axioms (to axiomatize the properties of these features) and the logic itself should be uncommitted (so that any programming language can be *axiomatized*).

The need for a mathematical semantics of programming languages led to the Scott-Strachey approach to denotational semantics and the development of **domain theory** (see [Sco70, SS71, Sco76, Plo81]). Denotational semantics takes an abstract view of *computations*, based on **continuous functions** between certain *spaces* of *partial elements* (**cpos**), rather than partial recursive functions (on the natural numbers).

A first attempt at a logic for reasoning about cpos and continuous functions was the **logic for computable functions** (see [Sco69]), which was subsequently extended to the **polymorphic predicate λ-calculus** $PP\lambda$ (see [GMW79, Pau85]). The idea is first to consider an extension of the typed λ-calculus suitable for the (cartesian closed) category of cpos and continuous functions (the *intended model*), by adding constants for *fixed-point operators* and *least elements*, and then to axiomatize *some* properties of the intended model, like fixed point induction, in a (classical) first order logic, whose formulae are built up from inequations between λ-terms ($M \leq N$).

In $PP\lambda$ programs are *identified* by λ-terms and properties (of a programs) are ordinary first order formulae. As pointed out at Page 9 of [Sco69], $PP\lambda$ (as well as the logic of partial elements) is a theory of partial functions based on total functions: "... classical type theory supposes **total** (everywhere defined) functions, while algorithms in general produce **partial** functions. We do not wish to reject a program if the function defined is partial – because as everyone knows it is not possible to predict which programs will *loop* and which will define total functions. The solution to this problem of total vs. partial functions is to make a *mathematical model* for the theory of partial functions using ordinary total functions." More precisely, *divergence* is represented by the least element ($\perp$) of a cpo and a partial (recursive) function $f: \mathbf{N} \rightharpoonup \mathbf{N}$ becomes a *strict* function

$f_\perp \colon \mathbf{N}_\perp \to \mathbf{N}_\perp$.

There are various criticisms that can be made to $PP\lambda$ (and domain theory), more or less related to the way partial functions are represented:

- while programs (and similar programming languages) have a simple set-theoretic semantics as partial functions from stores to stores, but their denotational semantics (based on cpos and continuous functions) is *comparatively clumsy*.

- the way of expressing *termination* (*existence*) of a program (partial element) t is *indirect* "$t \neq \perp$" and intuitionistically incorrect, according to [Sco79].

- the (choice and) interpretation of type constructors is based on the typed λ-calculus (i.e. a theory of total functions), rather than considerations about (real) programming languages. For instance, in a *lazy* programming language the *values* of type $\tau_1 \times \tau_2$ are pairs of *unevaluated expressions* and $[\![\tau_1 \times \tau_2]\!]$ should be the *lifted product* $([\![\tau_1]\!] \times [\![\tau_2]\!])_\perp$, while in a *eager* programming language the *values* of type $\tau_1 \times \tau_2$ are pairs of *values* and $[\![\tau_1 \times \tau_2]\!]$ should be the *smash product* $[\![\tau_1]\!] \otimes [\![\tau_2]\!]$.

These considerations on domain theory and $PP\lambda$ (among others) led Gordon Plotkin to take a different approach to denotational semantics, based on **continuous partial functions**, and to reformulate $PP\lambda$ using intuitionistic logic of partial elements on top of a suitable term language, called *metalanguage* (see [Plo85]).

The metalanguage is a typed functional language with a rich collection of types (products, coproducts, partial function spaces and recursive types), it has an operational semantics, based on eager evaluation, and a denotational semantics, according to which types denote cpos (possibly without a least element) and terms denote continuous partial functions. The two semantics are related by a correspondence theorem between (operational) termination and (denotational) existence, inspired by a similar result due to Martin-Löf (see [ML83] and Chapter 6 of [Abr87]).

The choice of eager evaluation makes it easier to translate programming languages into the metalanguage *correctly*, i.e. so that the operational and denotational semantics obtained via the translation are the *intended* ones. In fact, it is straightforward to translate lazy programming languages into typed eager programming languages, while the reverse translation is in general impossible (or at least very clumsy).

Since many programming languages can be *correctly* translated into the metalanguage, it is not necessary to *extend* the metalanguage for *accommodating* them, and a logic for reasoning about them can be *derived* from that for the metalanguage, while in $PP\lambda$ it has to be *axiomatized*, by creating a new theory. For the same reason, the correspondence theorem between operational and denotational semantics, unlike similar results for the λ-calculus (e.g. [Wad76]), has **direct** application to programming languages.

0.2 Overview of the thesis

We achieve three main objectives: to investigate the relations among various formal systems, based on first order logic, for reasoning about partial (and total) functions, to develop an (in)equational presentation of the intuitionistic lambda calculus for (monotonic) partial functions, and to relate its (in)equality to a suitable β-reduction for partial functions.

Relating formal systems. First, we show that the logic of partial elements LPE is a *conservative extension* of the logic of partial terms LPT via a *natural* translation (see Section 3.3). Then, we consider the relations among various formal systems *axiomatized* in LPT. The kind of relations we establish is that a formal system either is a *conservative extension* of or has *more deductive power* than another formal system *over* a certain *fragment* (see Chapter 6). These formal systems are classified according to their *signature* (language) and for each signature according to three independent choices: intuitionistic logic vs. classical logic, free logic vs. inhabited logic, partial functions vs. total functions. The signatures correspond to the following classes of models: algebras, monotonic algebras, type structures, monotonic type structures, type structures with fixed-point operators and type structures with least fixed-point operators.

The formal systems are (more or less directly) inspired by the literature, for instance: the classical inhabited logic for total algebras is the *ordinary* first order logic, the intuitionistic free logic for partial type structures corresponds to the λ_p-calculus in [Mog86, Ros86], the classical inhabited logic for total type structures with least fixed-point operators corresponds to a substantial subset[2] of $PP\lambda$ (see [Pau85]), the intuitionistic free logic for partial type structures with least fixed-point operators corresponds to a subset of Plotkin's reformulation of $PP\lambda$ (see [Plo85]).

We show that there are four typed lambda calculi for partial functions with different *deductive power* on the equational fragment, and they are classified according to two independent choices: intuitionistic logic vs. classical logic, type structures vs. monotonic type structures.

The monotonic calculi are more useful for applications (in computer science). On the other hand, only the intuitionistic calculi seem to enjoy an *equational presentation*. These two considerations lead us to give more emphasis to the intuitionistic lambda calculus for monotonic partial functions.

Equational and inequational presentation. We develop a formal system, which extends p-equational logic (see [Obt86]), for deriving equations (between untyped *pλ-terms*) from a set of equational axioms. This formal system is sound and complete w.r.t. the models of the intuitionistic untyped lambda calculus for partial functions. Moreover, there is a set of inference rules similar to those for the λ-calculus, which *generates* the equations valid in all these models.

We develop also a similar formal system for deriving inequations from a set of inequational axioms, which is sound and complete w.r.t. the models of the intuitionistic untyped lambda calculus for monotonic partial functions.

Inequality and β-reduction In the (pure) λ-calculus the relation between $\beta\eta$-conversion and β-reduction can be summarized by:

> M and N are $\beta\eta$-convertible iff there exist two η-convertible[3] λ-terms M' and N' s.t. M and N β-reduce to M' and N' respectively.

This result relies on three lemmas: β-reduction commutes with itself (Church-Rosser property), β-reduction commutes with η-reduction and η-reduction can be postponed after β-reduction. In the intuitionistic untyped lambda calculus for monotonic partial functions the situation is very different (see Chapter 8):

[2] e.g. the axioms for fixed-point induction are missing

[3] or equivalently with the same η-normal form

$t_1 \restriction D_1 \lesssim t_2 \restriction D_2$ is derivable iff there exists a pλ-term $t \restriction D$ s.t.
$t_1 \restriction D_1 (\to_1 \lesssim_{d\eta})^* t \restriction D$ and $t_2 \restriction D_2 (\to_1 \gtrsim_{d\eta})^* t \restriction D$

where $\to_1$ is a suitable β-reduction for partial functions and the relation $\lesssim_{d\eta}$, called dη-preorder, combines η-conversion and a preorder relation on pλ-terms due to the *restriction operator*. This result relies on a lemma, namely $\to_1$ commutes with itself up to $\lesssim_{d\eta}$, that involves the *new notion* of "_ **commutes with _ up to** _" (see Definition 8.1.2).

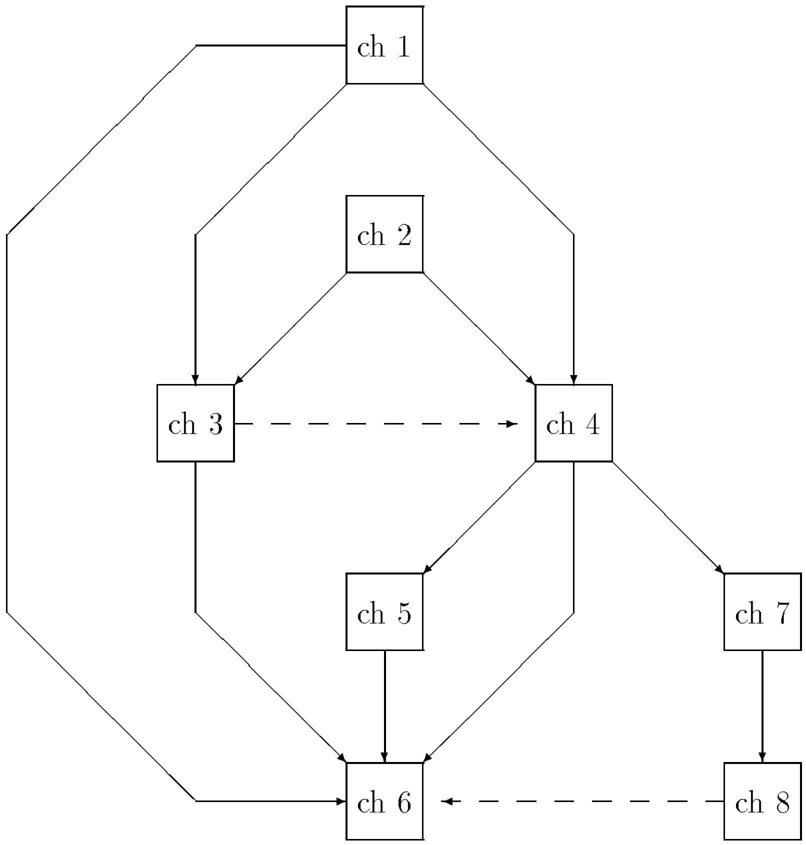
The decomposition of the inequality $\lesssim$ into β-reduction $\to_1$ and dη-preorder $\lesssim_{d\eta}$ is very natural from a operational point of view. In fact, β-reduction encapsulates the evaluation mechanism, while the dη-preorder is what is left of the inequality after removing (β). We expect the notion of "_ commutes with _ up to _" to be useful for studying other calculi, e.g. the non-deterministic lambda calculus (see [Sha84]).

0.2.1 Prerequisites

The reader is supposed to be familiar with elementary set theory (see first chapter of [Mon69]), first order logic (see [Ten78]) and the lambda calculus (see [Bar81, Bar84]). Only few basic notions of category theory (category, functor and natural transformation) are taken for granted (see [Mac71, LS86]).

0.2.2 Summary

The dependencies among chapters are given by the following diagram (where a dashed connection means that only a small part of the *target chapter* depends on the *source chapter*):



For each chapter we summarize its contents, and give some reading suggestions and a list of relevant literature.

Chapter 1. This chapter contains some basic definitions and facts about *formal systems* (considered as consequence relations), *inference rules* and *logics*, which are repeatedly used in the other chapters. Particularly important are the methods for defining formal systems, the criteria for comparing their deductive power and the ways inference rules and models can be applied to establish relations between formal systems. There are also few general remarks about *languages*, *formation rules* and *variables*.

At the beginning a superficial reading is enough.

References:
1.2 of [Tro73] (conservative and definitional extension),
[Acz77] (rules and inductive definitions),
Appendix C of [Bar81] (variables and substitution),
[Bar74] (abstract model theory).

Chapter 2. Two basic formal systems are introduced, the classical and intuitionistic *logic of partial terms*, on which most of the other formal systems depend. The logic of partial terms (LPT) is just many sorted first order logic with some *syntactic sugar* for partial functions, therefore the language, interpretation and formal system of classical and intuitionistic LPT are very similar to those of classical and intuitionistic first order logic (based on the sequent calculus). The proofs of soundness and completeness are *standard*.

It is enough to read the main definitions and theorems (without proofs).

References:
[Bee85] (logic of partial terms),
[Gen69, Kle52, MR77] (sequent calculus),
[Ten78] (interpretation, soundness and completeness),
5.1 of [Tro73] [vD86] (Kripke models, completeness for intuitionistic logic),
[Hen49, MR77] (completeness).

Chapter 3. Classical and intuitionistic LPT are compared, by considering some fragments on which intuitionistic and classical LPT do (or do not) coincide, this explains why the λ-calculus is sound and complete w.r.t. both *intuitionistic models* (say cartesian closed categories) and *classical models* (say type structures).

We prove some cut-elimination results, parametric in a set of axioms, for intuitionistic and classical LPT restricted to the *negative* fragment. A fallout of these results is, for instance, a procedure to get from a set of first order axioms to a set of *equational inference rules* that derive the same equations.

We introduce the logic of total terms and (the three *flavours* of) the logic of partial elements. We extend the metalogical results for LPT (i.e. the coincidence and cut-elimination results) to these other logics. Then, we compare LPT and (three *flavours* of) LPE.

This chapter is particularly interesting for applications, since the conservative extension and cut-elimination results proved can be applied to many formal systems, not just those in the thesis. These results are more or less implicit in the literature. However, motivated by particular applications, we have improved them slightly.

References:
[MR77] (coherent logic),
1.10 of [Tro73] (Harrop and negative formulae),
5.1 and 5.2 of [Tro73] (the operation $(\ ) \to (\ )'$),
[Gen69, Kle52, MR77] (cut elimination),
[Sco79] (logic of partial elements).

Chapter 4. We introduce various formal systems axiomatized in the logic of partial terms, ranging from partial equational logic (with restriction operators) to the λ_{p}-calculus (with least fixed-points), which is *equivalent* to extensional partial combinatory logic. For each formal system we analyse the expressive power of equational (or inequational) fragments.

P-equational logic is also considered, to provide the basis for the equational presentation of the λ_{p}-calculus, and the λ_{v}-calculus is characterized in terms of the *intensional logic of partial elements*.

This chapter has to be read, since most of the formal systems introduced here are considered in subsequent chapters.

References:
[Bur82] (equational fragments),
[Obt86] (p-equational logic),
[Bee85] (weak extensionality),
[Bar81, Mey82] (relation λ-calculus - extensional combinatory logic),
[Plo85] (the λ_{p}-calculus),
[Plo77] (**PCF**, the λY-calculus),
[Plo75] (the λ_{v}-calculus).

Chapter 5. Four model-theoretic techniques, applied in Chapter 6, are considered: (full) type hierarchies, logical relations, closed term models, the relation between termination and denotation. We give only the full type hierarchies used in Chapter 6. Logical relations between (partial algebras and) partial type structures are studied in the framework of first order logic, which allows a unified approach to logical relations and Kripke logical relations. An original closed term model construction for the $\lambda_p\mu$Y-calculus is presented. We recall the relation between termination and denotation, due to G. Plotkin, and apply it to characterize the existence-statements provable in the typed and untyped λ_p-calculus.

References:

[Fri75, Plo80] (full type hierarchy),

[Fri75, Sta85b, Sta85a, Plo80, Plo82, AL85, MM87] (logical relations),

[Sch87] (logical relations for partial algebras),

[Sta85a, BTC87, Mil77, Sto86] (closed term model),

[ML83, Plo85] (the relation between termination and denotation).

Chapter 6. The relations among (some of) the formal systems introduced in Chapter 4 are systematically investigated by applying the techniques developed in previous[4] chapters (see Chapters 3 and 5). The investigation is divided into three parts: *total* formal systems, *partial* formal systems, pure theories. The first part is essentially a review of scattered results from the literature, that have not been presented in a uniform framework before, while the last two parts are original.

This chapter is the main core of the thesis. The following chapters study in greater depth only one formal system, the intuitionistic (monotonic) λ_p-calculus.

Chapter 7. Pλ-terms are introduced in order to develop an (in)equational presentation for the intuitionistic (monotonic) λ_p-calculus along the lines of p-equational logic. The equational formal system is related to the formal system based on first order logic and *pure inference rules* are given, similar to those for the pure λ-calculus, to generate the smallest $\lambda\beta\eta$p-theory. As first application of the equational presentation, we show that the ξ-rule of the λ_p-calculus can be replaced by a set of equational axioms, and relate intuitionistic λ_p-models and λ_p-algebras.

The chapter is quite self-contained. In Chapter 8 only the properties of (pλ-terms and) the smallest $\lambda\beta\eta$monp-theory are used.

References:

[4] in few cases we rely on results about β-reduction, that are established subsequently

[Bar81] (analogies),
[Obt86] (p-equational logic),
[CO87] (alternative approach to the one taken here),
Lecture III of [Sta85a] and [Jac75] (Jacopini's lemma)

Chapter 8. As a second application of the (in)equational presentation, we express the preorder relation $\lesssim$, corresponding to the smallest $\lambda\beta\eta$p-theory, in terms of two other relations: the one-step parallel β-reduction $\rightarrow_1$ and the preorder relation $\lesssim_{d\eta}$ (which is decidable). We also prove that the relation $\lesssim$ between β-normal forms is decidable.

This chapter is the second main core of the thesis, in which new techniques are developed that generalize those applied to study β-reduction for the pure λ-calculus.

References:
[Bar81, Sha84] (analogies).

Chapter 1

Formal systems

In general by *formal system* we mean a pair $(L, \vdash)$, where L is a *language* and $\vdash$ is a *derivability* relation on L (called also an *entailment*). The elements of the language L are called *sentences*. The sets of sentences *closed w.r.t. derivability* are called *theories*, and when ordered by inclusion they form an *algebraic lattice*. Given a formal system, we define also the related notions of: *axiomatization*, *expressive power* and *restriction*.

Definition 1.0.1 Formal System

A **formal system** *is a pair* $(L, \vdash)$, *where* L *is a set and* $\vdash$ *is a subset of* $\mathcal{P}(L) \times L$ *which satisfies:*

reflexivity *if* $A \in T \in \mathcal{P}(L)$, *then* $T \vdash A$

transitivity *if* $T' \vdash A$ *for all* $A \in T$ *and* $T \vdash A'$, *then* $T' \vdash A'$

compactness *if* $T \vdash A$, *then there exists* $T_0 \subseteq_{fin} T$ *s.t.* $T_0 \vdash A$

A $\vdash$**-theory** *is a subset* T *of* L *closed w.r.t.* $\vdash$, *i.e.* $T = \{A \in L | T \vdash A\}$.

$\mathrm{TH}(\vdash)$ *is the poset of* $\vdash$*-theories ordered by inclusion.*

Notation 1.0.2 Let T_1 and T_2 be subsets of L:

- if $T_1 \vdash A$ for all $A \in T_2$, then we say that T_2 is **derivable** from T_2 in $\vdash$ (notation $T_1 \vdash T_2$)
- if $T_1 \vdash T_2$ and $T_2 \vdash T_1$, then we say that T_1 and T_2 are **equivalent** in $\vdash$.

Often sets of sentences on the right or left hand side of $\vdash$ will be treated as sequences. For instance, $T_1, A, T_2 \vdash A_1, A_2$ stands for $T_1 \cup \{A\} \cup T_2 \vdash \{A_1, A_2\}$.

If T is a subset of L, then we write $\mathrm{Th}_{\vdash}(T)$ for $\{A \in L | T \vdash A\}$. Since $\mathrm{Th}_{\vdash}(T)$ is the smallest $\vdash$-theory containing T, we call it the theory **axiomatized** by T in $\vdash$.

If F is a subset of L, then we write $F(\vdash)$ for $\mathrm{Th}_{\vdash}(\emptyset) \cap F$, and call it the **pure theory** of $\vdash$ over F.

A formal system is completely determined by the restriction $\vdash_0$ of the derivability relation $\vdash$ to finite sets, in fact:

- $L = \{A | \{A\} \vdash_0 A\}$
- $T \vdash A \iff$ there exists $T_0 \subseteq_{fin} T$ s.t. $T_0 \vdash_0 A$

The poset $\mathrm{TH}(\vdash)$ is an algebraic lattice, and its finite elements (in the domain-theoretic sense) are the theories axiomatized by finite subsets of L.

Remark 1.0.3 The construction of the algebraic lattice $\mathrm{TH}(\vdash)$ from the formal system $\vdash$, is an instance of a more general construction, that given an *information system* $(L, Con, \vdash_0)$ builds a *Scott's domain* (see [Sco82] and [WL84]). Unlike [Sco82], we do not assume that the set L has a distinguished element Δ, for the sentence *true* (see [WL84]), and we consider any finite subset of L to be *consistent*, so there is no need for Con.

Subsets of L can be viewed either as sets of *axioms* or as *sublanguages*. In the first case one is interested in what is derivable from a set of axioms, and derivability is the natural preorder on sets of axioms (see above). In the second case one is interested in what is *expressible* in a sublanguage, and the natural preorder is different (see below). Correspondingly, there are two ways of building new formal systems from a given one: by *axiomatization* and by *restriction*.

Definition 1.0.4 Axiomatization

If $(L, \vdash)$ is a formal system and T is a subset of L, then

- $(\_ + T \vdash \_)$ *is the formal system (on L)* **axiomatized** *by T in $\vdash$, i.e.* $T' + T \vdash A$ *iff* $T', T \vdash A$.

Definition 1.0.5 Expressive Power and Restriction

If $(L, \vdash)$ is a formal system and L_1 and L_2 are subsets of L, then

- *L_2 has more or equivalent* **expressive power** *than L_1 w.r.t. $\vdash$ ($L_1 \subseteq_\vdash L_2$) $\stackrel{\Delta}{\Longleftrightarrow}$ for all $A \in L_1$ there exists $T \subseteq_{fin} L_2$ s.t. T and $\{A\}$ are equivalent in $\vdash$.*
- *$(\_ \vdash_{L_1} \_)$ is the formal system $\vdash$* **restricted** *to L_1, i.e. $\vdash \cap(\mathcal{P}(L_1) \times L_1)$.*

Notation 1.0.6 We write $L_1 \equiv_\vdash L_2$, if L_1 and L_2 have the **same expressive power**, and $L_1 \subset_\vdash L_2$, if L_1 has **strictly more expressive power** than L_2. The subscript $\vdash$ is omitted when it is clear from the context.

There is an equivalent definition of $L_1 \subseteq_\vdash L_2$, namely: any theory axiomatized by a subset of L_1 can be axiomatized by a subset of L_2. In fact, if $\mathrm{Th}_\vdash(T)$ is axiomatized by a finite subset T', then we can find, by (compactness), $T_0 \subseteq_{fin} T$ which axiomatizes $\mathrm{Th}_\vdash(T)$.

1.1 Comparing formal systems

Given two formal systems, the simplest way to compare them is to look at their *behaviour* on a common sublanguage: *conservative* and *definitional extension* (compare with 1.2 of [Tro73]). However, when the formal systems do not have a common sublanguage, we need a more flexible way to compare them. For this purpose, we define *translation* and related notions: *relative interpretation* and *equivalence*.

Definition 1.1.1 Deductive Power and Conservative Extension

If $(L_1, \vdash_1)$ and $(L_2, \vdash_2)$ are formal systems s.t. $\vdash_2$ is an **extension** *of $\vdash_1$ (i.e. $L_1 \subseteq L_2$ and $\vdash_1 \subseteq \vdash_2$) and X is a subset of $\mathcal{P}(L_1) \times L_1$, then*

- *$\vdash_2$ has* **more deductive power** *than $\vdash_1$ over X ($\vdash_1 \subset_X \vdash_2$) $\stackrel{\Delta}{\Longleftrightarrow}$ $(\vdash_1 \cap X) \subset (\vdash_2 \cap X)$*
- *$\vdash_2$ is an extension of $\vdash_1$* **conservative** *over X ($\vdash_1 \subset^c_X \vdash_2$) $\stackrel{\Delta}{\Longleftrightarrow}$ $(\vdash_1 \cap X) = (\vdash_2 \cap X)$*

Notation 1.1.2 If F_1 and F_2 are subsets of L_1, then we write

- $\vdash_1 \subset_{F_1 \vdash F_2} \vdash_2$ for $\vdash_1 \subset_{\mathcal{P}(F_1) \times F_2} \vdash_2$
- $\vdash_1 \subset^c_{F_1 \vdash F_2} \vdash_2$ for $\vdash_1 \subset^c_{\mathcal{P}(F_1) \times F_2} \vdash_2$

Intuitively, F_1 is the set of sentences that can be used to axiomatize theories, while F_2 is the set of sentences we want to *observe*.

We write $\vdash_1 \subset^c \vdash_2$ for $\vdash_1 \subset^c_{L_1 \vdash L_1} \vdash_2$ and say that $\vdash_2$ is a **conservative extension** of $\vdash_1$.

Remark 1.1.3 The definition of "extension of _ conservative over _" is similar to the definition of "conservative over _ relative to _" for theories based on (intuitionistic) predicate logic (see 1.2 of [Tro73]).

The following facts about $\subset_X$ and $\subset^c_X$ can be used to derive other conservative extension results\failures from known ones:

Lemma 1.1.4 Properties of $\subset^c_X$

$$\text{restr}\ \frac{\vdash_1 \subset^c_Y \vdash_2}{\vdash_1 \subset^c_X \vdash_2}\ X \subseteq Y$$

$$\text{convex}\ \frac{\begin{array}{ccc} \vdash_1 & & \vdash_1 \\ \rotatebox{90}{$\subseteq$} & & \cap^c_X \\ \vdash_2 & \subseteq & \vdash_3 \end{array}}{\begin{array}{ccc} \vdash_1 & & \\ \cap^c_X & & \\ \vdash_2 & \subset^c_X & \vdash_3 \end{array}}$$

$$\text{trans}\ \frac{\vdash_1 \subset^c_X \vdash_2 \subset^c_X \vdash_3}{\vdash_1 \subset^c_X \vdash_3}$$

$$\frac{\begin{array}{ccc} \vdash_1 & \subset_X & \vdash_4 \\ \cap^c_X & & \\ \vdash_2 & \subseteq & \vdash_4 \end{array}}{\vdash_2 \subset_X \vdash_4}$$

Definition 1.1.5 Translation and Pullback

A **translation** *I from L_1 to L_2 is a function from L_1 to $\mathcal{P}_{fin}(L_2)$.*

If $\vdash$ is a formal system on L_2, then:

- *$I^*(\vdash)$ is the* **pullback** *of $\vdash$ along I, i.e. the formal system on L_1 s.t. $T I^*(\vdash) A$ iff $I(T) \vdash I(A)$, where I is extended additively to a function from $\mathcal{P}(L_1)$ to $\mathcal{P}(L_2)$.*

Remark 1.1.6 If $I(A)$ is allowed to be any subset of L_2, then the relation $I^*(\vdash)$ may no longer be a formal system, since (compactness) may fail. In the definition above, we cannot replace $\mathcal{P}_{fin}(L_2)$ by L_2, since L_2 may not be *closed w.r.t. conjunction*. For first order languages it is natural to impose some extra requirements on translations, e.g. preservation of free variables, *compositionality*, commutativity with connectives and quantifiers, preservation of atomic formulae.

Definition 1.1.7 Relative interpretation

If $(L_1, \vdash_1)$ and $(L_2, \vdash_2)$ are formal systems and I is a translation from L_1 to L_2, then

- *I is a* **relative interpretation** *of $\vdash_1$ in $\vdash_2$ $\stackrel{\Delta}{\iff}$*

  $T \vdash_1 A$ implies $I(T) \vdash_2 I(A)$

- *I is a* **conservative interpretation** *of $\vdash_1$ in $\vdash_2$ $\stackrel{\Delta}{\iff}$*

  $T \vdash_1 A$ iff $T \subseteq L_1$, $A \in L_1$ and $I(T) \vdash_2 I(A)$

Remark 1.1.8 $\vdash_2$ is a conservative extension of $\vdash_1$ iff the **identity translation** Id (i.e. $Id(A) = \{A\}$) is a conservative interpretation of $\vdash_1$ in $\vdash_2$. Conversely, I is a conservative interpretation of $\vdash_1$ in $\vdash_2$ iff $I^*(\vdash_2)$ is a conservative extension of $\vdash_1$.

Finally we want to define equivalence of formal systems. There is a very simple definition based on the algebraic lattice of theories:

> $\vdash_1$ is *equivalent* to $\vdash_2$ via f iff $f\colon \mathrm{TH}(\vdash_1) \to \mathrm{TH}(\vdash_2)$ is an isomorphism of posets (algebraic lattices).

However, we give a more direct (but equivalent) definition, which stresses the similarity with the notion of *definitional extension* (see 1.2.4 of [Tro73]):

Definition 1.1.9 Equivalence of Formal Systems

If $(L_1, \vdash_1)$ *and* $(L_2, \vdash_2)$ *are formal systems, then*

- $\vdash_2$ *is a* **definitional extension** *of* $\vdash_1$ ($\vdash_1 \subset^{def} \vdash_2$) $\stackrel{\Delta}{\Longleftrightarrow}$

  $\vdash_1 \subset^c \vdash_2$ *and* $L_2 \subseteq_{\vdash_2} L_1$, *i.e.* $\vdash_2$ *is a conservative extension of* $\vdash_1$ *and* L_1 *has the same expressive power as* L_2 *w.r.t.* $\vdash_2$.

- $\vdash_1$ *is* **equivalent** *to* $\vdash_2$ *via* I *and* J $\stackrel{\Delta}{\Longleftrightarrow}$

  - I *is a relative interpretation of* $\vdash_1$ *in* $\vdash_2$
  - J *is a relative interpretation of* $\vdash_2$ *in* $\vdash_1$
  - $\{A\}$ *and* $J(I(A))$ *are equivalent in* $\vdash_1$, *for all* $A \in L_1$
  - $\{A\}$ *and* $I(J(A))$ *are equivalent in* $\vdash_2$, *for all* $A \in L_2$

The relation between definitional extension, equivalence and isomorphism between algebraic lattices is summarized by the following propositions.

Proposition 1.1.10 Characterization of definitional extension

$\vdash_2$ *is a definitional extension of* $\vdash_1$ *iff there exists a translation* J *s.t.* $\vdash_1$ *is equivalent to* $\vdash_2$ *via* Id *and* J, *where* Id *is the* identity translation. *W.l.o.g. one can also require that* $J(A) = \{A\}$ *for all* $A \in L_1$.

Proof First we prove the implication from left to right. We define the translation J and prove that $J(A) \vdash_2 A$ and $A \vdash_2 J(A)$ for all $A \in L_2$:

- since $L_2 \subseteq_{\vdash_2} L_1$, then there exists $T \subseteq L_1$ which axiomatizes $\mathrm{Th}_{\vdash_2}(\{A\})$. Therefore

  $T \vdash_2 A$ and $A \vdash_2 T$

- by (compactness), there exists $J(A) \subseteq_{fin} T$ s.t.

  $J(A) \vdash_2 A$ and $A \vdash_2 J(A)$

The other properties for equivalence follow easily from $\vdash_1 \subset^c \vdash_2$.

Then we prove the other implication:

- since $\{A\}$ and $J(A) = Id(J(A))$ are equivalent in $\vdash_2$ for all $A \in L_2$, then $L_2 \subseteq_{\vdash_2} L_1$

- since Id is a relative interpretation of $\vdash_1$ in $\vdash_2$, then $\vdash_1 \subseteq \vdash_2$

We have still to prove that $T \subseteq L_1$, $A \in L_1$ and $T \vdash_2 A$ implies $T \vdash_1 A$:

- since J is a relative interpretation, then

  $J(T) \vdash_1 J(A)$

- since $J(T) = J(Id(T))$ is equivalent to T in $\vdash_1$ (for any T), then

  $T \vdash_1 A$

∎

Proposition 1.1.11 *f is an isomorphism of posets from* $\mathrm{TH}(\vdash_1)$ *to* $\mathrm{TH}(\vdash_2)$ *iff there there exist two translations I and J s.t. $f = I: \mathrm{TH}(\vdash_1) \to \mathrm{TH}(\vdash_2)$ and $\vdash_1$ is equivalent to $\vdash_2$ via I and J.*

Proof We prove only the implication from left to right. For each $A \in L_1$ we define a finite subset $I(A)$ of L_2 (the definition of J is similar):

- since f is an isomorphism of posets, then

  f preserves finite elements

- since $\mathrm{Th}_{\vdash_1}(A)$ is a finite elements, then $f(\mathrm{Th}_{\vdash_1}(A))$ is a finite element. Therefore there exists a finite set $I(A)$ s.t.

  $f(\mathrm{Th}_{\vdash_1}(A)) = \mathrm{Th}_{\vdash_2}(I(A))$

We prove that $f = I$:

- by definition

  the morphism I preserves lubs

- since f is an isomorphism of posets, then

  it preserves lubs

- since any $T \in \mathrm{TH}(\vdash_1)$ is the lub of $\{\mathrm{Th}_{\vdash_1}(A) | A \in T\}$, then

  $f(T) = \sqcup_{A \in T} f(\mathrm{Th}_{\vdash_1}(A)) =$ by definition of $I(A)$
  $\sqcup_{A \in T} \mathrm{Th}_{\vdash_2}(I(A)) =$ by definition of I
  $I(T)$

The properties for equivalence are straightforward to check. ∎

1.2 Inference and formation rules

Languages and formal systems will be defined by giving a set of *rules*. Associated with a set of rules there are certain *induction principles*, that can be used to prove properties of a theory or define functions over a language. The formal systems introduced in this thesis are based on Gentzen's *sequent calculus* (see [Gen69]) and their languages are (essentially) sets of *sequents* of *many sorted first order logic*.

Definition 1.2.1 ([Acz77]) Rules
If L is a set, then:

- *a (finitary)* **rule** *on L is a pair $\langle P, C\rangle \in \mathcal{P}_{fin}(L) \times L$*
- *the set $\mathrm{I}_R(T)$* **inductively defined** *from a subset T of L by a set R of rules on L is the smallest subset of L s.t.:*
  - *If $A \in T$, then $A \in \mathrm{I}_R(T)$*
  - *If $P \subseteq_{fin} \mathrm{I}_R(T)$ and $\langle P, C\rangle \in R$, then $C \in \mathrm{I}_R(T)$*
- *the formal system $\vdash^R$* **generated** *by a set R of rules on L is the smallest formal system on L containing R, i.e.*

$$T \vdash^R A \overset{\Delta}{\Longleftrightarrow} A \in \mathrm{I}_R(T)$$

- *a* **derivation** *of $A \in L$ from T by R is a finite tree Π of elements of L s.t.:*
  - *either $\Pi \equiv A$ and $A \in T$*
  - *or there exist $A_1, \ldots, A_n \in L$ and derivations Π_i of A_i from T by R s.t. $\langle\{A_1, \ldots, A_n\}, A\rangle \in R$ and*

$$\Pi \equiv \frac{\Pi_1 \ldots \Pi_n}{A}$$

- *a set R of rules is* **deterministic** $\overset{\Delta}{\Longleftrightarrow}$
  $\langle P, C\rangle, \langle P', C\rangle \in R$ implies $P = P'$

Notation 1.2.2 Rules will be presented in a *schematic* way, by using *metavariables* ranging over *syntactic entities*, like *terms* or *formulae*, which are clear from the context (see [Gen69, Kle52]). More precisely, the rule schema (referred as (name))

$$\text{name}\ \frac{P}{C}\ \text{side-condition}$$

corresponds to the set of all *instances* of $\langle P, C\rangle$ satisfying *side-condition.*

Inductively defined sets enjoy the following *R-induction principle* (see [Acz77]):

- if P is a *property* s.t.
  - if $A \in T$, then $P(A)$
  - if $\langle P, C\rangle \in R$ and $P(A)$ for all $A \in P$, then $P(C)$

  then $P(A)$ for all $A \in \mathrm{I}_R(T)$

Remark 1.2.3 *R-induction* is usually called **induction on the structure** of derivations. Sometimes we also use **induction on the size** (e.g. the number of nodes) of derivations.

1.2.1 Languages and formation rules

Languages will be sets inductively defined from the empty set by a **deterministic** set of rules (called **formation rules**). Since the set of formation rules is deterministic, functions (on a language) can be defined by **induction on the structure** of *elements*. In fact:

Proposition 1.2.4 ([Acz77]) *If R is a deterministic set of rules, then each element of* $\mathrm{I}_R(\emptyset)$ *has exactly one derivation, and the relation "A occurs in the derivation of B" is a* well-founded *partial order on* $\mathrm{I}_R(\emptyset)$*.*

Some syntactic entities (e.g. *terms* and *formulae*) are built up from *variables*, whose *sort* is specified by a *type environment* (i.e. a function from a finite set of variables to *sorts*), by means of *constructors* (i.e. constants, connectives, quantifiers, etc.), that may *bind* some of the variables. The formation rules for these entities define a set of *assertions* $s \in S(\eta)$ (the notation $\eta \vdash s\colon S$ is also widely used in the literature), whose *meaning* is:

s is a well-formed entity of *kind* S in the type environment η.

There are two kinds of formation rules (in the presence of variables) either (x) for some variable x

$$x \quad \frac{}{x \in S_\tau(\eta)} \; S_\tau = \eta(x)$$

or (c) for some constructor c, where the formation rule (c) is a *composition* of the following **basic** rules:

$$\frac{\{s_i \in S_i(\eta) | 1 \leq i \leq n\}}{c(s_1, \ldots, s_n) \in S(\eta)}$$

$$\frac{\Gamma \subseteq_{fin} S'(\eta)}{c(\Gamma) \in S(\eta)}$$

$$\frac{s \in S'(\eta \lfloor x\colon \tau)}{c(x\colon \tau.s) \in S(\eta)}$$

Notation 1.2.5 We use the following notations for sets of syntactic entities with free variables:

- $S(\eta)$ is the set of all syntactic entities s s.t. $s \in S(\eta)$ is derivable from the formation rules
- S is the set of all syntactic entities s s.t. $s \in S(\eta)$ for some η
- S_0 is the set of all syntactic entities s s.t. $s \in S(\emptyset)$

Unlike *terms* and *formulae*, sequents (and *sentences*) do not have free variables. Therefore, their formation rule is a composition of basic rules followed by a **universal closure** rule, which binds all variables in the type environment η (hence all free variables in the syntactic entity s):

$$\frac{s \in S(\eta)}{\eta.s \in \mathrm{Seq}}$$

Remark 1.2.6 To avoid the complications related to bound variables, we assume that *syntactic entities* are defined up to α-conversion (see Pages 26-28 and Appendix C of [Bar81]).

For a syntactic entity s one can define the set $\mathrm{FV}(s)$ of free variables in s and the substitution $s[x\colon = t]$ of t for x in s. This will be done for terms and formulae of first order logic, after being more specific about the language (see Section 2.1).

1.2.2 Formal systems and inference rules

When a set of rules is used to define a formal system, its elements are called **inference rules**.

Notation 1.2.7 We recall some standard terminology from Mathematical Logic (see 1.11 of [Tro73]). If $(L, \vdash)$ is a formal system, T is a subset of L and $\langle P, C\rangle$ is an inference rule on L, then

- $\langle P, C\rangle$ is **derivable** in $\vdash \overset{\Delta}{\Longleftrightarrow} P \vdash C$
- $\langle P, C\rangle$ is **admissible** in $T \overset{\Delta}{\Longleftrightarrow} P \subseteq_{fin} T$ implies $C \in T$

Remark 1.2.8 Formal systems based on *natural deduction* cannot be described by inference rules: they require instead *deduction rules*, that can *bind free variables* and *discharge assumptions* in a derivation.

If $(L_2, \vdash_2)$ is a definitional extension of $(L_1, \vdash_1)$, we describe a procedure that for any set R of inference rules for $\vdash_2$ (i.e. R generates $\vdash_2$) returns a set of inference rules for $\vdash_1$. This procedure depends on a translation J from L_2 onto L_1, whose existence is guaranteed by Proposition 1.1.10.

Proposition 1.2.9 Collapse of inference rules

Let R be a set of rules on L_2 and $L_1 \subseteq L_2$. If J is a translation from L_2 to L_1 s.t.

- *$J(A)$ and $\{A\}$ are equivalent in $\vdash^R$, for all $A \in L_2$*
- *$J(A) = \{A\}$, for all $A \in L_1$*

then the formal system $\vdash^R_{L_1}$ is generated by the set of rules

$$J(R) \triangleq \{\langle J(P), A\rangle | \exists \langle P, C\rangle \in R. A \in J(C)\}$$

Proof By the first property of J, the rules in $J(R)$ are derivable in $\vdash^R$ and therefore in $\vdash^R_{L_1}$. So we have to prove only $\vdash^R_{L_1} \subseteq \vdash^{J(R)}$:

- by induction on the derivation of A from T by R

  for all $T \subseteq L_2$ and $A \in L_2$ if $T \vdash^R A$, then $J(T) \vdash^{J(R)} J(A)$

- by the second property of J

  for all $T \subseteq L_1$ and $A \in L_1$ if $T \vdash^R A$, then $T \vdash^{J(R)} A$

∎

If $(L_1, \vdash_1)$ is the pullback of $(L_2, \vdash_2)$ along a translation I (with certain properties), we describe a procedure that for any set R of inference rules for $\vdash_2$ returns a set of inference rules for $\vdash_1$.

Proposition 1.2.10 Pullback of inference rules

Let I be an injective function from L_1 to L_2. If R is a set of rules on L_2 s.t.

- *$C \in I(L_1)$ implies $P \subseteq_{fin} I(L_1)$ for all $\langle P, C\rangle \in R$*

then the formal system $I^*(\vdash^R)$ is generated by the set of rules

$$I^*(R) \triangleq \{\langle P, C\rangle | \langle I(P), I(C)\rangle \in R\}$$

Remark 1.2.11 The result above is applied mainly when I is the inclusion of L_1 into L_2 and R does not contain *cut-like* inference rules. In fact, if R contains cut-like inference rules, then any *theorem* can occur in the derivation of any other theorem, therefore the image of I has to include $\mathrm{I}_R(\emptyset)$.

Proof By definition of $I^*(R)$, the rules in $I^*(R)$ are derivable in $I^*(\vdash^R)$. So we have to prove only $I^*(\vdash^R) \subseteq \vdash^{I^*(R)}$, or equivalently for all $T \subseteq L_1$ and $A \in L_1$ if $I(T) \vdash^R I(A)$, then $T \vdash^{I^*(R)} A$. We show, by induction on the derivation of A from $I(T)$ by R that for all $C \in L_1$ if $A = I(C)$ and $I(T) \vdash^R A$, then $T \vdash^{I^*(R)} C$:

- $I(C) \in I(T) \Longrightarrow C \in T$, by injectivity of I.
- $\langle P, A\rangle \in R \Longrightarrow$ by IH .
  - Since $A = I(C) \in I(L_1)$, by the property of R

    there exists $Q \subseteq_{fin} L_1$ s.t. $P = I(Q)$
  - by IH for P

    $T \vdash^{I^*(R)} Q$
  - by the rule $\langle Q, C\rangle \in I^*(\langle P, A\rangle) \subseteq I^*(R)$

    $T \vdash^{I^*(R)} C$

∎

1.3 Logics and models

By *logic* we mean a triple $(L, Mod, \models)$, where L is a *language*, Mod is a class of *models* and $\models$ is a *satisfaction relation*. We introduce also the derived notion of *logical consequence relation*. Logics will be used to *justify* a formal system, by proving that it is *sound* and *complete* for some logic.

Definition 1.3.1 Logic and logical consequence relation

A **logic** *is a triple $(L, Mod, \models)$, where L is a set, Mod is a class and $\models$ is a subclass of $Mod \times L$.*

The **logical consequence relation**, *also denoted by $\models$, is the subset of $\mathcal{P}(L) \times L$ s.t.*

$$T \models A \stackrel{\Delta}{\Longleftrightarrow} \textit{for all } M \in Mod \textit{ if } M \models A' \textit{ for all } A' \in T, \textit{ then } M \models A$$

Remark 1.3.2 In practice, L is a language inductively defined by a deterministic set of rules and $M \models A$ is defined by induction on the *structure* of A.

Notation 1.3.3 We introduce some terminology in relation to models. Let M be a model:

- if $M \models A$ for all $A \in T$, then we say that M is a **model** of T (notation $M \models T$), M **satisfies** T, T is **valid** in M or T is **true** in M.
- if $M \models P$ implies $M \models C$, then we say that the rule $\langle P, C\rangle$ is **admissible** (or **valid**) in M.

Definition 1.3.4 Soundness and Completeness

Let $(L, Mod, \models)$ be a logic and $(L, \vdash)$ a formal system:

- $\vdash$ *is* **sound** *(for $\models$) iff $\vdash \subseteq \models$*
- $\vdash$ *is* **complete** *(for $\models$) iff $\models \subseteq \vdash$*

Soundness and completeness results are very useful even when one is interested only in formal systems. In fact, one can establish relations between formal systems as a consequence of *model-theoretic* results.

Proposition 1.3.5 Embedding and Counterexamples

Let $(L_1, \vdash_1)$ and $(L_2, \vdash_2)$ be formal systems s.t. $\vdash_1 \subseteq \vdash_2$ and $\vdash_i$ is sound and complete for the logic $(L_i, Mod_i, \models_i)$ (for $i = 1, 2$)

- **embedding** $\vdash_1 \subset^c_{F_1 \vdash F_2} \vdash_2$ *provided*

  for all $M_1 \in Mod_1$ *there exists* $M_2 \in Mod_2$ *s.t.*

  1. *for all* $A \in F_1$ *if* $M_1 \models_1 A$*, then* $M_2 \models_2 A$
  2. *for all* $A \in F_2$ *if* $M_2 \models_2 A$*, then* $M_1 \models_1 A$

- **counterexample** $\vdash_1 \subset_X \vdash_2$ *provided*

  for some $(T, A) \in X$ *s.t.* $T \vdash_2 A$ *there exists* $M_1 \in Mod_1$ *s.t.* $M_1 \models_1 T$*, but not* $M_1 \models_1 A$*.*

Proposition 1.3.5 can be applied also to prove that a translation I from L_1 to L_2 is a conservative interpretation of $\vdash_1$ in $\vdash_2$ (equivalently $\vdash_1 = I^*(\vdash_2)$), provided we *pullback* the logic $(L_2, Mod_2, \models_2)$ along I in the *same* way as $\vdash_2$.

Definition 1.3.6 Pullback of Logics

Let I *be a translation from* L_1 *to* L_2*. If* $(L_2, Mod, \models)$ *is a logic, then its* **pullback** *along* I *is* $(L_1, Mod, I^*(\models))$*, where* $M I^*(\models) A \overset{\Delta}{\Longleftrightarrow} M \models I(A)$*.*

Proposition 1.3.7 *Let* I *be a translation from* L_1 *to* L_2*. If* $\vdash$ *is sound (complete) for the logic* $(L_2, Mod, \models)$*, then* $I^*(\vdash)$ *is sound (complete) for the logic* $(L_1, Mod, I^*(\models))$*.*

Proof Straightforward. ∎

Chapter 2

The logic of partial terms

In this chapter we introduce a first order logic for many sorted partial algebras and more generally relational structures with partial functions.

> Just considering partial functions as *single-valued* relations would be too clumsy, because equations would become formulae involving existential quantifiers and conjunctions. Moreover, for each partial function f in the structure one would have to add an axiom saying that the corresponding relation p_f is single-valued, i.e. if $p_f(\overline{x}, y)$ and $p_f(\overline{x}, z)$ then $y = z$.

The language is essentially that described in [Bur82] and Chapter VI of [Bee85] with sequents on top. For each *many sorted signature* we define two logics, *classical* and *intuitionistic* free logic of partial terms (LPT), that share the same language, but have different classes of models. The logical consequence relation of each logic can be described by a set of inference rules in a *sequent calculus style* (see [Gen69, Kle52]).

Remark 2.0.8 The sequent calculus presentation stresses the similarities between the formal systems for classical and intuitionistic logic better than any other presentation of first order logic (e.g. natural deduction or Hilbert-style). This is particularly useful when comparing classical and intuitionistic formal systems.

These logics provide a framework for defining other (*logics* and) *formal systems*. In fact, all formal systems introduced in this thesisare *equivalent* to a formal system *axiomatizable* in one of the two logics, possibly restricted to a *fragment* of the language. For instance, the language of the partial lambda calculus (λ_{p}-calculus) is not a first order language (because of lambda-abstraction), but the λ_p-calculus is *equivalent* to a formal system *directly* axiomatizable in first order logic.

As an alternative to LPT we could have chosen the Logic of Partial Elements (LPE) described in [Sco79, Fou77], where the fundamental concept is that of *partial element* and partial functions are represented as *strict* functions on partial elements. However, LPE is not such a simple-minded and straightforward choice for the kind of structures we want to consider, i.e. partial algebras. In particular, the set of partial elements of a (Kripke) partial algebra is not completely specified by the (Kripke) partial algebra itself. In Section 3.3 we will discuss in more details LPE and its relations with LPT.

2.1 The language of partial terms

Languages are sets of *sequents* and will be defined according to the following pattern:

1. a set of sorts
2. for each sort and type environment a set of terms
3. for each type environment a set of formulae
4. a set of sequents

This pattern is already familiar from many sorted first order logic (see [Ten78]), although in the literature it is more common to take as languages sets of formulae or closed formulae instead of sets of sequents.

Sequents are *syntactic entities* of the form $\eta.\Gamma_1 \Longrightarrow \Gamma_2$, where the **antecedent** Γ_1 and the **succedent** Γ_2 are finite sets of formulae whose free variables are *typed* in the type environment η.

There are *features* that are common to all many sorted first order languages (e.g. logical connectives and quantifiers), while others (e.g. sort or constant symbols) may change. A many sorted first order language will be specified by a *signature*. We may impose some constraints on signatures to capture *features* of the models, that we want to describe; e.g. in the case of applicative structures we want also function spaces and an application operation for each pair of sorts (see Definition 4.2.9).

Definition 2.1.1 Signature

A **signature** Σ *for* **partial algebras** *is:*

1. *a set* $\mathrm{Sort}(\Sigma)$ *of sort symbols* τ
2. *for each* $\tau \in \mathrm{Sort}(\Sigma)$ *a set* $\mathrm{Const}_\tau(\Sigma)$ *of constant symbols* c
3. *for each* $\overline{\tau} \in \mathrm{Sort}(\Sigma)^*$ *and* $\tau \in \mathrm{Sort}(\Sigma)$ *a set* $\mathrm{Funct}_{(\overline{\tau})\to\tau}(\Sigma)$ *of function symbols* f
4. *for each* $\overline{\tau} \in \mathrm{Sort}(\Sigma)^*$ *and a set* $\mathrm{Pred}_{\overline{\tau}}(\Sigma)$ *of predicate symbols* p

For simplicity we assume that **all these sets are disjoint and countable**.

Notation 2.1.2 We introduce some terminology for signatures:

- a signature Σ is **included** in Σ' (notation $\Sigma \subseteq \Sigma'$) iff $\mathrm{Sort}(\Sigma) \subseteq \mathrm{Sort}(\Sigma')$ and for all $\tau \in \mathrm{Sort}(\Sigma)$ and $\overline{\tau} \in \mathrm{Sort}(\Sigma)^*$:
  - $\mathrm{Const}_\tau(\Sigma) \subseteq \mathrm{Const}_\tau(\Sigma')$
  - $\mathrm{Funct}_{(\overline{\tau})\to\tau}(\Sigma) \subseteq \mathrm{Funct}_{(\overline{\tau})\to\tau}(\Sigma')$
  - $\mathrm{Pred}_{\overline{\tau}}(\Sigma) \subseteq \mathrm{Pred}_{\overline{\tau}}(\Sigma')$
- If Σ is a signature and $\tau \in \mathrm{Sort}(\Sigma)$, then we write Σ^τ for Σ **restricted** to τ, i.e. the *biggest* signature included in Σ, whose only sort symbol is τ
- If Σ and Σ' are signatures, then we write $\Sigma \uplus \Sigma'$ for the **union** of Σ and Σ', i.e. the *smallest* signature which includes both Σ and Σ'

Remark 2.1.3 In LPT it is convenient to consider only signatures Σ with a **existence** predicate $\_\downarrow_\tau \in \mathrm{Pred}_\tau(\Sigma)$ and an **equality** predicate $\_ =_\tau \_ \in \mathrm{Pred}_{\tau,\tau}(\Sigma)$ for any $\tau \in \mathrm{Sort}(\Sigma)$. These predicates have a standard interpretation (see Sections 2.2 and 2.3), namely: $t \downarrow_\tau$ means that the term t is defined, and $t =_\tau t'$ means that the terms t and t' are both defined and equal. The subscript τ will be omitted when it can be derived from the context.

Type environments are used to specify the sorts of free variables (alternatively we would have to attach the sort to each occurrence of a variable):

Definition 2.1.4 Type Environment

Given a signature Σ*, a* **type environment** η *for* Σ *is a function from a finite set of* variables *to* $\mathrm{Sort}(\Sigma)$*. For type environments we write* $x{:}\,\tau$ *(instead of* $x{:}=\tau$*) for the function* $\{\langle x,\tau\rangle\}$*.*

Definition 2.1.5 Language of LPT over Σ

Given a signature Σ*,*

1. *the set* $\mathrm{Term}^{\Sigma}_{\tau}(\eta)$ *of* **terms** *of sort* τ *in the type environment* η
2. *the set* $\mathrm{Form}^{\Sigma}(\eta)$ *of* **formulae** *in the type environment* η
3. *the set* Seq^{Σ} *of* **sequents**

are defined by the following formation rules*:*

$$x\quad \frac{}{x \in \mathrm{Term}^{\Sigma}_{\eta(x)}(\tau)}\ \tau = \eta(x)$$

$$c\quad \frac{}{c \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)}\ c \in \mathrm{Const}_{\tau}(\Sigma)$$

$$f\quad \frac{\{\overline{t}_i \in \mathrm{Term}^{\Sigma}_{\overline{\tau}_i}(\eta) | i \in |\overline{\tau}|\}}{f(\langle \overline{t}_i | i \in |\overline{\tau}|\rangle) \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)}\ f \in \mathrm{Funct}_{(\overline{\tau})\rightarrow\tau}(\Sigma)$$

$$\bot\quad \bot \in \mathrm{Form}^{\Sigma}(\eta)$$

$$p\quad \frac{\{\overline{t}_i \in \mathrm{Term}^{\Sigma}_{\overline{\tau}_i}(\eta) | i \in |\overline{\tau}|\}}{p(\langle \overline{t}_i | i \in |\overline{\tau}|\rangle) \in \mathrm{Form}^{\Sigma}(\eta)}\ p \in \mathrm{Pred}_{\overline{\tau}}(\Sigma)$$

$$\wedge\quad \frac{A_1 \in \mathrm{Form}^{\Sigma}(\eta) \qquad A_2 \in \mathrm{Form}^{\Sigma}(\eta)}{A_1 \wedge A_2 \in \mathrm{Form}^{\Sigma}(\eta)}$$

$$\vee\quad \frac{A_1 \in \mathrm{Form}^{\Sigma}(\eta) \qquad A_2 \in \mathrm{Form}^{\Sigma}(\eta)}{A_1 \vee A_2 \in \mathrm{Form}^{\Sigma}(\eta)}$$

$$\rightarrow\quad \frac{A_1 \in \mathrm{Form}^{\Sigma}(\eta) \qquad A_2 \in \mathrm{Form}^{\Sigma}(\eta)}{A_1 \rightarrow A_2 \in \mathrm{Form}^{\Sigma}(\eta)}$$

$$\forall\quad \frac{A \in \mathrm{Form}^{\Sigma}(\eta\lfloor x{:}\,\tau)}{\forall x{:}\,\tau.A \in \mathrm{Form}^{\Sigma}(\eta)}$$

$$\exists\ \frac{A \in \mathrm{Form}^{\Sigma}(\eta \lfloor x\colon \tau)}{\exists x\colon \tau.A \in \mathrm{Form}^{\Sigma}(\eta)}$$

$$\Longrightarrow\ \frac{\Gamma_1 \subseteq_{fin} \mathrm{Form}^{\Sigma}(\eta) \qquad \Gamma_2 \subseteq_{fin} \mathrm{Form}^{\Sigma}(\eta)}{\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}}$$

Notation 2.1.6 The definition of $\mathrm{Term}^{\Sigma}_{-}(\eta)$ is extended to indexed families of sort symbols (according to the general conventions). If η' is a type environment, then an element σ of $\mathrm{Term}^{\Sigma}_{\eta'}(\eta)$, i.e. a function mapping variables in $\mathrm{dom}(\eta')$ to terms of the appropriate type, is called η'-**substitution**.

Notation 2.1.7 $p(\bar{t})$ is called an **atomic formula** ($\perp$ is not atomic). *Negation* $\neg A$ is defined as $A \rightarrow \perp$, while the logical constant *true* $\top$ is defined as $\neg\perp$. Kleene equality $t_1 \simeq t_2$ is an abbreviation for $(t_1 \downarrow \rightarrow t_1 = t_2) \wedge (t_2 \downarrow \rightarrow t_1 = t_2)$.

If f and g are two I-indexed families of terms in $\mathrm{Term}^{\Sigma}_{h}$, then we write $f =_h g$ for the set of formulae $\{f_i =_{h_i} g_i | i \in I\}$. A similar convention is used for other predicate symbols of given arity.

We write $\wedge(\Gamma)$ for the conjunction (in some irrelevant order) of the formulae in the finite set Γ, and a similar convention is used for $\vee$.

We write $(\forall x_1\colon \tau_1, \ldots, x_n\colon \tau_n.A)$ for $(\forall x_1\colon \tau_1.\ldots\forall x_n\colon \tau_n.A)$ and $\forall\eta.A$ for universal quantification of A w.r.t. all the variables in $\mathrm{dom}(\eta)$ (in some irrelevant order). Similar conventions are used for $\exists$.

Notation 2.1.8 Often antecedents and succedents will be treated as sequences (rather than sets) of formulae. For instance, $\eta. \Longrightarrow \Gamma_1, A, \Gamma_2$ stands for the sequent $\eta.\emptyset \Longrightarrow \Gamma_1 \cup \{A\} \cup \Gamma_2$. We will also make use of the following shorthands:

- $\Gamma_1 \Longrightarrow \Gamma_2$ stands for the sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2$, where η is a type environment for the free variables of $\Gamma_1 \cup \Gamma_2$ s.t. $\Gamma_1 \cup \Gamma_2 \subseteq \mathrm{Form}^{\Sigma}(\eta)$. Since each predicate and function symbol has a unique *arity*, then η (if it exists) is **unique**

- $\eta.\Gamma$ stands for the sequent $\eta.\emptyset \Longrightarrow \Gamma$

- If A is a formula, then A may stand for the sequent $\emptyset \Longrightarrow A$

The following properties of the languages defined above are obvious and in the sequel they will be used implicitly:

Lemma 2.1.9 Signature Extension preserves Well-Formedness

If $\Sigma \subseteq \Sigma'$*, then for all type environments* η *and sorts* τ *for* $\mathrm{Sort}(\Sigma)$*:*

- $\mathrm{Term}^{\Sigma}_{\tau}(\eta) \subseteq \mathrm{Term}^{\Sigma'}_{\tau}(\eta)$
- $\mathrm{Form}^{\Sigma}(\eta) \subseteq \mathrm{Form}^{\Sigma'}(\eta)$
- $\mathrm{Seq}^{\Sigma} \subseteq \mathrm{Seq}^{\Sigma'}$

Lemma 2.1.10 Type Environment Extension preserves Well-Formedness

Given a type environment η *and a sort* τ *for* Σ*:*

- *if* $t \in \mathrm{Term}^{\Sigma}_{\tau'}(\eta)$ *then* $t \in \mathrm{Term}^{\Sigma}_{\tau'}(x\colon\tau\lfloor\eta)$
- *if* $A \in \mathrm{Form}^{\Sigma}(\eta)$ *then* $A \in \mathrm{Form}^{\Sigma}(x\colon\tau\lfloor\eta)$
- *if* $(\eta.\Gamma_1 \Longrightarrow \Gamma_2) \in \mathrm{Seq}^{\Sigma}$ *then* $(x\colon\tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2) \in \mathrm{Seq}^{\Sigma}$

We recall the familiar notion of free variable and substitution:

Definition 2.1.11 Free variables and Substitution

The set $\mathrm{FV}(\_)$ *of* **free variables** *of* $\_$ *is defined by induction on the structure of terms and formulae:*

- $\mathrm{FV}(x) \triangleq \{x\}$
- $\mathrm{FV}(c) \triangleq \emptyset$
- $\mathrm{FV}(f(\overline{t})) \triangleq \cup_{i\in|\overline{t}|}\mathrm{FV}(t_i)$ *and similarly for predicates and logical connectives*
- $\mathrm{FV}(\forall x\colon\tau.A) \triangleq \mathrm{FV}(A) - \{x\}$ *and similarly for the existential quantifier.*

If σ *is a* η*-substitution, then the* σ-**substitution** *instance* $\_[\sigma]$ *is defined by induction on the structure of terms and formulae:*

- $x[\sigma] \triangleq \begin{cases} \sigma(x) & \text{if } x \in \mathrm{dom}(\sigma) \\ x & \text{otherwise} \end{cases}$
- $c[\sigma] \triangleq c$
- $f(\overline{t})[\sigma] \triangleq f(\overline{t}[\sigma])$ *and similarly for predicates and logical connectives*

- $(\forall x\colon\tau.A)[\sigma] \stackrel{\Delta}{\equiv} \forall x\colon\tau.A[\sigma]$ *and similarly for the existential quantifier.*

  By α-conversion we can assume that x is not in $\mathrm{dom}(\sigma)$ and does not occur free in $\sigma(x')$ for any $x' \in \mathrm{dom}(\sigma)$

The definition of $\mathrm{FV}(\_)$ and $\_[\sigma]$ is extended, in the obvious way, to families and sets of terms or formulae.

The application of a substitution to a well-formed expression results in a well-formed expression provided every substituted variable has the same sort as the term it is substituted with.

Lemma 2.1.12 Substitution preserves Well-Formedness
Given a signature Σ and a η'-substitution $\sigma \in \mathrm{Term}^{\Sigma}_{\eta'}(\eta)$:

- *If $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta\lfloor\eta')$ then $t[\sigma] \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$*
- *If $A \in \mathrm{Form}^{\Sigma}(\eta\lfloor\eta')$ then $A[\sigma] \in \mathrm{Form}^{\Sigma}(\eta)$*
- *If $(\eta\lfloor\eta'.\Gamma_1 \Longrightarrow \Gamma_2) \in \mathrm{Seq}^{\Sigma}$ then $(\eta.\Gamma_1[\sigma] \Longrightarrow \Gamma_2[\sigma]) \in \mathrm{Seq}^{\Sigma}$*

Definition 2.1.13 Fragments

- *A* **fragment of the formulae** *is a subset F of Form^{Σ} closed w.r.t. substitution, i.e. if $A \in \mathrm{Form}^{\Sigma}(\eta)$ is in F and $\sigma \in \mathrm{Term}^{\Sigma}_{\eta}(\eta')$, then $A[\sigma]$ is in F.*

- *A* **fragment of the sequents** *is a subset F of Seq^{Σ}* **closed w.r.t. substitution***, i.e. if $(\eta.\Gamma \Longrightarrow \Gamma')$ is in F and $\sigma \in \mathrm{Term}^{\Sigma}_{\eta}(\eta')$, then $(\eta'.\Gamma[\sigma] \Longrightarrow \Gamma'[\sigma])$ is in F.*

Notation 2.1.14 We will make use of the following notations for fragments:

- If $\ldots$ is a set of connectives and quantifiers, i.e. $\ldots \subseteq \{\bot \wedge \vee \exists \forall \rightarrow\}$, then $\mathrm{Frag}^{\Sigma}(\ldots)$ is the fragment of Form^{Σ} built up from atomic formulae by using only $\ldots$ as *constructors*.

  Moreover, $\mathrm{Frag}^{\Sigma}(\ldots)(\eta)$ and $\mathrm{Frag}^{\Sigma}_0(\ldots)$ are the intersection of $\mathrm{Frag}^{\Sigma}(\ldots)$ with $\mathrm{Form}^{\Sigma}(\eta)$ and Form^{Σ}_0 respectively

- JSeq^{Σ} is the fragment of Seq^{Σ} whose elements are of the form $\eta.\Gamma \Longrightarrow A$, i.e. the succedent is a singleton.

- if F and F' are fragments of Form^Σ, then $\{F\} \Longrightarrow \{F'\}$ is the fragment of Seq^Σ whose elements are the sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ s.t. Γ_1 is a finite subset of F and Γ_2 is a finite subset of F'.

  Moreover, $\{F\} \Longrightarrow F'$ is the intersection of $\{F\} \Longrightarrow \{F'\}$ and JSeq^Σ.

2.2 Classical free logic of partial terms

The definition of Σ-structure is essentially that in [Bur82, Bee85] and differs from the usual definition for many sorted first order logic in two ways:

- the interpretation of a sort symbol may be the empty set (this is why the logic is called *free*). We allow empty sorts both for the sake of generality, and because they arise quite naturally
- the interpretation of a function symbol is a partial function, and therefore the interpretation of a term (given a denotation for its free variables) may be undefined

Morphisms between Σ-structures are defined as *weak homomorphisms* (see [Bur82]); this amounts to interpreting partial function symbols as (single-valued) relations and to taking the usual homomorphisms between relational structures as morphisms. This choice of morphism is crucial to get the *right* definition of Kripke Σ-structure (see Definition 2.3.1).

Definition 2.2.1 Structure, Morphism and Reduct Functor
Given a signature Σ, we define the **category of Σ-structures** $\mathrm{Struct}(\Sigma)$*:*

- *A* **Σ-structure** $\mathcal{A}$ *in* $\mathrm{Struct}(\Sigma)$ *is a function* $\_^{\mathcal{A}}$ *defined on the symbols of Σ and satisfying the following properties:*

  1. *if $\tau \in \mathrm{Sort}(\Sigma)$, then $\tau^{\mathcal{A}}$ is a set, possibly empty.*

     Since $\langle \tau^{\mathcal{A}} | \tau \in \mathrm{Sort}(\Sigma)\rangle$ is a $\mathrm{Sort}(\Sigma)$-indexed family of sets, we can use the general conventions and write $f^{\mathcal{A}}$ for the product $\prod_{i\in I} f_i^{\mathcal{A}}$, whenever f is an I-indexed family of sorts.
  2. *if $c \in \mathrm{Const}_\tau(\Sigma)$, then $c^{\mathcal{A}} \in \tau^{\mathcal{A}}$*
  3. *if $f \in \mathrm{Funct}_{(\overline{\tau})\rightharpoonup\tau}(\Sigma)$, then $f^{\mathcal{A}}\colon \overline{\tau}^{\mathcal{A}} \rightharpoonup \tau^{\mathcal{A}}$*
  4. *$\_\downarrow_\tau^{\mathcal{A}}$ is the set $\tau^{\mathcal{A}}$*

5. $\_ =_\tau \_^{\mathcal{A}}$ *is the diagonal relation on* $\tau^{\mathcal{A}}$*, i.e.* $\{\langle a, a\rangle | a \in \tau^{\mathcal{A}}\}$

6. *for the other predicate symbols, if* $p \in \mathrm{Pred}_{\overline{\tau}}(\Sigma)$*, then* $p^{\mathcal{A}} \subseteq \overline{\tau}^{\mathcal{A}}$

- *A* Σ-**morphism** ϕ *from* $\mathcal{A}_1$ *to* $\mathcal{A}_2$ *is a* $\mathrm{Sort}(\Sigma)$*-indexed family of total functions* $\phi_\tau : \tau^{\mathcal{A}_1} \to \tau^{\mathcal{A}_2}$ *s.t.:*

  1. *if* $c \in \mathrm{Const}_\tau(\Sigma)$*, then* $c^{\mathcal{A}_2} = \phi_\tau(c^{\mathcal{A}_1})$
  2. *if* $f \in \mathrm{Funct}_{(\overline{\tau})\to\tau}(\Sigma)$ *and* $f^{\mathcal{A}_1}(\overline{a})$ *is defined, then* $f^{\mathcal{A}_2}(\phi_{\overline{\tau}}(\overline{a})) = \phi_\tau(f^{\mathcal{A}_1}(\overline{a}))$
  3. *if* $p \in \mathrm{Pred}_{\overline{\tau}}(\Sigma)$ *and* $\overline{a} \in p^{\mathcal{A}_1}$*, then* $\phi_\tau(\overline{a}) \in p^{\mathcal{A}_2}$

Given two signatures Σ *and* Σ' *s.t.* $\Sigma \subseteq \Sigma'$*, the* **reduct functor** $\_\restriction\Sigma$ *is the functor from* $\mathrm{Struct}(\Sigma')$ *to* $\mathrm{Struct}(\Sigma)$ *s.t. for all* Σ'*-structures* $\mathcal{A}$*:*

- *if* $\tau \in \mathrm{Sort}(\Sigma)$*, then* $\tau^{\mathcal{A}\restriction\Sigma} \triangleq \tau^{\mathcal{A}}$
- *if* $c \in \mathrm{Const}_\tau(\Sigma)$*, then* $c^{\mathcal{A}\restriction\Sigma} \triangleq c^{\mathcal{A}}$
- *if* $f \in \mathrm{Funct}_{(\overline{\tau})\to\tau}(\Sigma)$*, then* $f^{\mathcal{A}\restriction\Sigma} \triangleq f^{\mathcal{A}}$
- *if* $p \in \mathrm{Pred}_{\overline{\tau}}(\Sigma)$*, then* $p^{\mathcal{A}\restriction\Sigma} \triangleq p^{\mathcal{A}}$

and for all Σ'*-morphisms* ϕ*:*

- $(\phi\restriction\Sigma)_\tau \triangleq \phi_\tau$*, for all* $\tau \in \mathrm{Sort}(\Sigma)$

Notation 2.2.2 If η is a type environment, then an element ρ of $\eta^{\mathcal{A}}$ is called an η-**environment**.

The interpretation of terms and formulae depends on an environment, which assigns an interpretation to the free variables that may occur in them.

Definition 2.2.3 Interpretation in a Structure

Given a Σ*-structure* $\mathcal{A}$*, the denotation* $[\![t]\!]^{\mathcal{A}}_\rho$ *of a term* $t \in \mathrm{Term}^{\Sigma}_\tau(\eta)$ *in an environment* $\rho \in \eta^{\mathcal{A}}$ *is defined by induction on the structure of* t*:*

- $[\![x]\!]^{\mathcal{A}}_\rho \overset{\Delta}{\simeq} \rho(x)$
- $[\![c]\!]^{\mathcal{A}}_\rho \overset{\Delta}{\simeq} c^{\mathcal{A}}$
- $[\![f(\overline{t})]\!]^{\mathcal{A}}_\rho \overset{\Delta}{\simeq} f^{\mathcal{A}}([\![\overline{t}]\!]^{\mathcal{A}}_\rho)$

$[\![t]\!]^{\mathcal{A}}_{\rho}$ *is either an element of* $\tau^{\mathcal{A}}$ *or is undefined.*

The assertion "$\rho \in \eta^{\mathcal{A}}$ **satisfies** *the formula* $A \in \mathrm{Form}^{\Sigma}(\eta)$*" (*$\mathcal{A}, \rho \models A$*) is defined by induction on the structure of* A*:*

- $\mathcal{A}, \rho \models p(\bar{t}) \overset{\Delta}{\iff} [\![\bar{t}]\!]^{\mathcal{A}}_{\rho}$ (is defined and) is in $p^{\mathcal{A}}$
- $\mathcal{A}, \rho \models A_1 \wedge A_2 \overset{\Delta}{\iff} \mathcal{A}, \rho \models A_1$ and $\mathcal{A}, \rho \models A_2$
- $\mathcal{A}, \rho \models A_1 \vee A_2 \overset{\Delta}{\iff} \mathcal{A}, \rho \models A_1$ or $\mathcal{A}, \rho \models A_2$
- $\mathcal{A}, \rho \models A_1 \to A_2 \overset{\Delta}{\iff}$ if $\mathcal{A}, \rho \models A_1$, then $\mathcal{A}, \rho \models A_2$
- $\mathcal{A}, \rho \models \forall x\colon \tau.A \overset{\Delta}{\iff} \mathcal{A}, \rho\lfloor x := a \models A$ for all $a \in \tau^{\mathcal{A}}$
- $\mathcal{A}, \rho \models \exists x\colon \tau.A \overset{\Delta}{\iff} \mathcal{A}, \rho\lfloor x := a \models A$ for some $a \in \tau^{\mathcal{A}}$

The sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ *is* **valid** *in* $\mathcal{A}$ *(*$\mathcal{A} \models \eta.\Gamma_1 \Longrightarrow \Gamma_2$*) iff:*

- for all $\rho \in \eta^{\mathcal{A}}$ if $\mathcal{A}, \rho \models \Gamma_1$, then $\mathcal{A}, \rho \models A$ for some $A \in \Gamma_2$

Notation 2.2.4 If ρ is an η-environment and f is a J-indexed family of terms in $\mathrm{Term}^{\Sigma}(\eta)$, then $[\![f]\!]^{\mathcal{A}}_{\rho}$ is (according to the general conventions) the J-indexed family $\langle [\![f_j]\!]^{\mathcal{A}}_{\rho} | j \in J \rangle$, provided $[\![f_j]\!]^{\mathcal{A}}_{\rho} \downarrow$ for all $j \in J$.

Moreover, if $\Gamma \subseteq \mathrm{Form}^{\Sigma}(\eta)$, then the assertion $\mathcal{A}, \rho \models \Gamma$ is true iff $\mathcal{A}, \rho \models A$ for all $A \in \Gamma$.

From the definition of interpretation in a model it is easy to show that the validity of a sequent in a Σ-structure does not depend on the interpretation of the symbols of Σ that do not occur in the sequent, in fact:

Lemma 2.2.5 Satisfaction Lemma

Given $\Sigma \subseteq \Sigma'$*, a* Σ'*-structure* $\mathcal{A}$*, a type environment* η *for* Σ *and an* η*-environment* $\rho \in \eta^{\mathcal{A}\restriction\Sigma} (= \eta^{\mathcal{A}})$*:*

- *if* $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$*, then* $[\![t]\!]^{\mathcal{A}\restriction\Sigma}_{\rho} \simeq [\![t]\!]^{\mathcal{A}}_{\rho}$
- *if* $A \in \mathrm{Form}^{\Sigma}(\eta)$*, then* $\mathcal{A}\restriction\Sigma, \rho \models A$ *iff* $\mathcal{A}, \rho \models A$
- *if* $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}$ *then* $\mathcal{A}\restriction\Sigma \models \eta.\Gamma_1 \Longrightarrow \Gamma_2$ *iff* $\mathcal{A} \models \eta.\Gamma_1 \Longrightarrow \Gamma_2$

2.3 Intuitionistic free logic of partial terms

For intuitionistic Logic there are various definitions of model: topological models, Kripke and Beth models (see [vD86]), logical categories and topoi (see [MR77, LS86]). All of them induce the same logical consequence relation, so the choice is quite arbitrary.

We have chosen a variant of Kripke models, which is equivalent to the Kripke-Joyal semantics in a functor category $\mathbf{SET}^{\mathrm{K}}$ (see [LS86]). With this definition any intuitionistic theory is the theory of a model (see Lemma 2.6.11); even the inconsistent theory has a model, namely the one with K empty. The *usual* definition of Kripke model (see [Tro73, vD86]) corresponds to those K that are partial orders with a least element. And most of the Kripke models defined in the sequel will be over a partial order with a least element anyway.

Definition 2.3.1 Kripke Structure and Reduct Function

Given a signature Σ *and a small category* K*, the* **category of Kripke** Σ**-structures over** K *is the functor category* $\mathrm{Struct}(\Sigma)^{\mathrm{K}}$*.*

Given two signatures Σ *and* Σ' *s.t.* $\Sigma \subseteq \Sigma'$*, the* **reduct functor** $\_\restriction\Sigma$ *from* $\mathrm{Struct}(\Sigma')$ *to* $\mathrm{Struct}(\Sigma)$ *induces a* **reduct functor** $(\_\restriction\Sigma)^{\mathrm{K}}$ *(also denoted by* $\_\restriction\Sigma$*) from* $\mathrm{Struct}(\Sigma')^{\mathrm{K}}$ *to* $\mathrm{Struct}(\Sigma)^{\mathrm{K}}$

Notation 2.3.2 An element α of K is usually called a **stage of knowledge**. We write $\alpha \leq \alpha'$ to indicate that there exists a morphism from α to α' in K.

We write $\tau^{\mathcal{B}}$ for the functor from K to **SET** s.t. $\tau^{\mathcal{B}}(\alpha) = \tau^{\mathcal{B}(\alpha)}$ and $\tau^{\mathcal{B}}(f) = \mathcal{B}(f)_{\tau}$. The functors $\overline{\tau}^{\mathcal{B}}$ and $\eta^{\mathcal{B}}$ are defined similarly.

If $f\colon \alpha \to \alpha'$ is a morphism in K and $a \in \tau^{\mathcal{B}}(\alpha)$, then we write $a_{\restriction f}$ for a **after** f, i.e. the element $\tau^{\mathcal{B}}(f)(a)$ of $\tau^{\mathcal{B}}(\alpha')$. We introduce similar shorthands for $\overline{\tau}^{\mathcal{B}}(f)(\overline{a})$ and $\eta^{\mathcal{B}}(f)(\rho)$.

The interpretation of terms and atomic formulae at a certain stage of knowledge α is exactly their interpretation in $\mathcal{B}(\alpha)$. But for non atomic formulae the interpretation at α may depend on that at $\alpha' \geq \alpha$ (in a suitably transformed environment).

Definition 2.3.3 Interpretation in a Kripke Structure

Given a Kripke Σ*-structure* $\mathcal{B}\colon \mathrm{K} \to \mathrm{Struct}(\Sigma)$*, the denotation* $[\![t]\!]^{\mathcal{B},\alpha}_{\rho}$ *of a term* $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$ *in an environment* $\rho \in \eta^{\mathcal{B}(\alpha)}$ *at a stage* α *is* $[\![t]\!]^{\mathcal{B}(\alpha)}_{\rho}$*.*

The assertion "$\rho \in \eta^{\mathcal{B}(\alpha)}$ **satisfies** *the formula $A \in \mathrm{Form}^{\Sigma}(\eta)$ at stage α" ($\mathcal{B}, \alpha, \rho \Vdash A$) is defined by induction on the structure of A:*

- $\mathcal{B}, \alpha, \rho \Vdash p(\bar{t}) \overset{\Delta}{\iff} \mathcal{B}(\alpha), \rho \models p(\bar{t})$
- $\mathcal{B}, \alpha, \rho \Vdash A_1 \wedge A_2 \overset{\Delta}{\iff} \mathcal{B}, \alpha, \rho \Vdash A_1$ and $\mathcal{B}, \alpha, \rho \Vdash A_2$
- $\mathcal{B}, \alpha, \rho \Vdash A_1 \vee A_2 \overset{\Delta}{\iff} \mathcal{B}, \alpha, \rho \Vdash A_1$ or $\mathcal{B}, \alpha, \rho \Vdash A_2$
- $\mathcal{B}, \alpha, \rho \Vdash A_1 \rightarrow A_2 \overset{\Delta}{\iff}$
  for all $f: \alpha \rightarrow \alpha'$ if $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A_1$, then $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A_2$
- $\mathcal{B}, \alpha, \rho \Vdash \forall x{:}\, \tau.A \overset{\Delta}{\iff}$
  $\mathcal{B}, \alpha', \rho_{\restriction f}\lfloor x := a \rfloor \Vdash A$ for all $f: \alpha \rightarrow \alpha'$ and $a \in \tau^{\mathcal{B}(\alpha')}$
- $\mathcal{B}, \alpha, \rho \Vdash \exists x{:}\, \tau.A \overset{\Delta}{\iff} \mathcal{B}, \alpha, \rho\lfloor x := a \rfloor \Vdash A$ for some $a \in \tau^{\mathcal{B}(\alpha)}$

The sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is **valid** *in $\mathcal{B}$ ($\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$) iff:*

- for all $\alpha \in \mathrm{K}$ and $\rho \in \eta^{\mathcal{B}(\alpha)}$ if $\mathcal{B}, \alpha, \rho \Vdash \Gamma_1$, then
  $\mathcal{B}, \alpha, \rho \Vdash A$ for some $A \in \Gamma_2$

The *functoriality* of Kripke Σ-structures is reflected also in the interpretation:

Lemma 2.3.4 Functoriality of the Interpretation w.r.t. α

Given a signature Σ, a type environment η for Σ, a Kripke Σ-structure $\mathcal{B}: \mathrm{K} \rightarrow \mathrm{Struct}(\Sigma)$, $f: \alpha \rightarrow \alpha'$ and an environment $\rho \in \eta^{\mathcal{B}(\alpha)}$:

- *If $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$ and $[\![t]\!]^{\mathcal{B},\alpha}_{\rho}$ is defined then $[\![t]\!]^{\mathcal{B},\alpha'}_{\rho_{\restriction f}} = ([\![t]\!]^{\mathcal{B},\alpha}_{\rho})_{\restriction f}$*
- *If $A \in \mathrm{Form}^{\Sigma}(\eta)$ and $\mathcal{B}, \alpha, \rho \Vdash A$ then $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A$*

Remark 2.3.5 Categorically the functoriality of $[\![t]\!]^{\mathcal{B},\alpha}_{\rho}$ means that the family $\langle (\lambda\rho \in \eta^{\mathcal{B}(\alpha)}.[\![t]\!]^{\mathcal{B},\alpha}_{\rho}) | \alpha \in \mathrm{K} \rangle$ is a *quasi-natural transformation* from $\eta^{\mathcal{B}}$ to $\tau^{\mathcal{B}}$ (see Definition 5.1.7). Similarly, the functoriality of $\mathcal{B}, \alpha, \rho \Vdash A$ means that the family $\langle \{\rho \in \eta^{\mathcal{B}(\alpha)} | \mathcal{B}, \alpha, \rho \Vdash A\} | \alpha \in \mathrm{K} \rangle$ is a *sub-functor* of $\eta^{\mathcal{B}}$.

From the definition of interpretation in a Kripke model and its functoriality it is easy to show that:

Proposition 2.3.6 *If A is a closed formula and $\bot$ is a* weakly initial *stage of knowledge (the least stage, if* K *is a partial order), then the sequent $\emptyset.\emptyset \Longrightarrow A$ is valid in $\mathcal{B}$, i.e. $\mathcal{B} \Vdash A$, iff the environment $\emptyset$ satisfies A at stage $\bot$, i.e. $\mathcal{B}, \bot, \emptyset \Vdash A$*

Also for Kripke Σ-structures, the validity of a sequent does not depend on the interpretation of the symbols of Σ that do not occur in the sequent:

Lemma 2.3.7 Satisfaction Lemma

Given $\Sigma \subseteq \Sigma'$, a Kripke Σ'-structure $\mathcal{B}\colon \mathrm{K} \to \mathrm{Struct}(\Sigma')$, a type environment η for Σ, a stage $\alpha \in \mathrm{K}$ and an η-environment $\rho \in \eta^{\mathcal{B}\restriction\Sigma(\alpha)} (= \eta^{\mathcal{B}(\alpha)})$:

- *if $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$, then $[\![t]\!]^{\mathcal{B}\restriction\Sigma,\alpha}_{\rho} \simeq [\![t]\!]^{\mathcal{B},\alpha}_{\rho}$*
- *if $A \in \mathrm{Form}^{\Sigma}(\eta)$, then $\mathcal{B}\restriction\Sigma, \alpha, \rho \Vdash A$ iff $\mathcal{B}, \alpha, \rho \Vdash A$*
- *if $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}$ then $\mathcal{B}\restriction\Sigma \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$ iff $\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$*

2.4 Formal systems

The *satisfaction relation* $\models$ between $\mathrm{Struct}(\Sigma)$ and Seq^{Σ} induces a **logical consequence** relation $\models^{\mathrm{K}}_{\Sigma}$ on Seq^{Σ} (see Definition 1.3.1). This logical consequence relation can be alternatively described by a formal system for deriving sequents from (non-logical) assumptions (see [MR77]). Sections 2.5 and 2.6 will be devoted to proving that the logical consequence relation and the formal system defined below are the same.

The formal system has the (cut) rule, which is not derivable from the other rules, although it is admissible in the theory generated by them (see the Hauptsatz of [Gen69]). We include the axioms for $\_\downarrow$ and $\_ = \_$, in order to capture the standard interpretation for these predicates. The inference rules are divided in three classes: general rules, rules that describe the *meaning* of terms (including the axioms for existence and equality) and rules that describe the *meaning* of formulae.

Definition 2.4.1 The Formal System *FreeK*LPT

$\vdash^{\mathrm{K}}_{\Sigma}$ *is the formal system on* Seq^{Σ} generated *by the following inference rules:*
General rules:

$$\text{log}\ \frac{}{\eta.\Gamma_1, A \Longrightarrow A, \Gamma_1'}$$

$$\text{thinning}\ \frac{\eta_1.\Gamma_1 \Longrightarrow \Gamma_1'}{\eta_2\lfloor\eta_1.\Gamma_1, \Gamma_2 \Longrightarrow \Gamma_1', \Gamma_2'}$$

$$\text{cut}\ \frac{\eta.\Gamma_1, A \Longrightarrow \Gamma_1' \qquad \eta.\Gamma_2 \Longrightarrow A, \Gamma_2'}{\eta.\Gamma_1, \Gamma_2 \Longrightarrow \Gamma_1', \Gamma_2'}$$

Rules for terms:

$$\text{subst}\ \frac{\eta.\Gamma_1 \Longrightarrow t\downarrow_\tau, \Gamma_1' \qquad \eta\lfloor x\colon\tau.\Gamma_2 \Longrightarrow \Gamma_2'}{\eta.\Gamma_1, \Gamma_2[x\colon= t] \Longrightarrow \Gamma_1', \Gamma_2'[x\colon= t]}$$

Axioms for existence*:*

$$\text{E.}x\ \ x\downarrow_\tau$$

$$\text{E.}c\ \ c\downarrow_\tau$$

$$\text{E.}f.\text{i}\ \ f(\overline{t})\downarrow_\tau \Longrightarrow \overline{t}_i\downarrow_{\overline{\tau}_i} \qquad i \in \mathrm{dom}(\overline{t})$$

$$\text{E.}p.\text{i}\ \ p(\overline{t}) \Longrightarrow \overline{t}_i\downarrow_{\overline{\tau}_i} \qquad i \in \mathrm{dom}(\overline{t})$$

Axioms for equality*:*

$$\text{refl}\ \ x =_\tau x$$

$$\text{symm}\ \ x_1 =_\tau x_2 \Longrightarrow x_2 =_\tau x_1$$

$$\text{trans}\ \ x_1 =_\tau x_2, x_2 =_\tau x_3 \Longrightarrow x_1 =_\tau x_3$$

$$\text{cong.}f\ \ \overline{x} =_{\overline{\tau}} \overline{x}', f(\overline{x})\downarrow_\tau \Longrightarrow f(\overline{x}) =_\tau f(\overline{x}')$$

$$\text{cong.}p\ \ \overline{x} =_{\overline{\tau}} \overline{x}', p(\overline{x}) \Longrightarrow p(\overline{x}')$$

Rules for formulae:

$$\bot \Longrightarrow\ \ \eta.\Gamma_1, \bot \Longrightarrow \Gamma_2$$

$$\wedge \Longrightarrow \quad \frac{\eta.\Gamma_1, A_i \Longrightarrow \Gamma_2}{\eta.\Gamma_1, A_1 \wedge A_2 \Longrightarrow \Gamma_2}$$

$$\Longrightarrow \wedge \quad \frac{\eta.\Gamma_1 \Longrightarrow A_1, \Gamma_2 \qquad \eta.\Gamma_1 \Longrightarrow A_2, \Gamma_2}{\eta.\Gamma_1 \Longrightarrow A_1 \wedge A_2, \Gamma_2}$$

$$\vee \Longrightarrow \quad \frac{\eta.\Gamma_1, A_1 \Longrightarrow \Gamma_2 \qquad \eta.\Gamma_1, A_2 \Longrightarrow \Gamma_2}{\eta.\Gamma_1, A_1 \vee A_2 \Longrightarrow \Gamma_2}$$

$$\Longrightarrow \vee \quad \frac{\eta.\Gamma_1 \Longrightarrow A_i, \Gamma_2}{\eta.\Gamma_1 \Longrightarrow A_1 \vee A_2, \Gamma_2}$$

$$\rightarrow\Longrightarrow \quad \frac{\eta.\Gamma_1 \Longrightarrow A_1, \Gamma_2 \qquad \eta.\Gamma'_1, A_2 \Longrightarrow \Gamma'_2}{\eta.\Gamma_1, \Gamma'_1, A_1 \rightarrow A_2 \Longrightarrow \Gamma_2, \Gamma'_2}$$

$$\Longrightarrow\rightarrow_K \quad \frac{\eta.\Gamma_1, A_1 \Longrightarrow A_2, \Gamma_2}{\eta.\Gamma_1 \Longrightarrow A_1 \rightarrow A_2, \Gamma_2}$$

$$\exists \Longrightarrow \quad \frac{\eta\lfloor x\colon \tau.\Gamma_1, A \Longrightarrow \Gamma_2}{\eta.\Gamma_1, \exists x\colon \tau.A \Longrightarrow \Gamma_2}$$

$$\Longrightarrow \exists \quad \frac{\eta.\Gamma_1 \Longrightarrow t\downarrow_\tau, \Gamma_2 \qquad \eta.\Gamma_1 \Longrightarrow A[x\colon= t], \Gamma_2}{\eta.\Gamma_1 \Longrightarrow \exists x\colon \tau.A, \Gamma_2}$$

$$\forall \Longrightarrow \quad \frac{\eta.\Gamma_1 \Longrightarrow t\downarrow_\tau, \Gamma_2 \qquad \eta.\Gamma'_1, A[x\colon= t] \Longrightarrow \Gamma'_2}{\eta.\Gamma_1, \Gamma'_1, \forall x\colon \tau.A \Longrightarrow \Gamma_2, \Gamma'_2}$$

$$\Longrightarrow \forall_K \quad \frac{\eta\lfloor x\colon \tau.\Gamma_1 \Longrightarrow A, \Gamma_2}{\eta.\Gamma_1 \Longrightarrow \forall x\colon \tau.A, \Gamma_2}$$

Remark 2.4.2 In first order logic it is not necessary to use sequents to describe the formal consequence relation. In fact, the sequent $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)$ is *equivalent* to the closed formula $(\forall\eta. \wedge(\Gamma_1) \rightarrow \vee(\Gamma_2))$.

The inference rules of the formal system $\vdash^{\mathrm{J}}_{\Sigma}$ for intuitionistic logic ($FreeJ$LPT) are only a minor modification of those for $\vdash^{\mathrm{K}}_{\Sigma}$. In fact, all the inference rules for classical logic are valid in Kripke Σ-structures, except $(\Longrightarrow\rightarrow_K)$ and $(\Longrightarrow \forall_K)$, which are valid only when Γ_2 is empty.

Definition 2.4.3 The Formal System $Free J$LPT

$\vdash^{\mathrm{J}}_{\Sigma}$ *is the formal system on* Seq^{Σ} generated *by the inference rules for* $\vdash^{\mathrm{K}}_{\Sigma}$*, with the rules* $(\Longrightarrow\!\rightarrow_K)$ *and* $(\Longrightarrow \forall_K)$ *replaced by:*

$$\Longrightarrow\!\rightarrow_I \ \frac{\eta.\Gamma_1, A_1 \Longrightarrow A_2}{\eta.\Gamma_1 \Longrightarrow A_1 \rightarrow A_2}$$

$$\Longrightarrow \forall_I \ \frac{\eta\lfloor x\colon \tau.\Gamma_1 \Longrightarrow A}{\eta.\Gamma_1 \Longrightarrow \forall x\colon \tau.A}$$

Alternatively to get the intuitionistic formal system on the JSeq^{Σ} fragment, one can restrict all the inference rules for classical logic to JSeq^{Σ} (see [Gen69] or [Kle52]), in fact:

Proposition 2.4.4 *If* $T \subseteq \mathrm{JSeq}^{\Sigma}$ *and* $T \vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$*, then there exists a formula* A *in* $\Gamma_2 \cup \{\bot\}$ *s.t.* $\eta.\Gamma_1 \Longrightarrow A$ *is derivable from* T *by using only inference rules restricted to* JSeq^{Σ}*.*

Proof By induction on the derivation of $\eta.\Gamma_1 \Longrightarrow \Gamma_2$. ∎

The formal system for classical logic can be axiomatized in that for intuitionistic logic by adding either the axiom (schema) of the *excluded middle* or that of *double-negation*. Also the formal system for the *usual* many-sorted structures, where all types are inhabited and all functions are total, can be easily axiomatized.

Definition 2.4.5 Variants of LPT

The logic of partial terms has eight variants:

$$\begin{array}{ccccc} Free & J & & & 0 \\ \cap & \cap & \mathrm{LPT} & + & \cap \\ Inhabited & K & & & \mathrm{tot} \end{array}$$

each of them is axiomatized in the free intuitionistic logic of partial terms ($Free J$LPT *over* Σ*) by a suitable combination of the following axioms:*

- *axioms for* **classical** *models, either*

  excl $A \vee \neg A$

or

$$\neg\neg\ \neg\neg A \Longrightarrow A$$

- *axioms for models with all sorts* **inhabited**

  inhab $\exists x\colon \tau.\top$

- *axioms for models with all functions* **total**

  tot $f(\overline{x})\downarrow_\tau$

2.5 Soundness

Soundness of the formal system $\vdash^{\mathrm{K}}_{\Sigma}$ w.r.t. the logical consequence relation $\models^{\mathrm{K}}_{\Sigma}$ amounts to showing that the inference rules for $\vdash^{\mathrm{K}}_{\Sigma}$ are valid in any Σ-structure. This is easily proved by using the following lemmas:

Lemma 2.5.1 Environment Extension preserves Meaning

Given a signature Σ, a type environment η for Σ, a Σ-structure $\mathcal{A}$, an environment $\rho \in \eta^{\mathcal{A}}$ and $a \in \tau^{\mathcal{A}}$:

- *If $t \in \mathrm{Term}^{\Sigma}_{\tau'}(\eta)$ then $[\![t]\!]^{\mathcal{A}}_{x:=a\lfloor\rho} \simeq [\![t]\!]^{\mathcal{A}}_{\rho}$*
- *If $A \in \mathrm{Form}^{\Sigma}(\eta)$ then $\mathcal{A}, x\colon= a\lfloor\rho \models A \Longleftrightarrow \mathcal{A},\rho \models A$*

Lemma 2.5.2 Substitution Lemma

Given a signature Σ, a type environment η for Σ, a Σ-structure $\mathcal{A}$, an environment $\rho \in \eta^{\mathcal{A}}$ and $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$ s.t. $[\![t]\!]^{\mathcal{A}}_{\rho} = a \in \tau^{\mathcal{A}}$:

- *If $t' \in \mathrm{Term}^{\Sigma}_{\tau'}(\eta\lfloor x\colon\tau)$ then $[\![t'[x\colon= t]]\!]^{\mathcal{A}}_{\rho} \simeq [\![t']\!]^{\mathcal{A}}_{\rho\lfloor x:=a}$*
- *If $A \in \mathrm{Form}^{\Sigma}(\eta\lfloor x\colon\tau)$ then $\mathcal{A},\rho \models A[x\colon= t] \Longleftrightarrow \mathcal{A},\rho\lfloor x\colon= a \models A$*

For intuitionistic logic, soundness of $\vdash^{\mathrm{J}}_{\Sigma}$ w.r.t. $\models^{\mathrm{J}}_{\Sigma}$ is proved in a similar way, but to show the validity of the rules ($\Longrightarrow\rightarrow_I$) and ($\Longrightarrow\forall_I$) we have to use also functoriality of the interpretation in Kripke Σ-structures (see Lemma 2.3.4).

Theorem 2.5.3 Soundness

$\vdash_{\Sigma}$ *is sound w.r.t.* $\models_{\Sigma}$

2.6 Completeness

The proof of completeness follows Henkin (see [Hen49]), but some extra care is required, because the addition of new constants forces some sorts to be inhabited.

Completeness of $\vdash^{K}$ w.r.t. $\models^{K}$ amounts to proving that: for all signatures Σ, sets $T \subseteq \mathrm{Seq}^{\Sigma}$ and $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}$ if $T \not\vdash^{K}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then there is a Σ-structure $\mathcal{A}$ which is a model of T but does not satisfy $\eta.\Gamma_1 \Longrightarrow \Gamma_2$.

- By the *Deduction Lemma* (see Lemma 2.6.2), this is equivalent to: for all signatures Σ, sets of sequents $T \subseteq \mathrm{Seq}^{\Sigma}$ and a finite set of closed formulae $\Gamma \subseteq_{fin} \mathrm{Form}_0^{\Sigma}$, if $T \not\vdash^{K}_{\Sigma} \emptyset.\Gamma$, then there is a Σ-structure $\mathcal{A}$ which is a model of T but does not satisfy any of the formulae in Γ

- Then we introduce the auxiliary notion of classically *T-complete* set and prove the *Separation Lemma* (see Lemma 2.6.4): if $T \not\vdash^{K}_{\Sigma} \emptyset.\Gamma$, then there is a classically T-complete set (on an extension of Σ), which does not contain any of the formulae in Γ

- Finally, the *Characterization* of the term model $\mathcal{A}_{(\Sigma,H)}$ corresponding to a classically T-complete set H on Σ (see Proposition 2.6.8) provides the last link for the proof of the *Completeness Theorem* (see Theorem 2.6.9)

For intuitionistic logic the definition of intuitionistically T-complete set and the proofs of the Deduction and Separation Lemmas are exactly the same as the ones for classical logic, provided $\vdash^{K}$ is replaced by $\vdash^{J}$. However, intuitionistically T-complete sets *correspond* to stages of knowledge in the Kripke term model $\mathcal{B}_{(\Sigma,T)}$, rather than to Kripke models of T. The *Characterization* of $\mathcal{B}_{(\Sigma,T)}$, unlike that of $\mathcal{A}_{(\Sigma,H)}$, makes use of the Separation Lemma (see Proposition 2.6.11). Moreover, the *Completeness Theorem* (see Theorem 2.6.12) becomes a rather simple consequence of the *Characterization* of $\mathcal{B}_{(\Sigma,T)}$.

In the separation lemma and for the term model constructions we have to extend a signature with extra constant symbols and use the following conventions:

Notation 2.6.1 A **set** C **of new constant symbols for** Σ is a set disjoint from (the sets in) Σ together with a function from C to $\mathrm{Sort}(\Sigma)$ assigning to each element of C its sort. Similarly, a **new constant symbol** c **of sort** τ **for** Σ is a symbol not in Σ.

The set C (symbol c) can be viewed as a signature s.t. $\mathrm{Sort}(C)$ is $\mathrm{Sort}(\Sigma)$ and $\mathrm{Const}_\tau(C)$ is the set of symbols in C whose sort is τ (the other sets are empty). Therefore, in the sequel we write $\Sigma \uplus C$ ($\Sigma \uplus c$) to denote the union of Σ and the signature corresponding to C (c).

Lemma 2.6.2 Deduction Lemma

Given a signature Σ *and a set* $T \subseteq \mathrm{Seq}^\Sigma$*:*

- *If* $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^\Sigma$ *and* $A \in \mathrm{Form}_0^\Sigma$*, then*

  $T \vdash_\Sigma \eta.A, \Gamma_1 \Longrightarrow \Gamma_2$ *iff* $T, A \vdash_\Sigma \eta.\Gamma_1 \Longrightarrow \Gamma_2$

- *If* $x \notin \mathrm{dom}(\eta)$*,* $x\colon \tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^\Sigma$ *and* c *is a new constant of sort* τ *for* Σ*, then*

  $T \vdash_\Sigma x\colon \tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2$ *iff* $T \vdash_{\Sigma\uplus c} \eta.\Gamma_1[x := c] \Longrightarrow \Gamma_2[x := c]$

Proof In both cases the two implications are proved separately. The implication from left to right is the easy direction, and it follows from:

- $(\eta.A, \Gamma_1 \Longrightarrow \Gamma_2), A \vdash_\Sigma \eta.\Gamma_1 \Longrightarrow \Gamma_2$
- $(x\colon \tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2) \vdash_{\Sigma\uplus c} \eta.\Gamma_1[x := c] \Longrightarrow \Gamma_2[x := c]$

The implication from right to left is by induction on the derivation of the r.h.s., and the only cases that require some care are the basic steps of the induction:

- - if $T, A \vdash_\Sigma \eta.\Gamma_1 \Longrightarrow \Gamma_2$ because
    $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is in T, then
    $T \vdash_\Sigma \eta.A, \Gamma_1 \Longrightarrow \Gamma_2$ by (thinning)
  - if $T, A \vdash_\Sigma \eta.\Gamma_1 \Longrightarrow \Gamma_2$ because
    $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is $\emptyset.A$, then
    $T \vdash_\Sigma \emptyset.A \Longrightarrow A$ by (log)
- - if $T \vdash_{\Sigma\uplus c} \eta.\Gamma_1[x := c] \Longrightarrow \Gamma_2[x := c]$ because
    $\eta.\Gamma_1[x := c] \Longrightarrow \Gamma_2[x := c]$ is in T, then
    x does not occur free in Γ_1 or Γ_2, so
    $T \vdash_\Sigma x\colon \tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2$ by (thinning)

– if $T \vdash_{\Sigma \uplus c} \eta.\Gamma_1[x:=c] \Longrightarrow \Gamma_2[x:=c]$ by (E.c), then
$\eta.\Gamma_1[x:=c] \Longrightarrow \Gamma_2[x:=c]$ is $\emptyset.c \downarrow_\tau$, so
$T \vdash_\Sigma x{:}\,\tau.x \downarrow_\tau$ by (E.x)

∎

Definition 2.6.3 Complete Set

*Given a signature Σ and a set $T \subseteq \mathrm{Seq}^\Sigma$, a set $H \subseteq \mathrm{Form}_0^\Sigma$ is T-**complete** on Σ iff:*

1. $\perp \notin H$
2. *For all $A \in \mathrm{Form}_0^\Sigma$ if $T, H \vdash_\Sigma A$, then $A \in H$*
3. *If $A_1 \vee A_2 \in H$, then $A_1 \in H$ or $A_2 \in H$*
4. *If $\exists x{:}\,\tau.A \in H$, then there exists $t \in \mathrm{Term}_\tau^\Sigma(\emptyset)$ s.t.*
   $t \downarrow \in H$ and $A[x:=t] \in H$

It is easy to show that a classically T-complete set is maximal consistent, i.e. for all $A \in \mathrm{Form}_0^\Sigma$ if $H, A \not\vdash_\Sigma^{\mathrm{K}} \perp$, then $A \in H$ (see [Hen49]). In fact, if $H, A \not\vdash_\Sigma^{\mathrm{K}} \perp$, then $\neg A \notin H$, but $\neg A \vee A$ is in H, because it is a tautology in classical logic (but not in intuitionistic logic), therefore $A \in H$, by the third property for complete sets. However, there are maximal consistent sets that are not classically complete, because they may not satisfy the fourth property for complete sets.

Lemma 2.6.4 Separation Lemma

Given a signature Σ, a set C of new constant symbols for Σ, with $\aleph_0$ symbols for each sort, and a set $T \subseteq \mathrm{Seq}^\Sigma$, then for all $\Gamma \subseteq_{fin} \mathrm{Form}_0^\Sigma$ s.t. $T \not\vdash_\Sigma^{\mathrm{K}} \emptyset.\Gamma$ there is a signature Σ_ω, with $\Sigma \subseteq \Sigma_\omega \subseteq \Sigma \uplus C$, and a T-complete set H on Σ_ω s.t. $A \notin H$ for all A in Γ

Proof Let A_n be an enumeration of $\mathrm{Form}_0^{\Sigma \uplus C}$, we define, by induction on n, an increasing sequence (Σ_n, H_n) (where $\Sigma_n \subseteq \Sigma \uplus C$ is a finite extension of Σ and $H_n \subset \mathrm{Form}_0^{\Sigma_n}$):

- basic step
  $H_0 \equiv \emptyset$
  $\Sigma_0 \equiv \Sigma$

- inductive step

  Let C_n be the set of new constant symbols for Σ_n occurring in A_n; there are two cases to be considered:

  - if $T, H_n, A_n \not\vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$ then there are two subcases:

    * If $A_n \equiv \exists x\colon \tau.A$ then

      $H_{n+1} \equiv H_n \cup \{A_n, A[x\colon = c]\}$

      $\Sigma_{n+1} \equiv \Sigma_n \uplus C_n \uplus c$

      where $c \in C$ is a new constant symbol for $\Sigma_n \uplus C_n$

    * else

      $H_{n+1} \equiv H_n \cup \{A_n\}$

      $\Sigma_{n+1} \equiv \Sigma_n \uplus C_n$

  - else

    $H_{n+1} \equiv H_n$

    $\Sigma_{n+1} \equiv \Sigma_n$

The rest of the proof will establish that $\Sigma_\omega \triangleq \cup_{n\in\omega}\Sigma_n$ and $H \triangleq \cup_{n\in\omega}H_n$ satisfy the statement of the lemma. For each of the four properties in the definition of T-complete set we prove a claim that essentially amounts to showing that H has that property.

Claim 2.6.4.1 $T, H_n \not\vdash_{\Sigma_n} \emptyset.\Gamma$

Proof By induction on n. The non trivial cases (in the inductive step) are:

- the first subcase of the first case. Let $A_n \equiv \exists x\colon \tau.A$, we have to show that if $T, H_n, \exists x\colon \tau.A \not\vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$, then

  $T, H_n, \exists x\colon \tau.A, A[x\colon = c] \not\vdash_{\Sigma_n \uplus C_n \uplus c} \emptyset.\Gamma$.

  - Assume that

    $T, H_n, \exists x\colon \tau.A, A[x\colon = c] \vdash_{\Sigma_n \uplus C_n \uplus c} \emptyset.\Gamma$

  - By the Deduction Lemma

    $T, H_n \vdash_{\Sigma_n \uplus C_n} x\colon \tau.\exists x\colon \tau.A, A \Longrightarrow \Gamma$

  - By the inference rule ($\exists \Longrightarrow$)

    $T, H_n \vdash_{\Sigma_n \uplus C_n} \exists x\colon \tau.A \Longrightarrow \Gamma$

- By the Deduction Lemma

  $T, H_n, \exists x{:}\,\tau.A \vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$

  contradiction

- the second case follows immediately from the IH (actually it is the only case where the IH is used)

∎

From the previous claim it follows easily that $\bot \notin H$ and $A \notin H$ for all A in Γ.

Claim 2.6.4.2 *If* $T, H \vdash_{\Sigma_\omega} A_n$*, then* $T, H_n, A_n \not\vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$

Proof

- By compactness of $\vdash$ there exists $m \geq n$ s.t.

  $T, H_m \vdash_{\Sigma_m} A_n$ (and therefore $\Sigma_n \uplus C_n \subseteq \Sigma_m$)

- assume that

  $T, H_n, A_n \vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$

- By the Deduction Lemma

  $T, H_n \vdash_{\Sigma_n \uplus C_n} A_n \Longrightarrow \Gamma$

- By compactness of $\vdash$, $\Sigma_n \uplus C_n \subseteq \Sigma_m$ and $H_n \subseteq H_m$

  $T, H_m \vdash_{\Sigma_m} A_n \Longrightarrow \Gamma$

- By the inference rule (cut)

  $T, H_m \vdash_{\Sigma_m} \emptyset.\Gamma$

  which contradicts the previous claim

∎

From the previous claim it follows easily that for all $A \in \mathrm{Form}_0^{\Sigma_\omega}$ if $T, H \vdash_{\Sigma_\omega} A$, then $A \in H$.

Claim 2.6.4.3 *Given m s.t. $A'_1 \vee A'_2$, A'_1 and A'_2 have at least one index $< m$ in the enumeration of* $\mathrm{Form}_0^{\Sigma \uplus C}$*, if $A'_1 \vee A'_2 \in H_m$, then $A'_1 \in H_m$ or $A'_2 \in H_m$*

Proof It is not possible that $T, H_m \vdash_{\Sigma_m} A'_1 \Longrightarrow \Gamma$ and $T, H_m \vdash_{\Sigma_m} A'_2 \Longrightarrow \Gamma$; otherwise by the inference rule ($\vee \Longrightarrow$), the Deduction Lemma and the hypothesis $A'_1 \vee A'_2 \in H_m$ it follows that $T, H_m \vdash_{\Sigma_m} \emptyset.\Gamma$, which contradicts the first claim. So taking $n < m$ s.t. A_n is A'_1 or A'_2 and $T, H_m \not\vdash_{\Sigma_m} A_n \Longrightarrow \Gamma$, we prove that: $T, H_n, A_n \not\vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$.

- Assume that

  $T, H_n, A_n \vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$

- By the Deduction Lemma

  $T, H_n \vdash_{\Sigma_n \uplus C_n} A_n \Longrightarrow \Gamma$

- By compactness of $\vdash$, $\Sigma_n \uplus C_n \subseteq \Sigma_m$ and $H_n \subseteq H_m$

  $T, H_m \vdash_{\Sigma_m} A_n \Longrightarrow \Gamma$

  contradiction

By construction and $n < m$ it follows $A_n \in H_m$ ∎

From the previous claim it follows that if $A'_1 \vee A'_2 \in H$, then $A'_1 \in H$ or $A'_2 \in H$.

Claim 2.6.4.4 *If* $\exists x{:}\,\tau.A \equiv A_n \in H$, *then there exists* $c \in \mathrm{Const}_\tau(\Sigma_\omega)$ *s.t.* $A[x{:}= c] \in H$

Proof

- By the second claim

  $T, H_n, A_n \not\vdash_{\Sigma_n \uplus C_n} \emptyset.\Gamma$

- By construction

  $A[x{:}= c] \in H$ and $c \in \Sigma_\omega$

- By the inference rule (E.c)

  $T, H \vdash_{\Sigma_\omega} c \downarrow_\tau$

- By a consequence of the second claim

  $c \downarrow_\tau \in H$

∎

From the second and fourth claim it follows that if $\exists x{:}\,\tau.A \in H$, then there exists $t \in \mathrm{Term}^{\Sigma_\omega}_\tau(\emptyset)$ s.t. $t \downarrow \in H$ and $A[x{:}=t] \in H$ ∎

The crucial property of a classically T-complete set H on Σ is that it is the set of closed formulae valid in a certain model $\mathcal{A}_{(\Sigma,H)}$ of T. However, the definition of $\mathcal{A}_{(\Sigma,H)}$ itself relies only on few *closure* properties of the atomic formulae in H.

Definition 2.6.5 Acceptable set

*A subset H of Form^Σ_0 is Σ-**acceptable** iff*

1. *if $c \in \mathrm{Const}_\tau(\Sigma)$, then $c \downarrow_\tau \in H$*
2. *if $f(\overline{t}) \downarrow_\tau \in H$, then $\overline{t}_i \downarrow_{\overline{\tau}_i} \in H$ for any $i \in \mathrm{dom}(\overline{t})$*
3. *if $p(\overline{t}) \in H$, then $\overline{t}_i \downarrow_{\overline{\tau}_i} \in H$ for any $i \in \mathrm{dom}(\overline{t})$*
4. *if $t \downarrow_\tau \in H$, then $t =_\tau t \in H$*
5. *if $t_1 =_\tau t_2 \in H$, then $t_2 =_\tau t_1 \in H$*
6. *if $t_1 =_\tau t_2$ and $t_2 =_\tau t_3 \in H$, then $t_1 =_\tau t_3 \in H$*
7. *if $\overline{t} =_{\overline{\tau}} \overline{t}' \subseteq_{fin} H$ and $f(\overline{t}) \downarrow_\tau \in H$, then $f(\overline{t}) =_\tau f(\overline{t}') \in H$*
8. *if $\overline{t} =_{\overline{\tau}} \overline{t}' \subseteq_{fin} H$ and $p(\overline{t}) \in H$, then $p(\overline{t}') \in H$*

Remark 2.6.6 The properties of an acceptable set ensure that functions and predicates are *strict* and $\_ = \_$ is a *congruence relation*. Both classically and intuitionistically complete sets are acceptable. In fact, the closure properties above are a consequence of the second property for complete sets, more precisely they can be *derived* from the general rules and the rules that describe the meaning of terms (see Definition 2.4.1).

Definition 2.6.7 The Term Model $\mathcal{A}_{(\Sigma,H)}$

Given a Σ-acceptable set H the Σ-structure $\mathcal{A}_{(\Sigma,H)}$ is defined by:

1. $\tau^{\mathcal{A}_{(\Sigma,H)}} \triangleq \{[t]_H | t \downarrow_\tau \in H\}$, *where* $[t]_H \triangleq \{t' | t = t' \in H\}$
2. $c^{\mathcal{A}_{(\Sigma,H)}} \triangleq [c]_H$
3. $f^{\mathcal{A}_{(\Sigma,H)}}([\overline{t}]_H) \overset{\Delta}{\simeq} \begin{cases} [f(\overline{t})]_H & \textit{if } f(\overline{t}) \downarrow \in H \\ undefined & \textit{otherwise} \end{cases}$

4. $[\overline{t}]_H \in p^{\mathcal{A}_{(\Sigma,H)}} \stackrel{\Delta}{\Longleftrightarrow} p(\overline{t}) \in H$

Proposition 2.6.8 Characterization of $\mathcal{A}_{(\Sigma,H)}$

Let H be a classically T-complete set on Σ and $\mathcal{A} \equiv \mathcal{A}_{(\Sigma,H)}$. Then, for all type environments η, environments $\rho \in \eta^{\mathcal{A}}$ and substitutions $\sigma \in \prod \rho$:

- *If $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$, then $[\![t]\!]^{\mathcal{A}}_{\rho} \simeq \begin{cases} [t[\sigma]]_H & \text{if } t[\sigma] \downarrow\in H \\ \text{undefined} & \text{otherwise} \end{cases}$*
- *If $A \in \mathrm{Form}^{\Sigma}(\eta)$, then $\mathcal{A}, \rho \models A \Longleftrightarrow A[\sigma] \in H$*

Moreover, $\mathcal{A} \models T$

Proof The first statement is proved by induction on t. The basic cases x and c are trivial (in fact, $\sigma \downarrow_{\eta} \subseteq_{fin} H$ as $\sigma \in \prod \rho$), so we consider only the case $f(\overline{t})$. First we prove that if $f(\overline{t})[\sigma] \downarrow\in H$, then $[\![f(\overline{t})]\!]^{\mathcal{A}}_{\rho} \simeq [f(\overline{t})[\sigma]]_H$

- by the second property of complete sets

  $\overline{t}_i[\sigma] \downarrow\in H$ for any $i \in |\overline{t}|$
- by IH applied to $\overline{t}_i$

  $[\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho} = [\overline{t}_i[\sigma]]_H$
- by definition of $[\![\_]\!]^{\mathcal{A}}_{\rho}$ and $f^{\mathcal{A}}$

  $[\![f(\overline{t})]\!]^{\mathcal{A}}_{\rho} \simeq f^{\mathcal{A}}(\langle [\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho} | i \in |\overline{t}| \rangle) = [f(\overline{t}[\sigma])]_H$

Then we prove that if $[\![f(\overline{t})]\!]^{\mathcal{A}}_{\rho}$ is defined, then $[\![f(\overline{t})]\!]^{\mathcal{A}}_{\rho} = [f(\overline{t})[\sigma]]_H$

- by definition of $[\![\_]\!]^{\mathcal{A}}_{\rho}$

  $[\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho}$ are defined for any $i \in |\overline{t}|$ and $f^{\mathcal{A}}(\langle [\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho} | i \in |\overline{t}| \rangle)$ is defined
- by IH applied to $\overline{t}_i$

  $[\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho} = [\overline{t}_i[\sigma]]_H$
- by definition of $f^{\mathcal{A}}$

  $(f(\overline{t}[\sigma]) \downarrow\in H$ and$)$ $f^{\mathcal{A}}(\langle [\![\overline{t}_i]\!]^{\mathcal{A}}_{\rho} | i \in |\overline{t}| \rangle) = [f(\overline{t}[\sigma])]_H$

The second statement is proved by induction on A. The basic case $p(\overline{t})$ is similar to the case $f(\overline{t})$ for terms. The second property of complete sets is used in all cases. The third and fourth properties of complete sets are used only in the proof of $(A_1 \vee A_2)[\sigma] \in H$ implies $\mathcal{A}, \rho \models A_1 \vee A_2$ and $(\exists x\colon \tau.A)[\sigma] \in H$ implies $\mathcal{A}, \rho \models \exists x\colon \tau.A$, respectively.
Since in classical logic $A_1 \rightarrow A_2$ is formally equivalent to $\neg A_1 \vee A_2$ and $\forall x\colon \tau.A$ is formally equivalent to $\neg\exists x\colon \tau.\neg A$, it is enough to show that $\mathcal{A}, \rho \models \neg A$ iff $\neg A[\sigma] \in H$ (this is the only place where the first property of complete sets is used):

- assume that $\mathcal{A}, \rho \models \neg A$, then

  $\mathcal{A}, \rho \models A$ is not true, by definition of $\mathcal{A}, \rho \models \_$
- By IH

  $A[\sigma] \notin H$
- By the second property of complete sets

  $A[\sigma] \vee \neg A[\sigma] \in H$
- By the third property of complete sets

  $\neg A[\sigma] \in H$

conversely

- assume that $\neg A[\sigma] \in H$, then

  $A \notin H$, by the first and second property of complete sets
- By IH

  $\mathcal{A}, \rho \models A$ is not true
- By definition of $\mathcal{A}, \rho \models \_$

  $\mathcal{A}, \rho \models \neg A$

$\mathcal{A} \models T$ follows easily from the second statement and the definition of $\mathcal{A}$ ∎

Theorem 2.6.9 Completeness
$\vdash^{\mathrm{K}}_{\Sigma}$ *is complete w.r.t.* $\models^{\mathrm{K}}_{\Sigma}$

Proof We prove that for all sets $T \subseteq \mathrm{Seq}^{\Sigma}$ and $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}$

- if $T \not\vdash^{\mathrm{K}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then there is a model of T which does not satisfy $\eta.\Gamma_1 \Longrightarrow \Gamma_2$

Take a set C of new constant symbols (of the appropriate sorts), one for each of the variables in $\mathrm{dom}(\eta)$, and a substitution $\gamma \in \mathrm{Term}^{\Sigma \uplus C}_{\eta}(\emptyset)$ that establish a bijection between $\mathrm{dom}(\eta)$ and the set C of new constant symbols for Σ.

- Assume that

  $T \not\vdash^{\mathrm{K}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$

- by the Deduction Lemma

  $T, \Gamma_1[\gamma] \not\vdash^{\mathrm{K}}_{\Sigma \uplus C} \emptyset.\Gamma_2[\gamma]$

- by the Separation Lemma

  there is an extension Σ' of $\Sigma \uplus C$ and a $T \cup \{\Gamma_1[\gamma]\}$-complete set H' on Σ' s.t. $A[\gamma] \notin H'$ for all A in Γ_2

- By the Characterization Lemma for $\mathcal{A}_{(\Sigma,H)}$

  $\mathcal{A}_{(\Sigma',H')} \models T$ and $\mathcal{A}_{(\Sigma',H')}, \emptyset \models \Gamma_1[\gamma]$, but

  not $\mathcal{A}_{(\Sigma',H')}, \emptyset \models A[\gamma]$ for any $A \in \Gamma_2$

  therefore $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is not valid in $\mathcal{A}_{(\Sigma',H')}$

- by the Satisfaction Lemma

  $\mathcal{A}_{(\Sigma',H')}\restriction\Sigma \models T$, but $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is not valid in $\mathcal{A}_{(\Sigma',H')}\restriction\Sigma$

∎

In order to prove completeness for intuitionistic logic, we define for any set T of intuitionistic sequents over Σ a Kripke model $\mathcal{B}_{(\Sigma,T)}$, whose theory is exactly the theory axiomatized by T in $\vdash^{\mathrm{J}}_{\Sigma}$. The stages of $\mathcal{B}_{(\Sigma,T)}$ *correspond* to intuitionistically T-complete sets on an extension of Σ:

Definition 2.6.10 The Kripke Term Model $\mathcal{B}_{(\Sigma,T)}$

Given a set T of sequents over Σ and a set $C = \{c_i | i \in \omega\times\omega\}$ of new constant symbols for Σ s.t. for all $n \in \omega$ the set $C_{n+1} - C_n$ has $\aleph_0$ symbols of each sort, where C_n is $\{c_i | i \in n \times \omega\}$; the Kripke Σ-structure $\mathcal{B}_{(\Sigma,T)}$ over $\mathrm{K}_{(\Sigma,T)}$ is defined by:

- $(\Sigma', H') \in \mathrm{K}_{(\Sigma,T)} \overset{\Delta}{\iff}$
  - $\Sigma \subseteq \Sigma'$ *and* $\Sigma' \subseteq \Sigma \uplus C_n$ *for some* $n \in \omega$
  - H' *is intuitionistically* T*-complete on* Σ'

  and $(\Sigma_1, H_1) \leq (\Sigma_2, H_2) \overset{\Delta}{\iff} \Sigma_1 \subseteq \Sigma_2$ *and* $H_1 \subseteq H_2$.
- $\mathcal{B}_{(\Sigma,T)}(\Sigma', H') \triangleq \mathcal{A}_{(\Sigma',H')}$
- $\mathcal{B}_{(\Sigma,T)}(f)_\tau([t]_{H_1}) \triangleq [t]_{H_2}$, *for all* $t \downarrow_\tau \in H_1$

Since an intuitionistically complete set on Σ' is Σ'-acceptable (see remarks after Definition 2.6.5), $\mathcal{B}_{(\Sigma,T)}$ is well-defined.

The characterization of $\mathcal{B}_{(\Sigma,T)}$ (unlike the one for $\mathcal{A}_{(\Sigma,H)}$) makes use of the Separation Lemma:

Proposition 2.6.11 Characterization of $\mathcal{B}_{(\Sigma,T)}$

Let T *be a set of sequents on* Σ, $\mathrm{K} \equiv \mathrm{K}_{(\Sigma,T)}$ *and* $\mathcal{B} \equiv \mathcal{B}_{(\Sigma,T)}$. *Then, for all type environments* η, $\alpha \equiv (\Sigma', H') \in \mathrm{K}$, *environments* $\rho \in \eta^{\mathcal{B}(\alpha)}$ *and substitutions* $\sigma \in \prod \rho$*:*

- *If* $t \in \mathrm{Term}^\Sigma_\tau(\eta)$, *then* $[\![t]\!]^{\mathcal{B},\alpha}_\rho \simeq \begin{cases} [t[\sigma]]_{H'} & \text{if } t[\sigma] \downarrow \in H' \\ \text{undefined} & \text{otherwise} \end{cases}$
- *If* $A \in \mathrm{Form}^\Sigma(\eta)$ *then* $\mathcal{B}, \alpha, \rho \Vdash A \iff A[\sigma] \in H'$

Moreover, $\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$ *iff* $T \vdash^{\mathrm{J}}_\Sigma \eta.\Gamma_1 \Longrightarrow \Gamma_2$

Proof The first statement is proved as in Proposition 2.6.8.

The second statement is proved by induction on A. The only cases that are treated differently from Proposition 2.6.8 are $\to$ and $\forall$. Since they are quite similar, we consider only $\to$. It is easy to prove that if $(A_1 \to A_2)[\sigma] \in H'$, then $\mathcal{B}, \alpha, \rho \Vdash A_1 \to A_2$; in fact for all $\alpha' \equiv (\Sigma'', H'') \in \mathrm{K}$ and $f\colon \alpha \to \alpha'$ (there is at most one such f):

- assume that

  $\mathcal{B}, \alpha', \rho_{\upharpoonright f} \Vdash A_1$
- By IH applied to A_1

  $A_1[\sigma] \in H''$

- By the assumption on H' and $H' \subseteq H''$

  $(A_1 \to A_2)[\sigma] \in H''$

- By the second property of complete sets

  $A_2[\sigma] \in H''$

- By IH applied to A_2

  $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A_2$

by definition of $\mathcal{B}, \alpha, \rho \Vdash \_$, it follows that $\mathcal{B}, \alpha, \rho \Vdash A_1 \to A_2$.

To prove that if $\mathcal{B}, \alpha, \rho \Vdash A_1 \to A_2$, then $(A_1 \to A_2)[\sigma] \in H'$, we need the separation lemma (see Lemma 2.6.4):

- assume that

  $(A_1 \to A_2)[\sigma] \notin H'$

- By the second property of complete sets

  $T, H' \not\vdash^{\mathrm{J}}_{\Sigma'} A_1[\sigma] \to A_2[\sigma]$

- By formal derivation

  $T, H' \not\vdash^{\mathrm{J}}_{\Sigma'} A_1[\sigma] \Longrightarrow A_2[\sigma]$

- By the Deduction Lemma

  $T, H', A_1[\sigma] \not\vdash^{\mathrm{J}}_{\Sigma'} A_2[\sigma]$

- take n s.t. $\Sigma' \subseteq \Sigma \uplus C_n$, then by the Separation Lemma

  there is a signature Σ'', with $\Sigma' \subseteq \Sigma'' \subseteq \Sigma \uplus C_{n+1}$, and an intuitionistically $T \cup H' \cup \{A_1[\sigma]\}$-complete set H'' on Σ'' not containing $A_2[\sigma]$

- let $\alpha' \triangleq (\Sigma'', H'')$ and $f\colon \alpha \to \alpha' \in \mathrm{K}$ be the unique from α to α', then by IH

  $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A_1$ but not $\mathcal{B}, \alpha', \rho_{\restriction f} \Vdash A_2$

- by definition of $\mathcal{B}, \alpha, \rho \Vdash \_$

  $\mathcal{B}, \alpha, \rho \Vdash A_1 \to A_2$ does not hold

By Soundness, to prove that if $T \vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then $\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$, it is enough to show that $\mathcal{B} \Vdash T$, which follows easily from the second statement and the definition of $\mathcal{B}$.

For the other implication we prove that if $T \not\vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then $\mathcal{B}$ does not satisfy $\eta.\Gamma_1 \Longrightarrow \Gamma_2$. Let C be the set of new constant symbols used in the definition of $\mathcal{B}_{(\Sigma,T)}$ (see Definition 2.6.10). Take a substitution $\gamma \in \mathrm{Term}^{\Sigma \uplus C}_{\eta}(\emptyset)$ that establishes a bijection between $\mathrm{dom}(\eta)$ and a finite set $C' \subseteq C$ of new constant symbols for Σ. Since C' is finite, there exists n s.t. C' is included in C_n.

- Assume that

  $T \not\vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$

- by the Deduction Lemma

  $T, \Gamma_1[\gamma] \not\vdash^{\mathrm{J}}_{\Sigma \uplus C'} \emptyset.\Gamma_2[\gamma]$,

- by the Separation Lemma there is an extension $\Sigma' \subseteq \Sigma \uplus C_{n+1}$ of $\Sigma \uplus C'$ and an intuitionistically $T \cup \{\Gamma_1[\gamma]\}$-complete set H' on Σ' s.t. $A[\gamma] \notin H'$ for all A in Γ_2

- let $\alpha \equiv (\Sigma', H') \in \mathrm{K}$ and $\rho \triangleq [\gamma]_{H'} \in \eta^{\mathcal{B}(\alpha)}$, by the second statement

  $\mathcal{B}, \alpha, \rho \Vdash A_1$, but not $\mathcal{B}, \alpha, \rho \Vdash A$ for any $A \in \Gamma_2$

  therefore $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is not valid in $\mathcal{B}$

∎

Theorem 2.6.12 Completeness

$\vdash^{\mathrm{J}}_{\Sigma}$ *is complete w.r.t.* $\models^{\mathrm{J}}_{\Sigma}$

Proof Given a set $T \subseteq \mathrm{Seq}^{\Sigma}$ and a sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{Seq}^{\Sigma}$, if $T \not\vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then we can apply the characterization of the Kripke Σ-structure $\mathcal{B}_{(\Sigma,T)}$ (see Proposition 2.6.11). Therefore, $\mathcal{B}_{(\Sigma,T)}$ is a model of T and it does not satisfy $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ ∎

Chapter 3

Meta-logical results

In this chapter we establish some general facts about LPT (see Sections 3.1 and 3.2) used for proving conservative extension results between formal systems by simply looking at the sequents that *axiomatize* them (in LPT) or at the inference rules that *generate* their *pure theories*.

In Section 3.3 we take a critical look at the logic of partial elements (LPE), by considering three *flavours* of LPE and by discussing the relations among them and LPT.

3.1 Coincidence of intuitionistic and classical logic

In this section we give two results on coincidence of intuitionistic and classical LPT (see Theorems 3.1.5 and 3.1.15). Theorem 3.1.5 is implicit from 5.2 of [MR77], but we did not find anything like Theorem 3.1.15 in the literature.

3.1.1 Coherent theories

Coherent formulae and coherent sequents have been investigated quite extensively, especially in relation to geometric morphisms between toposes. A formal system for *Coherent Logic* can be found in 5.2 of [MR77]. The most familiar theories axiomatizable on the coherent fragment are the equational, quasi-equational and essentially algebraic theories (see [KR77]). We are interested in only one property of the coherent fragment, namely that classical and intuitionistic logic induce the same logical consequence relation over it (see Theorem 3.1.5).

Definition 3.1.1 Coherent Formulae

A formula is **coherent** *if it is in the fragment* $\mathrm{Frag}^{\Sigma}(\bot \wedge \vee \exists)$

A sequent is **coherent** *if it is in the* **coherent fragment** $\{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee \exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot \wedge \vee \exists)$

Remark 3.1.2 The fragment $\{\mathrm{Frag}^{\Sigma}(\bot\wedge\vee\exists)\} \Longrightarrow \{\mathrm{Frag}^{\Sigma}(\bot\wedge\vee\exists)\}$ has the same expressive power as the coherent fragment, therefore the restriction to one-sided sequents can be dropped.

Proposition 3.1.3 Local property of Coherent Formulae

Given a Kripke Σ-structure $\mathcal{B}$ over K*, for all type environments η, coherent formulae $A \in \mathrm{Frag}^{\Sigma}(\bot \wedge \vee \exists)(\eta)$, stages $\alpha \in \mathrm{K}$ and environments $\rho \in \eta^{\mathcal{B}(\alpha)}$:*

- $\mathcal{B}, \alpha, \rho \Vdash A$ *iff* $\mathcal{B}(\alpha), \rho \models A$

Proof By induction on the structure of A ∎

An easy consequence of the local property of coherent formulae is:

Proposition 3.1.4 *If $\mathcal{B}$ is a Kripke model of a set T of coherent sequents and α is a stage of knowledge, then $\mathcal{B}(\alpha)$ is a classical model of T*

The two propositions above can be used to establish the meta-logical result we are interested in:

Theorem 3.1.5 Coincidence of the Free Logics

Let L be the coherent fragment $\{\mathrm{Frag}^{\Sigma}(\bot\wedge\vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot\wedge\vee\exists)$*, then intuitionistic and classical free logic coincide on L, i.e. $Free J\mathrm{LPT} \subset^{c}_{L\vdash L} Free K\mathrm{LPT}$.*

Proof We have to show that for all sets T of coherent sequents and coherent sequents $\eta.\Gamma_1 \Longrightarrow A_2$:

- $T \vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow A_2$ iff $T \vdash^{\mathrm{K}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow A_2$

The implication from left to right is obvious. By completeness for intuitionistic logic and the definition of $\mathcal{B}\Vdash\_$, to prove the other implication it is enough to show that $\mathcal{B}, \alpha, \rho \Vdash A_2$ for all Kripke models $\mathcal{B}$ of T, stages α and environments ρ s.t. $\mathcal{B}, \alpha, \rho \Vdash \Gamma_1$.

- by the local property of coherent formulae (see Proposition 3.1.3)

  $\mathcal{B}(\alpha), \rho \models \Gamma_1$

- since $\mathcal{B}(\alpha) \models T$ (see Proposition 3.1.4), by Soundness for classical logic

  $\mathcal{B}(\alpha), \rho \models A_2$

- by the local property of coherent formulae

  $\mathcal{B}, \alpha, \rho \Vdash A_2$

∎

3.1.2 Harrop theories

In the sequel, we will investigate theories that are not axiomatizable by coherent sequents. The typical sequents that fail to be coherent are those for *extensionality*, e.g. extensionality of total functions

$$\text{ext.} = \; (\forall z{:}\, \tau . xz = yz) \Longrightarrow x = y$$

However, these theories are axiomatizable by *Harrop sequents*.

Intuitionistic theories axiomatized by *Harrop formulae* have been widely investigated (see 1.10, 5.1 and 5.2 of [Tro73]) and they enjoy *nice* properties, the most familiar ones being the disjunctive and existential properties (see Proposition 3.1.11 and 5.2.10 of [Tro73]). These two properties provide also a foundation for higher order logic programming (see [MNS87]).

Classical and intuitionistic logic differ over the Harrop fragment. However, it is possible to make the two logical consequence relations coincide (see Theorem 3.1.15), by restricting them to a smaller fragment (and by using the inhabited logics, see Definition 2.4.5). The sequent for extensionality of total functions above is in this smaller fragment, but other sequents are not, e.g. extensionality of partial functions (see Definition 4.2.9)

$$\text{ext.} \simeq \; (\forall z{:}\, \tau . xz \simeq yz) \Longrightarrow x = y$$

The proofs of Theorems 3.1.11 and 3.1.15 rely on the properties of a special Kripke term model (see Proposition 3.1.10), whose connections with classical term model constructions are not made explicit in the literature:

- the initial algebra for an algebraic theory is an instance of the Initial Model for a $T \subseteq \{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ (see Proposition 3.1.12)

- the open term model for a lambda-theory (see [Bar81]) is an instance of the Free Model for a $T \subseteq \{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ (see Proposition 3.1.14).

Definition 3.1.6 Harrop Sequent

$\eta.\Gamma \Longrightarrow A$ *is an* **Harrop sequent** *if it is in the* **Harrop fragment** $\{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall \rightarrow)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$

Remark 3.1.7 Harrop formulae, as defined in 1.10.5 of [Tro73], and Harrop sequents are equivalent, in the sense that any theory axiomatizable by Harrop sequents is axiomatizable by Harrop formulae and conversely. For convenience we recall the definition of **Harrop formulae**:

- an atomic formula and $\bot$ are Harrop formulae
- if A_1 and A_2 are Harrop formulae, then $A_1 \wedge A_2$ is a Harrop formula
- if A is a Harrop formula, then $\forall x\colon \tau.A$ is a Harrop formula
- if A_2 is a Harrop formula, then $A_1 \rightarrow A_2$ is a Harrop formula

It is obvious that if A is a Harrop formula (in particular one in $\mathrm{Frag}^{\Sigma}(\bot)$), then the sequent $\eta.\Gamma \Longrightarrow A$ is equivalent to the Harrop formula $\eta.\wedge(\Gamma) \rightarrow A$. Conversely, if A is a Harrop formula, then the sequent $\eta.A$ is equivalent to a finite set $T_{\eta.A}$ of Harrop sequents defined by induction on A:

- if A is an atomic formula or $\bot$, then
  $T_{\eta.A} \triangleq \{\eta.A\}$
- if $A \equiv A_1 \wedge A_2$, then
  $T_{\eta.A} \triangleq T_{\eta.A_1} \cup T_{\eta.A_2}$
- if $A \equiv \forall x\colon \tau.A'$, then
  $T_{\eta.A} \triangleq T_{\eta\lfloor x:\tau.A'}$
- if $A \equiv A_1 \rightarrow A'$, then
  $T_{\eta.A} \triangleq \{\eta'.\Gamma_1, A_1 \Longrightarrow A_2 | \eta'.\Gamma_1 \Longrightarrow A_2 \in T_{\eta.A'}\}$

In classical logic any formula, say A, is equivalent to a Harrop formula, e.g. $(A \rightarrow \bot) \rightarrow \bot$, because of the double-negation axiom (see Definition 2.4.5).

Definition 3.1.8 The Kripke Term Model $\mathcal{B}^{\perp}_{(\Sigma,T)}$

Given a set T of sequents over Σ, the Kripke Σ-structure $\mathcal{B}^{\perp}_{(\Sigma,T)}$ over $\mathrm{K}^{\perp}_{(\Sigma,T)}$ is obtained by extending $\mathcal{B}_{(\Sigma,T)}$ of Definition 2.6.10 as follows:

- $\mathrm{K}^{\perp}_{(\Sigma,T)}$ *is* $\mathrm{K}_{(\Sigma,T)}$ *with a new stage of knowledge $\perp$ at the bottom*
- $\mathcal{B}^{\perp}_{(\Sigma,T)}(\perp) \triangleq \mathcal{A}_{(\Sigma,H)}$, *where H is the* Σ-acceptable *set* $\{A \in \mathrm{Form}^{\Sigma}_{0} | T \vdash^{\mathrm{J}}_{\Sigma} A\}$ *of closed formulae intuitionistically derivable from T.*

  At the other stages it is like $\mathcal{B}_{(\Sigma,T)}$
- $\mathcal{B}^{\perp}_{(\Sigma,T)}(f)_{\tau}([t]_H) \triangleq [t]_{H'}$ *for all $t \downarrow_{\tau} \in H$ and $f\colon \perp \to (\Sigma', H')$.*

  For the other morphisms of $\mathrm{K}_{(\Sigma,T)}$ *it is like* $\mathcal{B}_{(\Sigma,T)}$

Remark 3.1.9 Since H (above) is Σ-acceptable and it is included in any T-complete set, then $\mathcal{B}^{\perp}_{(\Sigma,T)}$ and $\mathcal{B}^{\perp}_{(\Sigma,T)}(f)_{\tau}$ are well-defined.

$\mathcal{B}^{\perp}_{(\Sigma,T)}$ is the result of applying the operation $(\ ) \to (\ )'$, defined in 5.2 of [Tro73], to the Kripke model $\mathcal{B}_{(\Sigma,T)}$. Similar definitions, used to prove the disjunctive and existential properties, can be found also in [vD86, Acz68].

If $\mathcal{B}^{\perp}_{(\Sigma,T)}$ satisfies a sequent, then that sequent is intuitionistically derivable from T; but $\mathcal{B}^{\perp}_{(\Sigma,T)}$, unlike $\mathcal{B}_{(\Sigma,T)}$, may not be a model of T.

Proposition 3.1.10 Characterization of $\mathcal{B}^{\perp}_{(\Sigma,T)}$

Let T be a consistent set of Harrop sequents on Σ. Then, $\mathcal{B}^{\perp}_{(\Sigma,T)} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$ *iff* $T \vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$

Proof The implication from left to right follows from the properties of $\mathcal{B}_{(T,\Sigma)}$ (see Proposition 2.6.11).

Let $\mathrm{K} \equiv \mathrm{K}^{\perp}_{(\Sigma,T)}$ and $\mathcal{B} \equiv \mathcal{B}^{\perp}_{(\Sigma,T)}$. By Soundness (see Theorem 2.5.3), to prove that $T \vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$ implies $\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow \Gamma_2$, it is enough to show that $\mathcal{B} \Vdash T$, i.e. for all axioms $\eta.\Gamma_1 \Longrightarrow A_2 \in T$, stages $\alpha \in \mathrm{K}$ and environments $\rho \in \eta^{\mathcal{B}(\alpha)}$ if $\mathcal{B}, \alpha, \rho \Vdash \Gamma_1$, then $\mathcal{B}, \alpha, \rho \Vdash A_2$. If α is not $\perp$, then it follows from Proposition 2.6.11. So we have to check only the case when α is $\perp$.

Take a substitution $\sigma \in \prod \rho$:

- assume that

  $\mathcal{B}, \perp, \rho \Vdash \Gamma_1$

- by the Substitution Lemma (see Lemma 2.5.2)

  $\mathcal{B}, \perp, \emptyset \Vdash \Gamma_1[\sigma]$

- by Functoriality (see Lemma 2.3.4) for all $(\Sigma', H') \in \mathrm{K}$

  $\Gamma_1[\sigma] \subseteq_{fin} H'$

- by the Separation Lemma (see Lemma 2.6.4)

  $T \vdash^{J}_{\Sigma} \Gamma_1[\sigma]$

- by formal derivation

  $T \vdash^{J}_{\Sigma} A_2[\sigma]$, i.e. $A_2[\sigma] \in H$

- since A_2 is atomic, by definition of $\mathcal{B}$ (see Definition 3.1.8)

  $\mathcal{B}, \perp, \emptyset \Vdash A_2[\sigma]$

- by the Substitution Lemma (see Lemma 2.5.2)

  $\mathcal{B}, \perp, \rho \Vdash A_2$

∎

$\mathrm{K}^{\perp}_{(\Sigma,T)}$ (unlike $\mathrm{K}_{(\Sigma,T)}$) has always a least stage of knowledge, as required by the more restrictive definition of Kripke model (see [Tro73, vD86]). From this, it follows easily that the theory of $\mathcal{B}^{\perp}_{(\Sigma,T)}$ has the disjunctive and existence properties. These properties can be established also by proof-theoretic methods (see [Pra65]).

Theorem 3.1.11 Disjunctive and Existence properties

The theory generated by a consistent set T of Harrop sequents satisfies the disjunctive and existence properties, i.e.:

$\vee$ *if* $T \vdash^{J}_{\Sigma} \emptyset.A_1 \vee A_2$, *then* $T \vdash^{J}_{\Sigma} \emptyset.A_1$ *or* $T \vdash^{J}_{\Sigma} \emptyset.A_2$

$\exists$ *if* $T \vdash^{J}_{\Sigma} \emptyset.\exists x{:}\,\tau.A$, *then* $T \vdash^{J}_{\Sigma} \emptyset.A[x{:=}t]$ *for some* t *s.t.* $T \vdash^{J}_{\Sigma} \emptyset.t\downarrow_{\tau}$

Proof The two properties are proved in a similar way, so we consider only the existence property. We make essential use of the characterization of $\mathcal{B}^{\perp}_{(\Sigma,T)}$ (see Proposition 3.1.10).

Let $\mathcal{B} \equiv \mathcal{B}^{\perp}_{(\Sigma,T)}$ and $H \equiv \{A \in \mathrm{Form}^{\Sigma}_{0} | T \vdash^{J}_{\Sigma} A\}$:

- assume that $T \vdash^{\mathrm{J}}_{\Sigma} \emptyset.\exists x{:}\,\tau.A$
- by the characterization of $\mathcal{B}$

  $\mathcal{B} \Vdash \emptyset.\exists x{:}\,\tau.A$
- by Proposition 2.3.6

  $\mathcal{B}, \bot, \emptyset \Vdash \exists x{:}\,\tau.A$
- by definition of $\mathcal{B}, \alpha, \rho \Vdash \_$ and $\mathcal{B}$, there exists t s.t.

  $t \downarrow_\tau \in H$ (i.e. $T \vdash^{\mathrm{J}}_{\Sigma} \emptyset.t \downarrow_\tau$) and

  $\mathcal{B}, \bot, x{:}= [\![t]\!]^{\mathcal{B},\bot}_{\emptyset} \Vdash A$
- by substitution lemma

  $\mathcal{B}, \bot, \emptyset \Vdash A[x{:}= t]$
- since $\bot$ is the least stage of knowledge, by functoriality

  $\mathcal{B}, \alpha, \emptyset \Vdash A[x{:}= t]$, for all stages α
- by definition of $\mathcal{B} \Vdash \_$

  $\mathcal{B} \Vdash \emptyset.A[x{:}= t]$
- by the characterization of $\mathcal{B}$

  $T \vdash^{\mathrm{J}}_{\Sigma} \emptyset.A[x{:}= t]$

∎

If we consider a consistent theory axiomatized by a set of sequents T in the intersection of the coherent and Harrop fragments (e.g. **quasi-equations** $\overline{t} = \overline{t}' \Longrightarrow t = t'$), then the Σ-structure $\mathcal{B}^{\bot}_{(\Sigma,T)}(\bot)$ is a model of T (usually called the *initial* model of T), which satisfies only those closed atomic formulae derivable from T, more precisely:

Proposition 3.1.12 The Initial Model for Harrop Theories without $\forall$ and $\to$

Let $T \subseteq \{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ be a consistent set of sequents and $\mathcal{A} \equiv \mathcal{B}^{\bot}_{(\Sigma,T)}(\bot)$:

- *if $A \in \mathrm{Frag}^{\Sigma}_0(\bot \wedge \vee\exists)$, then $T \vdash^{\mathrm{J}}_{\Sigma} A \Longleftrightarrow \mathcal{A} \models A$*

Moreover, $\mathcal{A} \models T$

Proof The first statement follows immediately from Lemma 2.3.6 and Propositions 3.1.10 and 3.1.3.

$\mathcal{A} \models T$ is an immediate consequence of Propositions 3.1.10 and 3.1.4 ∎

The previous result can be *improved* by considering consistent theories axiomatized by a set of sequents T in $\{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ plus the axiom (inhab) for models with all sorts inhabited (see Definition 2.4.5). The exact statement is:

> there is a model of $T + \mathrm{inhab}$, which satisfies only those formulae in $\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall)$ derivable from $T + \mathrm{inhab}$

The axiom (inhab) is not in the Harrop fragment, but it is easy to show that:

Proposition 3.1.13 *If C is a set of new constant symbols for Σ, with at least one symbol for each sort , then* LPT *(over $\Sigma \uplus C$) is an extension of Inhabited*LPT *(over Σ) conservative over $\mathcal{P}(\mathrm{Seq}^{\Sigma}) \times \mathrm{Seq}^{\Sigma}$.*

Proof If $\mathcal{A}$ is a $\Sigma \uplus C$-structure, then $\mathcal{A}\restriction\Sigma$ is inhabited (by the assumptions on C). On the other hand, if $\mathcal{A}$ is an inhabited Σ-structure, then it is always possible to interpret additional constant symbols. In particular, $\mathcal{A}$ can be extended to a $\Sigma \uplus C$-structure. By Proposition 1.3.5, the claim follows from the two facts above. ∎

Then the statement above is a consequence of the following proposition:

Proposition 3.1.14 The Free Model for Harrop Theories without $\rightarrow$

Given a set C of new constant symbols for Σ, with $\aleph_0$ symbols for each sort, let $T \subseteq \{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ be a consistent set of sequents over $\Sigma \uplus C$, i.e. $T \not\vdash^{\mathrm{J}}_{\Sigma\uplus C} \bot$, and $\mathcal{A} \equiv \mathcal{B}^{\bot}_{(\Sigma\uplus C,T)}(\bot)$. Then:

- *if $A \in \mathrm{Frag}_0^{\Sigma\uplus C}(\bot \wedge \vee\exists\forall)$, then $T \vdash^{\mathrm{J}}_{\Sigma\uplus C} A \Longleftrightarrow \mathcal{A}, \emptyset \models A$*

Moreover, $\mathcal{A} \models T$

Proof The first statement is proved by induction on the number of connectives and quantifiers. The implication from left to right follows from Proposition 3.1.10 and a *semi-local* property for formulae in $\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists\forall)$:

Claim 3.1.14.1 *Given a Kripke Σ-structure $\mathcal{B}$ over K, for all type environments η, $A \in \mathrm{Frag}^{\Sigma}(\perp \wedge \vee\exists\forall)(\eta)$, stages $\alpha \in \mathrm{K}$ and environments $\rho \in \eta^{\mathcal{B}(\alpha)}$:*

- *if $\mathcal{B}, \alpha, \rho \Vdash A$, then $\mathcal{B}(\alpha), \rho \models A$*

Proof By induction on A ∎

For the implication from right to left the only non trivial case is when A is a universally quantified formula. First we establish the following fact:

Claim 3.1.14.2 *If $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} A[x := c]$ and $c \in C_\tau$ does not occur in A, then $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} \forall x{:}\,\tau.A$*

Proof Assume that $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} A[x := c]$

- By compactness of $\vdash^{\mathrm{J}}$, there exists $C_0 \subseteq_{fin} C$ s.t. $c \notin C_0$ and

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C_0 \uplus c} A[x := c]$
- By the Deduction Lemma and formal derivation

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C_0} \forall x{:}\,\tau.A$
- By compactness of $\vdash^{\mathrm{J}}$

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} \forall x{:}\,\tau.A$

∎

Now we prove that $\mathcal{A}, \emptyset \models \forall x{:}\,\tau.A$ implies $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} \forall x{:}\,\tau.A$:

- assume that $\mathcal{A}, \emptyset \models \forall x{:}\,\tau.A$
- since C_τ is infinite, there exists $c \in C_\tau$ which does not occur in A, by definition of $\mathcal{A}, \rho \models \_$ and Substitution Lemma

  $\mathcal{A}, \emptyset \models A[x := c]$
- by IH applied to $A[x := c]$

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} A[x := c]$
- by the claim and the assumptions about c

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} \forall x{:}\,\tau.A$

To show that $\mathcal{A}$ is a model of T we prove that for any $\eta.\Gamma_1 \Longrightarrow A_2 \in T$ and $\rho \in \eta^{\mathcal{A}}$ if $\mathcal{A}, \rho \models \Gamma_1$, then $\mathcal{A}, \rho \models A_2$. Take a substitution $\sigma \in \prod \rho$:

- by the Substitution Lemma (see Lemma 2.5.2)

  $\mathcal{A}, \emptyset \models \Gamma_1[\sigma]$

- since $\Gamma[\sigma] \subseteq_{fin} \mathrm{Frag}_0^{\Sigma \uplus C}(\perp \wedge \vee \exists \forall)$, by the first statement

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} \Gamma_1[\sigma]$

- by formal derivation

  $T \vdash^{\mathrm{J}}_{\Sigma \uplus C} A_2[\sigma]$

- since A_2 is atomic, by definition of $\mathcal{A}$ (see Definition 3.1.8)

  $\mathcal{A}, \emptyset \models A_2[\sigma]$

- by the Substitution Lemma (see Lemma 2.5.2)

  $\mathcal{A}, \rho \models A_2$

∎

Theorem 3.1.15 Coincidence of the Inhabited Logics

Let L be $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$, then intuitionistic and classical inhabited logic coincide on L, i.e. Inhabited$J$LPT $\subset^{c}_{L \vdash L}$ Inhabited$K$LPT.

It is clear from the proof below that the coincidence result for inhabited logics can be improved as follows: $T \vdash^{\mathrm{J}}_{\Sigma} A$ iff $T \vdash^{\mathrm{K}}_{\Sigma} A$ for all $T \subseteq \{\mathrm{Frag}^{\Sigma}(\perp \wedge \vee \exists \forall)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$ and $A \in \mathrm{Frag}_0^{\Sigma}(\perp \wedge \vee \exists \forall)$.

Proof We have to prove that *Inhabited$J$*LPT and *Inhabited$K$*LPT have the same consequence relation on the fragment $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$. The inclusion from left to right is obvious.

The other inclusion is proved by contraposition.

Take a set C' of new constant symbols (of the appropriate sorts) one for each of the variables in $\mathrm{dom}(\eta)$ and a substitution $\gamma \in \mathrm{Term}_{\eta}^{\Sigma \uplus C'}(\emptyset)$ that establish a bijection between $\mathrm{dom}(\eta)$ and the set C' of new constant symbols for Σ. Take also a set C of new constant symbols for $\Sigma \uplus C'$, with $\aleph_0$ symbols for each sort:

- assume that

  $T + \text{inhab} \not\vdash^{\mathrm{J}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow A_2$

- by the Deduction Lemma

  $T, \Gamma_1[\gamma] + \text{inhab} \not\vdash^{\mathrm{J}}_{\Sigma \uplus C'} A_2[\gamma]$

- by Proposition 3.1.13

  $T, \Gamma_1[\gamma] \not\vdash^{\mathrm{J}}_{\Sigma \uplus C' \uplus C} A_2[\gamma]$

- because of the equivalence between Harrop sequents and Harrop formulae (see Remark 3.1.7), we can treat $\Gamma_1[\gamma]$ as a set of sequents in $\{\emptyset\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$ Since $T \cup \{\Gamma_1[\gamma]\}$ is consistent over $\Sigma \uplus C' \uplus C$, we can apply Proposition 3.1.14 and get a classical model $\mathcal{A}$ of $T \cup \{\Gamma_1[\gamma]\}$ s.t.

  $\mathcal{A} \models A_2[\gamma]$ does not hold

- By Soundness

  $T, \Gamma_1[\gamma] \not\vdash^{\mathrm{K}}_{\Sigma \uplus C' \uplus C} A_2[\gamma]$

- by Proposition 3.1.13

  $T, \Gamma_1[\gamma] + \text{inhab} \not\vdash^{\mathrm{K}}_{\Sigma \uplus C'} A_2[\gamma]$

- by the Deduction Lemma

  $T + \text{inhab} \not\vdash^{\mathrm{K}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow A_2$

∎

3.1.3 Counterexamples and discussion

By some counterexamples, we show that there is little space for improving the results on coincidence of classical and intuitionistic logic (see Theorems 3.1.5 and 3.1.15).

Proposition 3.1.16 *Intuitionistic and classical free (inhabited) logic do not coincide on the fragment* $\{\mathrm{Frag}(\rightarrow)\} \Longrightarrow \mathrm{Frag}(\emptyset)$

Proof Take a signature Σ with (at least) three propositional constants p_1, p_2 and p_3. Let T be the set of sequents

$$\{((p_1 \rightarrow p_2) \Longrightarrow p_3), (p_1 \Longrightarrow p_3)\}$$

$T \vdash^{\mathrm{K}}_{\Sigma} p_3$ by formal derivation. On the other hand, any Kripke Σ-structure over $\{\perp < \top\}$ s.t.

- at stage $\perp$ none of the predicates is true
- at stage $\top$ both p_1 and p_3 becomes true, while p_2 remains not true

is a model of T, but does not satisfy p_3 ∎

Proposition 3.1.17 *Intuitionistic and classical free (inhabited) logic do not coincide on the fragment* $\{\mathrm{Frag}(\forall)\} \Longrightarrow \mathrm{Frag}(\vee)$

Proof Take a signature Σ with (at least) one sort symbol τ, two propositional constants p_1 and p_3 and an unary predicate p_2. Let T be the set of sequents

$$\{(x{:}\,\tau.p_2(x) \vee p_1), ((\forall x{:}\,\tau.p_2(x)) \Longrightarrow p_3), (p_1 \Longrightarrow p_3)\}$$

$T \vdash^{\mathrm{K}}_{\Sigma} p_3$ by formal derivation. On the other hand, any Kripke Σ-structure $\mathcal{B}$ over $\{\perp < \top\}$ s.t.

- at stage $\perp$ neither p_1 nor p_3 is true, while $p_2(a)$ is true, for all $a \in \tau^{\mathcal{B}(\perp)}$
- at stage $\top$ both p_1 and p_3 becomes true and there is a new element b of sort τ that comes into existence s.t. $p_2(b)$ is not true

is a model of T, but does not satisfy p_3 ∎

Proposition 3.1.18 *Intuitionistic and classical free logic do not coincide on the fragment* $\{\mathrm{Frag}(\forall)\} \Longrightarrow \mathrm{Frag}(\emptyset)$

Proof Take a signature Σ with *only* one sort symbol τ and two propositional constants p_1 and p_2. Let T be the set of sequents

$$\{((\forall x{:}\,\tau.p_1) \Longrightarrow p_2), (x{:}\,\tau.p_2)\}$$

$T \vdash^{\mathrm{K}}_{\Sigma} p_2$ by formal derivation. On the other hand, any Kripke Σ-structure $\mathcal{B}$ over $\{\perp < \top\}$ s.t.

- at stage $\perp$ none of the predicates is true and $\tau^{\mathcal{B}(\perp)}$ is empty
- at stage $\top$ p_2 becomes true and $\tau^{\mathcal{B}(\perp)}$ becomes inhabited, while p_1 remains not true

is a model of T, but does not satisfy p_2 ∎

We conjecture that intuitionistic and classical inhabited logic do not coincide on the fragment $\{\mathrm{Frag}(\forall)\} \Longrightarrow \mathrm{Frag}(\exists)$. However, they coincide on $\{\mathrm{Frag}(\forall)\} \Longrightarrow \mathrm{Frag}(\exists!)$, because the $\exists!$ can be removed by introducing a partial function which picks up the **unique** witness (this restricted form of *skolemization* is intuitionistically correct).

All the axioms of **LCF** (see [Sco69]) are in $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\bot)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$, except the axiom for fixed-point induction:

$$A(\bot), (\forall x.A(x) \rightarrow A(fx)) \Longrightarrow A(Yf)$$

where A is a conjunction of inequalities. We do not know whether classical and intuitionistic **LCF** are the same on the atomic fragment.

3.2 Restriction of LPT to fragments

When one looks at the inference rules for the λ-calculus, (partial) equational logic and so on, the *impression* is that they are just the inference rules of a first order theory (with some syntactic sugar, e.g. λ-abstraction) restricted to the equational fragment. However, to check that the inference rules *formalize* what they are supposed to, one has to prove *soundness* and *completeness* w.r.t. a class of models (for a first order theory). The proof involves fairly standard techniques, but it is quite long (as we have seen for LPT).

Most of the formal systems considered in this thesis are on first order languages rather than on smaller languages (like the equational fragment), but in this section

> we set up a method that given a set of axioms $T \subseteq \mathrm{Seq}^{\Sigma}$ and a T-*closed* fragment F of Seq^{Σ} (see Definition 3.2.6), produces a set of inference rules on F to generate the elements of F that are derivable from T in LPT (see Definition 3.2.9 and Theorem 3.2.12).

If there are no axioms (i.e. $T = \emptyset$), then the simple minded idea is to take the inference rules of LPT *restricted* to F (i.e. premisses and conclusion must be in F). Gentzen's cut-elimination result (see [Gen69]) implies that this simple method works, when $F = \{F'\} \Longrightarrow \{F'\}$ and F' is *closed under subformulae*.

When T is not empty, we have to add the axioms (in T) as inference rules and we need only a *restricted cut-rule*, where the *cut formula* is a substitution instance of a formula in T (see Chapter 5 of [MR77]). Then, the simple method works, when F satisfies the condition above and $T \subseteq F$.

However, these results are not completely satisfactory. For instance, if T is the set of axioms for extensional combinatory logic, then F' must contain $\mathrm{Frag}^{\Sigma}(\forall)$ (because of the extensionality axiom), while we can easily give a set of inference rules on the *equational fragment* $\{\emptyset\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\emptyset)$.

The basic idea to improve the results above is that the sequent

$$\emptyset.\forall x\colon \tau.p_1(x) \wedge p_2(x) \rightarrow p(x) \Longrightarrow q$$

can be viewed *equivalently* as the inference rule (schema)

$$\frac{\eta\lfloor x\colon \tau.\Gamma, p_1(x), p_2(x) \Longrightarrow p(x)}{\eta.\Gamma \Longrightarrow q}$$

(see $(\mathrm{R}T_{pt})$ in Definition 3.2.9). In order to carry out this transformation, we consider only sequents in the fragment $\{\mathrm{Frag}^{\Sigma}(\wedge\forall \rightarrow)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\emptyset)$.

Remark 3.2.1 From a pragmatic point of view this restriction is not so drastic, for instance all formal systems in the thesis can be axiomatized by this kind of sequent. Moreover, we can replace the three constructors $\wedge$, $\forall$ and $\rightarrow$ with a compound one $(\forall\_.\wedge(\_) \rightarrow \_)$, which makes explicit the connection between formulae in $\mathrm{Frag}^{\Sigma}(\wedge\forall \rightarrow)$ and *higher order one-sided sequents* (see [SH84]) and drastically reduces the number of formation and inference rules, although they become more complex (see Definitions 3.2.2 and 3.2.9).

3.2.1 Language revisited

We introduce a new set of formulae (*s-formulae*, s- for sequent) and a new set of sequents (*s-sequents*). There is only one way of combining s-formulae together to build a s-formula $(\eta.\Gamma \rightarrow A)$, which is essentially the way of combining s-formulae to build a one-sided s-sequent. We define also a translation $\mathrm{I_s}$ from s-sequents to sequents. By pulling back along it (see Definition 1.3.6) we can define intuitionistic and classical logic on s-sequents.

Definition 3.2.2 Revisited Language of LPT

Given a signature Σ,

1. *the set* $\mathrm{AForm}^{\Sigma}(\eta)$ *of* **atomic formulae** *in the type environment* η

2. *the set* $\mathrm{SForm}^{\Sigma}(\eta)$ *of* **s-formulae** *in the type environment* η

3. *the set* SSeq^{Σ} *of* **s-sequents**

are defined by the following formation rules*:*

$$p \ \frac{\{\overline{t}_i \in \mathrm{Term}^{\Sigma}_{\overline{\tau}_i}(\eta) | i \in |\overline{\tau}|\}}{p(\langle \overline{t}_i | i \in |\overline{\tau}|\rangle) \in \mathrm{AForm}^{\Sigma}(\eta)} \ p \in \mathrm{Pred}_{\overline{\tau}}(\Sigma)$$

$$\rightarrow \ \frac{\Gamma_1 \subseteq_{fin} \mathrm{SForm}^{\Sigma}(\eta \lfloor \eta') \qquad A_2 \in \mathrm{AForm}^{\Sigma}(\eta \lfloor \eta')}{(\eta'.\Gamma_1 \rightarrow A_2) \in \mathrm{SForm}^{\Sigma}(\eta)}$$

$$\Longrightarrow \ \frac{\Gamma_1 \subseteq_{fin} \mathrm{SForm}^{\Sigma}(\eta) \qquad \Gamma_2 \subseteq_{fin} \mathrm{AForm}^{\Sigma}(\eta)}{\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{SSeq}^{\Sigma}}$$

Notation 3.2.3 JSSeq^{Σ} is the fragment of SSeq^{Σ} whose elements are s-sequents with exactly one formula in the succedent. In the sequel we will make use of the following shorthands for s-formulae and s-sequents (see also Notation 2.1.8):

- If A is an atomic formula, then it may stand for the s-formula $\emptyset.\emptyset \rightarrow A$
- If $\eta.\Gamma \rightarrow A$ is a closed s-formula, then it may stand for the s-sequent $\eta.\Gamma \Longrightarrow A$

Remark 3.2.4 The *intended* meaning of an s-formula $(\eta.\Gamma \rightarrow A)$ is given by a translation from SForm^{Σ} to $\mathrm{Frag}^{\Sigma}(\wedge\forall\rightarrow)$ defined by induction on the structure of s-formulae (and extended to SSeq^{Σ} in the obvious way):

- $\mathrm{I_s}(\eta.\Gamma_1 \rightarrow A_2) \triangleq (\forall\eta. \wedge \mathrm{I_s}(\Gamma_1) \rightarrow A_2)$
- $\mathrm{I_s}(\eta.\Gamma_1 \Longrightarrow \Gamma_2) \triangleq (\eta.\mathrm{I_s}(\Gamma_1) \Longrightarrow \Gamma_2)$

$\mathrm{I_s}$ has an *inverse translation*, i.e. any A in $\mathrm{Frag}^{\Sigma}(\wedge\forall\rightarrow)$ is equivalent (in $JLPT$) to the conjunction of a finite set S_A of s-formulae (more precisely their $\mathrm{I_s}$-translation) defined by induction on A:

- if A is an atomic formula, then
$S_A \triangleq \{A\}$

- if $A \equiv A_1 \wedge A_2$, then
$S_A \triangleq S_{A_1} \cup S_{A_2}$

- if $A \equiv \forall x\colon\tau.A'$, then
$S_A \triangleq \{x\colon\tau\lfloor\eta.\Gamma_1 \to A_2|\eta.\Gamma_1 \to A_2 \in S_{A'}\}$

- if $A \equiv A_1 \to A'$, then
$S_A \triangleq \{\eta.\Gamma_1, S_{A_1} \to A_2|\eta.\Gamma_1 \to A_2 \in S_{A'}\}$. It may be necessary to rename the bound variables in A' to make sure that the variables in η do not clash with the free variables in S_{A_1}.

Moreover, any sequent $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ in $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\to)\} \Longrightarrow \{\mathrm{Frag}^{\Sigma}(\emptyset)\}$ is equivalent to the ($\mathrm{I_s}$-translation of the) s-sequent $\eta.\cup\{S_A|A \in \Gamma_1\} \Longrightarrow \Gamma_2$.

The translation $\mathrm{I_s}$ induces an interpretation of s-formulae (and s-sequents) in a (Kripke) Σ-structure:

Definition 3.2.5 Interpretation in a Structure

Given a Σ-structure $\mathcal{A}$, the satisfaction relation $\mathcal{A}, \rho \models A$ is defined by induction on the s-formula A:

- $\mathcal{A}, \rho \models \eta_1.\Gamma_1 \to A_1 \stackrel{\Delta}{\Longleftrightarrow}$ for all $\rho_1 \in \eta_1^{\mathcal{A}}$
if $\mathcal{A}, \rho\lfloor\rho_1 \models \Gamma_1$, then $\mathcal{A}, \rho\lfloor\rho_1 \models A_1$

Given a Kripke Σ-structure $\mathcal{B}\colon \mathrm{K} \to \mathrm{Struct}(\Sigma)$, the satisfaction relation $\mathcal{B}, \alpha, \rho \Vdash A$ is defined by induction on the s-formula A:

- $\mathcal{B}, \alpha, \rho \Vdash \eta_1.\Gamma_1 \to A_1 \stackrel{\Delta}{\Longleftrightarrow}$ for all $f\colon \alpha \to \alpha'$ and $\rho_1 \in \eta_1^{\mathcal{B}(\alpha')}$
if $\mathcal{A}, \alpha', \rho_{\restriction f}\lfloor\rho_1 \Vdash \Gamma_1$, then $\mathcal{A}, \alpha', \rho_{\restriction f}\lfloor\rho_1 \Vdash A_1$

Certain fragments of SSeq^{Σ} enjoy an important closure property w.r.t. the revisited inference rules (see Theorem 3.2.12).

Definition 3.2.6 T-closed Fragment

Let $T \subseteq \mathrm{JSSeq}^{\Sigma}$. A fragment F of SForm^{Σ} *is T-**closed** iff:*

1. *if $(\eta.\Gamma \to A) \in F$ and $(\eta'.\Gamma' \to A') \in \Gamma$, then $\Gamma' \subseteq_{fin} F$*

2. if $(\eta.\Gamma \Longrightarrow A) \in T$ *and* $(\eta'.\Gamma' \rightarrow A') \in \Gamma$*, then* $\Gamma' \subseteq_{fin} F$

A fragment F *of* SSeq^Σ *is* T-**closed** *iff* $F = \{F'\} \Longrightarrow \{\mathrm{Frag}^\Sigma(\emptyset)\}$ *for some* T*-closed fragment* F' *of* SForm^Σ*.*

Example 3.2.7 Let $T = \{F\} \Longrightarrow \mathrm{Frag}^\Sigma(\emptyset)$, where F is a set of s-formulae. We consider three possibilities for F and for each of them we give the smallest T-closed fragment of SSeq^Σ together with some *well-known* axiom in T.

- If every formula in F is of the form $\eta.\emptyset \rightarrow A$, then the set of atomic formulae $\eta.\emptyset \Longrightarrow A$ is T-closed and extensionality for total functions is in T

$$(\forall z.xz = yz) \Longrightarrow x = y$$

- If every formula in F is of the form $\eta.t_1 \downarrow, \ldots, t_n \downarrow \rightarrow A$, then the set of *existentially conditioned atomic formulae* $\eta.t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow A$ is T-closed and extensionality for monotonic partial functions is in T

$$(\forall z.xz \downarrow \rightarrow xz \leq yz) \Longrightarrow x \leq y$$

- If every formula $\eta.\Gamma \rightarrow A$ in F is s.t. all formulae in Γ are atomic, then the set of *quasi-atomic formulae* $\eta.\Gamma \Longrightarrow A$, where Γ is a set of atomic formulae, is T-closed and the axiom of fixed-point induction (see [Sco69]) is in T

$$A(\bot), (\forall x.A(x) \rightarrow A(fx)) \Longrightarrow A(Yf)$$

3.2.2 Formal systems revisited

Any formal system $\vdash$ on Seq^Σ induces, via the $\mathrm{I_s}$-translation, a formal system on JSSeq^Σ, which we will call $\vdash$ as well. In this section we consider the formal systems on JSSeq^Σ induced by classical and intuitionistic logic of partial terms: $\vdash^{\mathrm{K}}_\Sigma$ and $\vdash^{\mathrm{J}}_\Sigma$. For each set $T \subseteq \mathrm{JSSeq}^\Sigma$ we define two sets of inference rules on

SSeq^{Σ} to generate the set of s-sequents derivable from T in $\vdash^{\mathrm{K}}_{\Sigma}$ and $\vdash^{\mathrm{J}}_{\Sigma}$ respectively (see Section 3.2.3).

The main feature of these sets is the absence of cut-like inference rules, i.e. rules that have at least one formula in the premisses which is not a *subformula* in the conclusion (or in T). More precisely, to derive a s-sequent in a T-closed fragment F from the revisited inference rules (for T) it is enough to use the inference rules *restricted* to F (see Theorem 3.2.12).

Remark 3.2.8 The restriction to JSSeq^{Σ} implies that:

- any s-sequent is *equivalent* to an s-formula and it is $\mathrm{I_s}$-translated to a Harrop sequent
- any theory is consistent, since the trivial model, where all atomic formulae $p(\overline{t})$ are true, satisfies all s-sequents.

From the point of view of JLPT the restriction is particularly natural (see 2.4.4). However, for KLPT we have to allow, at least in the intermediate steps of a derivation, s-sequents with more than one formula in the succedent.

Definition 3.2.9 Revisited Inference rules for LPT

Let $T \subseteq \mathrm{JSSeq}^{\Sigma}$ *and* T_{pt} *be* T *plus the axioms for* equality *(see Definition 2.4.1). The inference rules for* $\mathrm{Th}_{K\mathrm{LPT}}(T)$ *on* SSeq^{Σ} *are:*

E.x $\eta\lfloor x\colon\tau.\Gamma_1 \Longrightarrow x\downarrow_{\tau}, \Gamma'_1$

E.c $\eta.\Gamma_1 \Longrightarrow c\downarrow_{\tau}, \Gamma'_1$

$$\text{E.}f.\text{i}\ \frac{\eta.\Gamma_1 \Longrightarrow f(\overline{t})\downarrow_{\tau}, \Gamma'_1}{\eta.\Gamma_1 \Longrightarrow \overline{t}_i\downarrow_{\overline{\tau}_i}, \Gamma'_1}\ i \in \mathrm{dom}(\overline{t})$$

$$\text{E.}p.\text{i}\ \frac{\eta.\Gamma_1 \Longrightarrow p(\overline{t}), \Gamma'_1}{\eta.\Gamma_1 \Longrightarrow \overline{t}_i\downarrow_{\overline{\tau}_i}, \Gamma'_1}\ i \in \mathrm{dom}(\overline{t})$$

$$\to\Longrightarrow\ \frac{\begin{array}{c}\{\eta_1.\Gamma_1 \Longrightarrow \sigma(x)\downarrow, \Gamma'_1 | x \in \mathrm{dom}(\eta_2)\}\\ \{\eta_1\lfloor\eta_3.\Gamma_1, \Gamma_3[\sigma] \Longrightarrow A'_3[\sigma], \Gamma'_1 |\\ (\eta_3.\Gamma_3 \to A'_3) \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A'_2[\sigma], \Gamma'_1}\ \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1)\\ \eta_2.\Gamma_2 \to A'_2 \in \Gamma_1\end{array}$$

$$\mathrm{R}T_{pt}\ \frac{\begin{array}{c}\{\eta_1.\Gamma_1 \Longrightarrow \sigma(x)\downarrow, \Gamma_1' | x \in \mathrm{dom}(\eta_2)\} \\ \{\eta_1\lfloor\eta_3.\Gamma_1, \Gamma_3[\sigma] \Longrightarrow A_3'[\sigma], \Gamma_1' | \\ (\eta_3.\Gamma_3 \to A_3') \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A_2'[\sigma], \Gamma_1'}\ \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1) \\ \eta_2.\Gamma_2 \Longrightarrow A_2' \in T_{pt}\end{array}$$

- $T \vdash^{\mathrm{RK}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2 \stackrel{\Delta}{\Longleftrightarrow} \eta.\Gamma_1 \Longrightarrow \Gamma_2$ *is derivable by the inference rules for* $\mathrm{Th}_{K\mathrm{LPT}}(T)$.
- $T \vdash^{\mathrm{RJ}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow A_2 \stackrel{\Delta}{\Longleftrightarrow} \eta.\Gamma_1 \Longrightarrow A_2$ *is derivable by the revisited inference for* $\mathrm{Th}_{J\mathrm{LPT}}(T)$, *i.e. the inference rules for* $\mathrm{Th}_{K\mathrm{LPT}}(T)$ *restricted to* JSSeq^{Σ}, *i.e. with* $\Gamma_1' \equiv \emptyset$.

Remark 3.2.10 If $T \vdash^{\mathrm{RK}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, then Γ_2 is nonempty, because in all inference rules the succedent of the conclusion is nonempty.

The set of inference rules ($\mathrm{R}T_{pt}$) seems quite complicated, but if one considers specific examples of T, then the rules in ($\mathrm{R}T_{pt}$) become more familiar. For instance, if T is $\{(\forall z.xz\downarrow\to xz \leq yz) \Longrightarrow x \leq y\}$, i.e. the axiom of extensionality for monotonic partial functions, then ($\mathrm{R}T_{pt}$) includes the following rule

$$\frac{\begin{array}{l}\{\eta.\Gamma \Longrightarrow t_i\downarrow, \Gamma' | i = 1,2\} \\ \eta\lfloor z.\Gamma, t_1 z\downarrow \Longrightarrow t_1 z \leq t_2 z, \Gamma'\end{array}}{\eta.\Gamma \Longrightarrow t_1 \leq t_2, \Gamma'}$$

Moreover, $(\to\Longrightarrow)$ and ($\mathrm{R}T_{pt}$) can be explained as a combination of the more elementary inference rules for LPT given in Definition 2.4.1. This will be done through an example.

Example 3.2.11 Let us consider the instance of the inference rule ($\mathrm{R}T_{pt}$) with $\eta_1 = \emptyset$ and $\Gamma_1 \equiv \Gamma_1' \equiv \emptyset$. To justify it we derive the conclusion $A_2'[\sigma]$ from the premisses and the axiom $\eta_2.\Gamma_2 \Longrightarrow A_2'$ by the inference rules of KLPT (see Definition 2.4.1).

- since $\sigma(x)\downarrow$ for all $x \in \mathrm{dom}(\eta_2)$ (this is the first set of premisses in ($\mathrm{R}T_{pt}$)), we can repeatedly apply (subst) and derive the σ-substitution instance $\Gamma_2[\sigma] \Longrightarrow A_2'[\sigma]$ of the axiom in T_{pt}.

- we derive A for all (closed) formulae in the antecedent $\Gamma_2[\sigma]$ and by repeatedly applying (cut) we derive the required formula.

We still have to show how to derive $A \in \Gamma_2[\sigma]$ from the second set of premisses in (RT_{pt}). Since Γ_2 is a set of s-formulae, A is the σ-substitution instance of an s-formula $(\eta_3.\Gamma_3 \to A'_3)$ in Γ_2. Therefore, we want to derive the s-formula $\eta_3.\Gamma_3[\sigma] \to A'_3[\sigma]$:

- by the second set of premisses in (RT_{pt})

  $\eta_3.\Gamma_3[\sigma] \Longrightarrow A'_3[\sigma]$

- by repeatedly applying $(\wedge \Longrightarrow)$

  $\eta_3. \wedge (\Gamma_3[\sigma]) \Longrightarrow A'_3[\sigma]$

- by $(\Longrightarrow\to)$

  $\eta_3. \wedge (\Gamma_3[\sigma]) \to A'_3[\sigma]$

- by repeatedly applying $(\Longrightarrow \forall)$

  $\forall\eta_3. \wedge (\Gamma_3[\sigma]) \to A'_3[\sigma]$, which is the $\mathrm{I_s}$-translation of s-formula

  $\eta_3.\Gamma_3[\sigma] \to A'_3[\sigma]$

The next result is a particular instance of Proposition 1.2.10.

Theorem 3.2.12 Closure w.r.t. Derivation

Let $T \subseteq \mathrm{JSSeq}^{\Sigma}$ and F be a T-closed fragment of SSeq^{Σ}.
If $T \vdash^{\mathrm{R}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$, and $\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in F$, then any s-sequent in any derivation of $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is in F.

Proof The statement is proof by induction on the derivation of $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ by the revisited inference rules (restricted to JSSeq^{Σ}) of Definition 3.2.9. ∎

3.2.3 Correctness and completeness

We claim that $\vdash^{\mathrm{R}}_{\Sigma}$ and $\vdash_{\Sigma}$ are the same relation on JSSeq^{Σ}.

- The inclusion $\vdash^{\mathrm{R}}_{\Sigma} \subseteq \vdash_{\Sigma}$ means that the revisited inference rules are **correct** for deriving consequences of T in $\vdash_{\Sigma}$.

- The other inclusion, $\vdash_\Sigma \subseteq \vdash^{\mathrm{R}}_\Sigma$, means that the revisited inference rules are also **complete**, i.e. they can derive all consequences of T in $\vdash_\Sigma$.

Theorem 3.2.13 Correctness
If $T \subseteq \mathrm{JSSeq}^\Sigma$ *and* $T \vdash^{\mathrm{R}}_\Sigma \eta.\Gamma_1 \Longrightarrow A_2$, *then* $T \vdash_\Sigma \eta.\Gamma_1 \Longrightarrow A_2$.

Proof The $\mathrm{I_s}$-translations (see Remark 3.2.4) of the revisited inference rules for $\mathrm{Th}_{\mathrm{LPT}}(T)$ are derivable from the inference rules for LPT and the set of axioms $\mathrm{I_s}(T)$. ∎

The proof of completeness is based on a term model construction similar to the one used in Section 2.6 to prove completeness of LPT.

Because of completeness of KLPT, the inclusion $\vdash^{\mathrm{K}} \subseteq \vdash^{\mathrm{RK}}$ amounts to proving that: for all signatures Σ, $T \subseteq \mathrm{JSSeq}^\Sigma$ and $\eta.\Gamma_1 \Longrightarrow A_2 \in \mathrm{JSSeq}^\Sigma$, if $T \not\vdash^{\mathrm{RK}}_\Sigma \eta.\Gamma_1 \Longrightarrow A_2$, then there is a Σ-structure $\mathcal{A}$ which is a model of T but does not satisfy $\eta.\Gamma_1 \Longrightarrow A_2$.

- By the *Deduction Lemma* (see Lemma 3.2.16), this is equivalent to: for all signatures Σ, $T \subseteq \mathrm{JSSeq}^\Sigma$ and $A \in \mathrm{AForm}^\Sigma_0$, if $T \not\vdash^{\mathrm{RK}}_\Sigma \emptyset.A$, then there is a Σ-structure $\mathcal{A}$ which is a model of T but does not satisfy any of the formulae in A

- Then we introduce the auxiliary notion of classically *T-s-complete* set and prove the *Separation Lemma* (see Lemma 3.2.19): if $T \not\vdash^{\mathrm{RK}}_\Sigma \emptyset.A$, then there is an intuitionistic T-s-complete set (on an extension of Σ), which does not contain the atomic formula A

  The definition of classical T-s-complete set differs from that of classical T-complete set, because the conditions for existential formulae and disjunctions are replaced by a condition for s-formulae. This difference affects also the proof of the Separation Lemma.

- Finally, the *Characterization* of the term model $\mathcal{A}_{(\Sigma,H)}$ corresponding to a classically T-s-complete set H on Σ (see Proposition 3.2.21) provides the required model.

  The definition of $\mathcal{A}_{(\Sigma,H)}$ and (the proof of) its characterization are as in Section 2.6.

Throughout the proofs we rely on the derivability (in $\vdash^{\mathrm{RK}}$) of the inference rules (log), (thinning), (subst) and (cut). Since they are not among the revisited inference rules, their derivability has to be proved (see Lemma 3.2.14).

The proofs of the Admissibility and Deduction Lemma for $\vdash^{\mathrm{RJ}}$ are essentially the same as the ones for $\vdash^{\mathrm{RK}}$. The proof of the Separation Lemma (see Lemma 3.2.20) is simpler, because the definition of intuitionistically T-s-complete set does not involve any condition for s-formulae. The definition of the Kripke term model $\mathcal{A}_{(\Sigma,T)}$ and its characterization are, *mutatis mutandis*, as in Section 2.6.

Lemma 3.2.14 Admissible Inference Rules

For any $T \subseteq \mathrm{JSSeq}^{\Sigma}$ the following inference rules are admissible in the set generated by the revisited inference rules of Definition 3.2.9

$$\text{log}\quad \eta_1\lfloor\eta_2.\Gamma_1, (\eta_2.\Gamma_2 \to A_2'), \Gamma_2 \Longrightarrow A_2', \Gamma_1'$$

$$\text{thinning}\quad \frac{\eta_1.\Gamma_1 \Longrightarrow \Gamma_1'}{\eta_2\lfloor\eta_1.\Gamma_1, \Gamma_2 \Longrightarrow \Gamma_1', \Gamma_2'}$$

$$\text{subst}\quad \frac{\eta.\Gamma_1 \Longrightarrow t\downarrow_\tau, \Gamma_1' \qquad \eta\lfloor x{:}\,\tau.\Gamma_2 \Longrightarrow \Gamma_2'}{\eta.\Gamma_1, \Gamma_2[x{:}=t] \Longrightarrow \Gamma_1', \Gamma_2'[x{:}=t]}$$

$$\text{cut}\quad \frac{\begin{array}{c}\eta.\Gamma_1, (\eta'.\Gamma \to A') \Longrightarrow \Gamma_1' \\ \eta\lfloor\eta'.\Gamma_2, \Gamma \Longrightarrow A', \Gamma_2'\end{array}}{\eta.\Gamma_1, \Gamma_2 \Longrightarrow \Gamma_1', \Gamma_2'}$$

Proof (log) is proved by induction on the **degree** $\deg(\eta_2.\Gamma_2 \to A_2')$, where

$$\deg(\eta.\Gamma \to A') \triangleq \begin{cases} 0 & \text{if } \Gamma \equiv \emptyset \\ \max\{\deg(A) | A \in \Gamma\} + 1 & \text{otherwise} \end{cases}$$

- By (E.x), for all $x \in \mathrm{dom}(\eta_2)$

  $\eta_1\lfloor\eta_2.\Gamma_1, (\eta_2.\Gamma_2 \to A_2'), \Gamma_2 \Longrightarrow x\downarrow, \Gamma_1'$

- By IH , for all $(\eta_3.\Gamma_3 \to A_3') \in \Gamma_2$
  (since their degree is less than $\deg(\eta_2.\Gamma_2 \to A_2')$)

  $\eta_1\lfloor\eta_2\lfloor\eta_3.\Gamma_1, (\eta_2.\Gamma_2 \to A_2'), \Gamma_2, \Gamma_3 \Longrightarrow A_3', \Gamma_1'$

- By ($\rightarrow\Longrightarrow$) (for the identity substitution)

  $\eta_1\lfloor\eta_2.\Gamma_1, (\eta_2.\Gamma_2 \rightarrow A'_2), \Gamma_2 \Longrightarrow A'_2, \Gamma'_1$

(thinning) is proved by a straightforward induction on the derivation of $\eta_1.\Gamma_1 \Longrightarrow \Gamma'_1$:

- (E.x) $\Longrightarrow$ immediate by (E.x).
- (E.c) $\Longrightarrow$ immediate by (E.c).
- (E.f.i) $\Longrightarrow$ by IH and (E.f.i).

  Let $\eta_1.\Gamma_1 \Longrightarrow f(\bar{t}) \downarrow_\tau, \Gamma'$ be the premiss of the last rule (i.e. (E.f.i)) in the proof of $\eta_1.\Gamma_1 \Longrightarrow \Gamma'_1$.

  - by IH

    $\eta_2\lfloor\eta_1.\Gamma_1, \Gamma_2 \Longrightarrow f(\bar{t}) \downarrow_\tau, \Gamma', \Gamma'_2$
  - by (E.f.i)

    $\eta_2\lfloor\eta_1.\Gamma_1, \Gamma_2 \Longrightarrow \bar{t}_i \downarrow_{\bar{\tau}_i}, \Gamma', \Gamma'_2$
- (E.p.i) $\Longrightarrow$ similar to the case (E.f.i).
- ($\rightarrow\Longrightarrow$) $\Longrightarrow$ similar to the case (E.f.i).
- (R.T_{pt}) $\Longrightarrow$ similar to the case (E.f.i).

(subst) is proved by induction on the derivation of $\eta\lfloor x{:}\,\tau.\Gamma_2 \Longrightarrow \Gamma'_2$. The only case where the proof differs from that for (thinning) is:

$$\text{E.}x \quad \eta\lfloor x{:}\,\tau.\Gamma_2 \Longrightarrow x \downarrow_\tau, \Gamma'$$

when we must derive $\eta.\Gamma_1, \Gamma_2[x{:}=t] \Longrightarrow t \downarrow_\tau, \Gamma'_1, \Gamma'[x{:}=t]$:

- from the first premiss $\eta.\Gamma_1 \Longrightarrow t \downarrow_\tau, \Gamma'_1$ of (subst), by (thinning)

  $\eta.\Gamma_1, \Gamma_2[x{:}=t] \Longrightarrow t \downarrow_\tau, \Gamma'_1, \Gamma'[x{:}=t]$

(cut) is proved by lexicographic induction on the *degree* $\deg(\eta'.\Gamma \rightarrow A')$ of the *cut formula* and the derivation of $\eta.\Gamma_1, (\eta'.\Gamma \rightarrow A') \Longrightarrow \Gamma'_1$. We proceed by case analysis on the last rule in the derivation of $\eta.\Gamma_1, (\eta'.\Gamma \to A') \Longrightarrow \Gamma'_1$ and IH . The only case where the proof differs from that for (thinning) is:

$$\to\Longrightarrow \quad \frac{\begin{array}{l}\{\eta.\Gamma_1, (\eta'.\Gamma \to A') \Longrightarrow \sigma(x)\downarrow, \Gamma' | \\ \qquad\qquad\qquad x \in \mathrm{dom}(\eta')\} \\ \{\eta\lfloor\eta_3.\Gamma_1, (\eta'.\Gamma \to A'), \Gamma_3[\sigma] \Longrightarrow A'_3[\sigma], \Gamma' | \\ \qquad\qquad\qquad (\eta_3.\Gamma_3 \to A'_3) \in \Gamma\}\end{array}}{\eta.\Gamma_1, (\eta'.\Gamma \to A') \Longrightarrow A'[\sigma], \Gamma'} \ \sigma \in \mathrm{Term}^{\Sigma}_{\eta'}(\eta)$$

when we must derive $\eta.\Gamma_1, \Gamma_2 \Longrightarrow A'[\sigma], \Gamma', \Gamma'_2$:

- by IH and (thinning), for all $x \in \mathrm{dom}(\eta')$

  $\eta.\Gamma_1, \Gamma_2 \Longrightarrow \sigma(x)\downarrow, \Gamma', \Gamma'_2$

- from the second premiss $\eta\lfloor\eta'.\Gamma_2, \Gamma \Longrightarrow A', \Gamma'_2$ of (cut), by (thinning)

  $\eta\lfloor\eta'.\Gamma_1, \Gamma_2, \Gamma \Longrightarrow A', \Gamma', \Gamma'_2$

- by repeatedly applying (subst)

  $\eta.\Gamma_1, \Gamma_2, \Gamma[\sigma] \Longrightarrow A'[\sigma], \Gamma', \Gamma'_2$

- If $\deg(\eta'.\Gamma \to A') = 0$, then $\Gamma \equiv \emptyset$ and we have derived the required s-sequent. Otherwise let $n+1 = \deg(\eta'.\Gamma \to A') = n+1$, by IH

  $\eta\lfloor\eta_3.\Gamma_1, \Gamma_2, \Gamma_3[\sigma] \Longrightarrow A'_3[\sigma], \Gamma', \Gamma'_2$ for all $(\eta_3.\Gamma_3 \to A'_3) \in \Gamma$

- since $\deg(\eta_3.\Gamma_3[\sigma] \to A'_3[\sigma]) \leq n$ for all $(\eta_3.\Gamma_3 \to A'_3) \in \Gamma$, then by IH we can repeatedly apply (cut) with *cut formula* in $\Gamma[\sigma]$

  $\eta.\Gamma_1, \Gamma_2 \Longrightarrow A'[\sigma], \Gamma', \Gamma'_2$

∎

Remark 3.2.15 For the set generated by the revisited inference rules restricted to JSSeq^{Σ} only the restricted versions of the inference rules (log), (thinning), (subst) and (cut) are admissible.

The admissibility of (cut) is proved by the same kind of induction used by Gentzen to prove cut-elimination (see [Gen69, Kle52]).

Lemma 3.2.16 Deduction Lemma

Given a signature Σ and $T \subseteq \mathrm{JSSeq}^{\Sigma}$:

- *If* $\eta.\Gamma_1 \Longrightarrow \Gamma'_1 \in \mathrm{SSeq}^\Sigma$ *and* $A \in \mathrm{SForm}_0^\Sigma$, *then*

  $T \vdash^{\mathrm{R}}_{\Sigma} \eta_1.A, \Gamma_1 \Longrightarrow \Gamma'_1$ *iff* $T, A \vdash^{\mathrm{R}}_{\Sigma} \eta_1.\Gamma_1 \Longrightarrow \Gamma'_1$

- *If* $x \notin \mathrm{dom}(\eta)$, $x\colon\tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2 \in \mathrm{SSeq}^\Sigma$ *and* c *is a new constant of sort* τ *for* Σ, *then*

  $T \vdash^{\mathrm{R}}_{\Sigma} x\colon\tau\lfloor\eta.\Gamma_1 \Longrightarrow \Gamma_2$ *iff* $T \vdash^{\mathrm{R}}_{\Sigma \uplus c} \eta.\Gamma_1[x\colon= c] \Longrightarrow \Gamma_2[x\colon= c]$

Proof The proof is similar to that of Lemma 2.6.2. ∎

Definition 3.2.17 s-Complete Set
Let Σ *be a signature and* $T \subseteq \mathrm{JSSeq}^\Sigma$.

- *a set* $H \subseteq \mathrm{SForm}_0^\Sigma$ *is* **classically** T**-s-complete** *on* Σ *iff*

  1. *for all* $(\eta.\Gamma \to A) \in \mathrm{SForm}_0^\Sigma$ *if* $T, H \vdash^{\mathrm{RK}}_{\Sigma} \eta.\Gamma \Longrightarrow A$, *then* $(\eta.\Gamma \to A) \in H$
  2. *for all* $(\eta.\Gamma \to A) \in \mathrm{SForm}_0^\Sigma$ *if* $(\eta.\Gamma \to A) \notin H$, *then there exists* $\sigma \in \mathrm{Term}_\eta^\Sigma(\emptyset)$ *s.t.* $\sigma \downarrow_\eta \subseteq_{fin} H$ *and* $\Gamma[\sigma] \subseteq_{fin} H$, *but* $A[\sigma] \notin H$

- *a set* $H \subseteq \mathrm{SForm}_0^\Sigma$ *is* **intuitionistically** T**-s-complete** *on* Σ *iff*

  1. *for all* $(\eta.\Gamma \to A) \in \mathrm{SForm}_0^\Sigma$ *if* $T, H \vdash^{\mathrm{RJ}}_{\Sigma} \eta.\Gamma \Longrightarrow A$, *then* $(\eta.\Gamma \to A) \in H$

Remark 3.2.18 The second condition for classically T-s-complete sets is the counterpart of the third and fourth condition in the definition of classically complete set (see Definition 2.6.3). In fact, in classical logic $\neg(\forall x\colon\tau.A_1 \to A_2)$ is equivalent to $(\exists x\colon\tau.\neg A_1 \vee A_2)$. Since this equivalence is not provable in intuitionistic logic one should not expect to have a second condition in the definition of intuitionistically T-s-complete set.

Lemma 3.2.19 Classical Separation Lemma

Given a signature Σ, *a set* C *of new constant symbols for* Σ, *with* $\aleph_0$ *symbols for each sort, and* $T \subseteq \mathrm{JSSeq}^\Sigma$, *then for all* $\Gamma' \subseteq_{fin} \mathrm{AForm}_0^\Sigma$ *s.t.* $T \not\vdash^{\mathrm{RK}}_{\Sigma} \emptyset.\Gamma'$ *there is a signature* Σ_ω, *with* $\Sigma \subseteq \Sigma_\omega \subseteq \Sigma \uplus C$, *and a classically* T*-s-complete set* H *on* Σ_ω *s.t.* $A \notin H$ *for all* A *in* Γ'

Proof Let A_n be an enumeration of $\mathrm{SForm}_0^{\Sigma \uplus C}$, we define, by induction on n, an increasing sequence $(\Sigma_n, H_n, \Gamma_n)$ (where $\Sigma_n \subseteq \Sigma \uplus C$ is a finite extension of Σ, $H_n \subseteq_{fin} \mathrm{SForm}_0^{\Sigma_n}$ and $\Gamma_n \subseteq_{fin} \mathrm{AForm}_0^{\Sigma_n}$):

- basic step
  $H_0 \equiv \emptyset$
  $\Gamma_0 \equiv \Gamma'$
  $\Sigma_0 \equiv \Sigma$

- inductive step
  Let C_n be the set of new constant symbols for Σ_n occurring in $A_n \equiv (\eta.\Gamma \to A)$, there are two cases to be considered:

  - if $T \not\vdash^{\mathrm{RK}}_{\Sigma_n \uplus C_n} \emptyset.H_n, A_n \Longrightarrow \Gamma_n$, then
    $H_{n+1} \equiv H_n, A_n$
    $\Gamma_{n+1} \equiv \Gamma_n$
    $\Sigma_{n+1} \equiv \Sigma_n \uplus C_n$

  - else
    $H_{n+1} \equiv H_n, \Gamma[\gamma]$
    $\Gamma_{n+1} \equiv \Gamma_n, A[\gamma]$
    $\Sigma_{n+1} \equiv \Sigma_n \uplus C_n \uplus \mathrm{cod}(\gamma)$
    where γ is a η-substitution which assigns to each $x \in \mathrm{dom}(\eta)$ a different new constant symbol for $\Sigma_n \uplus C_n$

The rest of the proof will establish that $\Sigma_\omega \triangleq \cup_{n\in\omega}\Sigma_n$ and $H \triangleq \cup_{n\in\omega}H_n$ satisfy the statement of the lemma. For each of the two properties in the definition of classically T-s-complete set we prove a claim that essentially amounts to showing that H has that property.

Claim 3.2.19.1 $T \not\vdash^{\mathrm{RK}}_{\Sigma_n} \emptyset.H_n \Longrightarrow \Gamma_n$

Proof By induction on n. The only non trivial case (in the inductive step) is the second one. Let $A_n \equiv (\eta.\Gamma \to A)$, we have to show that if $T \vdash_{\Sigma_n \uplus C_n} \emptyset.H_n, (\eta.\Gamma \to A) \Longrightarrow \Gamma_n$, then

$$T \not\vdash_{\Sigma_n \uplus C_n \uplus \mathrm{cod}(\gamma)} \emptyset.H_n, \Gamma[\gamma] \Longrightarrow A[\gamma], \Gamma_n$$

- Assume that

  $T \vdash_{\Sigma_n \uplus C_n \uplus \mathrm{cod}(\gamma)} \emptyset.H_n, \Gamma[\gamma] \Longrightarrow A[\gamma], \Gamma_n$

- By the Deduction Lemma

  $T \vdash_{\Sigma_n \uplus C_n} \eta.H_n, \Gamma \Longrightarrow A, \Gamma_n$

- By the assumption and the inference rule (cut)

  $T \vdash_{\Sigma_n \uplus C_n} \emptyset.H_n \Longrightarrow \Gamma_n$

  which contradicts the IH

∎

From the previous claim it follows immediately that $A \notin H$ for all A in Γ'.

Claim 3.2.19.2 *Let* $(\eta.\Gamma \to A) \equiv A_n$*. If* $T, H \vdash^{\mathrm{RK}}_{\Sigma_\omega} \eta.\Gamma \Longrightarrow A$*, then* $T \not\vdash^{\mathrm{RK}}_{\Sigma_n \uplus C_n} \emptyset.H_n, A_n \Longrightarrow \Gamma_n$

Proof By compactness of $\vdash^{\mathrm{RK}}$ there exists $m \geq n$ s.t. $T, H_m \vdash^{\mathrm{RK}}_{\Sigma_m} A_n$ (and therefore $\Sigma_n \uplus C_n \subseteq \Sigma_m$). By the Deduction Lemma, $T \vdash^{\mathrm{RK}}_{\Sigma_m} \emptyset.H_m \Longrightarrow A_n$.

- Assume that

  $T \vdash^{\mathrm{RK}}_{\Sigma_n \uplus C_n} \emptyset.H_n, A_n \Longrightarrow \Gamma_n$

- since $\Sigma_n \uplus C_n \subseteq \Sigma_m$, by compactness of $\vdash^{\mathrm{RK}}$

  $T \vdash^{\mathrm{RK}}_{\Sigma_m} \emptyset.H_n, A_n \Longrightarrow \Gamma_n$

- since $H_n \subseteq H_m$ and $\Gamma_n \subseteq \Gamma_m$, by (thinning)

  $T \vdash^{\mathrm{RK}}_{\Sigma_m} \emptyset.H_m, A_n \Longrightarrow \Gamma_m$

- By the inference rule (cut)

  $T, H_m \vdash^{\mathrm{RK}}_{\Sigma_m} \emptyset.H_n \Longrightarrow \Gamma_m$

  which contradicts the first claim

∎

Claim 3.2.19.3 *If* $(\eta.\Gamma \to A) \equiv A_n \notin H$*, then there exists* $\gamma \in \mathrm{Term}^{\Sigma_\omega}_\eta(\emptyset)$ *mapping variables to constants s.t.* $\Gamma[\gamma] \subseteq_{fin} H$*, but* $A[\gamma] \notin H$*.*

Proof Since $A_n \notin H$, then $T \vdash^{\mathrm{RK}}_{\Sigma_n \uplus C_n} \emptyset.H_n, A_n \Longrightarrow \Gamma_n$. Therefore, by construction there exists a η-substitution γ mapping variables to constants s.t.

- $\Gamma[\gamma] \subseteq_{fin} H$
- $A[\gamma] \in \Gamma_{n+1}$
- $\gamma \in \mathrm{Term}_\eta^{\Sigma_\omega}(\emptyset)$

We still have to show that $A[\gamma] \notin H$ or equivalently $A[\gamma] \notin H_m$ for all $m > n$:

- assume that for an $m > n$

  $A[\gamma] \in H_m$

- since $A[\gamma] \in \Gamma_{n+1} \subseteq \Gamma_m$, by (log)

  $T \vdash^{\mathrm{RK}}_{\Sigma_m} \emptyset.H_m \Longrightarrow \Gamma_m$

  which contradicts the first claim

∎

∎

Lemma 3.2.20 Intuitionistic Separation Lemma

Given a signature Σ, a set C of new constant symbols for Σ, with $\aleph_0$ symbols for each sort, and $T \subseteq \mathrm{JSSeq}^\Sigma$, then for all $A' \in \mathrm{AForm}_0^\Sigma$ s.t. $T \not\vdash^{\mathrm{RJ}}_{\Sigma} \emptyset.A'$ there is a signature Σ_ω, with $\Sigma \subseteq \Sigma_\omega \subseteq \Sigma \uplus C$, and an intuitionistically T-s-complete set H on Σ_ω s.t. $A' \notin H$.

Proof Let A_n be an enumeration of $\mathrm{SForm}_0^{\Sigma \uplus C}$, we define, by induction on n, an increasing sequence (Σ_n, H_n) (where $\Sigma_n \subseteq \Sigma \uplus C$ is a finite extension of Σ and $H_n \subseteq_{fin} \mathrm{SForm}_0^{\Sigma_n}$):

- basic step

  $H_0 \equiv \emptyset$

  $\Sigma_0 \equiv \Sigma$

- inductive step

  Let C_n be the set of new constant symbols for Σ_n occurring in $A_n \equiv (\eta.\Gamma \rightarrow A)$, there are two cases to be considered:

– if $T \not\vdash^{\mathrm{RJ}}_{\Sigma_n \uplus C_n} \emptyset.H_n, A_n \Longrightarrow A'$, then

$H_{n+1} \equiv H_n, A_n$

$\Sigma_{n+1} \equiv \Sigma_n \uplus C_n$

– else

$H_{n+1} \equiv H_n$

$\Sigma_{n+1} \equiv \Sigma_n$

For intuitionistically T-s-complete sets there is only one property to check, and we need only the analogue of the first two claims in Lemma 3.2.19 (with Γ_n replaced by A'). The first claim is immediate, while the second claim is proved as in Lemma 3.2.19. ∎

From now on we proceed as in Section 2.6. The term models $\mathcal{A}_{(\Sigma,H)}$ and $\mathcal{B}_{(\Sigma,T)}$ (below) are as in Definitions 2.6.7 and 2.6.10, but with complete sets replaced by s-complete sets.

Proposition 3.2.21 Characterization of $\mathcal{A}_{(\Sigma,H)}$

Let $T \subseteq \mathrm{JSSeq}^{\Sigma}$, H be a classically T-s-complete set on Σ and $\mathcal{A} \equiv \mathcal{A}_{(\Sigma,H)}$. Then, for all type environments η, environments $\rho \in \eta^{\mathcal{A}}$ and substitutions $\sigma \in \prod \rho$:

- *If $A \in \mathrm{SForm}^{\Sigma}(\eta)$, then $\mathcal{A}, \rho \models A \Longleftrightarrow A[\sigma] \in H$*

Moreover, $\mathcal{A} \models T$.

Proof See proof of Proposition 2.6.8. ∎

Proposition 3.2.22 Characterization of $\mathcal{B}_{(\Sigma,T)}$

Let $T \subseteq \mathrm{JSSeq}^{\Sigma}$, $\mathrm{K} \equiv \mathrm{K}_{(\Sigma,T)}$ and $\mathcal{B} \equiv \mathcal{B}_{(\Sigma,T)}$. Then, for all type environments η, $\alpha \equiv (\Sigma', H') \in \mathrm{K}$, environments $\rho \in \eta^{\mathcal{B}(\alpha)}$ and substitutions $\sigma \in \prod \rho$:

- *If $A \in \mathrm{SForm}^{\Sigma}(\eta)$ then $\mathcal{B}, \alpha, \rho \Vdash A \Longleftrightarrow A[\sigma] \in H'$*

Moreover, $\mathcal{B} \Vdash \eta.\Gamma_1 \Longrightarrow A_2$ iff $T \vdash^{\mathrm{RJ}}_{\Sigma} \eta.\Gamma_1 \Longrightarrow \Gamma_2$

Proof See proof of Proposition 2.6.11. ∎

3.2.4 Discussion

It is possible to make some simple improvements to Theorem 3.2.12, e.g.

- by allowing $\perp$ as an atomic formula, so that s-formulae have the same expressive power as **negative formulae** ($\mathrm{Frag}^{\Sigma}(\wedge\forall \to \perp)$, see 1.10.6 of [Tro73])
- by *sharpening* the definition of T-closed set.

It would be more interesting to extend the revisited inference rules well beyond the negative fragment, possibly to the entire language of LPT. An idea in this direction is to use a more complex compound constructor

$$(\eta.\Gamma \to \{\eta_i.A_i | 1 \leq i \leq n\})$$

whose intended meaning is

$$(\forall \eta. \wedge(\Gamma) \to \vee\{\exists \eta_i.A_i | 1 \leq i \leq n\})$$

Then, any formula is equivalent (in JLPT) to one built up from atomic formulae by this compound constructor, but on the r.h.s. of $\to$ we must allow non atomic formulae.

3.3 LPE and its relation with LPT

In this section we compare the logic of partial terms (LPT) with the **logic of partial elements** (LPE) and set the framework for relating λ_{p}- and λ_{v}-calculus (see Section 4.4). En passant we mention also the **logic of total terms** (LTT, a *different* name for the *familiar* first order logic), give revisited inference rules for LTT and LPE (similar to those for LPT) and state results about coincidence of intuitionistic and classical logic in LTT and LPE.

For simplicity we consider only **single sorted inhabited** logics and give more emphasis to classical logics, but the results of this section hold, mutatis mutandis, in more general settings. We fix three signatures (one for each logic): Σ_{tt}, Σ_{pe} and Σ_{pt}.

Definition 3.3.1 Conventions for signatures

$$\begin{array}{ccc} \Sigma_{tt} & \subset & \Sigma_{pt} \\ & & \\ & & \Sigma_{pe} \end{array}$$

- *Σ is a signature with one sort ι, one constant c, one unary function f and one unary predicate p*
- *Σ_{tt} is Σ plus equality $\_ = \_$*
- *Σ_{pe} is Σ plus* **existence** $\mathrm{E}(\_)$ *and* **equivalence** $\_ \equiv \_$ *for* partial elements
- *Σ_{pt} is Σ_{tt} plus existence $\_ \downarrow$ for* partial terms

Remark 3.3.2 The language for **single sorted inhabited logics** is *simplified* as follows (compare with Definition 2.1.5):

- We remove sort symbols, because there is only one sort.
- In a sequent $\eta.\Gamma \Longrightarrow \Gamma'$ we *could* remove the type assignment η, because there is only one sort and it is inhabited. However, in some inference rules we would have to add side conditions to ensure that *certain* variables are not free in the conclusion.

  For clarity we prefer to keep the type environments and force inhabitation by having a constant in the signature.

3.3.1 The logic of total terms

LTT can be defined as LPT + tot (see Definition 2.4.5) restricted to the set of sequents without $\_\downarrow$. Clearly LPT + tot is a definitional extension of LTT, since the *term existence statements $t\downarrow$ convey no information* (i.e. they are *tautologies* of LPT + tot). However, we want to go further and give a *procedure* to derive Gentzen's inference rules for LTT (see [Gen69, Kle52]) from those for LPT. Then, we can apply the same procedure to the revisited inference rules for LPT and get much simpler revisited inference rules for LTT (see Definition 3.2.9).

Definition 3.3.3 The translation from LPT + tot onto LTT

The translation $\_^{\mathrm{T}}$ from the language over Σ_{pt} to the language over Σ_{tt} is:

- *$(t\downarrow)^{\mathrm{T}}$ is $\top$ and A^{T} is A for the other atomic formulae*
- *$\_^{\mathrm{T}}$ commutes with logical constants, connectives and quantifiers*
- $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)^{\mathrm{T}} = \begin{cases} \{\eta.erase(\Gamma_1)^{\mathrm{T}} \Longrightarrow {\Gamma_2}^{\mathrm{T}}\} & \text{if } \Gamma_2 = erase(\Gamma_2) \\ \emptyset & \text{otherwise} \end{cases}$

  where $erase(\Gamma)$ is $\Gamma - \{t\downarrow \mid t \in \mathrm{Term}\}$.

Remark 3.3.4 $\_^{\mathrm{T}}$ satisfies the conditions of Proposition 1.2.9 when R is the set of inference rules for $\mathrm{LPT} + \mathrm{tot}$, therefore LTT is generated by the $\_^{\mathrm{T}}$-translation of the inference rules for LPT. Moreover, these inference rules are exactly those for Gentzen's sequent calculus (see [Gen69, Kle52]).

Definition 3.3.5 Revisited Inference rules for LTT

The translation $\_^{\mathrm{T}}$ *is extended to the revisited language as follows:*

- $(\eta.\Gamma \to A)^{\mathrm{T}}$ *is* $\eta.erase(\Gamma)^{\mathrm{T}} \to A^{\mathrm{T}}$

Given a signature Σ *for* LTT *and* $T \subseteq \mathrm{JSSeq}^{\Sigma}$, *let* T_{tt} *be* ${T_{pt}}^{\mathrm{T}}$, *i.e.* T *plus the axioms for* equality *in* LTT. *The inference rules for* $\mathrm{Th}_{KLTT}(T)$ *on* SSeq^{Σ} *are:*

$$\to\Longrightarrow \quad \frac{\begin{array}{r}\{\eta_1\lfloor\eta_3.\Gamma_1, \Gamma_3[\sigma] \Longrightarrow A'_3[\sigma], \Gamma'_1 |\\ (\eta_3.\Gamma_3 \to A'_3) \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A'_2[\sigma], \Gamma'_1} \quad \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1)\\ \eta_2.\Gamma_2 \to A'_2 \in \Gamma_1\end{array}$$

$$\mathrm{R}T_{tt} \quad \frac{\begin{array}{r}\{\eta_1\lfloor\eta_3.\Gamma_1, \Gamma_3[\sigma] \Longrightarrow A'_3[\sigma], \Gamma'_1 |\\ (\eta_3.\Gamma_3 \to A'_3) \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A'_2[\sigma], \Gamma'_1} \quad \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1)\\ \eta_2.\Gamma_2 \Longrightarrow A'_2 \in T_{tt}\end{array}$$

The coincidence results for intuitionistic and classical LTT are similar to those for LPT (see Section 3.1).

Proposition 3.3.6 Coincidence results for LTT

- *Intuitionistic and classical free* LTT *coincide on the fragment* $\{\mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot \wedge \vee\exists)$.
- *Intuitionistic and classical inhabited* LTT *coincide on the fragment* $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\bot)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$.

Proof The results follow immediately from Theorems 3.1.5 and 3.1.15, because LTT is axiomatized in LPT by a set of sequents (tot) included in both fragments.

∎

At the beginning of Chapter 2 we sketched a translation of LPT in LTT, where partial functions are represented by *single-valued* relations. Because of this, the coincidence results for free LTT and free LPT are interderivable. The coincidence result for inhabited LPT cannot be derived from the corresponding result for LTT, because atomic formulae of LPT are translated to formulae in $\mathrm{Frag}^{\Sigma}(\exists\wedge)$. However, the following *improved* coincidences for inhabited Logics are interderivable:

> $InhabitedK$LPT is an extension of $InhabitedJ$LPT conservative over $\mathcal{P}(\{\mathrm{Frag}^{\Sigma}(\perp\wedge\vee\exists\forall)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\exists!\wedge\perp)) \times \mathrm{Frag}^{\Sigma}(\perp\wedge\vee\exists\forall)$, and similarly for LTT.

3.3.2 The logic of partial elements

The logic of partial elements is usually presented as an higher order logic with a topos-theoretic interpretation (see [Fou77]). However, in [Sco79] the logic is given incrementally: existence (and quantifiers), equivalence and equality, functions and relations, descriptions, higher order. By LPE we mean the logic described in the first three sections of [Sco79], but we consider two other *flavours* (see Definition 3.3.7): iLPE and LPE + I. Roughly speaking, the difference among the three flavours is:

- in LPE + I there is exactly one *nonexisting element*
- in LPE there is at most one
- in iLPE there are no restrictions on their number.

iLPE is not considered in the literature, since the topos-theoretic semantics takes (intuitionistic) *indiscernability* of partial elements for granted. But it plays an important role as the logic *behind* the λ_{v}-calculus and makes very clear the connection with the λ_{p}-calculus (see Section 4.4). We describe the logic of partial elements (LPE) in an indirect way.

> First, we translate the language of LPE into that of LTT, via an injective function $\_^{\mathrm{E}}$. Then, we define the *bare* LPE as the pullback of LTT along $\_^{\mathrm{E}}$ (see 1.1.5). Finally, we *axiomatize* LPE in the *bare* LPE.

The translation $\_^{\mathrm{E}}$ *explains* bare LPE in terms of LTT by saying that *existing elements* are a subset of the *partial elements* and quantifiers are *relativized* to existing elements. $\_^{\mathrm{E}}$ gives also a *procedure* to derive the revisited inference rules for LPE from those for LTT.

Definition 3.3.7 The formal systems LPE, iLPE and LPE + I

The translation $\_^{\mathrm{E}}$ *from the language over* Σ_{pe} *into the language over* Σ_{tt} *extended with the unary predicate* E *is:*

- $(t_1 \equiv t_2)^{\mathrm{E}}$ *is* $t_1 = t_2$ *and* A^{E} *is* A *for the other atomic formulae*
- $\_^{\mathrm{E}}$ *commutes with logical constants and connectives*
- $(\forall x.A)^{\mathrm{E}}$ *is* $(\forall x.\mathrm{E}(x) \to A^{\mathrm{E}})$
- $(\exists x.A)^{\mathrm{E}}$ *is* $(\exists x.\mathrm{E}(x) \wedge A^{\mathrm{E}})$
- $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)^{\mathrm{E}}$ *is* $(\eta.{\Gamma_1}^{\mathrm{E}} \Longrightarrow {\Gamma_2}^{\mathrm{E}})$

The **bare** *logic of partial elements (bare* LPE*) is the pullback of* LTT *along the* translation $\_^{\mathrm{E}}$.

The **intensional** *logic of partial elements (*iLPE*) is the formal system axiomatized in bare* LPE *by the following axioms for* existence*:*

E.c $\mathrm{E}(c)$

E.f $\mathrm{E}(f(x)) \Longrightarrow \mathrm{E}(x)$

E.p $p(x) \Longrightarrow \mathrm{E}(x)$

The **logic of partial elements** *(*LPE*) is the formal system axiomatized in* iLPE *by the following axiom for* equivalence*:*

in $(\mathrm{E}(x) \to x \equiv y), (\mathrm{E}(y) \to x \equiv y) \Longrightarrow x \equiv y$

The logic of partial elements with **descriptions** *(*LPE + I*) is defined in* Section 6 of [Sco79].

Remark 3.3.8 On page 681 of [Sco79] there is a slightly different definition of $\_^{\mathrm{E}}$, s.t. $(t_1 \equiv t_2)^{\mathrm{E}}$ is $\mathrm{E}(t_1) \vee \mathrm{E}(t_2) \to t_1 = t_2$. This translation rules out the *intensional* approach, by forcing (in) even in the bare LPE, and makes equivalence and *equality* interdefinable.
The axiom (E.c) has been added to make LPE closer to LPT, but in doing so we can no longer view free variables of LPE (that range over partial elements) as *generic constants*. (E.f) and (E.p) are the axioms (str) for *strictness*. There are equivalent formulations for the axiom (in) of *indiscernability*, but they are not s-formulae (and therefore unsuitable for the revisited inference rules).

To obtain the revisited inference rules for LPE we can simply *pull back* the revisited inference rules for LTT along $\_^{\mathrm{E}}$ (extended to the revisited language).

Definition 3.3.9 Revisited Inference rules for LPE

The translation $\_^{\mathrm{E}}$ *is extended to the revisited language as follows:*

- $(\eta.\Gamma \to A)^{\mathrm{E}}$ *is* $(\eta.\mathrm{E}(\eta), \Gamma^{\mathrm{E}} \to A^{\mathrm{E}})$

  where $\mathrm{E}(\eta)$ *is the set of formulae* $\{\mathrm{E}(x) | x \in \mathrm{dom}(\eta)\}$.

Given a signature Σ *for* LPE *and* $T \subseteq \mathrm{JSSeq}^{\Sigma}$, *let* T_{pe} *be the pullback of* ${T^{\mathrm{E}}}_{tt}$ *along* $\_^{\mathrm{E}}$, *i.e.* T *plus the axioms for* equivalence *in bare* LPE.

The inference rules for bare $\mathrm{Th}_{K\mathrm{LPE}}(T)$ *on* SSeq^{Σ} *are:*

$$\to\Longrightarrow \quad \frac{\begin{array}{r}\{\eta_1.\Gamma_1 \Longrightarrow \mathrm{E}(\sigma(x)), \Gamma_1' | x \in \mathrm{dom}(\eta_2)\} \\ \{\eta_1 \lfloor \eta_3.\mathrm{E}(\eta_3), \Gamma_1, \Gamma_3[\sigma] \Longrightarrow A_3'[\sigma], \Gamma_1' | \\ (\eta_3.\Gamma_3 \to A_3') \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A_2'[\sigma], \Gamma_1'} \quad \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1) \\ \eta_2.\Gamma_2 \to A_2' \in \Gamma_1\end{array}$$

$$\mathrm{R}T_{pe} \quad \frac{\begin{array}{r}\{\eta_1 \lfloor \eta_3.\mathrm{E}(\eta_3), \Gamma_1, \Gamma_3[\sigma] \Longrightarrow A_3'[\sigma], \Gamma_1' | \\ (\eta_3.\Gamma_3 \to A_3') \in \Gamma_2\}\end{array}}{\eta_1.\Gamma_1 \Longrightarrow A_2'[\sigma], \Gamma_1'} \quad \begin{array}{l}\sigma \in \mathrm{Term}^{\Sigma}_{\eta_2}(\eta_1) \\ \eta_2.\Gamma_2 \Longrightarrow A_2' \in T_{pe}\end{array}$$

For $\mathrm{Th}_{K\mathrm{iLPE}}(T)$ *one has to add the inference rules:*

$$\mathrm{E}.c \quad \eta.\Gamma_1 \Longrightarrow \mathrm{E}(c), \Gamma_1'$$

$$\mathrm{E}.f.\mathrm{i} \quad \frac{\eta.\Gamma_1 \Longrightarrow \mathrm{E}(f(\bar{t})), \Gamma_1'}{\eta.\Gamma_1 \Longrightarrow \mathrm{E}(\bar{t}_i), \Gamma_1'} \quad i \in \mathrm{dom}(\bar{t})$$

$$\mathrm{E}.p.\mathrm{i}\ \frac{\eta.\Gamma_1 \Longrightarrow p(\overline{t}),\Gamma'_1}{\eta.\Gamma_1 \Longrightarrow \mathrm{E}(\overline{t}_i),\Gamma'_1}\ i \in \mathrm{dom}(\overline{t})$$

Finally for $\mathrm{Th}_{K\mathrm{LPE}}(T)$ *one has to add also the inference rule:*

$$\mathrm{Rin}\ \frac{\eta_1.\Gamma_1,\mathrm{E}(t_1) \Longrightarrow t_1 \equiv t_2,\Gamma'_1 \qquad \eta_1.\Gamma_1,\mathrm{E}(t_2) \Longrightarrow t_1 \equiv t_2,\Gamma'_1}{\eta_1.\Gamma_1 \Longrightarrow t_1 \equiv t_2,\Gamma'_1}$$

Remark 3.3.10 The set of revisited inference rules for LTT satisfies the conditions of Proposition 1.2.10, when I is $\_^{\mathrm{E}}$, therefore bare $\mathrm{Th}_{K\mathrm{LPE}}(T)$ is generated by the *pullback* of the revisited inference rules for $\mathrm{Th}_{K\mathrm{LTT}}(T^{\mathrm{E}})$ along the translation $\_^{\mathrm{E}}$. We cannot apply Proposition 1.2.10 to pullback the inference rules of LTT, because in the cut rule the cut formula can always be taken outside the image of I. However, we can overcome this problem by using a restricted cut rule s.t. the cut formula is in the image of I: this will still be enough to derive all consequences of a set of axioms in the image of I (see Chapter 5 of [MR77]).

The coincidence results for intuitionistic and classical LPE are *dependent* on the flavour, and only in the case of iLPE they are similar to those for LPT (see Section 3.1).

Proposition 3.3.11 Coincidence results for iLPE

- *Intuitionistic and classical free* iLPE *coincide on the fragment* $\{\mathrm{Frag}^{\Sigma}(\perp \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp \wedge \vee\exists)$.
- *Intuitionistic and classical inhabited* iLPE *coincide on the fragment* $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$.

Proof The coincidence of intuitionistic and classical free bare LPE on the coherent fragment $\{\mathrm{Frag}^{\Sigma}(\perp \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp \wedge \vee\exists)$ follows from the first statement in Proposition 3.3.6. In fact, bare LPE is the pullback of LTT along $\_^{\mathrm{E}}$ (see Definition 3.3.7) and the pullback of the coherent fragment (over Σ_{tt}) along $\_^{\mathrm{E}}$ is the coherent fragment (over Σ_{pe}). Since the axioms of iLPE are in the coherent fragment, the same coincidence result holds for iLPE.

To prove the second statement we cannot use $\_^{\mathrm{E}}$, because the pullback of $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$ along $\_^{\mathrm{E}}$ is a proper subset of $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow$

$\mathrm{Frag}^{\Sigma}(\perp)$. Therefore, we introduce a new conservative interpretation (of bare LPE in a formal system axiomatized in LPT).

Let Σ be the signature Σ_{pt} extended with a sort ι_E and two unary partial functions $in: \iota_E \to \iota$ and $out: \iota \rightharpoonup \iota_E$. We define a translation $\_^{\mathrm{P}}$ from the language over Σ_{pe} into the language over Σ:

- $(\mathrm{E}(t))^{\mathrm{P}}$ is $out(t)\downarrow_{\iota_E}$
- $(t_1 \equiv t_2)^{\mathrm{P}}$ is $t_1 =_\iota t_2$ and A^{P} is A for the other atomic formulae
- $\_^{\mathrm{P}}$ commutes with logical constants and connectives
- $(\forall x.A)^{\mathrm{P}}$ is $(\forall y{:}\,\iota_E.A^{\mathrm{P}}[x:=in(y)])$
- $(\exists x.A)^{\mathrm{P}}$ is $(\exists y{:}\,\iota_E.A^{\mathrm{P}}[x:=in(y)])$
- $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)^{\mathrm{P}}$ is $(\eta.{\Gamma_1}^{\mathrm{P}} \Longrightarrow {\Gamma_2}^{\mathrm{P}})$

This translation, unlike $\_^{\mathrm{E}}$, does *almost* commute with quantifiers, in particular it maps $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$ into $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$.

Let Ax_{pe} be the following set of axioms

in $\quad y{:}\,\iota_E.out(in(y)) = y$

out $\quad x{:}\,\iota.out(x)\downarrow \Longrightarrow in(out(x)) = x$

tot.$f \quad f(x)\downarrow$

It is easy to see, by model-theoretic considerations, that $\_^{\mathrm{P}}$ is a conservative interpretation of bare LPE in LPT$+Ax_{pe}$, i.e. bare LPE is the pullback of LPT$+$ Ax_{pe} along $\_^{\mathrm{P}}$. Then we can proceed as in the proof of the first statement, because intuitionistic and classical inhabited LPT $+ Ax_{pe}$ coincide on $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\perp)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\perp)$ (see Theorem 3.1.15). ∎

The axiom (in) of LPE is outside both fragments, therefore we do not expect to get the coincidence result of iLPE. In fact:

Example 3.3.12 $f(x) \equiv g(x)$ is derivable from $f(\perp) \equiv g(\perp)$, $E(\perp) \Longrightarrow E(x)$ and $E(x) \Longrightarrow f(x) \equiv g(x)$ in KLPE but not in JLPE.

In LPE + I any fragment containing all term-existence statements $\mathrm{E}(t)$ has the same expressive power as the entire language, in fact (Theorem 6.6 of [Sco79]):

$A \longleftrightarrow \mathrm{E}(c{\restriction}A)$ is derivable in LPE + I, for any formula A.

Therefore, it is impossible to have any coincidence result (without imposing restrictions on the terms).

3.3.3 The relation between LPT and LPE

The relation between LPT and LPE can be summarized by saying that there is a conservative interpretation of LPT in all three *flavours* of LPE (see Proposition 3.3.14).

Definition 3.3.13 Translation of LPT in LPE

The translation $\_^\circ$ *from the language over* Σ_{pt} *into the language over* Σ_{pe} *is:*

- $(t_1 = t_2)^\circ$ *is the* **equality** $\mathrm{E}(t_1) \wedge t_1 \equiv t_2$ *of bare* LPE *(shortly* $t_1 = t_2$*) and* A° *is* A *for the other atomic formulae*
- $\_^\circ$ *commutes with logical constants, connectives and quantifiers*
- $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)^\circ$ *is* $(\eta.\mathrm{E}(\eta), A_1{}^\circ \Longrightarrow \Gamma_2{}^\circ)$

Let Σ *be a signature for* LPE*:*

- ESeq^Σ *is the fragment of* Seq^Σ *s.t. the sequent* $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ *is in it iff* $\mathrm{E}(\eta) \subseteq \Gamma_1$ *(i.e. all free variables are required to range over* existing elements*).*
- EqSeq^Σ *is the fragment of* Seq^Σ *built up by using the equality* $=$ *(of bare* LPE*) instead of equivalence* $\equiv$*.*
- EEqSeq^Σ *is the intersection of the fragments* EqSeq^Σ *and* ESeq^Σ*.*

The fragment EEqSeq is actually the image of $\_^\circ$.

Proposition 3.3.14 Embedding of LPT in iLPE

$\_^\circ$ *is a conservative interpretation of* LPT *in* iLPE*.*

Proof It is easy to prove that $\_^\circ$ is a relative interpretation, therefore we have only to prove the implication: if $(\eta.\Gamma_1 \Longrightarrow \Gamma_2)^\circ$ is derivable from T° in iLPE, then $\eta.\Gamma_1 \Longrightarrow \Gamma_2$ is derivable from T in LPT.
By Proposition 1.3.5, it is enough to show that for all models $\mathcal{A}$ of LPT there exists a model $\mathcal{A}'$ of iLPE s.t. $\mathcal{A}$ satisfies a sequent iff $\mathcal{A}'$ satisfies its translation.
We claim that the required $\mathcal{A}'$ is the model $\mathcal{A}_\perp$ obtained by *adding* to $\mathcal{A}$ a *nonexisting* element $\perp$. ∎

Remark 3.3.15 $\mathcal{A}_\perp$ is actually a model of LPE + I (and LPE), therefore the proof of Proposition 3.3.14 shows that $\_^\circ$ is also a conservative interpretation of LPT in LPE + I (and LPE).

Proposition 3.3.16 Embedding among the flavours of LPE

$$\mathrm{iLPE} \quad \subset^c_{\mathrm{EqSeq}\vdash\mathrm{EqSeq}} \quad \mathrm{LPE} \quad \subset^c_{\mathrm{ESeq}\vdash\mathrm{Seq}} \quad \mathrm{LPE} + \mathrm{I}$$

Proof We prove iLPE $\subset^c_{\mathrm{EqSeq}\vdash\mathrm{EqSeq}}$ LPE first. The inclusion iLPE $\subseteq$ LPE is obvious, therefore we have only to prove the other inclusion for the formal systems restricted to EqSeq.
By Proposition 1.3.5, it is enough to show that for all models $\mathcal{A}$ of iLPE there exists a model $\mathcal{A}'$ of LPE s.t. $\mathcal{A}$ and $\mathcal{A}'$ satisfy the same sequents in EqSeq.
We claim that the required $\mathcal{A}'$ is the model $\mathcal{A}/\equiv_e$, i.e. $\mathcal{A}$ *modulo* the *congruence* $a \equiv_e b \overset{\Delta}{\Longleftrightarrow} \mathrm{E}(a)$ or $\mathrm{E}(b)$ implies $a \equiv b$.

To prove LPE $\subset^c_{\mathrm{ESeq}\vdash\mathrm{Seq}}$ LPE + I, it is enough to show that for all models $\mathcal{A}$ of LPE there exists a model $\mathcal{A}'$ of LPE + I s.t.

- $\mathcal{A}'$ satisfies all sequents in ESeq satisfied by $\mathcal{A}$
- $\mathcal{A}$ satisfies all sequents satisfied by $\mathcal{A}'$

We claim that the required $\mathcal{A}'$ is the model $\mathcal{A}\restriction_\perp$ obtained by first *stripping* $\mathcal{A}$ of all *nonexisting* elements and then *adding* a *nonexisting* element $\perp$. ∎

A proof of LPE $\subset^c_{\mathrm{ESeq}\vdash\mathrm{Seq}}$ LPE + I for intuitionistic logic is sketched in 6.4 of [Sco79].

3.3.4 Concluding remarks

By Proposition 3.3.14, we can identify LPT with LPE restricted to the EEqSeq fragment. On this fragment LPE is *uncontroversial*, in the sense that the consequence relation is *flavour independent* (see Proposition 3.3.16). However, outside EEqSeq the differences become apparent, e.g.:

- $\emptyset \vdash (\mathrm{E}(x) \vee \mathrm{E}(y) \to x \equiv y) \Longrightarrow x \equiv y$ in LPE but not in iLPE
- $\mathrm{E}(x) \vdash \perp$ in LPE + I but not in LPE.

Among the three flavours, only LPE seems a suitable alternative to LPT, in fact:

- iLPE does not capture the set-theoretic view of partial functions as single-valued relations, because two strict functions may yield the same result when applied to existing elements, while being different on some *nonexisting elements*.
- from a computer science prospective, intuitionistic LPE + I is unsatisfactory, because there are too many partial elements to *view* all of them as denotation of programs (see [Ros86] for a more flexible approach based on *dominances* for *classifying* only some partial elements).

At these point it is possible to see another (more technical) reason, besides those mentioned in the introduction, for preferring LPT to LPE, namely the absence of coincidence results for LPE.

On the other hand, the notion of partial element (taken as primitive) opens new possibilities, that are not *thinkable* in the framework of LPT. For instance, iLPE is a very natural weakening of LPE, obtained by simply dropping one axiom, but it seems impossible have a similar weakening of LPT.

Chapter 4

Formal systems for applicative structures

In Definition 2.4.5 we have defined, for each signature Σ, eight formal systems on the same language Seq^{Σ}:

$$\begin{array}{ccccc} \mathit{Free} & J & & & 0 \\ \cap & \cap & \mathrm{LPT} & + & \cap \\ \mathit{Inhabited} & K & & & \mathrm{tot} \end{array}$$

All of them are axiomatizable in free intuitionistic logic of partial terms ($\mathit{Free}J$LPT over Σ).

In this chapter we introduce other formal systems, that are either directly axiomatizable in LPT or equivalent (at the level of terms) to a formal system axiomatizable in LPT. These formal systems are defined only for signatures Σ that satisfy certain closure properties. For simplicity we consider only formal systems for inhabited models, but we will mention the modifications required when empty types are allowed. The formal systems are divided into two groups:

- formal systems for partial algebras (that are directly axiomatizable in LPT), i.e. LPT and its variants: LPT + $\upharpoonright$ (LPT with restriction operator) and monLPT (monotonic LPT)

- formal systems for *extensional partial combinatory algebras* (that are directly axiomatizable in the $\lambda_{\mathrm{p}}\beta\eta$-calculus), i.e. the $\lambda_{\mathrm{p}}\beta\eta$-calculus (the partial lambda calculus) and its variants: the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus (the monotonic partial lambda calculus), the $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$-calculus (the fixed-point partial lambda calculus) and the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus (the least fixed-point partial lambda calculus).

We consider also the untyped partial lambda calculus, although there is a *natural* conservative interpretation of it in the typed partial lambda calculus (similar to that given in [Sco80] for the untyped lambda calculus).

In the study of reduction we will consider also the partial lambda calculi without the axioms (β) or (η).

We are mainly interested in the partial lambda calculus, but it is useful to consider partial algebras as well, since they provide a simpler case study, that has been extensively investigated in the literature (see [Bur82]), and that can suggest ways of tackling problems in the partial lambda calculus. We will study expressive power of *equational* fragments and *equational presentation* with this *incremental* approach. Moreover, in order to understand the differences between partial and total lambda calculus, it is important to compare embeddability of partial algebras in *models of the typed partial lambda calculus* versus embeddability of (total) algebras in models of the typed (total) lambda calculus.

The variants of the partial lambda calculus are inspired by the denotational semantics of a functional language in the category of cpos and continuous partial functions (see [Plo85]). Note that cpos cannot be completely described in first order logic, i.e. they are not the class of models of a first order theory, therefore we can only axiomatize some of their properties. The $\lambda_\mathrm{p}\mu\mathrm{Y}\beta\eta$-calculus is the variant closest to the denotational model, but the other variants, namely the mon$\lambda_\mathrm{p}\beta\eta$-calculus and the $\lambda_\mathrm{p}\mathrm{Y}\beta\eta$-calculus, enjoy an alternative characterization in terms of a reduction.

We consider two other formal systems: **p-equational logic** (see [Obt86]) and the **λ_v-calculus** (see [Plo75]). We prove that p-equational logic amounts to $\mathrm{LPT}+\upharpoonright$ restricted to a suitable fragment. We give an account of the λ_v-calculus in terms of the intensional logic of partial elements (see Section 3.3) and stress its connections with the partial lambda calculus.

4.1 Equational fragments and variants of LPT

In the literature on many sorted algebras and the lambda calculus one is usually concerned only with *equational theories*, i.e. sets of equations between terms closed w.r.t. the inference rules. For partial algebras there are different notions of *equation* and *term existence statements* become of interest as well. These notions of equation can be described in the language of LPT and their common feature is that they *specialize* to equations in the case of total algebras (see [Bur82]). In this section we compare the expressive power of these fragments in LPT. The overall picture is quite confusing, but we will argue that this is due to a *lack of terms*. In fact, if we add a binary *restriction operator* to LPT, then there are only two equational fragments with different expressive power. In the $\lambda_{\mathrm{p}}\beta$-calculus the picture is even simpler, since all equational fragments have the same expressive power (see Theorem 4.2.8).

Definition 4.1.1 Term Existence and Equational Fragments

- **t***erm* **e***xistence statements (TE-statements)* $t \downarrow$
- **e***xistentially* **c***onditioned* **TE***-statements (ECTE-statements)*
  $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \downarrow$
- **e***xistence equations (E-equations)* $t_1 = t_2$
- **s***trong equations (S-equations)* $t_1 \simeq t_2$*, i.e.*
  $(t_1 \downarrow \rightarrow t_1 = t_2) \wedge (t_2 \downarrow \rightarrow t_1 = t_2)$
- **e***xistentially* **c***onditioned* **E***-equations (ECE-equations)*
  $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t = t'$
- **e***xistentially* **c***onditioned* **S***-equations (ECS-equations)*
  $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \simeq t'$

There are six *term existence* and *equational* fragments with different expressive power (in LPT). The clear conclusion that one can draw by comparing them is that ECE-equations are the most expressive, although for stating equivalence or termination of programs, S-equations and TE-statements are more natural.

Theorem 4.1.2 Expressive power in LPT

$$\begin{array}{ccccc} TE & \subset & E & & \\ & & \cap & & \\ S & \subset & TE \cup S & & \\ \cap & & \cap & & \\ ECS & \subset & ECE & \equiv & ECTE \cup ECS \end{array}$$

Proof the inclusions are quite easy to prove:

- $TE \subseteq E$, because $t \downarrow$ is equivalent to the E-equation $t = t$
- $E \subseteq TE \cup S$, because $t_1 = t_2$ is equivalent to the conjunction $t_1 \downarrow \wedge t_1 \simeq t_2$
- $ECS \subseteq ECE$, because $(A \rightarrow t \simeq t')$ is equivalent to the conjunction $(A \wedge t \downarrow \rightarrow t = t') \wedge (A \wedge t' \downarrow \rightarrow t = t')$

To show that no other inclusion or equivalence (which is not derivable by transitivity from the ones indicated) can be added to the picture, it is enough to show that $TE \not\subseteq ECS$ and $S \not\subseteq E$.

To show that $TE \not\subseteq ECS$, we have to find a formula $A \in TE$ which is not derivable from $\{A' \in ECS | A \vdash A'\} \not\vdash A$ (this is a general method). Consider the signature Σ whose only symbol is a unary function f. We prove that the formula $f(x) \downarrow$ has the required property. In fact, for any ECS-equation $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \simeq t'$

- $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \simeq t'$ is derivable from $f(x) \downarrow$ implies
- $t \simeq t'$ is valid in the total free Σ-algebra implies
- t and t' are syntactically equal implies
- $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \simeq t'$ is derivable in LPT

Therefore, $f(x) \downarrow$ is not derivable from $\{A' \in ECS | f(x) \downarrow \vdash A'\}$, because it is not a theorem of LPT.

Similarly, for $S \not\subseteq E$ we can show that the formula $f(x) \simeq f(y)$ has the required property. In fact, the only E-equations derivable from it are of the form $x = x$, i.e. the E-equations valid in the partial free Σ-algebra (where f is the everywhere diverging function) ∎

4.1.1 LPT with restriction operator

In a formal system with more deductive power than LPT some of the fragments may become equivalent, for instance in LPT + tot all equational fragments have the same expressive power. However, if we just add more axioms to LPT the resulting formal system is in general no longer sound for partial algebras. Instead, the general idea we follow is to consider a definitional extension of LPT over Σ axiomatized, say by T, in LPT over a signature Σ' s.t. $\Sigma \subseteq \Sigma'$. This is equivalent to saying that any Σ-structure can be extended in a *unique* way to a model of T. Therefore, the new symbols in Σ' have a *standard* interpretation in a model of T, like $\_\downarrow$ and $\_ = \_$ in a Σ-structure, and the axioms T capture formally this standard interpretation, like the axioms for existence and equality.

In this section we consider a definitional extension of LPT obtained by adding a binary *restriction operator* (used also in [CO87]) and reexamine the expressive power of the equational fragments.

Definition 4.1.3 LPT + $\restriction$

The **logic of partial terms with restriction operator** *(*LPT + $\restriction$*) over* $\Sigma_\restriction$ *is the formal system axiomatized by* $\restriction$ *in* LPT *over* $\Sigma_\restriction$*, where*

- $\Sigma_\restriction$ *is a signature for* partial algebras **with restriction**, *i.e. a signature for* partial algebras *with a* **restriction operator** $\restriction_{\tau_1,\tau_2} \in \mathrm{Funct}_{(\tau_1,\tau_2)\to\tau_1}(\Sigma_\restriction)$ *for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma_\restriction)$
- $\restriction$ *is the axiom*

$$\restriction \quad x_1 \restriction x_2 = x_1$$

Notation 4.1.4 We write $t_1 \restriction t_2$ for $\restriction(t_1, t_2)$ and use the convention of *associating* the restriction operator to the left.

Moreover, if D is a finite set of terms, then the expression $t \restriction D$ will be treated as a shorthand for $t \restriction t_1 \restriction \ldots \restriction t_n$, where $t_1, \ldots, t_n$ is a list of the elements in D (possibly with repetitions). The vagueness in the definition of $t \restriction D$ does not cause problems, because the terms $t \restriction t_1 \restriction \ldots \restriction t_n$ and $t \restriction t'_1 \restriction \ldots \restriction t'_m$ are provably equivalent in LPT + $\restriction$ provided $\{t_1, \ldots, t_n\} = \{t'_1, \ldots, t'_m\}$.

Remark 4.1.5 In LPE the restriction operator may actually change the class of models, because it forces them to have certain partial elements (like the description operator does). The restriction operator $t \restriction A$ considered in Definition 6.5 of [Sco79] (where A is a formula) is more expressive than that considered here, and some partial elements definable by it cannot be regarded as *programs* (see Section 3.3 for similar considerations about LPE + I).

Throughout this section we fix a signature $\Sigma_\restriction$ for partial algebras with restriction and Σ will be $\Sigma_\restriction$ without restriction operators.

Proposition 4.1.6 *For any Σ-structure $\mathcal{A}$ there exists a unique model $\mathcal{A}'$ of* LPT + $\upharpoonright$ *over* $\Sigma_\upharpoonright$ *s.t.* $\mathcal{A} = \mathcal{A}' \upharpoonright \Sigma$

Proof To get a model of LPT + $\upharpoonright$ the interpretation of the restriction operator $\upharpoonright_{\tau_1,\tau_2}$ has to be the first projection from $\tau_1^{\mathcal{A}} \times \tau_2^{\mathcal{A}}$ to $\tau_1^{\mathcal{A}}$ ∎

Theorem 4.1.7 Expressive power in LPT + $\upharpoonright$

$$\begin{array}{ccccc} TE & \subset & E & & \\ & & \cap & & \\ & & S & \equiv & ECE \end{array}$$

Proof By Theorem 4.1.2 $S \subseteq ECE \equiv ECTE \cup ECS$, therefore to prove $S \equiv ECE$ it is enough to show that $ECTE \cup ECS \subseteq S$:

- $ECTE \subseteq S$, because $\wedge\{t_1 \downarrow, \ldots, t_n \downarrow\} \to t \downarrow$ is equivalent to the S-equation $x \upharpoonright \{t_1, \ldots, t_n\} \simeq x \upharpoonright \{t_1, \ldots, t_n, t\}$
- $ECS \subseteq S$, because $\wedge\{t_1 \downarrow, \ldots, t_n \downarrow\} \to t \simeq t'$ is equivalent to the S-equation $t \upharpoonright \{t_1, \ldots, t_n\} \simeq t' \upharpoonright \{t_1, \ldots, t_n\}$

$S \not\subseteq E$ is proved as in Theorem 4.1.2 ∎

The relations between the equational fragments of LPT and LPT + $\upharpoonright$ are (see Proposition 4.1.17): $TE^{\Sigma} \equiv TE^{\Sigma_\upharpoonright}$, $E^{\Sigma} \equiv E^{\Sigma_\upharpoonright}$ and $ECE^{\Sigma} \equiv S^{\Sigma_\upharpoonright}$. In other formal systems the restriction operator is already definable and does not increase the expressive power of the equational fragments. For instance, in $\mathrm{CL_p}$ (see Definition 4.2.9) $t_1 \upharpoonright t_2$ can be defined as $\mathrm{K}t_1t_2$.

Remark 4.1.8 For free logic the addition of the restriction operator is not enough to prove Theorem 4.1.7, since $\wedge\{t_1 \downarrow, \ldots, t_n \downarrow\} \to t \downarrow$ may not be equivalent to the S-equation $x \upharpoonright \{t_1, \ldots, t_n\} \simeq x \upharpoonright \{t_1, \ldots, t_n, t\}$. However, if we add also a sort 1 (and a constant $*$ of that sort), whose standard interpretation is the singleton $\{*\}$, then the $ECTE$-statement above is equivalent to $* \upharpoonright \{t_1, \ldots, t_n\} \simeq * \upharpoonright \{t_1, \ldots, t_n, t\}$.

4.1.2 P-equational logic

P-equational logic is introduced in [Obt86] as a method for *classifying* partial algebras, which has common features with equational logic: p-term, $*$-substitution, p-equation, p-theory. As in Obtulowicz, we consider only single-sorted algebras, but it is straightforward to extend p-equational logic to the many-sorted case. In Chapter 7 we will *extend* p-equational logic to give an *equational presentation* of the intuitionistic partial lambda calculus. In this section we investigate the relation between p-equational logic and LPT + $\restriction$, namely we prove that

- every $\Sigma_{\restriction}$-term t is provably equivalent to a Σ-p-term $\mathrm{s}(t)$. Therefore p-equations have the same expressive power as S-equations in LPT + $\restriction$.
- p-equational logic is LPT + $\restriction$ restricted to the fragment of p-equations.

A byproduct of these two results is an algorithm to decide equivalence of terms in LPT + $\restriction$, namely $t_1 \simeq t_2$ is derivable in LPT + $\restriction$ iff $\mathrm{s}(t_1)$ and $\mathrm{s}(t_2)$ are syntactically equal. In other words Σ-p-terms are like *canonical forms* for $\Sigma_{\restriction}$-terms.

Definition 4.1.9 ([Obt86]) P-equational logic pEQL

- *D is a Σ-***d-term** *iff*
  - *D is a finite set of Σ-terms of the form $f(\overline{t})$*
  - *any* subterm *of D of the form $f(\overline{t})$ is in D*

  *$M\restriction D$ is a Σ-***p-term** *iff*
  - *D is a Σ-d-term*
  - *M is a Σ-term, and if M is of the form $f(\overline{t})$, then M is in D*
- **$*$-substitution** *(for p-terms):*

$$(M\restriction D)*[x := N\restriction E] \stackrel{\Delta}{\equiv} M[x := N]\restriction D[x := N] \cup E$$

- *a* **p-equation** *is a pair $(M\restriction D \simeq N\restriction E)$ of p-terms and pS is the set of p-equations.*

- *a set T of p-equations (considered as ordered pairs of p-terms) is a* **p-theory** *iff it is a* $*$-substitutive *equivalence relation, i.e. closed w.r.t. the inference rule*

$$*\text{subst}\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2 \qquad t_3{\restriction}D_3 \simeq t_4{\restriction}D_4}{(t_1{\restriction}D_1)*[x:=t_3{\restriction}D_3] \simeq (t_2{\restriction}D_2)*[x:=t_4{\restriction}D_4]}$$

  $T \vdash M{\restriction}D \simeq N{\restriction}E$ *in p-equational logic* $\stackrel{\Delta}{\Longleftrightarrow}$ $M{\restriction}D \simeq N{\restriction}E$ *is in any p-theory which contains T*

Notation 4.1.10 We identify D with $y{\restriction}D$, where y is a *fixed* variable. For instance, $D \simeq E$ means that $y{\restriction}D \simeq y{\restriction}E$.

Remark 4.1.11 The intuition behind a d-term D is that it is closed w.r.t. *derivability*, i.e. if $\{t\downarrow |t \in D\} \Longrightarrow f(\overline{t})\downarrow$ is derivable in LPT, then $f(\overline{t}) \in D$

It is well-known that an equational theory can be described by a category with finite products, where morphisms are (equivalence classes of) terms and composition *is* substitution (see [KR77]). For p-equational logic there is a similar description that uses a category with *ordered strict pre-cartesian structure*, where morphisms are p-terms and composition is $*$-*substitution* (see [Obt86]). This description relies on few properties of $*$-substitution, that are an immediate consequence of the definitions above:

Proposition 4.1.12 Properties of $*$-substitution

- *p-terms are closed w.r.t. $*$-substitution, i.e.*
  if $M{\restriction}D$ and $N{\restriction}E$ are p-terms, then
  $(M{\restriction}D)*[x:=N{\restriction}E]$ *is a p-term*

- $*$-*substitution is associative, i.e.*
  if $y \notin (\mathrm{FV}(M{\restriction}D) - \{x\})$, *then*
  $((M{\restriction}D)*[x:=N{\restriction}E])*[y:=P{\restriction}F] \equiv (M{\restriction}D)*[x:=(N{\restriction}E)*[y:=P{\restriction}F]]$

- *two* independent $*$-*substitutions commute, i.e.*
  if $x \not\equiv y$, $x \notin \mathrm{FV}(P{\restriction}F)$ *and* $y \notin \mathrm{FV}(N{\restriction}E)$, *then*
  $((M{\restriction}D)*[x:=N{\restriction}E])*[y:=P{\restriction}F] \equiv ((M{\restriction}D)*[y:=P{\restriction}F])*[x:=N{\restriction}E]$

Remark 4.1.13 The last property justifies the use of *parallel* $*$-substitution:

$$(t{\restriction}D)*[x_1:=t_1{\restriction}D_1\lfloor\ldots\lfloor x_n:=t_n{\restriction}D_n]$$

We define an algorithm that given a $\Sigma_{\restriction}$-term t returns a Σ-p-term $\mathrm{s}(t)$, which is provably equivalent to t:

Definition 4.1.14 Saturation

The **saturation** $\mathrm{s}(t) \equiv \mathrm{st}(t)\restriction \mathrm{sd}(t)$ *of t is defined by induction on the structure of the $\Sigma_{\restriction}$-term t:*

- $\mathrm{s}(x) \stackrel{\Delta}{\equiv} x\restriction\emptyset$
- $\mathrm{s}(c) \stackrel{\Delta}{\equiv} c\restriction\emptyset$
- $\mathrm{s}(f(\bar{t})) \stackrel{\Delta}{\equiv} f(\mathrm{st}(\bar{t}))\restriction\{f(\mathrm{st}(\bar{t}))\} \cup (\cup \mathrm{sd}(\bar{t}))$
- $\mathrm{s}(t_1\restriction t_2) \stackrel{\Delta}{\equiv} \mathrm{st}(t_1)\restriction \mathrm{sd}(t_1) \cup \mathrm{sd}(t_2)$

Proposition 4.1.15 P-terms are saturated

If t is a Σ-term, then $\mathrm{sd}(t)$ is the set of subterms of t of the form $f(\bar{t})$.

If $t\restriction D$ is a p-term, then $\mathrm{s}(t\restriction D) \equiv t\restriction D$.

Proof The first statement is proved by induction on the structure of the Σ-term t.

The second statement follows immediately from the first one and the definition of p-terms. ∎

Proposition 4.1.16 *If t is a $\Sigma_{\restriction}$-term, then $\mathrm{s}(t)$ is a Σ-p-term and $t \simeq \mathrm{s}(t)$ is derivable in* LPT + $\restriction$

Proof By induction on the structure of the $\Sigma_{\restriction}$-term t. ∎

Proposition 4.1.17 Expressive power in LPT + $\restriction$

$$\begin{array}{c} TE^{\Sigma} \equiv TE^{\Sigma_{\restriction}} \\ \cap \\ E^{\Sigma} \equiv E^{\Sigma_{\restriction}} \\ \cap \\ S^{\Sigma_{\restriction}} \equiv pS^{\Sigma} \equiv ECE^{\Sigma} \end{array}$$

Proof By Proposition 4.1.16, we can always replace a $\Sigma_{\restriction}$-term with a Σ-p-term, in particular any $S^{\Sigma_{\restriction}}$-equation is equivalent to a Σ-p-equation ($S^{\Sigma_{\restriction}} \subseteq pS^{\Sigma}$).

- $TE^{\Sigma} \subseteq TE^{\Sigma\restriction}$ is obvious. So we have to prove that a $TE^{\Sigma\restriction}$-statement is equivalent to a set of TE^{Σ}-statements. In fact, if $M\restriction D$ is a Σ-p-term, then $M\restriction D\downarrow$ is equivalent to the conjunction of the following TE^{Σ}-statements:
  - $t\downarrow$ for any $t \in D$
  - $M\downarrow$
- $E^{\Sigma} \subseteq E^{\Sigma\restriction}$ is obvious. So we have to prove that a $E^{\Sigma\restriction}$-equation is equivalent to a set of E^{Σ}-equations. In fact, if $M\restriction D$ and $N\restriction E$ are Σ-p-terms, then $M\restriction D = N\restriction E$ is equivalent to the conjunction of the following TE^{Σ}-statements and E^{Σ}-equation:
  - $t\downarrow$ for any $t \in D$
  - $t\downarrow$ for any $t \in E$
  - $M = N$
- From Theorem 4.1.7, it follows that $ECE^{\Sigma} \subseteq S^{\Sigma\restriction}$. So we have to prove only that any p-equation $M\restriction D \simeq N\restriction E$ is equivalent to a set of ECE^{Σ}-equations. In fact, a p-equation $M\restriction D \simeq N\restriction E$ is equivalent to the conjunction of the following $ECTE^{\Sigma}$-statements and ECE^{Σ}-equation:
  - $\wedge\{t\downarrow | t \in D\} \to t'\downarrow$ for any $t' \in E$
  - $\wedge\{t\downarrow | t \in E\} \to t'\downarrow$ for any $t' \in D$
  - $\wedge\{t\downarrow | t \in D\} \to M = N$

∎

We prove that p-equational logic and LPT$+\restriction$ are the same formal system on p-equations, by showing that any p-theory T is the set of p-equations valid in a Kripke Σ-structure $\mathcal{B}_T$.

Remark 4.1.18 Although JLPT$+\restriction$ and KLPT$+\restriction$ coincide on the p-equational fragment, there are p-theories which are not the theory of one *classical* partial algebra.

Definition 4.1.19 The Kripke Term Model $\mathcal{B}_T$

Given a p-theory T, the Kripke Σ-structure $\mathcal{B}_T$ over K_T is defined by:

- K_T *is the set of* Σ*-d-terms ordered by inclusion*
- *if* $D \in \mathrm{K}_T$*, then* $\mathcal{A} \equiv \mathcal{B}_T(D)$ *is the structure s.t.:*
  - $\iota^{\mathcal{A}} \stackrel{\Delta}{=} \{[t_1 \restriction D_1]_D | (D \simeq D_1 \cup D) \in T\}$ *, where*
    $[t_1 \restriction D_1]_D \stackrel{\Delta}{=} \{t_2 \restriction D_2 | (t_1 \restriction D_1 \cup D \simeq t_2 \restriction D_2 \cup D) \in T\}$
  - $c^{\mathcal{A}} \stackrel{\Delta}{=} [\mathrm{s}(c)]_D$
  - $f^{\mathcal{A}}(\langle [t_i \restriction D_i]_D | i \in |\overline{t}|\rangle) \stackrel{\Delta}{\simeq} \begin{cases} [f(\overline{t}) \restriction \{f(\overline{t})\} \cup (\cup_i D_i)]_D & \textit{if it is in } \iota^{\mathcal{A}} \\ \textit{undefined} & \textit{otherwise} \end{cases}$
- *if* $f : D_1 \to D_2$ *(i.e.* $D_1 \subseteq D_2$*), then*
  $\mathcal{B}_T(f)([t \restriction D]_{D_1}) \stackrel{\Delta}{=} [t \restriction D]_{D_2}$

Proposition 4.1.20 Characterization of $\mathcal{B}_T$

Let T *be a p-theory,* $D \in \mathrm{K}_T$ *and* $\mathcal{A} \equiv \mathcal{B}_T(D)$*. Then, for all type environments* η*, environments* $\rho \in \eta^{\mathcal{A}}$ *and substitutions* $\sigma \in \prod \rho$*:*

- *If* $t \in \mathrm{Term}_\iota^{\Sigma_\restriction}(\eta)$*, then* $[\![t]\!]_\rho^{\mathcal{A}} \simeq \begin{cases} [(\mathrm{s}(t))*[\sigma]]_D & \textit{if it is in } \iota^{\mathcal{A}} \\ \textit{undefined} & \textit{otherwise} \end{cases}$

Moreover, for any pair of $\Sigma_\restriction$*-terms* $\mathcal{B}_T \Vdash t_1 \simeq t_2$ *iff* $\mathrm{s}(t_1) \simeq \mathrm{s}(t_2) \in T$

Proof The first statement is proved by induction on t. For simplicity we assume that σ is $x := N \restriction E$, the general case is a straightforward generalization.

The basic cases x and c are similar, so we consider only the case c. To prove the statement it is enough to show that $c \restriction D \simeq c \restriction E \cup D$ is in T.

- by reflexivity
  $c \restriction D \simeq c \restriction D$
- since $[N \restriction E]_D$ is in $\iota^{\mathcal{A}}$, then
  $D \simeq E \cup D$
- by applying ($*$subst) to substitute a variable y that occurs neither in D nor in the E
  $c \restriction D \equiv (c \restriction D)*[y := D] \simeq (c \restriction D)*[y := E \cup D] \equiv c \restriction E \cup D$

The inductive steps $f(\overline{t})$ and $t_1{\restriction}t_2$ are similar, so we consider only the case $f(\overline{t})$. Let $n = |\overline{t}|$, $M_i{\restriction}D_i \equiv (\mathrm{s}(t_i))*[x_1{:=}\, N{\restriction}E]$ (for all $i \in n$) and $\overline{M} \equiv \langle M_i | i \in n\rangle$. By IH for the t_i and by definition of $f^{\mathcal{A}}$:

$$[\![f(\overline{t})]\!]^{\mathcal{A}}_{\rho} \simeq \begin{cases} [f(\overline{M}){\restriction}\{f(\overline{M})\} \cup (\cup_{i\in n} D_i)]_D & \text{if it is in } \iota^{\mathcal{A}} \\ undefined & \text{otherwise} \end{cases}$$

but $(\mathrm{s}(f(\overline{t})))*[y{:=}\, N{\restriction}E] \equiv f(\overline{M}){\restriction}\{f(\overline{M})\} \cup (\cup_{i\in n} D_i) \cup E$. Therefore, to prove the statement it is enough to show that

$$f(\overline{M}){\restriction}\{f(\overline{M})\} \cup (\cup_{i\in n} D_i) \cup D \simeq f(\overline{M}){\restriction}\{f(\overline{M})\} \cup (\cup_{i\in n} D_i) \cup E \cup D$$

The proof proceeds like that for the basic cases.

The implication from right to left of the second statement follows immediately from the first statement. For the other direction, let $M_i{\restriction}D_i \equiv \mathrm{s}(t_i)$, we consider the denotation of the terms in the environment $\rho_i \stackrel{\Delta}{=} [\sigma]_{D_i}$ corresponding to the *identity substitution* σ (i.e. $\sigma(x) = x{\restriction}\emptyset$) at the stages D_1 and D_2:

1. $[M_1{\restriction}D_1]_{D_1} =$ by the first statement
   $[\![t_1]\!]^{\mathcal{B}_T, D_1}_{\rho_1} =$ by the assumption
   $[\![t_2]\!]^{\mathcal{B}_T, D_1}_{\rho_1} =$ by the first statement
   $[M_2{\restriction}D_2]_{D_1}$

2. $[M_1{\restriction}D_1]_{D_2} =$ by the first statement
   $[\![t_1]\!]^{\mathcal{B}_T, D_2}_{\rho_2} =$ by the assumption
   $[\![t_2]\!]^{\mathcal{B}_T, D_2}_{\rho_2} =$ by the first statement
   $[M_2{\restriction}D_2]_{D_2}$

Now we have to derive the pλ-equation $M_1{\restriction}D_1 \simeq M_2{\restriction}D_2$:

- by the first identity above

  $M_1{\restriction}D_1 \simeq M_2{\restriction}D_1 \cup D_2$

- from the second identity above, by ($*$subst)

  $D_1 \cup D_2 \simeq D_2$

- by reflexivity

  $M_2{\restriction}D_2 \simeq M_2{\restriction}D_2$

- by applying ($*$subst)

  $M_2{\restriction}D_1 \cup D_2 \simeq M_2{\restriction}D_2$

- by transitivity

  $M_1{\restriction}D_1 \simeq M_2{\restriction}D_2$

∎

Theorem 4.1.21 *p-equational logic is* LPT + $\restriction$ *restricted to the fragment* pS*, i.e.* $T \vdash M{\restriction}D \simeq N{\restriction}E$ *in p-equational logic iff* $T \vdash M{\restriction}D \simeq N{\restriction}E$ *in* LPT + $\restriction$ *for every p-equation* $M{\restriction}D \simeq N{\restriction}E$*.*

Proof The implication from left to right is easy to prove, in fact it is enough to prove that the inference rules for p-equational logic ((refl), (symm), (trans) and ($*$subst)) are derivable in LPT + $\restriction$.

For the other implication we rely on an easy corollary of Lemma 4.1.20 (and Proposition 4.1.15): $\mathcal{B}_T \Vdash t_1{\restriction}D_1 \simeq t_2{\restriction}D_2$ iff $t_1{\restriction}D_1 \simeq t_2{\restriction}D_2 \in T$ for any pair of p-terms. We prove that $M{\restriction}D \simeq N{\restriction}E \in T$, provided $M{\restriction}D \simeq N{\restriction}E$ is derivable from T' in JLPT + $\restriction$ and T is a p-theory containing T':

- since T' is a set of p-equations, by Lemma 4.1.20

  $\mathcal{B}_T \Vdash T'$

- since $\mathcal{B}_T$ is a model of JLPT, by soundness

  $\mathcal{B}_T \Vdash M{\restriction}D \simeq N{\restriction}E$

- since $M{\restriction}D \simeq N{\restriction}E$ is a p-equation, by Lemma 4.1.20

  $M{\restriction}D \simeq N{\restriction}E \in T$

∎

4.1.3 The monotonic logic of partial terms

In this section we introduce monLPT, the *variant* of LPT for *ordered* partial algebras and investigate the expressive powers of the *inequational* fragments. monLPT provides a base for the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus (and the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus).

Definition 4.1.22 monLPT

The **monotonic logic of partial terms** *(*monLPT*) over* Σ_{mon} *is the formal system axiomatized by* $\mathrm{mon_p}$ *in* LPT *over* Σ_{mon}*, where*

- Σ_{mon} *is a signature for* **monotonic** partial algebras*, i.e. a signature for* partial algebras *with a* **partial order** $\leq_\tau \in \mathrm{Pred}_{\tau,\tau}(\Sigma_{\mathrm{mon}})$*, for any* $\tau \in \mathrm{Sort}(\Sigma_{\mathrm{mon}})$
- $\mathrm{mon_p}$ *is the set of axioms:*

$$\text{refl} \quad x \leq_\tau x$$

$$\text{asymm} \quad x_1 \leq_\tau x_2, x_2 \leq_\tau x_1 \Longrightarrow x_1 =_\tau x_2$$

$$\text{trans} \quad x_1 \leq_\tau x_2, x_2 \leq_\tau x_3 \Longrightarrow x_1 \leq_\tau x_3$$

$$\text{mon.}f \quad \overline{x} \leq_{\overline{\tau}} \overline{x}', f(\overline{x}) \downarrow_\tau \Longrightarrow f(\overline{x}) \leq_\tau f(\overline{x}')$$

Notation 4.1.23 We write $t \lesssim t'$ for the formula $(t \downarrow \rightarrow t \leq t')$.

For ordered partial algebras it is more natural to consider *inequations*; they are defined like the corresponding equations, but $=$ and $\simeq$ are replaced by $\leq$ and $\lesssim$ respectively:

Definition 4.1.24 Inequational Fragments

- **e***xistence* **in***equations (Ein-equations)* $t_1 \leq t_2$
- **s***trong* **in***equations (Sin-equations)* $t_1 \lesssim t_2$*, i.e.* $t_1 \downarrow \rightarrow t_1 \leq t_2$
- **e***xistentially* **c***onditioned* **E**-**in***equations (ECEin-equations)* $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \leq t'$
- **e***xistentially* **c***onditioned* **S**-**in***equations (ECSin-equations)* $t_1 \downarrow, \ldots, t_n \downarrow \Longrightarrow t \lesssim t'$

In monLPT and monLPT$+\upharpoonright$ the expressive powers of the inequational fragments are related according to the same pattern given for the equational fragments.

Theorem 4.1.25 Expressive power in monLPT

$$\begin{array}{ccccc} TE & \subset & Ein & & \\ & & \cap & & \\ Sin & \subset & TE \cup Sin & & \\ \cap & & \cap & & \\ ECSin & \subset & ECEin & \equiv & ECTE \cup ECSin \end{array}$$

Theorem 4.1.26 Expressive power in monLPT $+\upharpoonright$

$$\begin{array}{ccccc} TE & \subset & Ein & & \\ & & \cap & & \\ & & Sin & \equiv & ECEin \end{array}$$

There is also a variant of p-equational logic corresponding to monLPT $+\upharpoonright$ restricted to the fragment of *p-inequations* (i.e. Sin-equations between p-terms): mon**p-equational logic**. The same theorems (and corresponding proofs) for p-equational logic hold *mutatis mutandis* for monp-equational logic, so we will only give the main definitions and theorems.

Definition 4.1.27 monp-equational logic monpEQL

- *a* **p-inequation** *is a pair* $(M\upharpoonright D \lesssim N\upharpoonright E)$ *of p-terms and* $pSin$ *is the set of p-inequations.*

- *a set* T *of p-inequations (considered as ordered pairs of p-terms) is a* **monotonic p-theory** *iff it is a* $*$-substitutive preorder *relation, i.e. it satisfies the axiom*

  incl $D \lesssim \emptyset$

  and is closed w.r.t. the inference rule

  $$*\text{subst}\ \frac{t_1\upharpoonright D_1 \lesssim t_2\upharpoonright D_2 \qquad t_3\upharpoonright D_3 \lesssim t_4\upharpoonright D_4}{(t_1\upharpoonright D_1)*[x := t_3\upharpoonright D_3] \lesssim (t_2\upharpoonright D_2)*[x := t_4\upharpoonright D_4]}$$

  $T \vdash M\upharpoonright D \lesssim N\upharpoonright E$ *in* mon*p-equational logic* $\overset{\Delta}{\iff}$ $M\upharpoonright D \lesssim N\upharpoonright E$ *is in any monotonic p-theory which contains* T

Proposition 4.1.28 Expressive power in monLPT + $\restriction$

$$\begin{array}{c} E^{\Sigma}in \equiv E^{\Sigma\restriction}in \\ \cap \\ S^{\Sigma\restriction}in \equiv pS^{\Sigma}in \equiv ECE^{\Sigma}in \end{array}$$

Proof The proof is similar to that of Proposition 4.1.17. The only difference is in proving $pS^{\Sigma}in \equiv ECE^{\Sigma}in$. In fact, a p-inequation $M\restriction D \lesssim N\restriction E$ is equivalent to the conjunction of the following $ECTE^{\Sigma}$-statements and $ECE^{\Sigma}in$-equation:

- $\wedge\{t\downarrow | t \in D\} \to t' \downarrow$ for any $t \in E$
- $\wedge\{t\downarrow | t \in D\} \to M \leq N$

∎

Theorem 4.1.29 mon*p-equational logic is* monLPT + $\restriction$ *restricted to the fragment pSin, i.e.* $T \vdash M\restriction D \lesssim N\restriction E$ *in* mon*p-equational logic iff* $T \vdash M\restriction D \lesssim N\restriction E$ *in* monLPT + $\restriction$ *for every p-inequation* $M\restriction D \lesssim N\restriction E$.

4.2 The partial lambda calculus

In this section we introduce the partial lambda calculus as an *extension* of LPT obtained by adding a new formation rule (for lambda-abstraction) and some axioms. Since its language is not a *first order* one (because of lambda-abstraction), we cannot axiomatize it directly in LPT. Therefore, we introduce another formal system, extensional partial combinatory logic, defined by axiomatization in LPT, and show that it is equivalent to the $\lambda_p\beta\eta$-calculus *at the level of terms*, i.e. the two translations of the equivalence commute with predicates, logical connectives and quantifiers, e.g. equations are translated into equations. A similar equivalence will be stated also for the $\lambda_p\beta$-calculus. Because of this equivalence, one can apply the results about LPT to the study of the partial lambda calculus.

The partial lambda calculus comes in two *flavours*: **typed** and **untyped**. The typed partial lambda calculus is essentially many-sorted, while the untyped partial lambda calculus is single-sorted. We will study both the typed and untyped (partial) lambda calculus. Since the two calculi are very similar and have many properties in common, it would be redundant to prove them twice, therefore we will prove a common property only for the untyped calculus. In fact, when a property does not rely on the type structure, types are just a syntactic nuisance.

Definition 4.2.1 Language of the typed λ_{p}-calculus

A signature Σ_λ *for* **typed partial applicative structures** *is a signature for partial algebras with* **function spaces** $\tau_1 \rightharpoonup \tau_2 \in \mathrm{Sort}(\Sigma_\lambda)$ *and* **application** $\mathrm{app}_{\tau_1,\tau_2} \in \mathrm{Funct}_{(\tau_1 \rightharpoonup \tau_2, \tau_1) \rightharpoonup \tau_2}(\Sigma_\lambda)$ *for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma_\lambda)$*.*

The language of the λ_{p}*-calculus over* Σ_λ *is the language generated by the formation rules for* LPT *over* Σ_λ *(see Definition 2.1.5) and the formation rule*

$$\lambda \ \frac{t \in \mathrm{Term}^{\Sigma_\lambda}_{\tau'}(\eta \lfloor x \colon \tau)}{(\lambda x \colon \tau . t) \in \mathrm{Term}^{\Sigma_\lambda}_{\tau \rightharpoonup \tau'}(\eta)}$$

Notation 4.2.2 We say that a sort ι is a **base sort** if it is not a function space.

Definition 4.2.3 Language of the untyped λ_{p}-calculus

A signature Σ_λ *for* **untyped partial applicative structures** *is a single-sorted signature for partial algebras with one* **application** $\mathrm{app} \in \mathrm{Funct}_{(\iota,\iota) \rightharpoonup \iota}(\Sigma_\lambda)$.

The language of the λ_{p}*-calculus over* Σ_λ *is the language generated by the formation rules for* LPT *over* Σ_λ *(see Definition 2.1.5) and the formation rule*

$$\lambda \ \frac{t \in \mathrm{Term}^{\Sigma_\lambda}_{\iota}(\eta \lfloor x \colon \iota)}{(\lambda x \colon \iota . t) \in \mathrm{Term}^{\Sigma_\lambda}_{\iota}(\eta)}$$

Notation 4.2.4 We (remove the subscripts and) write $t_1 t_2$ for $\mathrm{app}(t_1, t_2)$, we also use the convention of *associating* application to the left. We (remove the sort of the bound variables and) write $\lambda x_1, \ldots, x_n . t$ for $\lambda x_1 . (\ldots \lambda x_n . t)$ (if $n \geq 1$).

Definition 4.2.5 The $\lambda_{\mathrm{p}}\beta\eta$-calculus

The typed (untyped) **partial lambda calculus** *(*$\lambda_{\mathrm{p}}\beta\eta$*-calculus) over* Σ_λ *is the formal system on the language of the* λ_{p}*-calculus over* Σ_λ generated *by the inference rules for* LPT *and the set of axioms* $\lambda_{\mathrm{p}}\beta\eta$*, where*

- Σ_λ *is a signature for typed (untyped) partial applicative structures*

- *$\lambda_{\mathrm{p}}\beta\eta$ is the set of axioms*

$$\text{E}.\lambda\ \ (\lambda x\colon\tau.t)\downarrow$$

$$\xi.\simeq\ \ (\forall x\colon\tau.t\simeq t')\Longrightarrow(\lambda x\colon\tau.t)=(\lambda x\colon\tau.t')$$

$$\beta\ \ (\lambda x\colon\tau.t)x\simeq t$$

$$\eta\ \ (\lambda x\colon\tau.fx)=f$$

Remark 4.2.6 Besides the $\lambda_{\mathrm{p}}\beta\eta$-calculus there are three *weaker* calculi obtained by dropping either (η) or (β) or both: the $\lambda_{\mathrm{p}}\beta$- the $\lambda_{\mathrm{p}}\eta$- and the λ_{p}-calculus. In analogy with LPT, both the typed and untyped $\lambda_{\mathrm{p}}\beta\eta$-calculus (as well as the weaker calculi) have the following variants on the same language:

$$\begin{array}{ccccc} \mathit{Free} & J & & & 0 \\ \cap & \cap & \lambda_{\mathrm{p}}\beta\eta & + & \cap \\ \mathit{Inhabited} & K & & & \mathrm{tot} \end{array}$$

Moreover, there is also the partial lambda calculus with restriction operator ($\lambda_{\mathrm{p}}\beta\eta+\upharpoonright$-calculus).

Since the axiom (inhab), for inhabited structures, is derivable in the untyped partial lambda calculus, we do not distinguish between *Free* and *Inhabited* untyped partial lambda calculus.

Notation 4.2.7 When convenient the suffix "-calculus" is dropped, e.g. $J\lambda_{\mathrm{p}}\beta$ stands for the $J\lambda_{\mathrm{p}}\beta$-calculus.

The total variant of the $\lambda_{\mathrm{p}}\beta\eta$-calculus is called either the $\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot}$-calculus or the $\lambda_{\mathrm{t}}\beta\eta$-calculus.

In the $\lambda_{\mathrm{p}}\beta$-calculus the equational fragments have the same expressive power:

Theorem 4.2.8 Expressive power in the $\lambda_{\mathrm{p}}\beta$-calculus

$$TE\ \subset\ E\ \equiv\ ECE$$

Proof In the $\lambda_{\mathrm{p}}\beta$-calculus $S\equiv ECE$, because the restriction operator $t_1\upharpoonright t_2$ is definable as $(\lambda x,y.x)t_1t_2$ (see Theorem 4.1.7). So we have to prove only that $S\subseteq E$. In fact, the S-equation $t_1\simeq t_2$ is equivalent to the E-equation $(\lambda x.t_1)=(\lambda x.t_2)$, provided x occurs neither in t_1 nor in t_2 ∎

4.2.1 Equivalence of the $\lambda_{\mathrm{p}}\beta\eta$-calculus and $\mathrm{CL_p} + \mathrm{ext.} \simeq$

We define extensional partial combinatory logic ($\mathrm{CL_p} + \mathrm{ext.} \simeq$) and show that there are two translations between λ-terms and CL-terms that induce an equivalence with the $\lambda_{\mathrm{p}}\beta\eta$-calculus. For simplicity we consider only the untyped $\lambda_{\mathrm{p}}\beta\eta$-calculus.

Definition 4.2.9 Extensional partial combinatory logic $\mathrm{CL_p} + \mathrm{ext.} \simeq$

Partial combinatory logic *(*$\mathrm{CL_p}$*) over* Σ_{CL} *is the formal system axiomatized by* $\mathrm{CL_p}$ *in* LPT *over* Σ_{CL}*.*

Extensional *partial combinatory (*$\mathrm{CL_p} + \mathrm{ext.} \simeq$*) over* Σ_{CL} *is the formal system axiomatized by the axiom* (ext. $\simeq$) *in* $\mathrm{CL_p}$ *over* Σ_{CL}*, where*

- Σ_{CL} *is a signature for* **partial combinatory algebras***, i.e. a signature for partial applicative structures with* **combinators** $\mathrm{K}, \mathrm{S} \in \mathrm{Const}_\iota(\Sigma_{\mathrm{CL}})$ *and* **canonical representatives** $c_f \in \mathrm{Const}_\iota(\Sigma_{\mathrm{CL}})$ *for any function symbol* f *in* Σ_{CL}
- $\mathrm{CL_p}$ *is the set of axioms*

$$\mathrm{K} \quad \mathrm{K}xy = x$$

$$\mathrm{S} \quad \mathrm{S}xyz \simeq xz(yz)$$

$$\mathrm{E.S} \quad \mathrm{S}xy \downarrow$$

$$\mathrm{E.}c_f \quad c_f\overline{x} \downarrow \qquad \textit{if } f \textit{ takes } |\overline{x}| + 1 \textit{ arguments}$$

$$c_f \quad c_f x \simeq f() \qquad \textit{if } f \textit{ takes no arguments}$$

$$c_f \quad c_f\overline{x} \simeq f(\overline{x}) \qquad \textit{if } f \textit{ takes } |x| \geq 1 \textit{ arguments}$$

- (ext. $\simeq$) *is the axiom*

$$\mathrm{ext.} \simeq \quad (\forall z{:}\,\iota.xz \simeq yz) \Longrightarrow x = y$$

Notation 4.2.10 We write I for the **identity combinator** SKK.

Remark 4.2.11 When there are function symbols, canonical representatives are essential to define abstraction in $\mathrm{CL_p}$ (see Definition 4.2.14). It is not necessary to have a canonical representative for application, since the I combinator can be used instead.

Remark 4.2.12 In Chapter VI of [Bee85] partial combinatory algebras are considered in conjunction with the extensionality principle

$$\text{ext.} = \quad (\forall z{:}\,\iota.xz = yz) \Longrightarrow x = y$$

The axiom (ext. $=$) is too weak to *support λ-notation*. In fact, there is not a *canonical* choice for the interpretation of $\lambda x.M$, unless $\forall x.M \downarrow$. However, (ext. $=$) is in the fragment of the language where intuitionistic and classical inhabited logic coincide, therefore $\mathrm{CL_p}$ + ext. $=$ is *easier* to study than $\mathrm{CL_p}$ + ext. $\simeq$. In an **extensional partial applicative structure** (i.e. a model of (ext. $\simeq$)) an element is uniquely determined by its *applicative behaviour*, therefore there is at most one element which satisfies the axioms of K (S or c_f).

In $\mathrm{CL_p}$ (and similarly in the $\lambda_\mathrm{p}\beta$-calculus) one can w.l.o.g. consider only terms with no function symbols apart from application.

Proposition 4.2.13 Elimination of function symbols in $\mathrm{CL_p}$

In $\mathrm{CL_p}$ every term t is equivalent to a term t' with the same free variables and no function symbols apart from application

Proof t' is defined by induction on the structure of t; the only interesting cases are:

- $(f())' \stackrel{\Delta}{\equiv} c_f\mathrm{I}$
- $(f(t_1, \ldots, t_n))' \stackrel{\Delta}{\equiv} c_f t'_1 \ldots t'_n$ if f takes at least one argument

∎

The proof of equivalence between the $\lambda_\mathrm{p}\beta\eta$-calculus and $\mathrm{CL_p}$ + ext. $\simeq$ follows quite closely that for the $\lambda\beta\eta$-calculus and extensional combinatory logic (see [Bar81]). The main difference is that we cannot define abstraction in $\mathrm{CL_p}$ as in

[Bar81], i.e. $\lambda^* x.P \equiv \mathrm{K}P$ if $x \notin P$, because in general $\mathrm{CL_p} \not\vdash (\lambda^* x.P)\downarrow$; so we have to use the other abstraction $\lambda_I x.P$, which does not have the property

$$(\lambda_I x.P)[y\colon= Q] \equiv \lambda_I x.(P[y\colon= Q]) \text{ if } y \not\equiv x \notin Q$$

In order to prove the equivalence we fix a signature Σ_{CL} for partial combinatory algebras and a signature Σ_λ for partial applicative structures obtained from Σ_{CL} by removing K, S and c_f. In the rest of this section CL-**term** means a CL-term with no function symbols apart from application, since any other CL-term is equivalent to one of them (by Proposition 4.2.13). We define two translations:

- $\_^{CL}$ from λ-terms over Σ_λ to CL-terms over Σ_{CL}
- $\_^{\lambda}$ from CL-terms over Σ_{CL} to λ-terms over Σ_λ

and prove that they establish an equivalence between λ-terms and CL-terms.

Definition 4.2.14 Abstraction in $\mathrm{CL_p}$ and Translations

- *the* **abstraction** $[x]t$ *in* $\mathrm{CL_p}$ *is defined by induction on the structure of* t*.*
  - $[x]x \stackrel{\Delta}{\equiv} \mathrm{I}$ *where* $\mathrm{I} \stackrel{\Delta}{\equiv} \mathrm{SKK}$
  - $[x]x' \stackrel{\Delta}{\equiv} \mathrm{K}x'$ *if* $x \not\equiv x'$
  - $[x]c \stackrel{\Delta}{\equiv} \mathrm{K}c$
  - $[x](t_1 t_2) \stackrel{\Delta}{\equiv} \mathrm{S}([x]t_1)([x]t_2)$
- *the* **translation** t^{CL} *from* λ*-terms to* CL*-terms is defined by induction on* t *and makes use of the abstraction in* $\mathrm{CL_p}$*:*
  - $x^{CL} \stackrel{\Delta}{\equiv} x$
  - $c^{CL} \stackrel{\Delta}{\equiv} c$
  - $f()^{CL} \stackrel{\Delta}{\equiv} c_f\mathrm{I}$ *if* f *takes no arguments*
  - $f(\overline{t})^{CL} \stackrel{\Delta}{\equiv} c_f\overline{t}^{CL}$ *if* f *takes at least one argument*
  - $(\lambda x.t)^{CL} \stackrel{\Delta}{\equiv} [x]t^{CL}$
- *the* **translation** t^λ *from* CL*-terms to* λ*-terms is defined by induction on* t*:*
  - $x^\lambda \stackrel{\Delta}{\equiv} x$

- $\mathrm{K}^\lambda \stackrel{\Delta}{\equiv} \lambda x, y.x$
- $\mathrm{S}^\lambda \stackrel{\Delta}{\equiv} \lambda x, y, z.xz(yz)$
- $c^\lambda \stackrel{\Delta}{\equiv} c$
- ${c_f}^\lambda \stackrel{\Delta}{\equiv} \lambda x.f()$ *if f takes no arguments*
- ${c_f}^\lambda \stackrel{\Delta}{\equiv} \lambda\overline{x}.f(\overline{x})$ *if f takes $|\overline{x}| > 0$ arguments*
- $(t_1 t_2)^\lambda \stackrel{\Delta}{\equiv} ({t_1}^\lambda {t_2}^\lambda)$

Remark 4.2.15 Both translations preserve free variables and can be extended to formulae and sequents in the obvious way. The translation $(\_)^\lambda$ commutes with substitution, i.e. ${t'}^\lambda[x\!:= t^\lambda] \equiv (t'[x\!:= t])^\lambda$, but the other translation does not, because of abstraction, e.g. if $t' \equiv \lambda y.x$ and $t \equiv (yz)$, then ${t'}^{CL}[x\!:= t^{CL}] \equiv \mathrm{K}(yz)$ but $(t'[x\!:= t])^{CL} \equiv \mathrm{S}(\mathrm{K}y)(\mathrm{K}z)$.

Lemma 4.2.16 Properties of abstraction in $\mathrm{CL_p}$
The following formulae are derivable from $\mathrm{CL_p}$*:*

$\mathrm{CL_p}.\mathrm{E}.\lambda$ $([x]t)\downarrow$

$\mathrm{CL_p}.\beta$ $([x]t)x \simeq t$

Proof By induction on the structure of the CL-term t ∎

Lemma 4.2.17 Properties of $(\_)^{CL}$ in $\mathrm{CL_p} + \mathrm{ext.}\simeq$
The following formulae are derivable from $\mathrm{CL_p} + \mathrm{ext.}\simeq$*:*

$$t\downarrow \Longrightarrow ([x']t')[x\!:= t] = [x'](t'[x\!:= t])$$

sub.$\mathrm{CL_p}$ $t^{CL}\downarrow \Longrightarrow {t'}^{CL}[x\!:= t^{CL}] \simeq (t'[x\!:= t])^{CL}$

Proof The first statement is proved by induction on the structure of the CL-term t'. The only non trivial case is $t' \equiv x \not\equiv x'$, when we have to prove that $\mathrm{K}t = [x']t$ under the assumption $t\downarrow$ (and $x' \notin t$):

- by $t \downarrow$ and the axiom (K)

  $\mathrm{K}tx' = t$

- by the derived axiom ($\mathrm{CL_p}.\beta$)

  $([x']t)x' = t$

- by transitivity

  $([x']t)x' = \mathrm{K}tx'$

- by extensionality and $x' \notin t$

  $([x']t) = \mathrm{K}t$

The second statement is proved by induction on the structure of the λ-term t'. The only non trivial case is abstraction, when we have to use the first statement ∎

Lemma 4.2.18 Properties of abstraction in the $\lambda_{\mathrm{p}}\beta$-calculus

The following formula is derivable in the $\lambda_{\mathrm{p}}\beta$-calculus:

- $(\lambda x.t^{\lambda}) = ([x]t)^{\lambda}$

Proof By induction on the structure of the CL-term t ∎

Theorem 4.2.19 Equivalence on Terms

- $\_^{CL}$ *is a relative interpretation of the $\lambda_{\mathrm{p}}\beta\eta$-calculus in* $\mathrm{CL_p} + \mathrm{ext.} \simeq$
- $\_^{\lambda}$ *is a relative interpretation of* $\mathrm{CL_p} + \mathrm{ext.} \simeq$ *in the $\lambda_{\mathrm{p}}\beta\eta$-calculus*
- $\mathrm{CL_p} + \mathrm{ext.} \simeq \vdash_{\Sigma_{\mathrm{CL}}} t \simeq t^{\lambda CL}$
- $\lambda_{\mathrm{p}}\beta\eta \vdash_{\Sigma_{\lambda}} t \simeq t^{CL\,\lambda}$

Proof The first statement is proved by showing that the translation of the inference rules for the $\lambda_{\mathrm{p}}\beta\eta$-calculus are derivable in $\mathrm{CL_p} + \mathrm{ext.} \simeq$. Each inference rule for LPT is translated into itself, except for (subst). However, the translation of this inference rule can be derived by using ($\mathrm{sub.CL_p}$) (see Lemma 4.2.17). The translation of the other axioms for the $\lambda_{\mathrm{p}}\beta\eta$-calculus can be easily derived by using ($\mathrm{CL_p.E.}\lambda$), ($\mathrm{CL_p}.\beta$) and (ext. $\simeq$) (see Lemma 4.2.16).

The second statement is proved by showing that the translation of the inference rules for $\mathrm{CL_p}$+ext. $\simeq$ are derivable in the $\lambda_{\mathrm{p}}\beta\eta$-calculus. Each inference rule for LPT is translated into itself. The inference rule (subst) does not cause any problem, because the translation $\_^{\lambda}$ commutes with substitution. The translation of the other axioms for $\mathrm{CL_p}$ + ext. $\simeq$ can be easily derived.

The third statement is proved by induction on the structure of CL-terms.

The fourth statement is proved by induction on the structure of λ-terms and we have to use Lemma 4.2.18, when t is an abstraction. ∎

Corollary 4.2.20 Equivalence of the $\lambda_{\mathrm{p}}\beta\eta$-calculus and $\mathrm{CL_p}$ + ext. $\simeq$

The $\lambda_{\mathrm{p}}\beta\eta$-calculus and $\mathrm{CL_p}$ + ext. $\simeq$ *are equivalent via the translations* $(\_)^{\lambda}$ *and* $(\_)^{CL}$

Also the $\lambda_{\mathrm{p}}\beta$-calculus is equivalent to a formal system axiomatized in LPT. The proof is similar to that for the $\lambda_{\mathrm{p}}\beta\eta$-calculus plus some extra complications due to the lack of extensionality, that have already been considered in the literature (see [Mey82]).

Notation 4.2.21 We write B for the **composition combinator** S(KS)K and ϵ for S(KI), while ϵ_n is defined by induction on n:

- $\epsilon_1 \overset{\Delta}{\equiv} \epsilon$
- $\epsilon_{n+1} \overset{\Delta}{\equiv} \mathrm{B}\epsilon(\mathrm{B}\epsilon_n)$

Definition 4.2.22 *Partial combinatory logic with choice operator,* $\mathrm{CL_p}$+ϵext. $\simeq$, *over* Σ_{CL} *is the formal system axiomatized by* ϵext. $\simeq$ *in* $\mathrm{CL_p}$ *over* Σ_{CL}*, where*

- Σ_{CL} *is a signature for partial combinatory algebras*
- ϵext. $\simeq$ *is the set of axioms*

$$\epsilon\text{ext.} \simeq \quad (\forall z{:}\,\iota.xz \simeq yz) \Longrightarrow \epsilon x = \epsilon y$$

$$\epsilon \quad \epsilon = \epsilon\epsilon$$

$$\mathrm{E}.\epsilon.1 \quad \epsilon x \downarrow$$

$$\epsilon\mathrm{K}\ \ \mathrm{K} = \epsilon_2\mathrm{K}$$

$$\epsilon\mathrm{S}\ \ \mathrm{S} = \epsilon_3\mathrm{S}$$

$$\epsilon c_f\ \ c_f = \epsilon_1 c_f \qquad \textit{if } f \textit{ takes no arguments}$$

$$\epsilon c_f\ \ c_f = \epsilon_{n+1} c_f \qquad \textit{if } f \textit{ takes } n+1 \textit{ arguments}$$

Theorem 4.2.23 Equivalence of the $\lambda_{\mathrm{p}}\beta$-calculus and $\mathrm{CL}_{\mathrm{p}} + \epsilon\mathrm{ext.} \simeq$

The $\lambda_{\mathrm{p}}\beta$*-calculus and* $\mathrm{CL}_{\mathrm{p}} + \epsilon\mathrm{ext.} \simeq$ *are equivalent via the translations* $(\_)^{\lambda}$ *and* $(\_)^{CL}$

4.2.2 λ-models

The translation from λ-terms to CL-terms provides an indirect way of interpreting a λ-term t in a (Kripke) Σ_{CL}-structure $\mathcal{A}$, namely $[\![t]\!]^{\mathcal{A}}_{\rho}$ is $[\![t^{CL}]\!]^{\mathcal{A}}_{\rho}$. In general such an interpretation is *unsatisfactory*, because (subst) is not admissible. However, when $\mathcal{A}$ is a (Kripke) extensional partial combinatory algebra (or a partial combinatory algebra with choice operator), all the inference rules of the $\lambda_{\mathrm{p}}\beta\eta$-calculus are admissible (by Corollary 4.2.20).

Actually, there is a more direct description of this interpretation in environment model style (see [Mey82, MM87]):

Definition 4.2.24 λ-interpretation in extensional partial combinatory algebras

Given a Kripke extensional partial combinatory algebra $\mathcal{B}$ *over* K*, the denotation* $[\![t]\!]^{\mathcal{B},\alpha}_{\rho}$ *of a* λ*-term* $t \in \mathrm{Term}^{\Sigma_\lambda}_{\iota}(\eta)$ *in* $\rho \in \eta^{\mathcal{B}(\alpha)}$ *at a stage* α *is defined by induction on the structure of* t*:*

- $[\![x]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{\simeq} \rho(x)$
- $[\![c]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{\simeq} c^{\mathcal{B}(\alpha)}$
- $[\![f(\overline{t})]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{\simeq} f^{\mathcal{B}(\alpha)}([\![\overline{t}]\!]^{\mathcal{B},\alpha}_{\rho})$

- $[\![(\lambda x.t)]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{\simeq} d$, *where d is the unique element of $\iota^{\mathcal{B}(\alpha)}$ s.t.*

  $\mathrm{app}^{\mathcal{B}(\alpha')}(d_{\upharpoonright f}, a) \simeq [\![t]\!]^{\mathcal{B},\alpha'}_{\rho_{\upharpoonright f}\lfloor x:=a}$ for all $f\colon \alpha \to \alpha'$ and $a \in \iota^{\mathcal{B}(\alpha')}$

For the λ_{p}-calculus without (β) we need a more general notion of λ-*interpretation*, that generalizes that of *functional domain* (see [Mey82]):

Definition 4.2.25 λ-structure and λ-interpretation

A **λ-structure** $\mathcal{A}$ *is a Σ_λ-structure plus a partial function (partial morphism) $\Psi^{\mathcal{A}}\colon (\iota^{\mathcal{A}} \rightharpoonup_{set} \iota^{\mathcal{A}}) \rightharpoonup \iota^{\mathcal{A}}$ (in the category $\mathbf{SET}_p$ of Definition 5.1.5). The denotation $[\![t]\!]^{\mathcal{A}}_{\rho}$ of a λ-term $t \in \mathrm{Term}^{\Sigma_\lambda}_{\iota}(\eta)$ in $\rho \in \eta^{\mathcal{A}}$ is defined as in Definition 2.2.3 plus an extra clause for λ-abstraction:*

$$[\![(\lambda x.t)]\!]^{\mathcal{A}}_{\rho} \stackrel{\Delta}{\simeq} \Psi^{\mathcal{A}}(\lambda a \in \iota^{\mathcal{A}}.[\![t]\!]^{\mathcal{A}}_{\rho\lfloor x:=a})$$

Similarly, a Kripke λ-structure $\mathcal{B}$ over K is a Kripke Σ_λ-structure over K plus a partial morphism $\Psi^{\mathcal{A}}\colon (\iota^{\mathcal{A}} \rightharpoonup_{\mathrm{K}set} \iota^{\mathcal{A}}) \rightharpoonup \iota^{\mathcal{A}}$ in the category $\mathrm{K}\mathbf{SET}_p$ (see Definition 5.1.9). The extra clause for λ-abstraction is:

> $[\![(\lambda x.t)]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{\simeq} \Psi^{\mathcal{A}}_{\alpha}(s)$, *where s is the consistent family of partial functions at α (see Definition 5.1.7) s.t. $s_f = \lambda a \in \iota^{\mathcal{B}(\alpha')}.[\![t]\!]^{\mathcal{B},\alpha'}_{\rho_{\upharpoonright f}\lfloor x:=a}$ for all $f\colon \alpha \to \alpha'$*

Remark 4.2.26 In a λ-structure all the inference rules of LPT are admissible and in addition the following axiom (similar to (cong.f)) is valid:

cong.λ $(\forall x\colon \iota.t \simeq t'), (\lambda x.t)\downarrow \Longrightarrow (\lambda x.t) = (\lambda x.t')$

but the axioms (E.λ) and ($\xi.\simeq$), may not be valid.

There is a canonical way of extending a (Kripke) extensional partial applicative structure $\mathcal{A}$ (i.e. a model of (ext. $\simeq$)) to a λ-structure, namely

> $\Psi^{\mathcal{A}}(f) \stackrel{\Delta}{\simeq} d$, where d is the **unique** element of $\iota^{\mathcal{A}}$ s.t. $\mathrm{app}^{\mathcal{A}}(d, a) \simeq f(a)$ for all $a \in \iota^{\mathcal{A}}$

4.3 Variants of the partial lambda calculus

In this section we introduce some *variants* of the typed partial lambda calculus (similar variants can be defined also for the untyped partial lambda calculus) inspired by the denotational semantics of a language, the *metalanguage*, in the category of cpos and continuous partial functions (see [Plo85]). We consider only the type constructor for *partial function spaces* and the *definition of functions by recursion*, the other type constructors of the metalanguage have been left out for the sake of brevity, and they do not cause any problem.

4.3.1 The monotonic partial lambda calculus

Once sorts are interpreted by posets, partial function spaces ought to have a partial order *uniquely* determined by those on their components. Following Plotkin we take the *most natural* extension to partial functions of the *pointwise* order for total functions, namely $f \leq g \overset{\Delta}{\iff}$ for all x if $f(x)\downarrow$ then $f(x) \leq g(x)$

Definition 4.3.1 The $\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus

The **monotonic partial lambda calculus** *(*$\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$*-calculus) over* $\Sigma_{\mathrm{mon}\lambda}$ *is the formal system axiomatized by* $\mathrm{mon_p} + \xi.\lesssim$ *in the* $\lambda_{\mathrm{p}}\beta\eta$*-calculus over* $\Sigma_{\mathrm{mon}\lambda}$*, where*

- $\Sigma_{\mathrm{mon}\lambda}$ *is a signature for monotonic partial applicative structures*
- $\mathrm{mon_p}$ *is given in Definition 4.1.22 and* $\xi.\lesssim$ *is the axiom*

$$\xi.\lesssim \quad (\forall x\colon \tau.t \lesssim t') \Longrightarrow (\lambda x\colon \tau.t) \leq (\lambda x\colon \tau.t')$$

Remark 4.3.2 One may well consider some other orders instead of the pointwise one (e.g. Berry's order for stable functions), or consider other orders consistent with the pointwise order for total functions, but then the *partial cartesian closed category* defined in [Plo85] is no longer suitable as *intended* model. For instance if the order on functions is

$$f \leq g \overset{\Delta}{\iff} \text{ for all } x \text{ if } f(x)\downarrow \text{ or } g(x)\downarrow, \text{ then } f(x) \leq g(x)$$

then the *appropriate* partial cartesian closed category of cpos is the one where partial morphisms are continuous partial functions (in the sense of Plotkin) whose domain is (not only an open set, but) also a closed set. In this category $1 \rightharpoonup A$ is isomorphic to $A + 1$.

The $\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus is equivalent to **extensional monotonic partial combinatory logic**, i.e. the formal system axiomatized by $\mathrm{mon_p} + \mathrm{CL_p} + \mathrm{ext}.\lesssim$

in LPT over a suitable signature Σ_{monCL} (for monotonic partial combinatory algebras), where (ext.$\lesssim$) is the axiom

$$\text{ext.}\lesssim \quad (\forall z\colon \tau. xz \lesssim yz) \Longrightarrow x \leq y$$

In the mon$\lambda_{\mathrm{p}}\beta$-calculus the inequational fragments have the same expressive power:

Theorem 4.3.3 Expressive power in mon$\lambda_{\mathrm{p}}\beta$

$$TE \;\subset\; Ein \;\equiv\; ECEin$$

4.3.2 Fixed-point operators

The canonical way of having recursive definitions in the setting of the lambda calculus is to introduce a fixed-point operator $\mathrm{Y}_\tau \in \mathrm{Funct}_{(\tau\to\tau)\to\tau}(\Sigma)$ for any sort τ (see **PCF** in [Plo77]).

Remark 4.3.4 In the *standard* model of **PCF** the fixed-point operator Y satisfies the axioms

$$\text{tot.Y} \quad \mathrm{Y}f = f(\mathrm{Y}f)$$

$$\text{tot.}\mu \quad fx \leq x \Longrightarrow \mathrm{Y}f \leq x$$

However, in the category $\mathbf{CPO}_p$ of cpos and continuous partial functions (see Definition 5.1.12) cpos are not required to have a least element, and some continuous function $f\colon D \to D$ may have no fixed-points, e.g. the successor function on the natural numbers with the *flat* order (i.e. $m \leq n$ iff $m = n$). On the other hand, partial function spaces have always a least element and for them one can define a *least fixed-point operator*. Therefore, the only fixed-point operators allowed in a signature are those for function spaces.

Definition 4.3.5 The $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$-calculus

The **fixed-point partial lambda calculus** *($\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$-calculus) over $\Sigma_{\lambda\mathrm{Y}}$ is the formal system axiomatized by* Y *in the $\lambda_{\mathrm{p}}\beta\eta$-calculus over $\Sigma_{\lambda\mathrm{Y}}$, where*

- $\Sigma_{\lambda Y}$ *is a signature for typed partial applicative structures* **with fixed-points**, *i.e. a signature for typed partial applicative structures with* **fixed-point operators** $Y_{\tau_1,\tau_2} \in \mathrm{Const}_{((\tau_1 \to \tau_2) \to \tau_1 \to \tau_2) \to \tau_1 \to \tau_2}(\Sigma_{\lambda Y})$ *for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma_{\lambda Y})$

- Y *is the axiom*

$$\mathrm{Y} \quad \mathrm{Y}F = (\lambda x.F(\mathrm{Y}F)x)$$

Remark 4.3.6 In the untyped $\lambda_p\beta$-calculus there is already a term satisfiying the axiom (Y), therefore this untyped variant is not particularly interesting.

The $\lambda_p \mathrm{Y}\beta\eta$-calculus is equivalent to the formal system axiomatized by the axioms $\mathrm{Y}F \downarrow$ and $\mathrm{Y}Fx \simeq F(\mathrm{Y}F)x$ in extensional partial combinatory logic over a suitable signature Σ_{CLY}.

The axiom (Y) is particularly convenient as a rewriting rule for extending the definition of one-step parallel reduction (see Section 8.2).

In a model of $\lambda_p\beta\eta$ there can be more than one way of interpreting the constant Y so that the axiom (Y) is satisfied. However, in models based on cpos the *intended* interpretation of Y is the *least* fixed-point operator, which is captured axiomatically by the following variant.

Definition 4.3.7 The $\lambda_p\mu\mathrm{Y}\beta\eta$-calculus

The **least fixed-point partial lambda calculus** *($\lambda_p\mu\mathrm{Y}\beta\eta$-calculus) over $\Sigma_{\mathrm{mon}\lambda Y}$ is the formal system axiomatized by $\mu + \mathrm{Y}$ in the* mon$\lambda_p\beta\eta$*-calculus over $\Sigma_{\mathrm{mon}\lambda Y}$, where*

- $\Sigma_{\mathrm{mon}\lambda Y}$ *is a signature for typed monotonic partial applicative structures with fixed-points*

- Y *is given in Definition 4.3.5 and* μ *is the axiom*

$$\mu \quad (\lambda x.Ffx) \leq f \Longrightarrow \mathrm{Y}F \leq f$$

Notation 4.3.8 We introduce some *syntactic sugar* for λY-terms (see [Plo85]), which is particularly convenient in connection with the $\lambda_p\mu\mathrm{Y}\beta\eta$-calculus:

- we write $\bot_{\tau_1,\tau_2}$ for $\mathrm{Y}_{\tau_1,\tau_2}(\mathrm{I})$
- we write $\mu f{:}\,\tau_1 \rightharpoonup \tau_2.\lambda x{:}\,\tau_1.t$ for $\mathrm{Y}_{\tau_1,\tau_2}(\lambda f{:}\,\tau_1 \rightharpoonup \tau_2.\lambda x{:}\,\tau_1.t)$

Remark 4.3.9 The partial lambda calculi with fixed-points are somehow arbitrary, because they try to axiomatize continuous models, that cannot be completely captured in first-order logic. The $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus is definitely preferable to the $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$-calculus, because the interpretation of the fixed-point operator Y is uniquely determined. However, there are very natural improvements for better capturing the intended model based on the category of cpos and continuous partial function, for instance **fixed-point induction**:

Y.ind $A[f{:}=\bot] \wedge (\forall f.A \rightarrow A[f{:}=(\lambda x.Ffx)]) \Longrightarrow A[f{:}=\mathrm{Y}F]$

where A is a conjunction of inequations (see [Sco69]) or more generally ω-*inductive* (see [Plo85]).

In a signature for typed partial applicative structures with fixed-points there is no symbol $\mathrm{Y}_\iota \in \mathrm{Funct}_{(\iota\rightharpoonup\iota)\rightharpoonup\iota}(\Sigma)$ for a fixed-point operator of base sort ι. We will show that such a fixed-point operator, satisfying (tot.Y) and (tot.μ), is definable in the $\lambda_{\mathrm{t}}\mu\mathrm{Y}\beta\eta$-calculus.

Lemma 4.3.10 Least weak fixed-point operator

In a model of the mon$\lambda_{\mathrm{p}}\beta\eta$*-calculus there is a unique element* Y *of sort* $(\iota \rightharpoonup \iota) \rightharpoonup \iota$*, called the* **least weak fixed-point operator***, satisfying the axioms*

w.Y $\mathrm{Y}f \simeq f(\mathrm{Y}f)$

w.μ $(ft) \lesssim t \Longrightarrow (\mathrm{Y}f) \lesssim t$

In the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$*-calculus the term* $\hat{\mathrm{Y}} \stackrel{\Delta}{\equiv} (\mu F{:}\,(\iota \rightharpoonup \iota) \rightharpoonup \iota.\lambda f.f(Ff))$ *satisfies the axioms* (w.Y) *and* (w.μ)

Proof If Y and Y$'$ are two least weak fixed-point operators, then $\mathrm{Y}f \simeq \mathrm{Y}'f$, therefore (by extensionality) $\mathrm{Y} = \mathrm{Y}'$.

It is immediate (from the definition of $\hat{\mathrm{Y}}$ and (Y)) that $\hat{\mathrm{Y}}$ satisfies (w.Y). To prove (w.μ) we introduce a *different* least weak fixed-point operator $\hat{\mathrm{Y}}'$ and show that $\hat{\mathrm{Y}} \leq \hat{\mathrm{Y}}'$. Therefore, $\hat{\mathrm{Y}}$ satisfies (w.Y) because $\hat{\mathrm{Y}}'$ does. Given a variable f

of sort $\iota \rightharpoonup \iota$, let g_f be the term $(\mu g\colon(\iota \rightharpoonup \iota) \rightharpoonup \iota.\lambda x.f(gx))$ and $\hat{\mathrm{Y}}'$ be the term $(\lambda f.g_f\bot)$.

First, we prove that $\hat{\mathrm{Y}}'$ is a least weak fixed-point operator. (w.Y) is immediate, because $\hat{\mathrm{Y}}'f \simeq g_f\bot \simeq f(g_f\bot) \simeq f(\hat{\mathrm{Y}}'f)$. To prove (w.$\mu$), let's assume $(ft)\lesssim t$ and derive $(\hat{\mathrm{Y}}'f)\lesssim t$:

- from $(ft)\lesssim t$, by (β) and $(\xi.\lesssim)$ one derives

  $\lambda y\colon(\iota \rightharpoonup \iota).f((\lambda y.t)y) \leq \lambda y\colon(\iota \rightharpoonup \iota).t$

- by (μ)

  $g_f \leq \lambda y\colon(\iota \rightharpoonup \iota).t$

- by monotonicity, $(\lambda y.t)\bot \simeq t$ (we assume $y \notin \mathrm{FV}(t)$) and $(ft)\lesssim t$

  $f(g_f\bot)\lesssim f((\lambda y\colon(\iota \rightharpoonup \iota).t)\bot) \simeq (ft)\lesssim t$

- by the definition of $\hat{\mathrm{Y}}'$ and (Y)

  $(\hat{\mathrm{Y}}'f) \simeq f(g_f\bot)\lesssim t$

Then, we show that $\hat{\mathrm{Y}}' \leq \hat{\mathrm{Y}}$:

- from the definition of $\hat{\mathrm{Y}}'$ and g_f, by (Y)

  $f(\hat{\mathrm{Y}}'f) \simeq f(f(g_f\bot)) \simeq f(g_f\bot)$

- by $(\xi.\lesssim)$ and definition of $\hat{\mathrm{Y}}'$

  $\lambda f.f(\hat{\mathrm{Y}}'f) \leq \hat{\mathrm{Y}}'$

- by (μ)

  $\hat{\mathrm{Y}} \leq \hat{\mathrm{Y}}'$

∎

Remark 4.3.11 In the model based on cpos and continuous partial functions (more generally, whenever $\bot x$ is *undefined*) $\hat{\mathrm{Y}}$ is the everywhere diverging function $\bot_{\iota \rightharpoonup \iota,\iota}$, this explains the appellation *weak* given to $\hat{\mathrm{Y}}$.

Proposition 4.3.12

The $\lambda_{\mathrm{t}}\mu\mathrm{Y}\beta\eta$*-calculus is equivalent to the* $\mathrm{mon}\lambda_{\mathrm{t}}\beta\eta$*-calculus with least fixed-point operators* $\mathrm{Y}_\tau \in \mathrm{Funct}_{(\tau\rightharpoonup\tau)\rightharpoonup\tau}(\Sigma)$ *for any sort* τ *satisfying the axioms* (tot.Y) *and* (tot.μ) *(see Remark 4.3.4).*

Proof The idea is to identify the fixed-point operator $\mathrm{Y}_{\tau_1\to\tau_2}$ with $\mathrm{Y}_{\tau_1,\tau_2}$ and Y_ι with $\hat{\mathrm{Y}}$. With this identification, it is easy to prove that in the $\lambda_{\mathrm{t}}\beta\eta$-calculus:

- (tot.$\mathrm{Y}_{\tau_1\to\tau_2}$) is equivalent to ($\mathrm{Y}_{\tau_1,\tau_2}$), because $F(\mathrm{Y}F) = (\lambda x{:}\,\tau_1.F(\mathrm{Y}F)x)$
- (tot.$\mu_{\tau_1\to\tau_2}$) is equivalent to (μ_{τ_1,τ_2}), because $Ff = (\lambda x{:}\,\tau_1.Ffx)$
- (tot.Y_ι) is equivalent to (w.Y)
- (tot.μ_ι) is equivalent to (w.μ)

To complete the proof it is enough to note that the axioms (tot.Y_τ) and (tot.μ_τ) (as well as (w.Y_ι) and (w.μ_ι)) determine a unique element Y_τ (see Lemma 4.3.10).
∎

4.4 The lambda calculus of partial elements

In [Plo75] Plotkin introduces the λ_{v}-calculus, a variant of the (pure untyped) λ-calculus *corresponding* to the call-by-value operational semantics and *correct* w.r.t. call-by-value operational equivalence. The inference rules given by Plotkin are directly inspired by the call-by-value operational semantics.

Although, the partial lambda calculus is based on model-theoretic (rather than operational) considerations, it is *correct* w.r.t. call-by-value operational equivalence (by Corollaries 5.4.11 and 5.4.14); moreover, it is able to derive more equivalences of programs than the λ_{v}-calculus, e.g. $\mathrm{I}(xy) \simeq (xy)$ is provable in the $\lambda_{\mathrm{p}}\beta$-calculus, but not in $\lambda_{\mathrm{v}}\beta$-calculus.

In this section we define the analogue of the partial lambda calculus for LPE and iLPE (see Definition 3.3.7): the $\lambda_{\mathrm{pe}}\beta\eta$- and $\lambda_{\mathrm{ipe}}\beta\eta$-calculus. We show that the $\lambda_{\mathrm{pe}}\beta\eta$-calculus *corresponds* to the λ_{p}-calculus (see Theorem 4.4.8), while the $\lambda_{\mathrm{ipe}}\beta$-calculus *corresponds* to Plotkin's $\lambda_{\mathrm{v}}\beta$-calculus (see Theorem 4.4.15).

Definition 4.4.1 The $\lambda_{\mathrm{pe}}\beta\eta$- and $\lambda_{\mathrm{ipe}}\beta\eta$-calculus

The ($\lambda_{\mathrm{ipe}}\beta\eta$-) $\lambda_{\mathrm{pe}}\beta\eta$-calculus is the formal system (on the language of LPE *for* untyped partial applicative structures*, see Definition 4.2.3)* generated *by the inference rules for (*iLPE*)* LPE *and the following set of axioms:*

E.λ $\mathrm{E}(\lambda x.t)$

$\xi.\equiv\ (\forall x.t \equiv t') \Longrightarrow (\lambda x.t) \equiv (\lambda x.t')$

$\beta\ \mathrm{E}(x) \Longrightarrow (\lambda x.t)x \equiv t$

$\eta\ \mathrm{E}(f) \Longrightarrow (\lambda x.fx) \equiv f$

In order to apply the techniques available for first order logic we have to prove that the $\lambda_{\mathrm{pe}}\beta\eta$-calculus is equivalent to a formal system axiomatized in LPE.

Definition 4.4.2 Extensional combinatory logic of partial elements

$\mathrm{CL_{pe}}$ + ext. $\equiv$ *(*$\mathrm{CL_{ipe}}$ + ext. $\equiv$*) is the formal system axiomatized in* LPE *(*iLPE*) by the following axioms:*

K $\mathrm{E}(x), \mathrm{E}(y) \Longrightarrow \mathrm{K}xy \equiv x$

S $\mathrm{E}(x), \mathrm{E}(y), \mathrm{E}(z) \Longrightarrow \mathrm{S}xyz \equiv xz(yz)$

E.S $\mathrm{E}(x), \mathrm{E}(y) \Longrightarrow \mathrm{E}(\mathrm{S}xy)$

ext. $\equiv$ $\mathrm{E}(x), \mathrm{E}(y), (\forall z.xz \equiv yz) \Longrightarrow x \equiv y$

$\mathrm{sCL_{pe}}$+ext. $\equiv$ *(*$\mathrm{sCL_{ipe}}$+ext. $\equiv$*) is the formal system (on the language extended with a* **non-strict** *unary function* f_{K}*) axiomatized in* LPE *(*iLPE*) by the axioms above plus:*

tot.f_{K} $\mathrm{E}(f_{\mathrm{K}}(x))$

f_{K} $\mathrm{E}(y) \Longrightarrow f_{\mathrm{K}}(x)y \equiv x$

Remark 4.4.3 $\mathrm{CL_{pe}}+\mathrm{ext.}\equiv$ is the natural definition of extensional combinatory logic of partial elements. In fact, its axioms are equivalent (in LPE) to the $\_^{\circ}$-translation of the axioms for $\mathrm{CL_p}+\mathrm{ext.}\simeq$. However, it is too weak to be equivalent to the $\lambda_{\mathrm{pe}}\beta\eta$-calculus, so we have introduced $\mathrm{sCL_{pe}}+\mathrm{ext.}\equiv$.

There is an alternative axiomatization of $\mathrm{sCL_{pe}}+\mathrm{ext.}\equiv$, which does not require an extension of the language with f_{K}:

strong $x.\exists y.\forall z.yz \equiv x$

i.e. any partial element x is *represented* by an existing element y. Therefore, the collection of partial elements is uniquely determined by the collection of existing elements.

Lemma 4.4.4 *the* $\lambda_{\mathrm{ipe}}\beta\eta$*-calculus and* $\mathrm{sCL_{ipe}}+\mathrm{ext.}\equiv$ *are equivalent at the level of terms.*

Proof The proof of equivalence follows the same pattern as Section 4.2.1, the only difference is the definition of abstraction $[x]_s t$ in $\mathrm{sCL_{ipe}}+\mathrm{ext.}\equiv$:

- $[x]_s x \stackrel{\Delta}{\equiv} \mathrm{I}$
- $[x]_s x' \stackrel{\Delta}{\equiv} f_{\mathrm{K}}(x')$ if $x \not\equiv x'$
- $[x]_s c \stackrel{\Delta}{\equiv} \mathrm{K}c$
- $[x]_s f_{\mathrm{K}}(t) \stackrel{\Delta}{\equiv} ([x,y,z]xy)([x]_s t)$, where
  $[\_]\_$ is abstraction in $\mathrm{CL_p}$ (see Definition 4.2.14)
- $[x]_s(t_1 t_2) \stackrel{\Delta}{\equiv} \mathrm{S}([x]_s t_1)([x]_s t_2)$

and an additional case in the definition of $\_^{\lambda}$:

- $(f_{\mathrm{K}}(t))^{\lambda} \stackrel{\Delta}{\equiv} (\lambda y.t^{\lambda})$, where $y \notin \mathrm{FV}(t)$

∎

Remark 4.4.5 The abstraction $[x]x' \stackrel{\Delta}{\equiv} \mathrm{K}x'$ satisfies the required properties under the assumption $\mathrm{E}(x')$, but not in general.

Now we can treat the ($\lambda_{\mathrm{ipe}}\beta\eta$-) $\lambda_{\mathrm{pe}}\beta\eta$-calculus as a formal system axiomatized in first order logic of partial elements.

Lemma 4.4.6 $\mathrm{sCL_{ipe}} + \mathrm{ext.} \equiv$ *is an extension of* $\mathrm{CL_{ipe}} + \mathrm{ext.} \equiv$ *conservative over* $\mathrm{Seq} \vdash \mathrm{ESeq}$.

Proof We argue as in Proposition 3.3.16. It is obvious that $\mathrm{sCL_{ipe}} + \mathrm{ext.} \equiv$ is an extension of $\mathrm{CL_{ipe}} + \mathrm{ext.} \equiv$, therefore we have only to prove the other inclusion for the formal systems restricted to $\mathrm{Seq} \vdash \mathrm{ESeq}$.
By Proposition 1.3.5, it is enough to show that for every model $\mathcal{A}$ of $\mathrm{CL_{ipe}}+\mathrm{ext.} \equiv$ there exists a model $\mathcal{A}'$ of $\mathrm{sCL_{ipe}} + \mathrm{ext.} \equiv$ s.t.

- $\mathcal{A}'$ satisfies all sequents satisfied by $\mathcal{A}$
- $\mathcal{A}$ satisfies all sequents in ESeq satisfied by $\mathcal{A}'$

We claim that the required $\mathcal{A}'$ is the model $Cl(\mathcal{A})$, i.e. the smallest substructure of $\mathcal{A}$ containing all existing elements. In fact, any partial element a of $Cl(\mathcal{A})$ is the interpretation of a CL-term t in an environment ρ from variables to **existing** elements, therefore f_{K} can be interpreted by the function mapping a to the interpretation of $[y]t$ ($y \notin \mathrm{FV}(t)$) in ρ. ∎

Remark 4.4.7 The conservative extension result for $\mathrm{CL_{pe}} + \mathrm{ext.} \equiv$ follows immediately from that for $\mathrm{CL_{ipe}} + \mathrm{ext.} \equiv$, because $\mathrm{CL_{pe}} + \mathrm{ext.} \equiv$ is axiomatized in $\mathrm{CL_{ipe}} + \mathrm{ext.} \equiv$ by (in).

Theorem 4.4.8 The correspondence between $\lambda_{\mathrm{p}}\beta\eta$ and $\lambda_{\mathrm{pe}}\beta\eta$

The translation $\_^\circ$ *(of Definition 3.3.13) is a conservative interpretation of the* $\lambda_{\mathrm{p}}\beta\eta$*-calculus in the* $\lambda_{\mathrm{pe}}\beta\eta$*-calculus.*

Proof Since the $\lambda_{\mathrm{p}}\beta\eta$-calculus is equivalent to $\mathrm{CL_p} + \mathrm{ext.} \simeq$ and the $\lambda_{\mathrm{pe}}\beta\eta$-calculus is equivalent to $\mathrm{sCL_{pe}} + \mathrm{ext.} \equiv$, it is enough to prove that $\_^\circ$ is a conservative interpretation of $\mathrm{CL_p} + \mathrm{ext.} \simeq$ in $\mathrm{sCL_{pe}} + \mathrm{ext.} \equiv$:

- the axioms for $\mathrm{CL_{pe}} + \mathrm{ext.} \equiv$ are equivalent (in LPE) to the $\_^\circ$-translation of the axioms for $\mathrm{CL_p} + \mathrm{ext.} \simeq$, therefore by (the remark after) Proposition 3.3.14

  $\_^\circ$ is a conservative interpretation of $\mathrm{CL_p} + \mathrm{ext.} \simeq$ in $\mathrm{CL_{pe}} + \mathrm{ext.} \equiv$

- the image of $\_^\circ$ is included in ESeq, therefore by Lemma 4.4.6

  $\_^\circ$ is a conservative interpretation of $\mathrm{CL_p} + \mathrm{ext.} \simeq$ in $\mathrm{sCL_{pe}} + \mathrm{ext.} \equiv$

To complete the proof we would have to show that the *direct* translation from the $\lambda_{\mathrm{p}}\beta\eta$-calculus to the $\lambda_{\mathrm{pe}}\beta\eta$-calculus is equivalent (in the $\lambda_{\mathrm{pe}}\beta\eta$-calculus) to: first translate in $\mathrm{CL_p}$ + ext. $\simeq$, then translate in $\mathrm{CL_{pe}}$ + ext. $\equiv$ and finally translate in the $\lambda_{\mathrm{pe}}\beta\eta$-calculus. ∎

Remark 4.4.9 Along similar lines, one can also prove that $\_^{\circ}$ is a conservative interpretation of the $\lambda_{\mathrm{p}}\beta$-calculus into the $\lambda_{\mathrm{pe}}\beta$-calculus. $\_^{\circ}$ is not a relative interpretation of the $\lambda_{\mathrm{p}}\beta\eta$-calculus in the $\lambda_{\mathrm{ipe}}\beta\eta$-calculus, as will be clear from the conservative interpretation of the $\lambda_{\mathrm{v}}\beta$-calculus in the $\lambda_{\mathrm{ipe}}\beta$-calculus.

Now we turn our attention to the $\lambda_{\mathrm{v}}\beta$-calculus and characterize it in terms of the $\lambda_{\mathrm{ipe}}\beta$-calculus.

Definition 4.4.10 ([Plo75]) The $\lambda_{\mathrm{v}}\beta$-calculus $\lambda_{\mathrm{v}}\beta$

refl $M = M$

$$\text{symm}\ \frac{M = N}{N = M}$$

$$\text{trans}\ \frac{M = N \qquad N = P}{M = P}$$

$$\text{cong.}\cdot\ \frac{M = N \qquad P = Q}{MP = NQ}$$

$$\xi\ \frac{M = N}{(\lambda x.M) = (\lambda x.N)}$$

β_v $(\lambda x.M)N = M[x{:=}N]$ *if N is a value*

where a **value** *is a λ-term which is either a variable or a λ-abstraction*

Remark 4.4.11 We do not consider extra constants and δ-rules, as done in [Plo75], but such an extension does not affect the results below (nor their proofs).

λ_{v}-models are not considered in [Plo75], but it is pointed out that: "in a model one would expect that free variables would be interpreted as universally quantified over a restricted domain". This remark suggests that $M = N$ in the λ_{v}-calculus should be interpreted as $\mathrm{E}(\eta) \Longrightarrow M \equiv N$ in the λ_{ipe}-calculus, where η is the set of free variables in M or N.

Definition 4.4.12 The revisited $J\lambda_{\text{ipe}}\beta$-calculus

Consider the following fragments of ESeq *(see Definition 3.3.13):*

- *ETE-statements are the sequents*
  $\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(t)$
- *ES-equations are the sequents*
  $\mathrm{E}(\eta) \Longrightarrow t_1 \equiv t_2$

The inference rules for $\mathrm{Th}_{\lambda_{\text{ipe}}}(\beta)\lambda_{\text{ipe}}\beta$ *on* $ETE \cup ES$ *are:*

$$\to\Longrightarrow \quad \mathrm{E}(\eta), \mathrm{E}(x) \Longrightarrow \mathrm{E}(x)$$

$$\mathrm{E}.\cdot.1 \quad \frac{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(M \cdot N)}{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(M)}$$

$$\mathrm{E}.\cdot.2 \quad \frac{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(M \cdot N)}{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(N)}$$

$$\mathrm{refl} \quad \mathrm{E}(\eta) \Longrightarrow M \equiv M$$

$$\mathrm{symm} \quad \frac{\mathrm{E}(\eta) \Longrightarrow M \equiv N}{\mathrm{E}(\eta) \Longrightarrow N \equiv M}$$

$$\mathrm{trans} \quad \frac{\mathrm{E}(\eta) \Longrightarrow M \equiv N \qquad \mathrm{E}(\eta) \Longrightarrow N \equiv P}{\mathrm{E}(\eta) \Longrightarrow M \equiv P}$$

$$\mathrm{cong}.\cdot \quad \frac{\mathrm{E}(\eta) \Longrightarrow M \equiv P \qquad \mathrm{E}(\eta) \Longrightarrow N \equiv Q}{\mathrm{E}(\eta) \Longrightarrow M \cdot N \equiv P \cdot Q}$$

$$\mathrm{cong.E} \quad \frac{\mathrm{E}(\eta) \Longrightarrow M \equiv N \qquad \mathrm{E}(\eta) \Longrightarrow \mathrm{E}(M)}{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(N)}$$

$$\mathrm{E}.\lambda \quad \mathrm{E}(\eta) \Longrightarrow \mathrm{E}(\lambda x.M)$$

$$\xi.\equiv \quad \frac{\mathrm{E}(\eta), \mathrm{E}(x) \Longrightarrow M \equiv N}{\mathrm{E}(\eta) \Longrightarrow \lambda x.M \equiv \lambda x.N} \quad x \textit{ not free in } \mathrm{E}(\eta)$$

$$\beta \quad \frac{\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(N)}{\mathrm{E}(\eta) \Longrightarrow (\lambda x.M)N \equiv M[x := N]}$$

$\mathrm{Th}_{\lambda_{\text{ipe}}}(\beta)$ *is the subset of* $ETE \cup ES$ *generated by the inference rules above.*

Remark 4.4.13 In the $\lambda_{\rm pe}$-calculus (but not in the $\lambda_{\rm ipe}$-calculus) the fragments ETE and ES have the same expressive power as the $\_^\circ$-translation of the fragments TE and S (given in Definition 4.1.1) respectively.

Proposition 4.4.14 *The set* $\mathrm{Th}_{\lambda_{\rm ipe}}(\beta)$ *is the set of sequents in* $ETE \cup ES$ *derivable in the* ($J\lambda_{\rm ipe}\beta$-) $K\lambda_{\rm ipe}\beta$*-calculus.*

Proof The inference rules for $\mathrm{Th}_{\lambda_{\rm ipe}}(\beta)$ are the restriction of the revisited inference rules for the $J\lambda_{\rm ipe}\beta$-calculus (see Definition 3.3.9) to the $ETE \cup ES$ fragment.

The revisited inference rules for the $J\lambda_{\rm pe}\beta$-calculus satisfy the conditions of Proposition 1.2.10, when I is the inclusion of $ETE \cup ES$ into SSeq, therefore both sets of rules derive the same sequents in the $ETE \cup ES$ fragment.

Since the revisited inference rules are sound and complete, $J\lambda_{\rm pe}(\beta)$ is the set of sequents in $ETE \cup ES$ valid in all models, i.e. derivable in the $J\lambda_{\rm ipe}\beta$-calculus (from no assumptions).

Finally, we can replace the $J\lambda_{\rm ipe}\beta$-calculus with the $K\lambda_{\rm ipe}\beta$-calculus, because they coincide on $ETE \cup ES$ (see Proposition 3.3.11). ∎

The connection between Plotkin's $\lambda_{\rm v}$-calculus and the $\lambda_{\rm ipe}\beta$-calculus is fully described by the following correspondence.

Theorem 4.4.15 The correspondence between $\lambda_{\rm v}\beta$ and $\lambda_{\rm ipe}\beta$

For all λ*-terms* M *and* N*:*

1. *if* $\lambda_{\rm v}\beta \vdash M = N$, *then* $\lambda_{\rm ipe}\beta \vdash \mathrm{E}(\eta) \Longrightarrow M \equiv N$

2.
   - *if* $\lambda_{\rm ipe}\beta \vdash \mathrm{E}(\eta) \Longrightarrow M \equiv N$, *then* $\lambda_{\rm v}\beta \vdash M = N$
   - *if* $\lambda_{\rm ipe}\beta \vdash \mathrm{E}(\eta) \Longrightarrow \mathrm{E}(M)$, *then* M **has a value**, *i.e. there exists a value* V *s.t.* $\lambda_{\rm v}\beta \vdash M = V$

where η *is a type environments, s.t.* $\mathrm{FV}(M,N) \subseteq \mathrm{dom}(\eta)$.

Proof The first statement is proved by induction on the derivation of $M = N$ by the inference rules for the $\lambda_{\rm v}\beta$-calculus (see Definition 4.4.10). The following admissible inference rules in $\mathrm{Th}_{\lambda_{\rm ipe}}(\beta)$ make the proof completely straightforward and clarify the role of the assumptions about η:

- $$\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(N) \qquad \text{if } N \text{ is a value}$$

  - if N is a variable, say x, then $x \in \mathrm{dom}(\eta)$, therefore
    $\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(x)$ by $(\rightarrow\Longrightarrow)$
  - if N is a λ-abstraction, then
    $\mathrm{E}(\eta) \Longrightarrow \mathrm{E}(N)$ by $(\mathrm{E}.\lambda)$

- $$\frac{\mathrm{E}(\eta') \Longrightarrow A}{\mathrm{E}(\eta) \Longrightarrow A}\ \mathrm{FV}(A) \subseteq \mathrm{dom}(\eta)$$

  by (thinning), we can assume w.l.o.g. that $\eta \subseteq \eta'$

  - by substituting all variables not in η with I, we derive (by (subst))
    $\mathrm{E}(\eta), \mathrm{E}(\mathrm{I}) \Longrightarrow A$
  - Since E(I) (by $(\mathrm{E}.\lambda)$), we derive (by (cut))
    $\mathrm{E}(\eta) \Longrightarrow A$.

The second statement is proved by induction on the derivation of $\mathrm{E}(\eta) \Longrightarrow A$ in $\mathrm{Th}_{\lambda_{\mathrm{ipe}}}(\beta)$ by the revisited inference rules (see Definition 4.4.12). The only interesting cases are:

- $(\rightarrow\Longrightarrow) \Longrightarrow$

  Since $\mathrm{E}(\eta) \Longrightarrow A$ is in $ETE \cup ES$, then A must be $\mathrm{E}(x)$ for some variable x, which clearly has a value.

- $(\mathrm{E}.\cdot.1)$ and $(\mathrm{E}.\cdot.2) \Longrightarrow$

  By IH MN has a value, say V, and we have to prove that also M and N have a value. The claim follows from results in [Plo75]:

  - since values are closed w.r.t. β_v-reduction, by the Church-Rosser property (see Theorem 2 of [Plo75]) there exists a value Z s.t.
    (MN) (and V) β_v-reduces to Z

- by Corollary 1 of [Plo75] there exists a value Z' s.t.
  $(MN) \to_v^* Z'$, where
  $\to_v$ is the **one-step head reduction** defined on Page 136 of [Plo75]
- by arguing like in the proof of Theorem 4 of [Plo75],
  there exist two values K and L s.t.
  $M \to_v^* K$, $N \to_v^* L$ (and $(KL) \to_v^* Z'$), therefore
  both M and N have a value.

• $(\beta) \Longrightarrow$

By IH N has a value, say V, and we have to prove that $(\lambda x.M)N = M[x{:=}N]$:

- since $N = V$, we can replace N with V in any context, in particular
  $(\lambda x.M)N = (\lambda x.M)V$ and $M[x{:=}V] = M[x{:=}N]$
- since V is a value, by (β_v)
  $(\lambda x.M)V = M[x{:=}V]$, therefore
  $(\lambda x.M)N = M[x{:=}N]$

∎

Remark 4.4.16 A correspondence, like that stated in Theorem 4.4.15, can be established between the $\lambda_{\mathrm{ipe}}\beta\eta$-calculus and the $\lambda_{\mathrm{v}}\beta\eta$-calculus, i.e. Plotkin's call-by-value lambda calculus with an instance (η_v) of the η-axiom which is correct w.r.t. call-by-value operational equivalence:

$$\eta_v \;\; (\lambda x.Nx) = N \qquad \text{if } N \text{ is a value and } x \notin \mathrm{FV}(N)$$

However, before proving this correspondence, one would have to extend the results in [Plo75] first, by taking into account (η_v).

Chapter 5

Model-theoretic results

We gather some general techniques and results applied in the following chapter for comparing the deductive power of some formal systems, but they may also have different applications. The common feature of these techniques is that they have a model-theoretic flavour, i.e. they involve constructing models or proving properties of models. Apart from this, they are quite unrelated, therefore a more detailed discussion of each technique is postponed to its corresponding section.

5.1 Type hierarchies

In this section we consider some model constructions for the typed partial lambda calculus and variants. Some of these constructions are familiar from the literature, e.g. the *full type hierarchy* (see [Fri75, Plo80]), while the others are straightforward modifications of it. All of them are particular instances of a more general construction, which amounts to fixing an *interpretation* of the base sort ι in a *partial cartesian closed category* C, i.e. an object $[\![\iota]\!]$ of C, and to extend $[\![\_]\!]$ to function spaces by interpreting $[\![\tau_1 \rightharpoonup \tau_2]\!]$ as the partial function space $[\![\tau_1]\!] \rightharpoonup [\![\tau_2]\!]$ in C (see [Mog86, Ros86]).

We introduce three cartesian closed categories of *sets with structure*:

- the category of **sets** and functions

  (the full type hierarchy)

- the category of **posets** and **monotonic** functions

  (the full **monotonic** type hierarchy)

- the category of **cpos** and **continuous** functions

  (the full **continuous** type hierarchy)

and we give an explicit definition of their objects, morphisms and function spaces. Each of these categories can be equipped with a suitable *domain structure*, i.e. a collection $\mathcal{M}$ of monos with certain properties, so that it becomes a *non-trivial* partial cartesian closed category, called:

- the category of _ and _ **partial** functions

  (the full _ **partial** type hierarchy)

Instead of giving the domain structure, we give an explicit definition of the partial morphisms (as partial functions) and partial function spaces.

Remark 5.1.1 In a category of sets with structure a partial morphism from a to b is, in general, a morphism from a sub-structure (subobject) of a to b. A partial function g from (the carrier) $|a|$ (of a) to $|b|$ may correspond to more than one partial morphism from a to b, because two sub-structures of a may have the set $\mathrm{dom}(g)$ as carrier. However, if $\mathcal{A}$ is a structure with underlying set X, then a subset Y of X will be identified (as common practice) with $\mathcal{A}$ **restricted** to Y, i.e. the **biggest** sub-structure of $\mathcal{A}$ with Y as carrier.

These categories of sets with structure have an intuitionistic counterpart, because the proof of their *relevant properties* can be carried out in **IZF**. In particular, if we replace **SET** with $\mathbf{SET}^{\mathrm{K}}$, then the resulting (partial) cartesian closed category of *Kripke sets with structure* is called:

- the category of **Kripke** _ over K and _

  (the **Kripke** full _ type hierarchy over K)

Remark 5.1.2 The classical type hierarchies are a particular case of Kripke type hierarchies, namely when K is the category with one object and one morphism.

For the rest of this section we use the operation Σ^{τ} introduced in Notation 2.1.2 and fix the following signatures:

Definition 5.1.3 Conventions for signatures

$$\Sigma_{set} \quad \subset \quad \Sigma_{po}$$

- *Σ_{set} is a signature for typed partial applicative structures, which does not contain partial order symbols*

- *Σ_{po} is a signature for typed monotonic partial applicative structures, which has only one base sort (ι), no constant symbols, the only function symbols are applications and the only predicate symbols are: $\_\downarrow$, $\_=\_$ and $\_\leq\_$*

Remark 5.1.4 The assumptions about base sorts of Σ_{po} have been made in order to get simpler proofs. The other assumptions (on constant, function and predicate symbols of Σ_{po}) make sure that any of the full _ hierarchies (defined in the sequel) are uniquely determined by the choice of base type.

5.1.1 Categories of (Kripke) structures

Definition 5.1.5 Sets

- **SET** *is the category of sets and functions*

  $X \rightarrow_{set} Y$ *is the* **set of functions** *from* X *to* Y

- $\mathbf{SET}_p$ *is the category of sets and partial functions*

  $X \rightharpoonup_{set} Y$ *is the* **set of partial functions** *from* X *to* Y

  The **extensional order** $\leq_e$ *on* $X \rightharpoonup_{set} Y$ *is*

$$f \leq_e g \stackrel{\Delta}{\Longleftrightarrow} \text{for all } x \in X \text{ if } f(x)\downarrow, \text{ then } f(x) = g(x)$$

Remark 5.1.6 The extensional order can be defined in any category of partial morphisms, namely $f \leq_e g \stackrel{\Delta}{\Longleftrightarrow}$

- the domain of f is included in the domain of g

- f is equal to g restricted to the domain of f

In order to define the *Kripke counterparts* of **SET** and $\mathbf{SET}_p$ we need some auxiliary definitions:

Definition 5.1.7 Consistent families and quasi-natural transformations

If K *is a small category,* X *and* Y *are functors from* K *to* **SET** *and* $a \in \mathrm{K}$, *then*

- *F is a* **natural transformation** *from X to Y ($F: X \to Y$)* $\overset{\Delta}{\iff}$
  $F \equiv < F_a | a \in \mathrm{K} > \in \prod_{f:a \in \mathrm{K}} X(a) \to_{set} Y(a)$ *and for all* $f: a \to b$

$$\begin{tikzcd} X(a) \arrow[r, "F_a"] \arrow[d, "X(f)"'] \arrow[dr, phantom, "="] & Y(a) \arrow[d, "Y(f)"] \\ X(b) \arrow[r, "F_b"'] & Y(b) \end{tikzcd}$$

- *x is a* **global element** *of X ($x \in X$)* $\overset{\Delta}{\iff}$
  $x \equiv < x_a | a \in \mathrm{K} > \in \prod_{a \in \mathrm{K}} X(a)$ *and for all* $f: a \to b$

$$\begin{tikzcd} x_a \in X(a) \arrow[d, "X(f)"'] \\ x_b \in X(b) \end{tikzcd}$$

- *F is a* **quasi-natural transformation** *from X to Y ($F: X \rightharpoonup Y$)* $\overset{\Delta}{\iff}$
  $F \equiv < F_a | a \in \mathrm{K} > \in \prod_{f:a \in \mathrm{K}} X(a) \rightharpoonup_{set} Y(a)$ *and for all* $f: a \to b$

$$\begin{tikzcd} X(a) \arrow[r, harpoon, "F_a"] \arrow[d, "X(f)"'] \arrow[dr, phantom, "\leq_e"] & Y(a) \arrow[d, "Y(f)"] \\ X(b) \arrow[r, harpoon, "F_b"'] & Y(b) \end{tikzcd}$$

- *s is a* **consistent family of functions** *from X to Y at a* $\overset{\Delta}{\iff}$
  $s \equiv < s_f | f: a \to b > \in \prod_{f:a \to b} X(b) \to_{set} Y(b)$ *and for all* $f: a \to b$ *and* $g: b \to c$

$$\begin{tikzcd} X(b) \arrow[r, "s_f"] \arrow[d, "X(g)"'] \arrow[dr, phantom, "="] & Y(b) \arrow[d, "Y(g)"] \\ X(c) \arrow[r, "s_{g \circ f}"'] & Y(c) \end{tikzcd}$$

- *s is a* **consistent family of partial functions** *from X to Y at a* $\overset{\Delta}{\iff}$
  $s \equiv < s_f | f: a \to b > \in \prod_{f:a \to b} X(b) \rightharpoonup_{set} Y(b)$ *and for all* $f: a \to b$ *and* $g: b \to c$

$$\begin{tikzcd} X(b) \arrow[r, harpoon, "s_f"] \arrow[d, "X(g)"'] \arrow[dr, phantom, "\leq_e"] & Y(b) \arrow[d, "Y(g)"] \\ X(c) \arrow[r, harpoon, "s_{g \circ f}"'] & Y(c) \end{tikzcd}$$

Remark 5.1.8 A more general definition of quasi-natural transformation, for an arbitrary *p-category*, can be found [Obt86].

A global element of $X\colon \mathrm{K} \to \mathbf{SET}$ corresponds to a natural transformation from the constant functor $\lambda a\colon \mathrm{K}.\{*\}$ (the terminal object of $\mathbf{SET}^{\mathrm{K}}$) to X.

A consistent family of (partial) functions from X to Y at a is just a (quasi-)natural transformation from $X \circ \pi$ to $Y \circ \pi$, where $\pi\colon (\mathrm{K} \downarrow a) \to \mathrm{K}$ is the projection functor from the comma category $(\mathrm{K} \downarrow a)$ to K.

Definition 5.1.9 Kripke sets

- K**SET** *is the category of Kripke sets over* K *and functions, i.e. the category* $\mathbf{SET}^{\mathrm{K}}$ *of functors from* K *to* **SET** *and natural transformations*

  $X \to_{\mathrm{K}set} Y$ *is the Kripke set of functions from* X *to* Y*, i.e. the functor from* K *to* **SET** *s.t.*

  - $(X \to_{\mathrm{K}set} Y)(a)$ *is the set of consistent families of functions from* X *to* Y *at* a
  - $(X \to_{\mathrm{K}set} Y)(f\colon a \to b)$ *is the function s.t.* $s \mapsto < s_{g \circ f} | g\colon b \to c >$

  The **evaluation morphism** $\mathrm{eval}\colon (X \to_{\mathrm{K}set} Y) \times X \to Y$ *is the natural transformation s.t.* $\mathrm{eval}_a(s, x) = s_{\mathrm{id}_a}(x)$

- K$\mathbf{SET}_p$ *is the category of Kripke sets over* K *and partial functions, i.e. the category of functors from* K *to* **SET** *and quasi-natural transformations*

  $X \rightharpoonup_{\mathrm{K}set} Y$ *is the Kripke set of partial functions from* X *to* Y*, i.e. the functor from* K *to* **SET** *s.t.*

  - $(X \rightharpoonup_{\mathrm{K}set} Y)(a)$ *is the set of consistent families of partial functions from* X *to* Y *at* a
  - $(X \rightharpoonup_{\mathrm{K}set} Y)(f\colon a \to b)$ *is the function s.t.* $s \mapsto < s_{g \circ f} | g\colon b \to c >$

  The **partial evaluation morphism** $\mathrm{eval}\colon (X \rightharpoonup_{\mathrm{K}set} Y) \times X \rightharpoonup Y$ *is the quasi-natural transformation s.t.* $\mathrm{eval}_a(s, x) \simeq s_{\mathrm{id}_a}(x)$

Remark 5.1.10 The *function space presheaf* $X \to_{\mathrm{K}set} Y$ is a standard construction in topos theory, while the *partial function space presheaf* $X \rightharpoonup_{\mathrm{K}set} Y$ is a crossing between the *function space* and *power set presheaf*

Similar categories can be defined by *replacing* sets with posets and (partial) functions with monotonic (partial) functions:

Definition 5.1.11 Posets

- **PO** *is the category of posets and monotonic functions, where*
  - f *is a* **monotonic function** *from* X *to* $Y \overset{\Delta}{\iff}$ *for all* $x, y \in X$ *if* $x \leq_X y$, *then* $f(x) \leq_Y f(y)$

  $X \to_{po} Y$ *is the* **poset of monotonic functions** *from* X *to* Y *with the* **pointwise** *partial order, i.e.* $f \leq_{X \to_{po} Y} g \overset{\Delta}{\iff} f(x) \leq_Y g(x)$ *for all* $x \in X$

- $\mathbf{PO}_p$ *is the category of posets and monotonic partial functions, where*
  - f *is a* **monotonic partial function** *from* X *to* $Y \overset{\Delta}{\iff}$ *for all* $x, y \in X$ *if* $x \leq_X y$ *and* $f(x)\downarrow$, *then* $f(x) \leq_Y f(y)$

  $X \rightharpoonup_{po} Y$ *is the* **poset of monotonic partial functions** *from* X *to* Y *with the* **pointwise** *partial order, i.e.* $f \leq_{X \rightharpoonup_{po} Y} g \overset{\Delta}{\iff}$ *for all* $x \in X$ *if* $f(x)\downarrow$, *then* $f(x) \leq_Y g(x)$

- K**PO** *is the category of Kripke posets over* K *and monotonic functions, i.e. the category* $\mathbf{PO}^{\mathrm{K}}$ *of functors from* K *to* **PO** *and natural transformations*

  $X \to_{\mathrm{K}po} Y$ *is the* **Kripke poset of monotonic functions** *from* X *to* Y, *i.e. the functor from* K *to* **PO** *s.t.*
  - $(X \to_{\mathrm{K}po} Y)(a)$ *is the poset of consistent families of monotonic functions from* X *to* Y *at* a *with the* **componentwise** *partial order, i.e.*

    $$s \leq s' \overset{\Delta}{\iff} s_f \leq_{X(b) \to_{po} Y(b)} s'_f \text{ for all } f\colon a \to b$$

  - $(X \to_{\mathrm{K}po} Y)(f\colon a \to b)$ *is the monotonic function s.t.* $s \mapsto < s_{g \circ f} | g\colon b \to c >$

- $\mathrm{K}\mathbf{PO}_p$ *is the category of Kripke posets over* K *and monotonic partial functions, i.e. the category of functors from* K *to* **PO** *and quasi-natural transformations*

  $X \rightharpoonup_{\mathrm{K}po} Y$ *is the* **Kripke poset of monotonic partial functions** *from* X *to* Y*, i.e. the functor from* K *to* **PO** *s.t.*

  - $(X \rightharpoonup_{\mathrm{K}po} Y)(a)$ *is the poset of consistent families of monotonic partial functions from* X *to* Y *at* a *with the* componentwise *partial order*
  - $(X \rightharpoonup_{\mathrm{K}po} Y)(f\colon a \to b)$ *is the monotonic function s.t.* $s \mapsto < s_{g\circ f} | g\colon b \to c >$

To get models with least fixed-points the categories in Definition 5.1.11 are not completely satisfactory, because even when the least fixed-points exists, e.g. in the full monotonic (partial) type hierarchy with base type a complete lattice (see Definitions 5.1.19 and 5.1.20), they do not satisfy fixed-point induction (see Section 5.1.3). Therefore we consider the following sub-category of **PO**:

Definition 5.1.12 Complete posets

- **CPO** *is the category of cpos and continuous functions, where*

  - X *is a* **cpo** $\stackrel{\Delta}{\Longleftrightarrow}$ X *is a poset s.t. any* ω*-chain* $x_1 \leq \ldots \leq x_i \leq \ldots$ *has a lub* $\sqcup_i x_i$
  - f *is a* **continuous function** *from* X *to* Y $\stackrel{\Delta}{\Longleftrightarrow}$ f *is monotonic and preserves the lub of* ω*-chains*

  $X \to_{cpo} Y$ *is the* **cpo of continuous functions** *from* X *to* Y *with the pointwise partial order*

- $\mathbf{CPO}_p$ *is the category of cpos and continuous partial functions (see [Plo85]), where*

  - f *is a* **continuous partial function** *from* X *to* Y $\stackrel{\Delta}{\Longleftrightarrow}$ f *is a monotonic partial function and for any* ω*-chain* x_i *if* $f(\sqcup_i x_i)\downarrow$*, then there exists* n *s.t.* $f(x_n)\downarrow$ *and* $\sqcup_i f(x_{n+i}) = f(\sqcup_i x_i)$

  $X \rightharpoonup_{cpo} Y$ *is the* **cpo of continuous partial functions** *from* X *to* Y *with the pointwise partial order*

- K**CPO** *is the category of Kripke cpos over* K *and continuous functions, i.e. the category* $\mathbf{CPO}^{\mathrm{K}}$ *of functors from* K *to* **CPO** *and natural transformations*

  $X \to_{\mathrm{K}cpo} Y$ *is the* **Kripke cpo of continuous functions** *from* X *to* Y*, i.e. the functor from* K *to* **CPO** *s.t.*

  - $(X \to_{\mathrm{K}cpo} Y)(a)$ *is the cpo of consistent families of continuous functions from* X *to* Y *at* a *with the* componentwise *partial order*
  - $(X \to_{\mathrm{K}cpo} Y)(f\colon a \to b)$ *is the continuous function s.t.* $s \mapsto < s_{g \circ f} | g\colon b \to c >$

- K$\mathbf{CPO}_p$ *is the category of Kripke cpos over* K *and continuous partial functions, i.e. the category of functors from* K *to* **CPO** *and quasi-natural transformations*

  $X \rightharpoonup_{\mathrm{K}cpo} Y$ *is the* **Kripke cpo of continuous partial functions** *from* X *to* Y*, i.e. the functor from* K *to* **CPO** *s.t.*

  - $(X \rightharpoonup_{\mathrm{K}cpo} Y)(a)$ *is the cpo of consistent families of continuous partial functions from* X *to* Y *at* a *with the* componentwise *partial order*
  - $(X \rightharpoonup_{\mathrm{K}cpo} Y)(f\colon a \to b)$ *is the continuous function s.t.* $s \mapsto < s_{g \circ f} | g\colon b \to c >$

5.1.2 Full type hierarchies

Definition 5.1.13 The Classical Full Type Hierarchy

Given a set X*, the* **classical full type hierarchy** *with base type* X *is the* Σ_{set}*-structure* $\mathcal{A}$ *defined by:*

- $\mathcal{A} \upharpoonright \Sigma_{set}^{\tau} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f, x) \triangleq f(x)$

where X_τ *is defined by induction on the structure of* τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \to \tau_2} \triangleq X_{\tau_1} \to_{set} X_{\tau_2}$

It is easy to see that $\mathcal{A}$ is extensional *(i.e. it satisfies the axiom* (ext. $\simeq$) *of Definition 4.2.9), and it can be extended* in a unique way *to a model of the $\lambda_{\mathrm{t}}\beta\eta$-calculus (i.e there is exactly one way to interpret the combinators* K *and* S *that makes this possible)*

Definition 5.1.14 The Classical Full Partial Type Hierarchy

Given a set X, the **classical full partial type hierarchy** *with base type X is the Σ_{set}-structure $\mathcal{A}$ defined by:*

- $\mathcal{A}\lceil\Sigma^{\tau}_{set} \stackrel{\Delta}{=} X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f, x) \stackrel{\Delta}{\simeq} f(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \stackrel{\Delta}{=} X$
- $X_{\tau_1\rightarrow\tau_2} \stackrel{\Delta}{=} X_{\tau_1} \rightharpoonup_{set} X_{\tau_2}$

$\mathcal{A}$ can be extended in a unique way to a model of the $\lambda_{\mathrm{p}}\beta\eta$-calculus

Definition 5.1.15 The Kripke Full Type Hierarchy over K

Given a Kripke set X over K*, the* **Kripke full type hierarchy** *over* K *with base type X is the Kripke Σ_{set}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}\lceil\Sigma^{\tau}_{set} \stackrel{\Delta}{=} X_\tau$
- $\mathrm{app}^{\mathcal{B}(\alpha)}_{\tau_1,\tau_2}(s, x) \stackrel{\Delta}{=} s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \stackrel{\Delta}{=} X$
- $X_{\tau_1\rightarrow\tau_2} \stackrel{\Delta}{=} X_{\tau_1} \rightarrow_{\mathrm{K}set} X_{\tau_2}$

$\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\lambda_{\mathrm{t}}\beta\eta$-calculus

Definition 5.1.16 The Kripke Full Partial Type Hierarchy over K

Given a Kripke set X over K*, the* **Kripke full partial type hierarchy** *over* K *with base type X is the Kripke Σ_{set}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}\lceil\Sigma^{\tau}_{set} \stackrel{\Delta}{=} X_\tau$

- $\mathrm{app}^{\mathcal{B}(\alpha)}_{\tau_1,\tau_2}(s,x) \stackrel{\Delta}{\simeq} s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \stackrel{\Delta}{=} X$
- $X_{\tau_1 \rightharpoonup \tau_2} \stackrel{\Delta}{=} X_{\tau_1} \rightharpoonup_{\mathrm{K}set} X_{\tau_2}$

$\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\lambda_{\mathrm{p}}\beta\eta$-calculus

Remark 5.1.17 The (Kripke) full partial type hierarchy gives models of the $\lambda_{\mathrm{p}}\beta\eta$-calculus with **equality test** D_τ: $\mathrm{Const}_{\tau\rightharpoonup\tau\rightharpoonup\tau}(\Sigma_{set})$ for any $\tau \in \mathrm{Sort}(\Sigma_{set})$ satisfying the axioms

$$Dxx \downarrow$$

$$Dxy \downarrow \Longrightarrow x = y$$

(see also the notion of **equoidal category with types** in 5.2.1 of [Obt86]). Note that mon$\lambda_{\mathrm{p}}\beta$-models with equality testing are trivial, i.e. each sort has at most one element.

5.1.3 Full monotonic type hierarchies

There are model constructions for the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus, similar to the ones described in Definitions 5.1.13, 5.1.14, 5.1.15 and 5.1.16, but with sets *replaced* by posets and (partial) functions *replaced* by monotonic (partial) functions.

If the base types are (Kripke) complete lattices, then the models below (see Definitions 5.1.19, 5.1.20, 5.1.21 and 5.1.22) can be extended (in a unique way) to a model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus. This possibility relies on two facts:

- if Y is a complete lattice, and X is a poset, then $X \rightarrow_{po} Y$ and $X \rightharpoonup_{po} Y$ are complete lattices
- if Z is a complete lattice and $f: Z \rightarrow Z$ is a monotonic function, then f has the least fixed-point z

For Kripke complete lattices there are similar results (compare with Section 5.1.4). Therefore in a (Kripke) full monotonic (partial) type hierarchy with base type a

(Kripke) complete lattice we can define the least fixed-point operator as the least fixed-point of the monotonic function

$$\lambda Y{:}\,(Z \to_{po} Z) \to_{po} Z.\lambda F{:}\,Z \to_{po} Z.\lambda x{:}\,X.F(YF)x$$

However, in some full monotonic type hierarchies with base type a complete lattice the least fixed-points do not satisfy fixed-point induction (Y.ind) (see Remark 4.3.9).

Example 5.1.18 The full monotonic type hierarchy with base type the complete lattice $\omega + 1$ (i.e. the solution to the domain equation $X \cong X_\perp$) has least fixed-points, but they do not satisfy fixed-point induction.

Let $s, g{:}\,\omega + 1 \to \omega + 1$ be the monotonic functions s.t.

$$s(n) = \begin{cases} n+1 & \text{if } n < \omega \\ \omega & \text{if } n = \omega \end{cases}$$

$$g(n) = \begin{cases} 0 & \text{if } n < \omega \\ 1 & \text{if } n = \omega \end{cases}$$

$g(0) = 0$ and $g(x) = 0 \to g(sx) = 0$, but $g(\mathrm{Y}s) \neq 0$ for any fixed-point operator, because the only fixed-point of s is ω.

Definition 5.1.19 The Classical Full Monotonic Type Hierarchy

Given a poset X, the **classical full monotonic type hierarchy** *with base type X is the Σ_{po}-structure $\mathcal{A}$ defined by:*

- $\mathcal{A}\restriction\Sigma^{\tau}_{po} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f, x) \triangleq f(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \to \tau_2} \triangleq X_{\tau_1} \to_{po} X_{\tau_2}$

$\mathcal{A}$ can be extended in a unique way to a model of the $\mathrm{mon}\lambda_{\mathrm{t}}\beta\eta$*-calculus. Moreover, if X is a complete lattice, then $\mathcal{A}$ can be extended* in a unique way *to a model of the $\lambda_{\mathrm{t}}\mu\mathrm{Y}\beta\eta$-calculus (i.e there is exactly one way to interpret the combinators* K, S *and* Y *that makes this possible)*

Definition 5.1.20 The Classical Full Monotonic Partial Type Hierarchy

Given a poset X, the **classical full monotonic partial type hierarchy** *with base type X is the Σ_{po}-structure $\mathcal{A}$ defined by:*

- $\mathcal{A}{\restriction}\Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f,x) \overset{\Delta}{\simeq} f(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \rightharpoonup \tau_2} \triangleq X_{\tau_1} \rightharpoonup_{po} X_{\tau_2}$

$\mathcal{A}$ can be extended in a unique way to a model of the $\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$*-calculus. Moreover, if X is a complete lattice, then $\mathcal{A}$ can be extended in a unique way to a model of the* $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$*-calculus*

Definition 5.1.21 The Kripke Full Monotonic Type Hierarchy over K

Given a Kripke poset X over K*, the* **Kripke full monotonic type hierarchy** *over* K *with base type X is the Kripke Σ_{po}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}{\restriction}\Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{B}(\alpha)}_{\tau_1,\tau_2}(s,x) \triangleq s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \rightharpoonup \tau_2} \triangleq X_{\tau_1} \rightarrow_{\mathrm{K}po} X_{\tau_2}$

$\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\mathrm{mon}\lambda_{\mathrm{t}}\beta\eta$*-calculus. Moreover, if X is a complete lattice, then $\mathcal{B}$ can be extended in a unique way to a Kripke model of the* $\lambda_{\mathrm{t}}\mu\mathrm{Y}\beta\eta$*-calculus*

Definition 5.1.22 The Kripke Full Monotonic Partial Type Hierarchy over K

Given a Kripke poset X over K*, the* **Kripke full monotonic partial type hierarchy** *over* K *with base type X is the Kripke Σ_{po}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}\restriction\Sigma^{\tau}_{po} \stackrel{\Delta}{=} X_\tau$
- $\mathrm{app}^{\mathcal{B}(\alpha)}_{\tau_1,\tau_2}(s,x) \stackrel{\Delta}{\simeq} s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \stackrel{\Delta}{=} X$
- $X_{\tau_1 \to \tau_2} \stackrel{\Delta}{=} X_{\tau_1} \rightharpoonup_{\mathrm{K}po} X_{\tau_2}$

$\mathcal{B}$ can be extended in a unique way to a Kripke model of the mon$\lambda_{\mathrm{p}}\beta\eta$*-calculus. Moreover, if X is a complete lattice, then $\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus*

5.1.4 Full continuous type hierarchies

There are model constructions for the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus satisfying fixed-point induction (see Remark 4.3.9) similar to the models described in Definitions 5.1.19, 5.1.20, 5.1.19 and 5.1.22, but with posets *replaced* by cpos and monotonic (partial) functions by continuous (partial) functions.

The possibility of interpreting Y relies on the fact that if Z is a cpo with a least element $\bot_Z$ and $f: Z \to Z$ is a continuous function, then f has the least fixed-point z, which is the lub of the ω-chain z_i s.t. $z_0 = \bot_Z$ and $z_{i+1} = f(z_i)$.

Any partial function space $Z = X \rightharpoonup_{cpo} Y$ has a least element, namely the everywhere divergent function $\bot_{X,Y}$ from X to Y. Therefore in a full continuous partial type hierarchy we can define the least fixed-point operator as the least fixed-point of the continuous function

$$\lambda Y\colon (Z \rightharpoonup_{cpo} Z) \rightharpoonup_{cpo} Z.\lambda F\colon Z \rightharpoonup_{cpo} Z.\lambda x\colon X.F(YF)x$$

For Kripke cpos one can proceed in exactly the same way, more explicitly:

- $\bot$ is the least element of a Kripke poset Z over K iff $\bot$ is a global element and $\bot_a$ is the least element of the poset $Z(a)$
- if a Kripke cpo Z has a least element and $f: Z \to Z$ is a *continuous* natural transformation, then the least fixed-point of f is the global element z of Z s.t. z_a is the least fixed-point of f_a

- any partial function space $Z = X \rightharpoonup_{\mathrm{K}cpo} Y$ has a least element, namely the global element $\bot_{X,Y}$ s.t. $(\bot_{X,Y})_a =< \bot_{X(b),Y(b)} | f: a \to b >$

Therefore, the least fixed-point operator can be defined also in a Kripke full continuous partial type hierarchy.

On the other hand, for (Kripke) full continuous type hierarchies we must start with a base type X that has a least element, in order to have an interpretation for Y, in fact:

- if X and Y are cpos and Y has a least element $\bot_Y$, then $X \to_{cpo} Y$ has a least element, namely the constant function $\lambda x \in X.\bot_Y$

Definition 5.1.23 The Classical Full Continuous Type Hierarchy

Given a cpo X, the **classical full continuous type hierarchy** *with base type X is the Σ_{po}-structure $\mathcal{A}$ defined by:*

- $\mathcal{A} \upharpoonright \Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f, x) \triangleq f(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \to \tau_2} \triangleq X_{\tau_1} \to_{cpo} X_{\tau_2}$

If X has a least element, then $\mathcal{A}$ can be extended in a unique way to a model of the $\lambda_t\mu Y\beta\eta$-calculus

Definition 5.1.24 The Classical Full Continuous Partial Type Hierarchy

Given a cpo X, The **classical full continuous partial type hierarchy** *with base type X is the Σ_{po}-structure $\mathcal{A}$ defined by:*

- $\mathcal{A} \upharpoonright \Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}^{\mathcal{A}}_{\tau_1,\tau_2}(f, x) \overset{\Delta}{\simeq} f(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1 \to \tau_2} \triangleq X_{\tau_1} \rightharpoonup_{cpo} X_{\tau_2}$

$\mathcal{A}$ can be extended in a unique way to a model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus

Definition 5.1.25 The Kripke Full Continuous Type Hierarchy over K

Given a Kripke cpo X over K*, the* **Kripke full continuous type hierarchy** *over* K *with base type X is the Kripke Σ_{po}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}\restriction\Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}_{\tau_1,\tau_2}^{\mathcal{B}(\alpha)}(s,x) \triangleq s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1\to\tau_2} \triangleq X_{\tau_1} \to_{\mathrm{K}cpo} X_{\tau_2}$

If X has a least element, then $\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\lambda_{\mathrm{t}}\mu\mathrm{Y}\beta\eta$-calculus

Definition 5.1.26 The Kripke Full Continuous Partial Type Hierarchy over K

Given a Kripke cpo X over K*, the* **Kripke full continuous partial type hierarchy** *over* K *with base type X is the Kripke Σ_{po}-structure $\mathcal{B}$ over* K *defined by:*

- $\mathcal{B}\restriction\Sigma_{po}^{\tau} \triangleq X_\tau$
- $\mathrm{app}_{\tau_1,\tau_2}^{\mathcal{B}(\alpha)}(s,x) \overset{\Delta}{\simeq} s_{\mathrm{id}_\alpha}(x)$

where X_τ is defined by induction on the structure of τ

- $X_\iota \triangleq X$
- $X_{\tau_1\to\tau_2} \triangleq X_{\tau_1} \rightharpoonup_{\mathrm{K}cpo} X_{\tau_2}$

$\mathcal{B}$ can be extended in a unique way to a Kripke model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus

5.2 Logical relations and partial homomorphisms

The following is a well-known property of algebras:

> a sub-algebra and an homomorphic image of an algebra $\mathcal{A}$ satisfy all equations satisfied by $\mathcal{A}$. In other words, if there exists a **partial surjective homomorphism** from $\mathcal{A}$ to $\mathcal{A}'$, i.e. a surjective homomorphism from a sub-algebra of $\mathcal{A}$ to $\mathcal{A}'$, then $\mathcal{A}'$ satisfies all equations satisfied by $\mathcal{A}$.

The notion of partial surjective homomorphism, called simply **partial homomorphism**, has been extended to type structures (see [Fri75]), where it cannot be split into the more basic notions of *extensional* sub-structure and surjective homomorphism.

Partial homomorphisms are a particular kind of **logical relation** (see [Sta85b, Sta83]). The general idea behind logical relations is that the relation at type $\sigma \rightharpoonup \tau$ is uniquely determined by the relations at type σ and τ.

In this section we define logical relations and partial homomorphisms for (partial algebras and) typed partial applicative structures. For our purposes we need also Kripke logical relations (see [MM87]), but they are quite cumbersome to define directly in terms of Kripke models, therefore

> all logical relations will be defined axiomatically in the language of LPT over suitable signatures and their properties will be proved by formal derivation in JLPT.

It is useful to introduce the relevant notions for partial algebras first, since most of their properties have nothing to do with the type structure.

Definition 5.2.1 The language of Σ-relations

*If Σ is a signature for partial algebras, then the signature Σ_{lr} for Σ-**relations** between two partial algebras is $\Sigma \uplus \Sigma'$ extended with a binary relation $\mathrm{R}_\tau \in \mathrm{Pred}_{\tau,\tau'}(\Sigma_{lr})$ for any $\tau \in \mathrm{Sort}(\Sigma)$, where*

- *Σ' is a* **disjoint copy** *of Σ*
- *$\_'$ is a* **bijective translation** *from the language over Σ to the language over Σ', which maps a (term, formula or) sequent onto the* **corresponding** *(term, formula or) sequent:*
  - $(x)' \triangleq x'$*, where x' is a* new variable
  - $(c)' \triangleq c'$*, where c' is the symbol of Σ'* corresponding *to c*

- $f(\overline{t})' \triangleq f'(\overline{t}')$
- $p(\overline{t})' \triangleq p'(\overline{t}')$
- *the translation commutes with connectives and quantifiers*

We write $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *for a (Kripke)* Σ_{lr}*-structure and say that* R *is a* Σ*-relation between* $\mathcal{B}$ *and* $\mathcal{B}'$*.*

Notation 5.2.2 We write $t_1 \underset{\sim}{\mathrm{R}}_\tau t'_2$ for the formula $t_1 \downarrow \rightarrow t_1 \mathrm{R}_\tau t'_2$ and $t_1 \widetilde{\mathrm{R}}_\tau t'_2$ for $(t_1 \downarrow \vee t'_2 \downarrow) \rightarrow t_1 \mathrm{R}_\tau t'_2$.

If f and g' are I-indexed families in Term_h^Σ and $\mathrm{Term}_h^{\Sigma'}$ respectively, then we write $f \mathrm{R}_h g'$ for the set of formulae $\{f_i \mathrm{R}_{h_i} g'_i | i \in I\}$ (see Notation 2.1.7). Moreover, if η is a type environment for Σ, then we write $\eta \mathrm{R} \eta'$ for the set of formulae $\{x \mathrm{R}_{\eta(x)} x' | x \in \mathrm{dom}(\eta)\}$.

5.2.1 Correspondences between partial algebras

There are two notions of logical relation between partial algebras, called **semicorrespondence** and **correspondence** in 3.1 and 3.5 of [Sch85] (see also [Sch87]), that generalize the notions of homomorphism and strong homomorphism between partial algebras (as defined in [Bur82]). These notions of correspondence have been introduced to study behavioural inclusion and equivalence of partial algebras.

We establish the main properties of correspondences and partial homomorphisms of partial algebras (see Lemma 5.2.5 and Theorem 5.2.6) that will be further developed when considering typed partial applicative structures. We define partial homomorphisms (of partial algebras) so that they *preserve* satisfaction of ECE-equations (see Definition 4.1.1), i.e. the most expressive among the equations. There are more general notions of partial homomorphism, that *preserve* satisfaction of E-equations only. Since the equational fragments have the same expressive power in the $\lambda_{\mathrm{p}}\beta\eta$-calculus (see Theorem 4.2.8), then there is only one *natural* notion of partial homomorphism for typed partial applicative structures.

Definition 5.2.3 Correspondences and Partial Homomorphisms

If Σ *is a signature for partial algebras, then*

- *a* Σ*-relation* R *is a* Σ-**semicorrespondence** *from* $\mathcal{B}$ *to* $\mathcal{B}'$ *iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* MLR_0*, where* MLR_0 *is the set of axioms*

  MLR.c $\; c\,\mathrm{R}_\tau\, c'$

  MLR.f $\; \overline{x}\,\mathrm{R}_{\overline{\tau}}\,\overline{x}' \Longrightarrow f(\overline{x})\,\utilde{\mathrm{R}}_\tau\, f'(\overline{x}')$

- *a* Σ*-relation* R *is a* Σ-**correspondence** *between* $\mathcal{B}$ *and* $\mathcal{B}'$ *iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* LR_0*, where* LR_0 *is the set of axioms*

  LR.c $\; c\,\mathrm{R}_\tau\, c'$

  LR.f $\; \overline{x}\,\mathrm{R}_{\overline{\tau}}\,\overline{x}' \Longrightarrow f(\overline{x})\,\widetilde{\mathrm{R}}_\tau\, f'(\overline{x}')$

- *a* Σ*-relation* R *is a* Σ-**partial homomorphism of partial algebras** *from* $\mathcal{B}$ *onto* $\mathcal{B}'$ *(notation* $\mathrm{R}\colon\mathcal{B} \rightharpoonup \mathcal{B}'$*) iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* PHom_0*, where* PHom_0 *is* LR_0 *plus the set of axioms*

  LR.p $\; \overline{x}\,\mathrm{R}_{\overline{\tau}}\,\overline{x}', p(\overline{x}) \Longrightarrow p'(\overline{x}')$

  surj $\; \forall x'\colon \tau'.\exists x\colon \tau. x\,\mathrm{R}_\tau\, x'$

Remark 5.2.4 Semicorrespondences can be viewed as the morphisms of a suitable category, whose objects are Σ-structures, and similarly for correspondences and partial homomorphisms of partial algebras.

A correspondence R between $\mathcal{B}$ and $\mathcal{B}'$ can be equivalently defined as a semicorrespondence from $\mathcal{B}$ to $\mathcal{B}'$ s.t. its *opposite* R^{op} is a semicorrespondence from $\mathcal{B}'$ to $\mathcal{B}$. From this definition it is obvious that the opposite of a correspondence is a correspondence.

The name partial homomorphism is justified by the fact that whenever $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ is a model of PHom_0, then each component R_τ is the graph of a surjective partial function (functionality follows from (LR.$=$)). We denote such a partial function by $H\_$.

Lemma 5.2.5 Corresponding terms are related

Given a term $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$*, then*

1. $\eta \mathrel{\mathrm{R}} \eta' \Longrightarrow t \mathrel{\underset{\sim}{\mathrm{R}}} t'$ *is derivable from* MLR_0 *in* $J\mathrm{LPT}$

   and similarly $\eta \mathrel{\mathrm{R}} \eta' \Longrightarrow t \mathrel{\tilde{\mathrm{R}}} t'$ *is derivable from* LR_0 *in* $J\mathrm{LPT}$

2. $\eta \mathrel{\mathrm{R}} \eta' \Longrightarrow t \downarrow \to t' \downarrow$ *is derivable from* MLR_0 *in* $J\mathrm{LPT}$

   and similarly $\eta \mathrel{\mathrm{R}} \eta' \Longrightarrow t \downarrow \longleftrightarrow t' \downarrow$ *is derivable from* LR_0 *in* $J\mathrm{LPT}$

where t' *is the term corresponding to* t

Proof The first two statements are proved by induction on the structure of t, while the other two statements are an immediate consequence of the first two, since $M \mathrel{\underset{\sim}{\mathrm{R}}} N' \to (M \downarrow \to N' \downarrow)$ and $M \mathrel{\tilde{\mathrm{R}}} N' \to (M \downarrow \longleftrightarrow N' \downarrow)$. ∎

Theorem 5.2.6 Existentially conditioned atomic formulae are preserved

Given an existentially conditioned atomic formula $A \in \mathrm{Seq}^{\Sigma}$*, i.e. a sequent* $\eta.\overline{M} \downarrow \Longrightarrow p(\overline{t}) \in \mathrm{Seq}^{\Sigma}$*, let* $A_1 \in \mathrm{Form}^{\Sigma}(\eta)$ *be the formula* $\wedge(\overline{M} \downarrow) \to p(\overline{t})$*, then*

1. $\eta \mathrel{\mathrm{R}} \eta' \Longrightarrow A_1 \to A'_1$ *is derivable from* $\mathrm{LR}_0 + \mathrm{LR}.p$ *in* $J\mathrm{LPT}$

2. A' *is derivable from* A *plus* PHom_0 *in* $J\mathrm{LPT}$

where A' *is the sequent corresponding to* A

Proof For simplicity we assume that η is $x\colon\tau$ and $\overline{M}$ is $\langle M_1 \rangle$, so that A is the sequent $x\colon\tau.M_1 \downarrow \Longrightarrow p(\overline{t})$ and A_1 is the formula $M_1 \downarrow \to p(\overline{t})$.

To prove the first statement we show that $p'(\overline{t}')$ is derivable from the conjunction of $x \mathrel{\mathrm{R}} x'$, $M_1 \downarrow \to p(\overline{t})$ and $M'_1 \downarrow$:

- since $M'_1 \downarrow$, by Lemma 5.2.5

  $M_1 \downarrow$

- since $M_1 \downarrow \to p(\overline{t})$, by Modus Ponens

  $p(\overline{t})$

- by $(\mathrm{LR}.p)$

  $p'(\overline{t}')$

To prove the second statement we identify the sequents $x\!:\tau.A_1$ and A :

- By the first statement

  $x \mathrel{\mathrm{R}} x' \Longrightarrow A_1 \to A_1'$

- By the inference rule ($\Longrightarrow\to$),

  $x \mathrel{\mathrm{R}} x', A_1 \Longrightarrow A_1'$

- By A and the inference rule (cut),

  $x \mathrel{\mathrm{R}} x' \Longrightarrow A_1'$

- By the inference rule ($\exists \Longrightarrow$),

  $(\exists x\!:\tau.x \mathrel{\mathrm{R}} x') \Longrightarrow A_1'$

- By $(surj)$ and the inference rule (cut),

  A'

∎

5.2.2 Logical relations between typed partial applicative structures

We extend the results in the previous section to typed partial applicative structures. To make this extension as smooth as possible:

- we fix a signature Σ_{CL} for typed partial combinatory algebras and a signature Σ for typed partial applicative structures obtained from Σ_{CL} by removing K, S and c_f

- we require that the translation $\_'$ (in Definition 5.2.1) from the language over Σ (Σ_{CL}) to the language over Σ' (Σ'_{CL}) commutes with function spaces. i.e. $(\tau_1 \rightharpoonup \tau_2)' \equiv \tau_1' \rightharpoonup \tau_2'$ (and canonical representatives, i.e. $(c_f)' \equiv c_{f'}$).

The following definition extends the notions introduced in Definition 5.2.3 to typed partial applicative structures.

Definition 5.2.7 Logical Relations and Partial Homomorphisms

If Σ is a signature for typed partial applicative structures, then

- *a* Σ*-relation* R *is a* Σ-**m-logical relation** *from* $\mathcal{B}$ *to* $\mathcal{B}'$ *iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* MLR*, where* MLR *is* MLR_0 *(of Definition 5.2.3) plus the axiom*

$$\text{ext.}\underset{\sim}{\mathrm{R}} \quad (\forall x, x'.x \,\mathrm{R}_{\tau_1}\, x' \to fx \,\underset{\sim}{\mathrm{R}}_{\tau_2}\, f'x') \Longrightarrow f \,\mathrm{R}_{\tau_1 \to \tau_2}\, f'$$

- *a* Σ*-relation* R *is a* Σ-**logical relation** *between* $\mathcal{B}$ *and* $\mathcal{B}'$ *iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* LR*, where* LR *is* LR_0 *plus the axiom*

$$\text{ext.}\tilde{\mathrm{R}} \quad (\forall x, x'.x \,\mathrm{R}_{\tau_1}\, x' \to fx \,\tilde{\mathrm{R}}_{\tau_2}\, f'x') \Longrightarrow f \,\mathrm{R}_{\tau_1 \to \tau_2}\, f'$$

- *a* Σ*-relation* R *is a* Σ-**partial homomorphism** *from* $\mathcal{B}$ *onto* $\mathcal{B}'$ *(notation* $\mathrm{R} : \mathcal{B} \rightharpoonup \mathcal{B}'$*) iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ *is a model of* PHom*, where* PHom *is* PHom_0 *plus the axiom* (ext.$\tilde{\mathrm{R}}$)

Remark 5.2.8 The "m", in "m-logical", stands for monotonic, because the axioms (MLR.f) and (ext.$\underset{\sim}{\mathrm{R}}$) are similar to that for monLPT and the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus. Moreover, a particular kind of m-logical relation will be used to construct models of the typed monotonic partial lambda calculus.

Unlike correspondences, logical relations may not be m-logical relations (because of (ext.$\underset{\sim}{\mathrm{R}}$)).

If $\mathcal{B}$ is a typed partial applicative structure with an everywhere divergent function $\perp$ in any function space, and there is an m-logical relation $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$ whose components are (graphs of) surjective partial functions, then $\mathcal{B}'$ has trivial function spaces, since $\perp \mathrm{R}\, f$ for any f.

Proposition 5.2.9 *A* Σ*-logical relation* R *is uniquely determined by its base components* R_ι*, where* ι *ranges over base sorts. More precisely for any* $\tau_1, \tau_2 \in$ $\mathrm{Sort}(\Sigma)$

$$f \,\mathrm{R}_{\tau_1 \to \tau_2}\, f' \Longleftrightarrow \forall x{:}\, \tau_1, x'{:}\, \tau_1'.x \,\mathrm{R}_{\tau_1}\, x' \to fx \,\tilde{\mathrm{R}}_{\tau_2}\, f'x'$$

Similarly if R *is a* Σ*-m-logical relation, then for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma)$

$$f \,\mathrm{R}_{\tau_1 \to \tau_2}\, f' \Longleftrightarrow \forall x{:}\, \tau_1, x'{:}\, \tau_1'.x \,\mathrm{R}_{\tau_1}\, x' \to fx \,\underset{\sim}{\mathrm{R}}_{\tau_2}\, f'x'$$

Proof $f \, \mathrm{R}_{\tau_1 \to \tau_2} \, f' \iff \forall x{:}\,\tau_1, x'{:}\,\tau_1'.x \, \mathrm{R}_{\tau_1} \, x' \;\to\; fx \, \tilde{\mathrm{R}}_{\tau_2} \, f'x'$ follows immediately from (LR.app) and (ext. $\tilde{\mathrm{R}}$). The corresponding result for m-logical relations follows from (MLR.app) and (ext. $\underset{\sim}{\mathrm{R}}$). ∎

The following lemma shows that for any pair of typed partial combinatory algebras $\mathcal{B}$ and $\mathcal{B}'$ a Σ-logical relation between $\mathcal{B}\restriction\Sigma$ and $\mathcal{B}'\restriction\Sigma$ is also a Σ_{CL}-correspondence (indeed a Σ_{CL}-logical relation) between $\mathcal{B}$ and $\mathcal{B}'$.

Lemma 5.2.10 *If Σ_{CL} is a signature for typed partial combinatory algebras, then*

- (LR.K), (LR.S) *are derivable from* ext. $\tilde{\mathrm{R}}$ + LR.app + $\mathrm{CL_p}$ + $\mathrm{CL'_p}$ *in* JLPT
- (LR.c_f) *is derivable from* ext. $\tilde{\mathrm{R}}$ + LR.app + LR.f + $\mathrm{CL_p}$ + $\mathrm{CL'_p}$ *in* JLPT

Proof We prove only (LR.c_f), for an unary function f. Because of (ext. $\tilde{\mathrm{R}}$), it is enough to show that $x \, \mathrm{R}_{\tau_1} \, x'$ implies $c_f x \, \tilde{\mathrm{R}}_{\tau_2} \, c'_f x'$.

- Assume that

  $x \, \mathrm{R}_{\tau_1} \, x'$
- by (LR.f)

  $f(x) \, \tilde{\mathrm{R}} \, f'(x')$
- by (c_f), i.e. $c_f x \simeq f(x)$,

  $c_f x \, \tilde{\mathrm{R}}_{\tau_2} \, c'_f x'$

∎

By the previous lemma (see Lemma 5.2.10) and the equivalence between the $\lambda_{\mathrm{p}}\beta\eta$-calculus over Σ and $\mathrm{CL_p}$ + ext. $\simeq$ over Σ_{CL} (see Theorem 4.2.19), it is straightforward to derive from Lemma 5.2.5 and Theorem 5.2.6 their analogues for logical relations and partial homomorphisms.

Lemma 5.2.11 Fundamental Lemma of logical relations
Given a λ-term $t \in \mathrm{Term}^{\Sigma}_{\tau}(\eta)$, then

- $\eta \, \mathrm{R} \, \eta' \Longrightarrow t \, \underset{\sim}{\mathrm{R}} \, t'$ *is derivable from* MLR *in the $J\lambda_{\mathrm{p}}\beta\eta$-calculus*

  and similarly $\eta \, \mathrm{R} \, \eta' \Longrightarrow t \, \tilde{\mathrm{R}} \, t'$ *is derivable from* LR *in the $J\lambda_{\mathrm{p}}\beta\eta$-calculus* *where t' is the λ-term corresponding to t*

Theorem 5.2.12 Existentially conditioned atomic formulae are preserved

Given an existentially conditioned atomic formula $A \in \mathrm{Seq}^{\Sigma}$, i.e. a sequent $\eta.\overline{M}\downarrow \Longrightarrow p(\overline{t}) \in \mathrm{Seq}^{\Sigma}$, then

- *A' is derivable from A plus* PHom *in the $\lambda_{\mathrm{p}}\beta\eta$-calculus*

where A' is the sequent corresponding to A

5.2.3 Logical preorders on typed partial applicative structures

In this section we consider a particular kind of m-logical relation, called **logical preorder** on a typed partial applicative structure. The main application of logical preorders is for constructing models of the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus, in the *same way* as models of the $\lambda_{\mathrm{t}}\beta\eta$-calculus (and its extensions) are constructed as quotients of a typed combinatory algebra by a *logical partial equivalence relation* on it (see [BTC87]). In Section 5.3 this technique is used to construct a *closed term model* of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus.

Definition 5.2.13 The language of Σ-relations

*If Σ is a signature for partial algebras, then the signature Σ_{lpo} for Σ-**relations** on a partial algebra is Σ extended with a binary relation $\mathrm{R}_{\tau} \in \mathrm{Pred}_{\tau,\tau}(\Sigma_{lpo})$ for any $\tau \in \mathrm{Sort}(\Sigma)$.*

We write $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}$ for a (Kripke) Σ_{lpo}-structure and say that R is a Σ-relation on $\mathcal{B}$.

Notation 5.2.14 By analogy with the notation for logical relations (see after Definition 5.2.1), we write $t_1 \underset{\sim}{\mathrm{R}}_{\tau} t_2$ for the formula $t_1\downarrow \rightarrow t_1 \mathrm{R}_{\tau} t_2$ and $t_1 \widetilde{\mathrm{R}}_{\tau} t_2$ for $(t_1\downarrow \vee t_2\downarrow) \rightarrow t_1 \mathrm{R}_{\tau} t_2$. If f and g are I-indexed families in $\mathrm{Term}^{\Sigma}_{h}$, then we write $f \mathrm{R}_{h}\, g$ for the set of formulae $\{f_i \mathrm{R}_{h_i} g_i | i \in I\}$.

Definition 5.2.15 Logical Preorder

If Σ is a signature for typed partial applicative structures, then

- *a Σ-relation R is a Σ-**logical preorder** on $\mathcal{B}$ iff $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}$ is a model of* LPO*, where* LPO *is the set of axioms*

  MLR.c $\;c\, \mathrm{R}_{\tau}\, c'$

$$\text{MLR}.f \quad \overline{x}\,\mathrm{R}_{\overline{\tau}}\,\overline{x}' \Longrightarrow f(\overline{x})\,\underset{\sim}{\mathrm{R}}_{\tau}\,f'(\overline{x}')$$

$$\text{ext}.\underset{\sim}{\mathrm{R}} \quad (\forall x, x'.x\,\mathrm{R}_{\tau_1}\,x' \to fx\,\underset{\sim}{\mathrm{R}}_{\tau_2}\,f'x') \Longrightarrow f\,\mathrm{R}_{\tau_1 \to \tau_2}\,f'$$

$$\text{refl}.\,\mathrm{R} \quad x\,\mathrm{R}_{\tau}\,x$$

$$\text{trans}.\,\mathrm{R} \quad x_1\,\mathrm{R}_{\tau}\,x_2, x_2\,\mathrm{R}_{\tau}\,x_3 \Longrightarrow x_1\,\mathrm{R}_{\tau}\,x_3$$

If Σ is a signature for monotonic partial algebras, then A Σ-relation R *on a $\mathcal{B}$ is* **consistent** *with $\leq$ iff* $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}$ *is a model of*

$$\text{cons}.\,\mathrm{R} \quad x \leq_{\tau} x', x'\,\mathrm{R}_{\tau}\,y, y \leq_{\tau} y' \Longrightarrow x\,\mathrm{R}_{\tau}\,y'$$

Remark 5.2.16 A Σ-logical preorder $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}$ would be a Σ-m-logical relation (s.t. each R_{τ} is a preorder), if it were not for the fact that in the signature Σ_{lpo} we do not have two disjoint signatures Σ and Σ', as in the signature Σ_{lr} for m-logical relations, because we want a preorder.

If Σ is a signature for monotonic partial algebras, then the axiom (cons. R) makes sense also for a Σ_{lr}-structure $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}'$. For a logical preorder R *consistency* with $\leq$ is equivalent to $\leq \subseteq \mathrm{R}$, because of reflexivity.

Proposition 5.2.17 *If* R *is a logical preorder* of partial applicative structures *on $\mathcal{B}$, then* $f\,\mathrm{R}_{\tau_1 \to \tau_2}\,g \Longleftrightarrow (\forall a.fa\,\underset{\sim}{\mathrm{R}}_{\tau_2}\,ga)$

Proof The implication from left to right follows immediately from (MLR.app) and reflexivity ($a\,\mathrm{R}_{\tau_1}\,a$). For the other implication it is enough (by (ext. $\underset{\sim}{\mathrm{R}}$)) to derive $fa\,\underset{\sim}{\mathrm{R}}_{\tau_2}\,gb$ from $a\,\mathrm{R}_{\tau_1}\,b$:

- by the assumption $\forall a.fa\,\underset{\sim}{\mathrm{R}}_{\tau_2}\,ga$

  $fa\,\underset{\sim}{\mathrm{R}}_{\tau_2}\,ga$

- by reflexivity

  $g\,\mathrm{R}_{\tau_1 \to \tau_2}\,g$

- by (MLR.app) and the assumption $a \mathrel{\mathrm{R}_{\tau_1}} b$

  $ga \mathrel{\underset{\sim}{\mathrm{R}}_{\tau_2}} gb$

- by transitivity

  $fa \mathrel{\underset{\sim}{\mathrm{R}}_{\tau_2}} gb$

∎

The following definition extends a standard construction for monotonic algebras (see Theorem 2.4.11 of [Sto86] for instance), and can be generalized straightforwardly to Kripke monotonic partial algebras.

Definition 5.2.18 The quotient of a structure

If $\mathcal{B}$ is a monotonic partial Σ-algebra (for simplicity we assume that the only predicate symbols of Σ are: $\_\downarrow$, $\_ = \_$ and $\_ \leq \_$) and $\mathrm{R} \subseteq \mathcal{B} \times \mathcal{B}$ is a consistent Σ-logical preorder, then the **quotient** *$\mathcal{B}/\mathrm{R}$ of $\mathcal{B}$ by R is the monotonic partial Σ-algebra defined by:*

1. $\tau^{\mathcal{B}/\mathrm{R}} \triangleq \{[a] | a \in \tau^{\mathcal{B}}\}$ *, where*

   $[a] \triangleq \{b | a \mathrel{\mathrm{R}} b \wedge b \mathrel{\mathrm{R}} a\}$

2. $c^{\mathcal{B}/\mathrm{R}} \triangleq [c^{\mathcal{B}}]$

3. $f^{\mathcal{B}/\mathrm{R}}([\overline{a}]) \overset{\Delta}{\simeq} \begin{cases} [f^{\mathcal{B}}(\overline{a})] & \text{if } f^{\mathcal{B}}(\overline{a})\downarrow \\ undefined & otherwise \end{cases}$

4. $[a] \leq^{\mathcal{B}/\mathrm{R}} [b] \overset{\Delta}{\iff} a \mathrel{\mathrm{R}} b$

The **projection morphism** *H^{R} is the Σ-partial homomorphism of partial algebras which maps a to $[a]$ (actually it is total)*

Theorem 5.2.19 Properties of $\mathcal{B}/\mathrm{R}$

If $\mathcal{B}$ is a monotonic partial applicative structure and R is a consistent logical preorder on $\mathcal{B}$, then $\mathcal{B}/\mathrm{R}$ is an extensional monotonic partial applicative structure *and the morphism H^{R}, which maps a to $[a]$, is a* partial homomorphism. *Moreover, if $\mathcal{B}$ is a partial combinatory algebra, then $\mathcal{B}/\mathrm{R}$ is a model of the* $\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$*-calculus.*

Proof We prove only that $\mathcal{B}/\mathrm{R}$ satisfies (ext.$\lesssim$) and that the morphism H^{R} satisfies (ext.$\tilde{\mathrm{R}}$). (ext.$\lesssim$) follows immediately from Proposition 5.2.17. Since $H^{\mathrm{R}}(a) = [a]$, in order to establish the validity of (ext.$\tilde{\mathrm{R}}$) it is enough to derive $[f] = [g]$ from $\forall a.[fa] \simeq [ga]$:

- by definition of $[\_]$ and the assumption

  $\forall a.fa \, \tilde{\mathrm{R}} \, ga$, which is equivalent to

  $(\forall a.fa \, \underset{\sim}{\mathrm{R}} \, ga \wedge ga \, \underset{\sim}{\mathrm{R}} \, fa)$, which is equivalent to

  $(\forall a.fa \, \underset{\sim}{\mathrm{R}} \, ga) \wedge (\forall a.ga \, \underset{\sim}{\mathrm{R}} \, fa)$

- by Proposition 5.2.17

  $f \, \mathrm{R} \, g \wedge g \, \mathrm{R} \, f$

- by definition of $[\_]$

  $[f] = [g]$

∎

5.2.4 Existence of partial homomorphisms

In this section we give sufficient conditions for the existence of partial homomorphisms between partial applicative structures. As a matter of fact, these partial homomorphisms H have a *right-inverse* In (see below), and the proofs of existence *construct* the partial homomorphism H and its right inverse In by combined induction on types.

Definition 5.2.20 Partial retraction

*(H, In) is a Σ-**partial retraction** from $\mathcal{B}'$ onto $\mathcal{B}$ iff*

- *H is a Σ-partial homomorphism from $\mathcal{B}$ to $\mathcal{B}'$ (see remark after Definition 5.2.3)*

- *In is a Σ-morphism from $\mathcal{B}'$ to $\mathcal{B}$*

- *In is a* right-inverse *of H, i.e $H_\tau \circ In_\tau = \mathrm{id}$ for any $\tau \in \mathrm{Sort}(\Sigma)$*

Remark 5.2.21 There are Σ-partial homomorphisms without a right-inverse, unless the category of (Kripke) Σ-structures (over K) satisfies the *axiom of choice*: "any surjective map has a right-inverse". For instance, the category of sets satisfies the axiom of choice, but the category of Kripke sets over $\{\bot < \top\}$ does not. Σ-partial retractions (where Σ is a signature for partial applicative structures) should be compared with the *partial retraction systems* in [AL85]

For the rest of this section we use the operation Σ^τ introduced in Notation 2.1.2 and the signatures Σ_{set} and Σ_{po} of Definition 5.1.3.

Theorem 5.2.22 Existence of Σ_{set}-partial homomorphisms

If $\mathcal{B}'$ is an extensional (Kripke) Σ_{set}-structure (over K*), i.e. it satisfies* (ext. $\simeq$)*, and $\mathcal{B}_\iota$ is a (Kripke) Σ^ι_{set}-structure (over* K*) s.t. there exists a partial retraction (H_ι, In_ι) from $\mathcal{B}'\restriction\Sigma^\iota_{set}$ onto $\mathcal{B}_\iota$, then it is possible to extend (H_ι, In_ι) to a partial retraction (H, In) from $\mathcal{B}'$ onto the (Kripke) full partial type hierarchy, say $\mathcal{B}$, with base type $\mathcal{B}_\iota$*

Proof We give only the definition of H and In. Since $\mathcal{B}'$ is extensional, $(\tau_1 \rightharpoonup \tau_2)^{\mathcal{B}'}$ can be treated as a set of partial functions.

- $H_{\tau_1 \rightharpoonup \tau_2}(g) = f \overset{\Delta}{\iff}$
  $\forall b, a. H_{\tau_1}(b) = a \rightarrow (gb \downarrow \vee fa \downarrow) \rightarrow H_{\tau_1}(gb) = fa$
- $In_{\tau_1 \rightharpoonup \tau_2}(f) \overset{\Delta}{=} In_{\tau_2} \circ f \circ H_{\tau_1}$

The proof that they are well-defined and satisfy the required properties is left out, since it is similar to that in [Fri75] and it follows the same pattern as Theorem 5.2.26 ∎

Remark 5.2.23 Theorem 5.2.22 holds even if we drop the constraints on Σ_{set}, by allowing extra constant, function and predicate symbols. In fact, given the interpretation $\_^{\mathcal{B}'}$ of a constant, function or predicate symbol in $\mathcal{B}'$, we can define the interpretation of such a symbol in $\mathcal{B}$ so that (H, In) is still a partial retraction, namely:

- $c^{\mathcal{B}} \overset{\Delta}{=} In(c^{\mathcal{B}'})$
- $f^{\mathcal{B}}(\overline{a}) \overset{\Delta}{\simeq} In(f^{\mathcal{B}'}(H(\overline{a})))$ (compare with the definition of $In_{\tau_1 \rightharpoonup \tau_2}$ in the proof of Theorem 5.2.22)

- $p^{\mathcal{B}}(\overline{a}) \stackrel{\Delta}{\Longleftrightarrow} p^{\mathcal{B}'}(H(\overline{a}))$

The extension of Theorem 5.2.22 to arbitrary signatures for partial applicative structures (and a similar extension is possible for Theorem 5.2.26) depends essentially on the existence of a partial retraction (rather than a partial homomorphism) to get the extra choice required to define $c^{\mathcal{B}}$.

In [Fri75] it is proved that the $\lambda\beta\eta$-calculus is complete for deriving the equations (between λ-terms) valid in the full type hierarchy with base type the set $\mathbf{N}$ of natural numbers. A similar result can be proved for the full partial type hierarchy with base type $\mathbf{N}$:

Corollary 5.2.24 *An ECE-equation is derivable in the $K\lambda_{\mathrm{p}}\beta\eta$-calculus iff it is valid in the full partial type hierarchy with base type the set $\mathbf{N}$ of natural numbers.*

Proof The implication from left to right is obvious. For the other implication it is enough to show (by the *downward Löwenheim-Skolem theorem*) that for any countable model $\mathcal{A}$ of the $K\lambda_{\mathrm{p}}\beta\eta$-calculus there exists a partial homomorphism from the full partial type hierarchy with base type $\mathbf{N}$ onto $\mathcal{A}$. But this is trivial (by Theorem 5.2.22), because for any surjective partial function H from $\mathbf{N}$ to X (which must exists because X is countable) there exists a right-inverse $In: X \to \mathbf{N}$ (by the axiom of choice) ∎

Remark 5.2.25 The $K\lambda_{\mathrm{p}}\beta\eta$-calculus does not have an *open term model*, like that for the $\lambda\beta\eta$-calculus, therefore the proof of Corollary 5.2.24 cannot rely on such a model (as done in [Fri75]).

The $J\lambda_{\mathrm{p}}\beta\eta$-calculus proves fewer equations than the $K\lambda_{\mathrm{p}}\beta\eta$-calculus, so it cannot be complete for the full partial type hierarchy with base type $\mathbf{N}$. However, it has an *initial model* (see Theorem 3.1.10).

Theorem 5.2.26 Existence of Σ_{po}-partial homomorphisms

If $\mathcal{B}'$ is an extensional monotonic (Kripke) Σ_{po}-structure (over K*), i.e. it satisfies* (ext.$\lesssim$)*, and $\mathcal{B}_\iota$ is a (Kripke) complete lattice (over* K*) s.t. there exists a partial retraction (H_ι, In_ι) from $\mathcal{B}'{\restriction}\Sigma^\iota_{po}$ onto $\mathcal{B}_\iota$, then it is possible to extend (H_ι, In_ι) to a partial retraction (H, In) from $\mathcal{B}'$ onto the (Kripke) full monotonic partial type hierarchy, say $\mathcal{B}$, with base type $\mathcal{B}_\iota$*

Proof The poset $\tau^{\mathcal{B}}$ is a complete lattice for any sort τ, because the poset of monotonic (partial) functions from a poset X to a complete lattice Y is a complete lattice (see Section 5.1.3). However, in the proof we must rely also on other basic facts (about complete lattices), whose proof can be carried out in **IZF**.

1. the poset $(\mathcal{P}(X), \subseteq)$ of subsets of a set X is a complete lattice

2. if $f: X \rightharpoonup Y$, then the function $f_*: \mathcal{P}(X) \to \mathcal{P}(Y)$ (**image along** f), s.t.
$$f_*: A \mapsto f(A)$$
is a monotonic function

3. if X is a complete lattice, then the function $\sup_\perp: \mathcal{P}(X) \rightharpoonup X$ (**strict supremum**), s.t.
$$\stackrel{\Delta}{\simeq} \begin{cases} \sqcup X & \text{if } X \text{ is inhabited} \\ \mathit{undefined} & \text{otherwise} \end{cases}$$
is a monotonic partial function

4. if $(X, \leq)$ is a partial order, then the function $\hat{\_}: X \to \mathcal{P}(X)$ (**set below**), s.t.
$$\hat{\_}: x \mapsto \{y \in X | y \leq x\}$$
is a monotonic function

The definition of H_τ and In_τ and the proof of the induction hypothesis (IH$_\tau$) "(H_τ, In_τ) is a partial retraction from $\mathcal{B}' \upharpoonright \Sigma^\tau_{po}$ onto $\mathcal{B} \upharpoonright \Sigma^\tau_{po}$" are by combined induction on τ. We show that if IH$_{\tau_1}$ and IH$_{\tau_2}$ hold, then we can define $H_{\tau_1 \rightharpoonup \tau_2}$ and $In_{\tau_1 \rightharpoonup \tau_2}$ satisfying IH$_{\tau_1 \rightharpoonup \tau_2}$.

- $H_{\tau_1 \rightharpoonup \tau_2}(g) = f \stackrel{\Delta}{\Longleftrightarrow} \forall b, a. H_{\tau_1}(b) = a \to (gb \downarrow \vee fa \downarrow) \to H_{\tau_1}(gb) = fa$

  This definition is forced by the axioms for logical relations (see remark after Definition 5.2.7). Moreover, $H_{\tau_1 \rightharpoonup \tau_2}$ is a partial function (rather than a relation), because H_{τ_1} is surjective (by IH$_{\tau_1}$) and $\mathcal{B}'$ is extensional

- $In_{\tau_1 \rightharpoonup \tau_2}(f) \stackrel{\Delta}{=} \sup_\perp \circ (In_{\tau_2} \circ f \circ H_{\tau_1})_* \circ \hat{\_}$

  $In_{\tau_1 \rightharpoonup \tau_2}(f)$ is a monotonic partial function (even if f is not monotonic), because $\sup_\perp$, $\_{}_*$ and $\hat{\_}$ are monotonic partial functions (as $\tau_2^{\mathcal{B}}$ is a complete lattice)

Claim 5.2.26.1 *If $H_{\tau_1}(b) = a$, then $In_{\tau_1 \to \tau_2}(f)b \simeq In_{\tau_2}(fa)$*

Proof First we prove that $In(fa) \lesssim In(f)b$:

- assume $In(fa)\downarrow$
- since $H_{\tau_1}(b) = a$

  $In(fa) \in (In \circ f \circ H)_*(\hat{b}) \equiv A$
- by definition of $In(f)b$

  $In(fa) \leq \sqcup A = \sup_\perp(A) = In(f)b$

The proof of $In(f)b \lesssim In(fa)$ uses monotonicity of f:

- assume $In(f)b\downarrow$, then exists $b' \leq b$

  $In_{\tau_2}(f(H_{\tau_1}(b')))\downarrow$
- by IH_{τ_1}

  $H_{\tau_1}(b') \leq H_{\tau_1}(b) = a$
- since $In(f(H(b')))\downarrow$, by monotonicity of In_{τ_2} (IH_{τ_2}) and application in $\mathcal{B}'$,

  $In_{\tau_2}(f(H_{\tau_1}(b'))) \leq In_{\tau_2}(fa)$
- by definition of $In(f)b$

  $In(f)b \leq In(fa)$

∎

We prove the following properties:

- $H_{\tau_1 \to \tau_2}$ preserves $\leq$, i.e. if $H(g_1) = f_1$, $H(g_2) = f_2$ and $g_1 \leq g_2$, then $f_1 \leq f_2$. Since $\mathcal{B}'$ is extensional, this is equivalent to $f_1 a \lesssim f_2 a$ for all a. By IH_{τ_1}, we can assume w.l.o.g. that there exists b s.t. $H(b) = a$.
  - assume that $f_1 a\downarrow$
  - $f_i a \simeq H(g_i b)$ (by definition of $H_{\tau_1 \to \tau_2}$), therefore

    $g_1 b\downarrow$
  - by monotonicity of application in $\mathcal{B}$

    $g_1 b \leq g_2 b$

- by IH_{τ_2}

  $f_1 a \simeq H(g_1 b) \leq H(g_2 b) \simeq f_2 a$

- $In_{\tau_1 \to \tau_2}$ preserves $\leq$, i.e. if $f_1 \leq f_2$, then $In(f_1) \leq In(f_2)$. Since $\mathcal{B}$ is extensional, this is equivalent to $In(f_1)b \lesssim In(f_2)b$ for any b. By definition of $\sup_\perp$, it is enough to prove that for any $a_1 \in (In \circ f_1 \circ H)_*(\hat{b}) \equiv A_1$ there exists $a_2 \in (In \circ f_2 \circ H)_*(\hat{b}) \equiv A_2$ s.t. $a_1 \leq a_2$.

  - by definition of A_1, there exists $b' \leq b$

    $a_1 = In_{\tau_2}(f_1(H_{\tau_1}(b')))$

  - by monotonicity of In_{τ_2} (IH_{τ_2}) and application in $\mathcal{B}'$,

    $a_1 = In_{\tau_2}(f_1(H_{\tau_1}(b'))) \leq In_{\tau_2}(f_2(H_{\tau_1}(b'))) \stackrel{\Delta}{=} a_2$

  - by definition of A_2

    $a_1 \leq a_2 \in A_2$

- $(In_{\tau_1 \to \tau_2}, H_{\tau_1 \to \tau_2})$ is a partial retraction, i.e. $H_{\tau_1 \to \tau_2}(In_{\tau_1 \to \tau_2}(f)) = f$. By definition of $H_{\tau_1 \to \tau_2}$, this amounts to proving $H(In(f)b) = fa$ from $H_{\tau_1}(b) = a$ and $In(f)b \downarrow \vee fa \downarrow$.

  - by the claim

    $In(f)b \simeq In(fa)$

  - since In_{τ_2} is total (by IH_{τ_2}), then

    $In(f)b = In(fa)$

  - by IH_{τ_2}

    $H(In(f)b) \simeq H(In(fa)) = fa$

From the definition of H and the properties proved so far, it is obvious that H is a Σ_{po}-partial homomorphism of partial applicative structures ((ext. $\simeq$) follows immediately from the definition of $H_{\tau_1 \to \tau_2}$). To prove that In is a Σ_{po}-morphism we have to show that In commutes with application, i.e. $In_{\tau_1 \to \tau_2}(f) In_{\tau_1}(a) \simeq fa$:

- by IH_{τ_1}

  $H(In(a)) = a$

- by the claim

  $In(f)In(a) \simeq In(fa)$

∎

There is an analogue of Corollary 5.2.24 for the $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus:

Corollary 5.2.27 *An ECEin-equation is derivable in the $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus iff it is valid in the full monotonic partial type hierarchy with base type the complete lattice $(\mathcal{P}(\mathbf{N}), \subseteq)$ of subsets of natural numbers.*

Proof The implication from left to right is obvious.

For the other implication it is enough to show (by the *downward Löwenheim-Skolem theorem*) that for any countable model $\mathcal{A}$ of the $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus there exists a partial homomorphism from the full monotonic partial type hierarchy with base type $(\mathcal{P}(\mathbf{N}), \subseteq)$ onto $\mathcal{A}$. By Theorem 5.2.26, it is enough to prove that for any countable poset $(X, \leq)$ there exists partial retraction (H, In) from $(X, \leq)$ onto $(\mathcal{P}(\mathbf{N}), \subseteq)$. In fact, let $f: \mathbf{N} \rightharpoonup X$ be a surjective partial function (f must exist because X is countable), then $In(a) = \{n \in \mathbf{N} | f(n) \leq a\}$ and $H(A) = a \iff A = In(a)$ is a partial retraction ∎

A (Kripke) full monotonic partial type hierarchy with base type a cpo is a model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus, but it is not continuous, in particular it does not satisfies fixed-point induction. In [Plo82] (see also [AL85] for a similar result) there is an improvement for the *total* variant of Corollary 5.2.24 (see [Fri75]), namely:

Theorem 5.2.28 ([Plo82]) Completeness w.r.t. the full continuous type hierarchy *The $\lambda\beta\eta$-calculus derives all the equations valid in the full continuous type hierarchy $\mathcal{A}$ with base type the flat cpo $\mathbf{N}_\perp$ of natural numbers with a bottom element.*

Proof We do not carry out a complete proof, instead we point out the crucial steps that makes it different from Friedman's proof (see Theorems 5.2.22 and 5.2.26).

The open term model of the $\lambda\beta\eta$-calculus satisfies the property that we are trying to prove for $\mathcal{A}$, therefore it is enough to construct a Σ_{po}-partial homomorphism from $\mathcal{A}$ to the open term model of the $\lambda\beta\eta$-calculus (with the trivial partial order). The induction hypothesis IH_τ required to construct the partial retraction is:

H_τ has a right-inverse In_τ and it **can be extended to a continuous partial function** Out_τ (note that this induction hypothesis is stronger than that in Theorem 5.2.26)

The definition of $In_{\tau_1 \to \tau_2}(f)$ relies on the extra strength of IH_{τ_1}. In fact, $In_{\tau_1 \to \tau_2}(f)$ is defined as the *least total extension* of the continuous partial function $In_{\tau_2} \circ f \circ Out_{\tau_1}$. Such a least total extension exists and is continuous, because each sort is of the full continuous type hierarchy with base type $\mathbf{N}_\perp$ is a cpo with a least element.

To derive $\mathrm{IH}_{\tau_1 \to \tau_2}$ from IH_{τ_1} and IH_{τ_2} one makes essential use of certain properties of the open term model, namely to define $Out_{\tau_1 \to \tau_2}(f)$ one proceeds as follows (w.l.o.g. we identify a term with its $\beta\eta$-normal form):

- fix an enumeration x_i of the variables of type τ_1
- let $P = Out_{\tau_2}(f(In_{\tau_1}(x_0)))$

  if the r.h.s. is undefined, then $Out_{\tau_1 \to \tau_2}(f)$ is undefined, otherwise
- let x_n is the first variable that does not occur free in P
- let $N = Out_{\tau_2}(f(In_{\tau_1}(x_n)))$

  if the r.h.s. is undefined, then $Out_{\tau_1 \to \tau_2}(f)$ is undefined, otherwise
- let $Out(f) = (\lambda x_n{:}\, \tau_1.N)$

∎

Remark 5.2.29 An analogue for the $\lambda_{\mathrm{p}}\beta\eta$-calculus of Plotkin's result is not possible, because any full continuous partial type hierarchy, being a model of the mon$\lambda_{\mathrm{p}}\beta\eta$-calculus, satisfies more equations than those provable in the $\lambda_{\mathrm{p}}\beta\eta$-calculus.

We do not know whether the inequations provable in the Kmon$\lambda_{\mathrm{p}}\beta\eta$-calculus are exactly those valid in all full continuous partial type hierarchies. However, there is no way of extending Plotkin's proof to this formal system, because the Kmon$\lambda_{\mathrm{p}}\beta\eta$-calculus has no open term model.

The Jmon$\lambda_{\mathrm{p}}\beta\eta$-calculus has an *open term model* $\mathcal{B}$ (see Proposition 7.3.12), so it is possible to carry out Plotkin's proof, provided the open term model is not only monotonic, but also continuous. Then, it should be possible to prove, as in Plotkin, that there is a Σ_{po}-partial homomorphism from the Kripke full continuous partial type hierarchy with base type the cpo $\iota^{\mathcal{B}}$ to the open term model $\mathcal{B}$. We conjecture that:

in $\mathcal{B}$ there are no *strictly increasing* ω-chains (i.e. $x_1 < \ldots < x_i < \ldots$)

therefore the open term model $\mathcal{B}$ is (trivially) continuous.

5.3 Closed term models

In this section we describe a closed term model construction based on the call-by-value operational semantics. We fix a monotonic partial algebra $\mathcal{A}$ and show that it is embedded in a model $\lambda\mathrm{Y}(\mathcal{A})$ of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus (which satisfies also fixed-point induction, see Remark 4.3.9); therefore, the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus is a conservative extension of monLPT. The construction of $\lambda\mathrm{Y}(\mathcal{A})$ is decomposed in two steps:

- first we define a monotonic partial combinatory algebra VALUE($\mathcal{A}$) with fixed-point operators (see Definition 5.3.15). VALUE($\mathcal{A}$) is based on the call-by-value operational semantics Eval for $\lambda\bot\mathrm{Y}$-terms over $\mathcal{A}$ (see Definition 5.3.10)

- then we introduce a suitable *logical preorder* R on VALUE($\mathcal{A}$) (see Definition 5.3.17), and prove that the quotient $\lambda\mathrm{Y}(\mathcal{A})$ of VALUE($\mathcal{A}$) by R is a model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus with least fixed-point operators satisfying fixed-point induction (see Theorem 5.3.22)

Remark 5.3.1 For simplicity, the construction of VALUE($\mathcal{A}$) and R (and ultimately $\lambda\mathrm{Y}(\mathcal{A})$) is carried out in **ZF**. However, it can be carried out *unchanged* in **IZF**, starting from a Kripke monotonic partial algebra $\mathcal{B}$.

The main step in the construction of $\lambda\mathrm{Y}(\mathcal{A})$, namely R *satisfies a certain family of least fixed-point constraints* (see Lemma 5.3.19), is borrowed from the area of Full Abstraction (see Definition 3.2.12 of [Sto86]). We conjecture that:

$\lambda\mathrm{Y}(\mathcal{A})$ is inequational fully abstract, i.e. $x \leq_\tau y$ iff $f(x) \lesssim_\iota f(y)$ for all $f\colon \tau \rightharpoonup \iota$ (see [Mil77]).

In Full Abstraction the construction of term models usually involves also a *completion step*, since the concern is with the relations between operational equivalence and semantic equivalence in *continuous models*. However, we are mainly concerned with the relations between formal systems axiomatizable in first order logic, therefore the restriction to equations (or inequations) and continuous models is inappropriate.

Example 5.3.2 For every monotonic partial algebra $\mathcal{A}$ there is a continuous partial algebra satisfying the same inequations (for instance, the ideal completion), but this algebra, in general, is not even elementary equivalent to $\mathcal{A}$.

Let $\mathcal{A}$ be the monotonic partial algebra of natural numbers (with the standard linear order) with constants c_n (one for each $n \in \omega$) and one unary function *succ* (the successor), then no continuous partial algebra is *elementary equivalent* to $\mathcal{A}$. In fact, *succ* has no fixed-point in $\mathcal{A}$, while in any elementary equivalent continuous algebra it would have a fixed-point, namely the lub of the ω-chain $< c_n | n \in \omega >$

Definition 5.3.3 Signature with fixed-point approximations

A signature $\Sigma_{\lambda\perp\mathrm{Y}}$ *for typed partial applicative structures with* **fixed-points approximations** *is a signature for typed monotonic partial applicative structures with fixed-points with*

- *an* **everywhere undefined function** $\perp_{\tau_1,\tau_2} \in \mathrm{Const}_{\tau_1 \to \tau_2}(\Sigma_{\lambda\perp\mathrm{Y}})$ *for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma_{\lambda\perp\mathrm{Y}})$
- *a* fixed-point approximation $\mathrm{Y}_{n\,\tau_1,\tau_2} \in \mathrm{Const}_{((\tau_1\to\tau_2)\to\tau_1\to\tau_2)\to\tau_1\to\tau_2}(\Sigma_{\lambda\perp\mathrm{Y}})$, *for any* $\tau_1, \tau_2 \in \mathrm{Sort}(\Sigma_{\lambda\perp\mathrm{Y}})$ *and natural number* n.

For the rest of this section we fix a Σ-structure $\mathcal{A}$, where Σ satisfies the restrictions stated below, and the following signatures:

Definition 5.3.4 Conventions for signatures

$$\Sigma \subset \Sigma_{\mathcal{A}} \subset \Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$$

- Σ *is the signature of* $\mathcal{A}$. *We assume that it has only one sort symbol* ι, *no constant symbols and the only predicate symbols are:* $\_\downarrow$, $\_ = \_$ *and* $\_ \leq \_$

- $\Sigma_{\mathcal{A}}$ *is the signature* Σ *extended with a symbol* $c_a \in \mathrm{Const}_\iota(\Sigma_{\mathcal{A}})$ *for each element* $a \in \iota^{\mathcal{A}}$

- $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$ *is the smallest signature for typed partial applicative structures with fixed-point approximations which includes* $\Sigma_{\mathcal{A}}$

Remark 5.3.5 The restrictions on Σ can be relaxed without affecting the construction of $\lambda\mathrm{Y}(\mathcal{A})$ in a relevant way, namely by requiring only that all sorts of Σ are base sorts.

5.3.1 The $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$-structure VALUE($\mathcal{A}$)

The aim of this section is to construct a monotonic partial applicative structure VALUE($\mathcal{A}$). The construction relies on certain properties of a call-by-value operational semantics for λ-terms over $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$. In the following definitions the signature $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$ is used implicitly.

Definition 5.3.6 Values and Programs

- Value_τ *is the set of* **values** *of type* τ*, i.e. the subset of* Term_τ *whose elements are variables, constants or* λ*-abstractions.*

- Prog_τ *is the set of* **programs** *of type* τ*, i.e. the set of closed* λ*-terms of type* τ*.*

- $\mathrm{ProgValue}_\tau$ *is the set of* **program values** *of type* τ*, i.e. the intersection of* Value_τ *and* Prog_τ

Remark 5.3.7 Values are closed w.r.t. substitution of variables by values, i.e. if V and V' are values, then $V[x{:=}V']$ is a value.

The definition of $\rightarrow_v$ below is the $\lambda\perp\mathrm{Y}(\mathcal{A})$-counterpart of Plotkin's $\rightarrow_v$ for the untyped call-by-value lambda calculus (see [Plo75]):

Definition 5.3.8 One-step Head Reduction

The **one-step head reduction** $\rightarrow_v$ *is the least relation on* Term *s.t.*

$$\text{i.v}\quad \frac{M \rightarrow_v N}{f(\overline{V}, M, \overline{P}) \rightarrow_v f(\overline{V}, N, \overline{P})}\ \overline{V}_i \in \mathrm{Value}\ \textit{for all}\ i \in |\overline{V}|$$

f can be either an application or a first order function in Σ ($\Sigma_{\mathcal{A}}$)

$$f.\mathrm{v} \quad f(c_{a_1}, \ldots, c_{a_n}) \rightarrow_v c_a \qquad \textit{if } f^{\mathcal{A}}(a_1, \ldots, a_n) = a$$

$$\beta.\mathrm{v} \quad (\lambda x\colon \tau.M)V \rightarrow_v M[x := V] \qquad V \in \mathrm{Value}$$

$$\mathrm{Y}_0.\mathrm{v} \quad \mathrm{Y}_0 V \rightarrow_v \bot \qquad V \in \mathrm{Value}$$

$$\mathrm{Y}_{n+1}.\mathrm{v} \quad \mathrm{Y}_{n+1} V \rightarrow_v (\lambda x\colon \tau.V(\mathrm{Y}_n V)x) \qquad V \in \mathrm{Value}$$

$$\mathrm{Y}.\mathrm{v} \quad \mathrm{Y} V \rightarrow_v (\lambda x\colon \tau.V(\mathrm{Y} V)x) \qquad V \in \mathrm{Value}$$

Lemma 5.3.9 *$\rightarrow_v$ is* single-valued, *programs are closed w.r.t. $\rightarrow_v$ and values are not $\rightarrow_v$-reducible, i.e.*

- *if $M \rightarrow_v N$ and $M \rightarrow_v N'$, then $N \equiv N'$*
- *if $M \rightarrow_v N$ and $M \in \mathrm{Prog}$, then $N \in \mathrm{Prog}$*
- *if $M \rightarrow_v N$, then $M \notin \mathrm{Value}$*

Proof The set of inference rules for $\rightarrow_v$ is deterministic, therefore $\rightarrow_v$ is single-valued.

The second statement is proved by induction on $M \rightarrow_v N$.

In the conclusion $M \rightarrow_v N$ of any of the inference rules the l.h.s. is never a value, therefore values are not $\rightarrow_v$-reducible ∎

Definition 5.3.10 Program Evaluation

The **one-step evaluation** *function* Eval_1 *is the partial function from* Term *to* Term *s.t.* $\mathrm{Eval}_1(M) = N \overset{\Delta}{\iff} M \rightarrow_v N$.

The **program evaluation** *function* Eval *is the partial function from* Prog *to* ProgValue *s.t.* $\mathrm{Eval}(M) = V \overset{\Delta}{\iff} M \rightarrow_v^* V$

Remark 5.3.11 Single-step evaluation is well-defined, by Lemma 5.3.9. Similarly program evaluation is well-defined, because for any M there is at most one value V s.t. $M \rightarrow_v^* V$.

Definition 5.3.12 Syntactic Approximation

- *the* **approximation order** $\sqsubseteq_{-} \sqsubseteq_{-}$ _ *is the least relation on* Term *s.t.*

$$x \quad x \sqsubseteq x$$

$$c \quad c \sqsubseteq c$$

$$\text{cong.}f \quad \frac{\{M_i \sqsubseteq N_i | 1 \leq i \leq n\}}{f(M_1, \ldots, M_n) \sqsubseteq f(N_1, \ldots, N_n)} \quad f \textit{ has } \text{arity } n$$

$$\text{cong.}\lambda \quad \frac{M \sqsubseteq N}{(\lambda x\colon \tau.M) \sqsubseteq (\lambda x\colon \tau.N)}$$

$$\perp \quad \perp \sqsubseteq V \qquad V \in \text{Value}$$

$$\mathrm{Y}_n \quad \mathrm{Y}_n \sqsubseteq \mathrm{Y}_{n+m} \qquad m \in \omega$$

$$\mathrm{Y}.n \quad \mathrm{Y}_n \sqsubseteq \mathrm{Y}$$

$\leq_\tau$ *is the* **approximation order** $\sqsubseteq_\tau$ **restricted to** ProgValue_τ

- *the* n-**approximation** *of* M *is the* λ-*term* $M\lceil_n$ *(where all occurrences of* Y *have subscript* $\leq n$*) defined by induction on* M*:*

  - $x\lceil_n \equiv x$
  - $\mathrm{Y}_m\lceil_n \equiv \mathrm{Y}_{min(m,n)}$
  - $\mathrm{Y}\lceil_n \equiv \mathrm{Y}_n$
  - $\_\lceil_n$ *commutes with all other* term-constructors

Lemma 5.3.13 Properties of $\sqsubseteq$

1. $\sqsubseteq$ *is a partial order on* Term
2. Value *is upwards and downwards closed, i.e.*

   if $M \sqsubseteq N$, *then* $M \in \text{Value} \iff N \in \text{Value}$

3. *$\sqsubseteq$ is substitutive, i.e.*

   if $M \sqsubseteq M'$ and $V' \sqsubseteq V'' \in$ Value, then $M[x{:}= V'] \sqsubseteq M'[x{:}= V'']$

4. $M{\restriction_n} \sqsubseteq M{\restriction_{n+1}} \sqsubseteq M$

5. *$\_{\restriction_n}$ is monotonic, i.e. if $M \sqsubseteq N$, then $M{\restriction_n} \sqsubseteq N{\restriction_n}$*

Proof

1. reflexivity ($M \sqsubseteq M$) is proved by induction on M.

   Also asymmetry ($M \sqsubseteq N$ and $N \sqsubseteq M$ implies $M \equiv N$) is proved by induction on M, but it requires an extensive case analysis on the last rules applied to prove $M \sqsubseteq N$ and $N \sqsubseteq M$. The proof of transitivity ($M \sqsubseteq N$ and $N \sqsubseteq P$ implies $M \sqsubseteq P$) is similar to that of asymmetry.

2. easy

3. substitutivity is proved by induction on the proof of $M \sqsubseteq M'$. In the case ($\bot$), i.e. $\bot \sqsubseteq V$, we use the assumption $V'' \in$ Value to prove that $V[x{:}= V'']$ is a value

4. by (an easy) induction on the structure of M

5. monotonicity is proved by induction on the proof of $M \sqsubseteq N$

∎

Lemma 5.3.14 Monotonicity and Approximability of $\rightarrow_v$

1. *if $M \sqsubseteq N$ and $M \rightarrow_v M'$, then there exists (unique) N' s.t.*

   $N \rightarrow_v N'$ and $M' \sqsubseteq N'$

2. *if $M \rightarrow_v M'$ and $n \in \omega$, then there exists (unique) N s.t.*

   $M{\restriction_{n+1}} \rightarrow_v N$ and $M'{\restriction_n} \sqsubseteq N$

Proof Both statements rely on Lemma 5.3.13 and are proved by induction on the proof of $M \rightarrow_v M'$ (the IH is used only in the case (i.v)) and case analysis on M. We consider only the case (Y.v) for the second statement, and write M_n for $M{\restriction_n}$.

- by definition of $\rightarrow_v$ and $\_\restriction_n$

  $M \equiv \mathrm{Y}V \rightarrow_v (\lambda x{:}\,\tau.V(\mathrm{Y}V)x) \equiv M'$

  $M'_n \equiv (\lambda x{:}\,\tau.V_n(\mathrm{Y}_n V_n)x)$

  $M_{n+1} \equiv \mathrm{Y}_{n+1}V_{n+1} \rightarrow_v (\lambda x{:}\,\tau.V_{n+1}(\mathrm{Y}_n V_{n+1})x) \equiv N$

- therefore, we have to prove that

  $(\lambda x{:}\,\tau.V_n(\mathrm{Y}_n V_n)x) \sqsubseteq (\lambda x{:}\,\tau.V_{n+1}(\mathrm{Y}_n V_{n+1})x)$

- let P be $(\lambda x{:}\,\tau.y(\mathrm{Y}_n y)x)$, then by $V_n \sqsubseteq V_{n+1}$ and substitutivity

  $(\lambda x{:}\,\tau.V_n(\mathrm{Y}_n V_n)x) \equiv P[y{:}= V_n] \sqsubseteq P[y{:}= V_{n+1}] \equiv (\lambda x{:}\,\tau.V_{n+1}(\mathrm{Y}_n V_{n+1})x)$

∎

Definition 5.3.15 VALUE($\mathcal{A}$)

VALUE($\mathcal{A}$) *is the* $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$*-structure of program values over* $\mathcal{A}$*, i.e.*

- $\tau^{\mathrm{VALUE}(\mathcal{A})} \stackrel{\Delta}{\equiv} \mathrm{ProgValue}_\tau$
- $c^{\mathrm{VALUE}(\mathcal{A})} \stackrel{\Delta}{\equiv} c$
- $f^{\mathrm{VALUE}(\mathcal{A})}$ *is the partial function s.t.* $f^{\mathrm{VALUE}(\mathcal{A})}(\overline{V}) \stackrel{\Delta}{\simeq} \mathrm{Eval}(f(\overline{V}))$
- $V_1 \leq_\tau^{\mathrm{VALUE}(\mathcal{A})} V_2 \stackrel{\Delta}{\Longleftrightarrow} V_1 \leq_\tau V_2$ *(or equivalently* $V_1 \sqsubseteq_\tau V_2$*)*

Proposition 5.3.16 Properties of VALUE($\mathcal{A}$)

- VALUE($\mathcal{A}$) *is a* monotonic *partial applicative structure*
- VALUE($\mathcal{A}$) *can be extended to a partial combinatory algebra*
- VALUE($\mathcal{A}$) *satisfies the axioms* $\mathrm{Y}F\downarrow$ *and* $\mathrm{Y}Fx \simeq F(\mathrm{Y}F)x$

Proof The only non trivial step to prove the first statement is to show that application is monotonic and it makes essential use of Lemma 5.3.14. Suppose that $V_i \leq V'_i$ (for $i = 1, 2$) and $V_1V_2 \rightarrow_v^n V \equiv \mathrm{Eval}(V_1V_2)$, we have to show that $V' \equiv \mathrm{Eval}(V'_1V'_2)$ exists and $V \leq V'$:

- by Lemma 5.3.14, there exists V' s.t.

  $V'_1V'_2 \rightarrow_v^n V'$ and $V \sqsubseteq V'$

- since values are upwards closed, then V' is a value, therefore (by definition of Eval)

  $\mathrm{Eval}(V_1 V_2) \leq \mathrm{Eval}(V_1' V_2')$

The second and third statements are proved by a straightforward computation (use the definition of $\rightarrow_v$). ∎

5.3.2 The $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$-logical preorder R

In this section we define a $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$-relation R on $\mathrm{VALUE}(\mathcal{A})$ and prove that it is a consistent $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$-logical preorder (according to Definition 5.2.15).

Definition 5.3.17 The relation R

The $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$*-relation* R *on* $\mathrm{VALUE}(\mathcal{A})$*, i.e. a family of binary relations* R_τ *on* $\mathrm{ProgValue}_\tau$*, is defined by induction on* τ*:*

- $c_a \, \mathrm{R}_\iota \, c_b \overset{\Delta}{\Longleftrightarrow} a \leq^{\mathcal{A}} b$ *for all* $a, b \in \iota^{\mathcal{A}}$
- $V \, \mathrm{R}_{\tau_1 \rightarrow \tau_2} \, V' \overset{\Delta}{\Longleftrightarrow} \forall V_1, V_1'. V_1 \, \mathrm{R}_{\tau_1} \, V_1' \rightarrow V V_1 \, \underset{\sim}{\mathrm{R}}_{\tau_2} \, V' V_1'$

Notation 5.3.18 If M and N are programs, then

- $M \, \mathrm{R} \, N$ means that both $\mathrm{Eval}(M) \downarrow$ and $\mathrm{Eval}(N) \downarrow$ are defined and, moreover $\mathrm{Eval}(M) \, \mathrm{R} \, \mathrm{Eval}(N)$.
- $M \, \underset{\sim}{\mathrm{R}} \, N$ means that $\mathrm{Eval}(M) \, \mathrm{R} \, \mathrm{Eval}(N)$ when $\mathrm{Eval}(M)$ is defined

For showing that R is a consistent $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$-logical preorder on $\mathrm{VALUE}(\mathcal{A})$, the main difficulty is to prove that R satisfies (MLR.Y). The proof of (MLR.Y) relies on an auxiliary property, namely "R satisfies the least fixed-point constraints $(\sqcup_n M{\restriction_n}) = M$" (see Lemma 5.3.19).

Lemma 5.3.19 Properties of R

The $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$*-relation* $\mathrm{R} \subseteq \mathrm{VALUE}(\mathcal{A}) \times \mathrm{VALUE}(\mathcal{A})$ *satisfies* (MLR.app) *and* (ext. $\underset{\sim}{\mathrm{R}}$) *of Definition 5.2.15, moreover:*

1. R_ι *satisfies* (MLR.c_a) *for any element* $a \in \iota^{\mathcal{A}}$*, and* (MLR.$f$) *for any function symbol* f *of* $\Sigma_{\mathcal{A}}$
2. R *satisfies* (MLR.$\perp$) *and* (MLR.Y_n)

3. R_τ *is transitive*

4. R_τ *satisfies* (cons. R), *i.e. for all* $M, N, P, Q \in \mathrm{ProgValue}_\tau$

if $M \leq_\tau N \,\mathrm{R}_\tau\, P \leq_\tau Q$, *then* $M \,\mathrm{R}_\tau\, Q$

5. R_τ satisfies the least fixed-point constraints $(\sqcup_n M\restriction_n) = M$, *i.e.*

for all $M, N \in \mathrm{ProgValue}_\tau$ *if* $M\restriction_n \mathrm{R}_\tau\, N$ *for all* $n \in \omega$, *then* $M \,\mathrm{R}_\tau\, N$

6. R *satisfies* (MLR.Y)

Proof The proof of (MLR.app) and (ext. $\underset{\sim}{\mathrm{R}}$) is immediate from the definition of R .

1. (MLR.c_a) is obvious from the definition of R_ι. To prove (MLR.f) we use the monotonicity of $f^{\mathcal{A}}$, for simplicity let's consider the case of an unary function. We have to derive $f(c_a)\,\mathrm{R}_\iota\, f(c_b)$ from $\mathrm{Eval}(f(c_a))\downarrow$ and $c_a \,\mathrm{R}_\iota\, c_b$:

   - by definition of R_ι
     $a \leq^{\mathcal{A}} b$
   - by definition of Eval and the assumption $\mathrm{Eval}(f(c_a))\downarrow$, there exists c s.t.
     $c = f^{\mathcal{A}}(a)$ and $\mathrm{Eval}(f(c_a)) \equiv c_c$
   - by monotonicity of $f^{\mathcal{A}}$, there exists d s.t.
     $c \leq^{\mathcal{A}} d = f^{\mathcal{A}}(b)$
   - by definition of R_ι and Eval
     $\mathrm{Eval}(f(c_a)) \equiv c_c \,\mathrm{R}_\iota\, c_d \equiv \mathrm{Eval}(f(c_b))$

2. (MLR.$\perp$) is immediate, because $\mathrm{Eval}(\perp V)$ is undefined. The proof of (MLR.Y_n) is by induction on n. The base case is immediate, because $\perp \mathrm{R} \perp$ and $\mathrm{Eval}(\mathrm{Y}_0 V) = \perp$ for each value V. For the inductive step, we use the fact that the program value $\Phi \equiv (\lambda Y.\lambda F.\lambda x.F(YF)x)$ is related to itself, i.e. $\Phi \,\mathrm{R}\, \Phi$ (the proof is straightforward), and we prove that $\mathrm{Eval}(\mathrm{Y}_{n+1}A)\,\mathrm{R}\,\mathrm{Eval}(\mathrm{Y}_{n+1}B)$ is derivable from $A \,\mathrm{R}\, B$:

   - by definition of $\rightarrow_v$
     $\mathrm{Eval}(\mathrm{Y}_{n+1}A) = \mathrm{Eval}(\Phi \mathrm{Y}_n A)$ and $\mathrm{Eval}(\Phi \mathrm{Y}_n B) = \mathrm{Eval}(\mathrm{Y}_n B)$

- since $\Phi \mathrel{\mathrm{R}} \Phi$, by the induction hypothesis $(\mathrm{Y}_n \mathrel{\mathrm{R}} \mathrm{Y}_n)$ and the assumption $(A \mathrel{\mathrm{R}} B)$
  $\mathrm{Eval}(\mathrm{Y}_{n+1}A) = \mathrm{Eval}(\Phi \mathrm{Y}_n A) \mathrel{\underset{\sim}{\mathrm{R}}} \mathrm{Eval}(\Phi \mathrm{Y} B) = \mathrm{Eval}(\mathrm{Y} B)$

3. transitivity is proved by a straightforward induction on τ

4. (cons. R) is proved by a straightforward induction on τ. The base case is obvious, because $\leq_\iota$ is the trivial partial order. For the inductive step we prove that $MA \mathrel{\mathrm{R}_{\tau_2}} QB$ is derivable from $M \leq_{\tau_1 \to \tau_2} N \mathrel{\mathrm{R}_{\tau_1 \to \tau_2}} P \leq_{\tau_1 \to \tau_2} Q$, $A \mathrel{\mathrm{R}_{\tau_1}} B$ and $\mathrm{Eval}(MA) \downarrow$:

   - by monotonicity of application (see Proposition 5.3.16) and $\mathrm{Eval}(MA) \downarrow$
     $MA \leq_{\tau_2} NA$
   - by definition of $\mathrm{R}_{\tau_1 \to \tau_2}$ and $\mathrm{Eval}(NA) \downarrow$
     $NA \mathrel{\mathrm{R}_{\tau_2}} PB$
   - by monotonicity of application and $\mathrm{Eval}(PB) \downarrow$
     $PB \leq_{\tau_2} QB$
   - by IH
     $MA \mathrel{\mathrm{R}_{\tau_2}} QB$

5. R_ι trivially satisfies the least fixed-point constraints, since $M{\restriction_n} \equiv M$ if $M \in \mathrm{ProgValue}_\iota$ (i.e. M is of the form c_a). For the inductive step we prove that for any $m \in \omega$ $\mathrm{Eval}(MA){\restriction_m} \mathrel{\mathrm{R}_{\tau_2}} \mathrm{Eval}(NB)$ is derivable from $M{\restriction_{m+k}} \mathrel{\mathrm{R}_\tau} N$, $A \mathrel{\mathrm{R}_{\tau_1}} B$ and $\mathrm{Eval}(MA) \downarrow$, where k is the number of steps necessary to evaluate $\mathrm{Eval}(MA)$, i.e. $MA \to_v^k \mathrm{Eval}(MA)$. In fact, $\mathrm{Eval}(MA) \mathrel{\mathrm{R}_{\tau_2}} \mathrm{Eval}(NB)$ is derivable from $\forall m \in \omega. \mathrm{Eval}(MA){\restriction_m} \mathrel{\mathrm{R}_{\tau_2}} NB$ by IH .

   - by Lemma 5.3.14
     $\mathrm{Eval}(MA){\restriction_m} \sqsubseteq_{\tau_2} {\mathrm{Eval}_1}^k((MA){\restriction_{m+k}})$
   - since $\mathrm{Eval}(MA){\restriction_m} \in \mathrm{ProgValue}_{\tau_2}$ and ProgValue is upwards closed, by definition of Eval
     ${\mathrm{Eval}_1}^k((MA){\restriction_{m+k}}) \equiv \mathrm{Eval}((MA){\restriction_{m+k}})$
   - since $A{\restriction_{m+k}} \leq_{\tau_1} A \mathrel{\mathrm{R}_{\tau_1}} B$, by (cons. R)
     $A{\restriction_{m+k}} \mathrel{\mathrm{R}_{\tau_1}} B$

- by definition of $\mathrm{R}_{\tau_1 \to \tau_2}$
  $\mathrm{Eval}(MA)\restriction_m \leq_{\tau_2} \mathrm{Eval}((MA)\restriction_{m+k}) \equiv \mathrm{Eval}(M\restriction_{m+k} A\restriction_{m+k})\,\mathrm{R}_{\tau_2}\, NB$
- by (cons. R)
  $\mathrm{Eval}(MA)\restriction_m \mathrm{R}_{\tau_2}\, NB$

6. Since R satisfies the least fixed-point constraints $(\sqcup_n \mathrm{Y}\restriction_n) = \mathrm{Y}$, to prove (MLR.Y) it is enough to derive $\mathrm{Y}_n\,\mathrm{R}\,\mathrm{Y}$ for all n, which follows from $\mathrm{Y}_n\,\mathrm{R}\,\mathrm{Y}_n$ (i.e. (MLR.Y_n)) and $\mathrm{Y}_n \leq \mathrm{Y}$, by (cons. R)

∎

Proposition 5.3.20 R is a logical preorder

R *is a consistent* $\Sigma_{\lambda\perp\mathrm{Y}(M)}$*-logical preorder on* $\mathrm{VALUE}(\mathcal{A})$.

Proof Lemma 5.3.19 says that R is a consistent transitive *m-logical relation* on $\mathrm{VALUE}(\mathcal{A})$, therefore the only property still to prove is reflexivity, i.e. $V\,\mathrm{R}\,V$ for all $V \in \mathrm{ProgValue}$. We prove that for all $M \in \mathrm{Term}_\tau(\eta)$ and substitutions $\sigma_1, \sigma_2 \in \mathrm{ProgValue}_\eta$ (mapping variables to program values) if $\sigma_1\,\mathrm{R}_\eta\,\sigma_2$ and $\mathrm{Eval}(M[\sigma_1])\downarrow$, then $M[\sigma_1]\,\mathrm{R}_\tau\, M[\sigma_2]$.

The proof is by induction on the derivation of $M \in \mathrm{Term}_\tau(\eta)$ and it is similar to the proof that the Fundamental Lemma of logical relations (see Lemma 5.2.11):

- if M is a variable, then it is obvious
- if $M \equiv c$, then it follows immediately from (MLR.c)
- if $M \equiv f(M_1, \ldots, M_n)$, then it follows from (MLR.f) and the induction hypothesis for M_i $(1 \leq i \leq n)$
- if $M \equiv (\lambda x.N)$, then it follows from (ext. $\underset{\sim}{\mathrm{R}}$) and the induction hypothesis for N

∎

5.3.3 The closed term model $\lambda\mathrm{Y}(\mathcal{A})$

We show that the quotient $\lambda\mathrm{Y}(\mathcal{A}) \stackrel{\Delta}{\equiv} \mathrm{VALUE}(\mathcal{A})/\,\mathrm{R}$ is a model of the $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus. From the properties of $\mathrm{VALUE}(\mathcal{A})$ and R , it is straightforward to show that $\lambda\mathrm{Y}(\mathcal{A})$ is a model of the $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$-calculus. Therefore, the only difficulty is to establish (μ).

Lemma 5.3.21 R satisfies (μ)

Let $\Phi \equiv (\lambda F, f, x.F f x)$*. Then for all program values* M *and* N

- $(\lambda x.MNx) \equiv \mathrm{Eval}(\Phi MN) \mathrel{\mathrm{R}} N$ *implies* $(\lambda x.M(\mathrm{Y}M)x) \equiv \mathrm{Eval}(\mathrm{Y}M) \mathrel{\mathrm{R}} N$

Proof We write M_n for $M{\upharpoonright_n}$. By Lemma 5.3.19, it is enough to prove that $\mathrm{Eval}(\mathrm{Y}M){\upharpoonright_n} \mathrel{\mathrm{R}} N$, i.e. $(\lambda x.M_n(\mathrm{Y}_n M_n)x) \mathrel{\mathrm{R}} N$, for all n. We prove (by induction on n) that: $(\lambda x.M_m(\mathrm{Y}_n M_m)x) \mathrel{\mathrm{R}} N$, for all m and n. For the base case (0) we prove that $(\lambda x.M_m(\mathrm{Y}_0 M_m)x)A \mathrel{\underset{\sim}{\mathrm{R}}} NB$ is derivable from $A \mathrel{\mathrm{R}} B$:

- by $\perp \leq N$, $M_m \leq M$, reflexivity and consistency

  $\Phi \mathrel{\mathrm{R}} \Phi$, $M_m \mathrel{\mathrm{R}} M$ and $\perp \mathrel{\mathrm{R}} N$

- by definition of R and the assumption $\mathrm{Eval}(\Phi MN) \mathrel{\mathrm{R}} N$

  $\Phi M_m \perp \mathrel{\mathrm{R}} \Phi MN \mathrel{\mathrm{R}} N$

- by transitivity

  $\Phi M_m \perp \mathrel{\mathrm{R}} N$

- by definition of R and the assumption $A \mathrel{\mathrm{R}} B$

  $\Phi M_m \perp A \mathrel{\underset{\sim}{\mathrm{R}}} NB$

- since $\mathrm{Eval}((\lambda x.M_m(\mathrm{Y}_0 M_m)x)A) \simeq \mathrm{Eval}(\Phi M_m \perp A)$

  $(\lambda x.M_m(\mathrm{Y}_0 M_m)x)A \mathrel{\underset{\sim}{\mathrm{R}}} NB$

For the inductive step $(n+1)$ we prove that $(\lambda x.M_m(\mathrm{Y}_{n+1} M_m)x)A \mathrel{\underset{\sim}{\mathrm{R}}} NB$ is derivable from $A \mathrel{\mathrm{R}} B$ and $(\lambda x.M_m(\mathrm{Y}_n M_m)x) \mathrel{\mathrm{R}} N$:

- by $M_m \leq M$, reflexivity, consistency and one assumption

  $\Phi \mathrel{\mathrm{R}} \Phi$, $M_m \mathrel{\mathrm{R}} M$ and $(\lambda x.M_m(\mathrm{Y}_n M_m)x) \mathrel{\mathrm{R}} N$

- by definition of R and the assumption $\mathrm{Eval}(\Phi MN) \mathrel{\mathrm{R}} N$

  $\Phi M_m(\lambda x.M_m(\mathrm{Y}_n M_m)x) \mathrel{\mathrm{R}} \Phi MN \mathrel{\mathrm{R}} N$

- by transitivity

  $\Phi M_m(\lambda x.M_m(\mathrm{Y}_n M_m)x) \mathrel{\mathrm{R}} N$

- by definition of R and the assumption $A \, \mathrm{R} \, B$
  $\Phi M_m(\lambda x.M_m(\mathrm{Y}_n M_m)x)A \, \mathrm{R}_\sim \, NB$
- since $\mathrm{Eval}((\lambda x.M_m(\mathrm{Y}_{n+1}M_m)x)A) \simeq \mathrm{Eval}(\Phi M_m(\lambda x.M_m(\mathrm{Y}_n M_m)x)A)$
  $(\lambda x.M_m(\mathrm{Y}_{n+1}M_m)x)A \, \mathrm{R}_\sim \, NB$

∎

Theorem 5.3.22 Properties of $\lambda\mathrm{Y}(\mathcal{A})$ *indexdterm model, closed*
The $\Sigma_{\lambda\perp\mathrm{Y}(\mathcal{A})}$*-structure* $\lambda\mathrm{Y}(\mathcal{A}) \stackrel{\Delta}{\equiv} \mathrm{VALUE}(\mathcal{A})/\,\mathrm{R}$ *is a model of the* $\lambda_\mathrm{p}\mu\mathrm{Y}\beta\eta$*-calculus and* $\lambda\mathrm{Y}(\mathcal{A})\restriction\Sigma$ *is (isomorphic to)* $\mathcal{A}$

Proof It is immediate from the definition of VALUE($\mathcal{A}$) and R (see Definitions 5.3.15 and 5.3.17) that $\lambda\mathrm{Y}(\mathcal{A})\restriction\Sigma$ is (isomorphic to) $\mathcal{A}$. Since VALUE($\mathcal{A}$) is a monotonic partial combinatory algebra (Proposition 5.3.16) and R is a consistent logical preorder on VALUE($\mathcal{A}$) (Proposition 5.3.20), then (by Theorem 5.2.19) $\lambda\mathrm{Y}(\mathcal{A})$ is a model of the mon$\lambda_\mathrm{p}\beta\eta$-calculus.

The interpretation of Y in $\lambda\mathrm{Y}(\mathcal{A})$ satisfies the equations $\mathrm{Y}F\downarrow$ and $\mathrm{Y}Fx \simeq F(\mathrm{Y}F)x$, because VALUE($\mathcal{A}$) satisfies them and there exists a partial homomorphism from VALUE($\mathcal{A}$) to $\lambda\mathrm{Y}(\mathcal{A})$ (see Theorem 5.2.12). Since $\lambda\mathrm{Y}(\mathcal{A})$ is extensional, then $\mathrm{Y}^{\lambda\mathrm{Y}(\mathcal{A})}$ is a fixed-point operator (i.e. it satisfies (Y)). By Lemma 5.3.21, $\mathrm{Y}^{\lambda\mathrm{Y}(\mathcal{A})}$ satisfies also (μ), therefore it is a least fixed-point operator ∎

5.4 The relation between operational and denotational semantics

In this section we state a result due to G. Plotkin (see [Plo85] and Theorem 5.4.8), namely:

> a program *terminates*, according to the call-by-value operational semantics of the *metalanguage*, iff it *denotes*, according to the denotational semantics of the *metalanguage* in the category of cpos and continuous partial functions.

Instead of the metalanguage in [Plo85] we consider a subset of it, namely the set of λ-terms over a signature Σ_{ml} for typed monotonic partial applicative structures with fixed-point operatorsand *recursive* types.

Remark 5.4.1 The metalanguage can be viewed as:

- a programming language (essentially the functional part of **ML**)
- a metalanguage for cpos and continuous partial functions
- the term language of a logic (for computable functions)

Actually the third view of the metalanguage is quite vague, since one can think of different logics to put on top of the metalanguage.

Plotkin's result is used to studying both the typed and untyped partial lambda calculus (and it can be used to studying other formal systems) by giving a *translation* into the language over Σ_{ml}.

Definition 5.4.2 The signature for the metalanguage

Let Type be the set of open type expressions *with* partial function spaces *and* **recursive types**, *i.e.:*

$$\tau \in Type ::= \quad t \quad | \quad \tau_1 \rightharpoonup \tau_2 \quad | \quad (\mu t.\tau)$$

the signature Σ_{ml} for the metalanguage is the smallest signature for typed monotonic partial applicative structure with fixed-point operators s.t.

- $\mathrm{Sort}(\Sigma_{ml})$ *is the set of closed type expressions (in Type)*
- $\mathrm{intro} \in \mathrm{Funct}_{(\tau[t:=\mu t.\tau]) \rightharpoonup \mu t.\tau}(\Sigma_{ml})$ *and* $\mathrm{elim} \in \mathrm{Funct}_{(\mu t.\tau) \rightharpoonup \tau[t:=\mu t.\tau]}(\Sigma_{ml})$ *for any* $(\mu t.\tau) \in \mathrm{Sort}(\Sigma_{ml})$

Notation 5.4.3 In the sequel we use the following shorthands for type expressions:

- O is the type expression $(\mu t.t)$
- 1 is the type expression $\mathrm{O} \rightharpoonup \mathrm{O}$

We also use the syntactic sugar for λY-terms (see Notation 4.3.8), mainly $\perp$.

Remark 5.4.4 In the category of cpos and continuous functions O and 1 are interpreted by the initial and the terminal object, respectively. There is a straightforward translation from λ-terms over Σ_{ml} to expressions of the metalanguage in [Plo85], which maps Y to the expression $(\mu Y.\lambda F.\lambda x.F(YF)x)$ of the metalanguage and commutes with the other term constructors.

In the following definitions the signature Σ_{ml} is used implicitly (see Section 5.3.1 for similar definitions).

Definition 5.4.5 Programs and program values

- Prog_τ *is the set of* **programs** *of type* τ*, i.e. the set of closed* λ*-terms of type* τ.
- $\mathrm{ProgValue}_\tau$ *is the set of* **program values** *of type* τ*, i.e. the subset of* Prog_τ *whose elements are* λ*-abstractions or terms of the form* $\mathrm{intro}(c)$*, where* c *is a program value.* c *is used to denote program values.*

Remark 5.4.6 Although the constant Y is not a program value, it *has a value* (see below).

Now we define the operational semantics of the metalanguage, i.e. a partial function Eval from programs to program values.

Definition 5.4.7 Program evaluation

We write $e \Longrightarrow c$ *for* $\mathrm{Eval}(e) = c$ *and* $e \Downarrow$ *if* e **has a value** , *i.e. there exists a program value* c *s.t.* $\mathrm{Eval}(e) = c$

$$\mathrm{Y} \Longrightarrow \; \mathrm{Y} \Longrightarrow (\lambda F.\lambda x.F(\mathrm{Y}F)x)$$

$$\lambda \Longrightarrow \; (\lambda x.e) \Longrightarrow (\lambda x.e)$$

$$\mathrm{app} \Longrightarrow \; \frac{e_1 \Longrightarrow \lambda x.e \qquad e_2 \Longrightarrow c \qquad e[x{:}=c] \Longrightarrow c'}{e_1 e_2 \Longrightarrow c'}$$

$$\mathrm{intro} \Longrightarrow \; \frac{e \Longrightarrow c}{\mathrm{intro}(e) \Longrightarrow \mathrm{intro}(c)}$$

$$\mathrm{elim} \Longrightarrow \; \frac{e \Longrightarrow \mathrm{intro}(c)}{\mathrm{elim}(e) \Longrightarrow c}$$

For our purposes it is not relevant to know the details of the intended interpretation $\mathcal{A}_{ml}$ for the metalanguage, only some properties expressible in first order logic are relevant. These properties together with the operational characterization of $\_\downarrow$ in $\mathcal{A}_{ml}$ are stated in the following theorem:

Theorem 5.4.8 ([Plo85])

There exists a Σ_{ml}-structure $\mathcal{A}_{ml}$ which is a model of the $FreeK\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus and satisfies the properties:

- *in $\mathcal{A}_{ml}$ the type $\mathrm{O} \equiv (\mu t.t)$ is empty and any recursive type $(\mu t.\tau)$ is isomorphic to $\tau[t{:}= \mu t.\tau]$, i.e.*

  empty $x{:}\,\mathrm{O}.\bot$

  rec.1 $x{:}\,(\mu t.\tau).\mathrm{elim}(\mathrm{intro}(x)) = x$

  rec.2 $x{:}\,\tau[t{:}= \mu t.\tau].\mathrm{intro}(\mathrm{elim}(x)) = x$

- *for any program e the interpretation of e in $\mathcal{A}_{ml}$ exists iff e has a value,i.e. $\mathcal{A}_{ml} \models e\downarrow$ iff $e \Downarrow$*

Proof See [Plo85] ∎

In order to use Theorem 5.4.8 for analysing the pure lambda-theories on the term-existence statements (see Section 6.4), we have to define a translation in the language over Σ_{ml}, so that $\mathcal{A}_{ml}$ becomes a model of the pure theories (this can be proved formally by using the properties of $\mathcal{A}_{ml}$ stated in Theorem 5.4.8). However, this indirect interpretation gives only an *upper bound* on the term-existence statements provable in a pure theory. To get a *lower bound* we show that the operational semantics is *represented* in the (weakest among the) pure theories, i.e. $e \Longrightarrow c$ implies $\vdash e = c$.

5.4.1 The relation between operational semantics and the typed λ_{v}-calculus

Let Σ_λ be **the signature for the pure typed λ-calculus with one base sort** ι, then the translation $\_^{ml}$ from the language over Σ_λ to the language over Σ_{ml} is s.t.

- $\iota^{ml} \equiv 1$

and it commutes with all other constructors.

Notation 5.4.9 If η is a type environment for Σ_λ, then $\mathrm{least}_\eta \in \mathrm{ProgValue}^{\Sigma_{ml}}_{\eta^{ml}}$ is the substitution which maps each variable x to the *least element* of sort $\eta(x)^{ml}$. In the sequel we write neither the translation $\_^{ml}$ nor the subscript η.

Lemma 5.4.10 *For all λ-terms $M \in \mathrm{Term}^{\Sigma_\lambda}(\eta)$ if $M[\mathrm{least}] \Downarrow$, then there exists a value $V \in \mathrm{Term}^{\Sigma_\lambda}(\eta)$ s.t. $\lambda_v\beta \vdash M \simeq V$ and $M[\mathrm{least}] \Longrightarrow V[\mathrm{least}]$.*

Proof The proof is by induction on the proof that $M[\mathrm{least}] \Downarrow$ and case analysis on the structure of M.

- the base case (Y $\Longrightarrow$) is impossible, by definition of $\_^{ml}$
- in the base case ($\lambda \Longrightarrow$), M is either a variable or a λ-abstraction, therefore we can take $V \equiv M$.
- in the inductive step (app $\Longrightarrow$), M must be an application, say M_1M_2, therefore we can apply the induction hypothesis to the M_i and get some V_i (as $M_i[\mathrm{least}] \Downarrow$).

  V_1 can be either a variable or a λ-abstraction $(\lambda x.M')$, but the first possibility has to be excluded, otherwise $\mathrm{Eval}(M[\mathrm{least}]) \simeq \mathrm{Eval}(\mathrm{Y}(\mathrm{I})V_2[\mathrm{least}])$ would be undefined (contradicting the assumption $M[\mathrm{least}] \Downarrow$). Therefore $V_1 \equiv (\lambda x.M')$ and
  $\mathrm{Eval}(M[\mathrm{least}]) =$
  $\mathrm{Eval}(M_1[\mathrm{least}]M_2[\mathrm{least}]) =$
  $\mathrm{Eval}((\lambda x.M')[\mathrm{least}]M_2[\mathrm{least}]) =$
  $\mathrm{Eval}((M'[x:=V_2])[\mathrm{least}])$

  By applying the induction hypothesis for $M'[x:=V_2]$, we get a value V s.t. $M[\mathrm{least}] \Longrightarrow V[\mathrm{least}]$.

  We want to show that V satisfies also the other requirement, namely $\lambda_v\beta \vdash M \simeq V$. In fact:
  $M \equiv M_1M_2 \simeq$ by (cong.$\cdot$) and IH
  $V_1V_2 \equiv (\lambda x.M')V_2 \simeq$ by (β_v)
  $M'[x:=V_2] \simeq$ by IH
  V

- the remaining inductive steps, namely (intro $\Longrightarrow$) and (elim $\Longrightarrow$), are impossible, by definition of $\_^{ml}$

∎

Corollary 5.4.11 *If M is a typed λ-term (over Σ_λ), then the following assertions are equivalent:*

1. $M[\text{least}] \Downarrow$
2. $\lambda_{\text{v}}\beta \vdash M \downarrow$
3. $K\lambda_{\text{p}}\mu\text{Y}\beta\eta \vdash M \downarrow$
4. $\mathcal{A}_{ml} \models M \downarrow$

Proof

$1 \Longrightarrow 2$, by Lemma 5.4.10.

$2 \Longrightarrow 3$, because the $K\lambda_{\text{p}}\mu\text{Y}\beta\eta$-calculus has more deductive power than the $\lambda_{\text{v}}\beta$-calculus.

$3 \Longrightarrow 4$, because $\mathcal{A}_{ml}$ is a model of the $K\lambda_{\text{p}}\mu\text{Y}\beta\eta$-calculus.

$4 \Longrightarrow 3$. In fact, $\mathcal{A}_{ml} \models M \downarrow$ implies $\mathcal{A}_{ml} \models M[\text{least}] \downarrow$ (by Substitution Lemma) and this implies $M[\text{least}] \Downarrow$ (by Theorem 5.4.8) ∎

5.4.2 The relation between operational semantics and the untyped λ_{v}-calculus

For the untyped pure theories it is possible to prove a result similar to Lemma 5.4.10 and Corollary 5.4.11, the only difference being that the source language is changed and the translation $\_^{ml}$ has to be changed accordingly.

Let Σ_λ be **the signature for the pure untyped λ-calculus**, i.e. one sort ι and a binary partial function app, then the translation $\_^{ml}$ from the language over Σ_λ to the language over Σ_{ml} is s.t.

- $\iota^{ml} \equiv \text{V} \stackrel{\Delta}{\equiv} (\mu t.t \rightharpoonup t)$
- $(MN)^{ml} \equiv \text{elim}(M^{ml})N^{ml}$
- $(\lambda x\colon \iota.M)^{ml} \equiv \text{intro}(\lambda x\colon \text{V}.M^{ml})$

and it commutes with all other constructors.

Notation 5.4.12 If η is a type environment for Σ_λ, then $\mathrm{least}_\eta \in \mathrm{ProgValue}_{\eta^{ml}}^{\Sigma_{ml}}$ is the substitution which maps each variable x to the *least element* of type V. Again, we write neither the translation $\_^{ml}$ nor the subscript η.

Lemma 5.4.13 *For all λ-terms $M \in \mathrm{Term}^{\Sigma_\lambda}(\eta)$ if $M[\mathrm{least}] \Downarrow$, then there exists a value $V \in \mathrm{Term}^{\Sigma_\lambda}(\eta)$ s.t. $\lambda_v\beta \vdash M \simeq V$ and $M[\mathrm{least}] \Longrightarrow V[\mathrm{least}]$.*

Proof The proof is by induction on the proof that $M[\mathrm{least}] \Downarrow$ and case analysis on the structure of M.

- both base cases, namely $(\mathrm{Y} \Longrightarrow)$ and $(\lambda \Longrightarrow)$, are impossible, by definition of $\_^{ml}$

- in the inductive step $(\mathrm{app} \Longrightarrow)$, M must be an application, say M_1M_2, therefore we can apply the induction hypothesis to the M_i and get some V_i (as $M_i[\mathrm{least}] \Downarrow$).

  V_1 can be either a variable or a λ-abstraction $(\lambda x.M')$, but the first possibility has to be excluded, otherwise $\mathrm{Eval}(M[\mathrm{least}]) \simeq \mathrm{Eval}(\mathrm{Y}(\mathrm{I})V_2[\mathrm{least}])$ would be undefined (contradicting the assumption $M[\mathrm{least}] \Downarrow$). Therefore $V_1 \equiv (\lambda x.M')$ and
  $\mathrm{Eval}(M[\mathrm{least}]) =$
  $\mathrm{Eval}(\mathrm{elim}(M_1[\mathrm{least}])M_2[\mathrm{least}]) =$
  $\mathrm{Eval}(\mathrm{elim}(\mathrm{intro}((\lambda x.M')[\mathrm{least}]))M_2[\mathrm{least}]) =$
  $\mathrm{Eval}((\lambda x.M')[\mathrm{least}]V_2[\mathrm{least}]) =$
  $\mathrm{Eval}((M'[x := V_2])[\mathrm{least}])$

  By applying the induction hypothesis for $M'[x := V_2]$, we get a value V s.t. $M[\mathrm{least}] \Longrightarrow V[\mathrm{least}]$.

  We want to show that V satisfies also the other requirement, namely $\lambda_v\beta \vdash M \simeq V$. In fact:
  $M \equiv M_1M_2 \simeq$ by (cong.$\cdot$) and IH
  $V_1V_2 \equiv (\lambda x.M')V_2 \simeq$ by (β_v)
  $M'[x := V_2] \simeq$ by IH
  V

- in the inductive step (intro $\Longrightarrow$), M is either a variable or a λ-abstraction, therefore we can take $V \equiv M$.
- the inductive step (elim $\Longrightarrow$) is impossible, by definition of $\_^{ml}$

∎

Corollary 5.4.14 *If M is an untyped λ-term (over Σ_λ), then the following assertions are equivalent:*

1. $M[\mathrm{least}] \Downarrow$
2. $\lambda_{\mathrm{v}}\beta \vdash M \downarrow$
3. $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \vdash M \downarrow$
4. $\mathcal{A}_{ml} \models M \downarrow$

Proof

$1 \Longrightarrow 2$, by Lemma 5.4.13.

$2 \Longrightarrow 3$, because the $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus has more deductive power than the $\lambda_{\mathrm{v}}\beta$-calculus.

$3 \Longrightarrow 4$, because $\mathcal{A}_{ml}$ is a model of the $K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus after the translation $\_^{ml}$ from untyped λ-terms (over Σ_λ) to λ-terms over Σ_{ml}

$4 \Longrightarrow 3$. In fact, $\mathcal{A}_{ml} \models M \downarrow$ implies $\mathcal{A}_{ml} \models M[\mathrm{least}] \downarrow$ (by Substitution Lemma) and this implies $M[\mathrm{least}] \Downarrow$ (by Theorem 5.4.8) ∎

Chapter 6

Relations among formal systems

We investigate *conservative extension results* among some formal systems introduced in previous chapters and gathered in Definition 6.1.3. To avoid an excessive duplication of results, we give more emphasis to the relations among *inhabited* formal systems. However, when there are major differences, the relations among *free* formal systems will be pointed out.

First we review the relations among the *total* formal systems and then we analyse the relations among the *partial* formal systems. The reason for having a wide review of the *total* formal systems is twofold:

- the results on which it relies on are quite scattered throughout the literature
- it makes it possible to appreciate the different relations among the *partial* formal systems

There are two major differences between the *partial* and *total* case, that deserve to be mentioned here:

1. the equations between λ-terms derivable in any of the *total* lambda calculi are the same, while in the *partial* case there are four distinct equational theories (see Theorem 6.4.5)
2. the least fixed-point *partial* lambda calculus is a conservative extension of the logic of partial terms, while the least fixed-point *total* lambda calculus is not a conservative extension of equational logic

The first difference has important implications for the development of subsequent chapters on equational presentation and reduction for the partial lambda calculus, in fact:

- it is very unlikely that the *classical* partial lambda calculi admit an equational presentation, because they involve some kind of *case analysis* even at the equational level (see proof of Lemma 6.3.1)

- we have to develop an equational (actually inequational) presentation also for the *intuitionistic monotonic* partial lambda calculus, because the equational presentation for the *intuitionistic* partial lambda calculus is not *complete* (see Lemma 6.3.3). Besides, *monotonic* (and *continuous*) models are important for computer science applications.

Most of the techniques used to prove these conservative extension results are quite general and have also other applications, therefore they have been developed independently.

6.1 Preliminary definitions and conventions

In this section we fix 6 signatures and 48 (actually 44) formal systems axiomatized either in the logic of partial terms or in the typed partial lambda calculus. In the rest of this chapter, the deductive power of these formal systems is compared according to the criteria introduced in Definition 1.1.1.

Definition 6.1.1 Signatures

$$\begin{array}{ccc}
\Sigma & \subset & \Sigma_{\mathrm{mon}} \\
\cap & & \cap \\
\Sigma_\lambda & \subset & \Sigma_{\mathrm{mon}\lambda} \\
\cap & & \cap \\
\Sigma_{\lambda\mathrm{Y}} & \subset & \Sigma_{\mathrm{mon}\lambda\mathrm{Y}}
\end{array}$$

- Σ *is a signature for partial algebras, which does not contain function spaces, i.e. sort symbols of the form* $\tau_1 \rightharpoonup \tau_2$*, or partial orders, i.e. predicate symbols of the form* $\leq_\tau$

- Σ_{mon} *is a signature for monotonic partial algebras, which does not contain function spaces*

- Σ_λ *is a signature for typed partial applicative structures, which does not contain partial orders or fixed-point operators, i.e. constant symbols of the form* $\mathrm{Y}_{\tau_1,\tau_2}$

- $\Sigma_{\mathrm{mon}\lambda}$ *is a signature for typed monotonic partial applicative structures, which does not contain fixed-point operators*

- $\Sigma_{\lambda\mathrm{Y}}$ *is a signature for typed partial applicative structures with fixed-points, which does not contain partial orders*

- $\Sigma_{\mathrm{mon}\lambda\mathrm{Y}}$ *is a signature for typed monotonic partial applicative structures with fixed-points, which has only one base sort* ι *and the only predicate symbols are:* $\_\downarrow$, $\_ = \_$ *and* $\_ \leq \_$

Remark 6.1.2 For the *free* formal systems the constraints for the signatures ought to be stricter, because the extension of a signature with new constant symbols may force some sorts to be inhabited. The assumptions about base sorts and predicate symbols of $\Sigma_{\mathrm{mon}\lambda\mathrm{Y}}$ have been made in order to get simpler proofs, but they can be dropped without invalidating the results.

In Definitions 2.4.3 and 4.2.5 we have introduced two formal systems: the logic of partial terms ($Free J$LPT over Σ) and the typed partial lambda calculus ($Free J\lambda_{\mathrm{p}}\beta\eta$-calculus over Σ_λ). Each of them has 8 variants on the same signature (see Definition 2.4.5):

$$\begin{array}{cccccc} Free & J & \mathrm{LPT} & & 0 \\ \cap & \cap & \cap & + & \cap \\ Inhabited & K & \lambda_{\mathrm{p}}\beta\eta & & \mathrm{tot} \end{array}$$

We have also introduced variants of LPT and the $\lambda_{\mathrm{p}}\beta\eta$-calculus on suitably extended signatures (see Chapter 4):

$$\begin{array}{ccc} \mathrm{LPT} & \subset & \mathrm{monLPT} \\ \cap & & \cap \\ \lambda_{\mathrm{p}}\beta\eta & \subset & \mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \\ \cap & & \cap \\ \lambda_{\mathrm{p}}\mathrm{Y}\beta\eta & \subset & \lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta \end{array}$$

By combining the various possibilities together we get 48 formal systems:

Definition 6.1.3 Formal systems

$$\begin{array}{ccccccc} & & \mathrm{LPT} & \subset & \mathrm{monLPT} & & \\ Free & J & \cap & & \cap & & 0 \\ \cap & \cap & \lambda_{\mathrm{p}}\beta\eta & \subset & \mathrm{mon}\lambda_{\mathrm{p}}\beta\eta & + & \cap \\ Inhabited & K & \cap & & \cap & & \mathrm{tot} \\ & & \lambda_{\mathrm{p}}\mathrm{Y}\beta\eta & \subset & \lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta & & \end{array}$$

The **partial (total) formal systems** *are* $\chi\psi\mathrm{FS}$ *(*$\chi\psi\mathrm{FS}+\mathrm{tot}$*), where* χ*,* ψ *and* FS *are* **metavariables** *s.t.* $\chi \in \{Free, Inhabited\}$ *(***free** *versus* **inhabited** *logic),* $\psi \in \{J, K\}$ *(***intuitionistic** *versus* **classical** *logic) and* FS **ranges** *over the 6* **formal systems***:*

- LPT *is the logic of partial elements over* Σ *(see Definition 2.4.3)*
- monLPT *is the monotonic logic of partial elements axiomatized by* $\mathrm{mon_p}$ *in* LPT *over* Σ_{mon} *(see Definition 4.1.22)*
- $\lambda_{\mathrm{p}}\beta\eta$ *is the typed partial lambda calculus over* Σ_λ *(see Definition 4.2.5)*
- $\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$ *is the typed monotonic partial lambda calculus axiomatized by* $\mathrm{mon_p} + \xi.\lesssim$ *in the* $\lambda_{\mathrm{p}}\beta\eta$*-calculus over* $\Sigma_{\mathrm{mon}\lambda}$ *(see Definition 4.3.1)*
- $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$ *is the typed fixed-point partial lambda calculus axiomatized by* Y *in the* $\lambda_{\mathrm{p}}\beta\eta$*-calculus over* $\Sigma_{\lambda\mathrm{Y}}$ *(see Definition 4.3.5)*
- $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ *is the typed least fixed-point partial lambda calculus axiomatized by* $\mathrm{mon_p} + \xi.\lesssim + \mu + \mathrm{Y}$ *in the* $\lambda_{\mathrm{p}}\beta\eta$*-calculus over* $\Sigma_{\mathrm{mon}\lambda\mathrm{Y}}$ *(see Definition 4.3.7)*

Remark 6.1.4 Actually, there are 44 formal systems, because $Free\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta + \mathrm{tot}$ and $Free\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$ are inhabited. In fact, the axiom inhab is derivable in the $FreeJ\lambda_{\mathrm{t}}\mathrm{Y}\beta\eta$-calculus.

6.2 The total formal systems

In this section we study the relations among the **total formal systems** $\chi\psi\mathrm{FS}+\mathrm{tot}$. This will be mainly a review of results already available in the literature on the lambda calculus.

6.2.1 Intuitionistic versus classical logic

The following lemma *explains* why the λ-calculus, as equational formal system, is *independent* from the logic.

Lemma 6.2.1 Coincidence of the Inhabited $\lambda_{\mathrm{t}}\beta\eta$-calculi

Let L *be* $\{\mathrm{Frag}^{\Sigma}(\wedge\forall\bot)\} \Longrightarrow \mathrm{Frag}^{\Sigma}(\bot)$*, then intuitionistic and classical inhabited* $\lambda_{\mathrm{t}}\beta\eta$*-calculi coincide on* L*, i.e.* $InhabitedJ\lambda_{\mathrm{t}}\beta\eta \subset^{c}_{L\vdash L} InhabitedK\lambda_{\mathrm{t}}\beta\eta$*.*

Proof By Theorem 4.2.19, the $\lambda_t\beta\eta$-calculus is equivalent at the level of terms to the formal system axiomatized by $\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot}$ in LPT. Therefore, it is enough to prove that

$$\mathit{Inhabited}J\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot} \subset^c_{L\vdash L} \mathit{Inhabited}K\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot}$$

- since the axioms (tot), ($\mathrm{CL_p}$) and (ext. $=$) are included in L and (ext. $\simeq$ +tot) is equivalent to (ext. $=$ +tot) in LPT, then

  extensional total combinatory logic $\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot}$ is axiomatized in LPT by a set of sequents T included in L

- by Theorem 3.1.15 $\mathit{Inhabited}J\mathrm{LPT} \subset^c_{L\vdash L} \mathit{Inhabited}K\mathrm{LPT}$, therefore

  $\mathit{Inhabited}J\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot} \equiv \mathit{Inhabited}J\mathrm{LPT} + T \subset^c_{L\vdash L}$ $\mathit{Inhabited}K\mathrm{LPT} + T \equiv \mathit{Inhabited}K\mathrm{CL_p} + \mathrm{ext.} \simeq +\mathrm{tot}$

∎

Theorem 6.2.2 *Let L be the fragment* $\{\mathrm{Frag}(\wedge\forall\bot)\} \Longrightarrow \mathrm{Frag}(\bot)$*, then for any formal system* FS *(see Definition 6.1.3)*

$$\mathit{Inhabited}J\mathrm{FS} + \mathrm{tot} \subset^c_{L\vdash L} \mathit{Inhabited}K\mathrm{FS} + \mathrm{tot}$$

Proof If FS is either LPT or monLPT, then we proceed as follow:

- since the axioms (tot) and ($\mathrm{mon_p}$) are included in L, then

  $\mathrm{FS} + \mathrm{tot}$ is axiomatized in LPT by a set of sequents T included in L.

- by Theorem 3.1.15 $\mathit{Inhabited}J\mathrm{LPT} \subset^c_{L\vdash L} \mathit{Inhabited}K\mathrm{LPT}$, therefore

  $\mathit{Inhabited}J\mathrm{FS} + \mathrm{tot} \equiv \mathit{Inhabited}J\mathrm{LPT} + T \subset^c_{L\vdash L}$ $\mathit{Inhabited}K\mathrm{LPT} + T \equiv \mathit{Inhabited}K\mathrm{FS} + \mathrm{tot}$

If FS is either $\lambda_p\beta\eta$, mon$\lambda_p\beta\eta$, $\lambda_p\mathrm{Y}\beta\eta$ or $\lambda_p\mu\mathrm{Y}\beta\eta$, we replace LPT with the $\lambda_t\beta\eta$-calculus and proceed in a similar way:

- since the axioms ($\mathrm{mon_p}$), $(\xi, \leq)$, (Y) and (μ) are included in L and the axiom $(\xi.\lesssim)$ is equivalent to $(\xi.\leq)$ in the $\lambda_t\beta\eta$-calculus, then

  $\mathrm{FS} + \mathrm{tot}$ is axiomatized in the $\lambda_t\beta\eta$-calculus by a set of sequents T included in L.

- by Lemma 6.2.1 $InhabitedJ\lambda_t\beta\eta \subset^c_{L\vdash L} InhabitedK\lambda_t\beta\eta$, therefore
  $InhabitedJ\mathrm{FS}+\mathrm{tot} \equiv InhabitedJ\lambda_t\beta\eta + T \subset^c_{L\vdash L}$
  $InhabitedK\lambda_t\beta\eta + T \equiv InhabitedK\mathrm{FS}+\mathrm{tot}$

∎

Remark 6.2.3 The counterexamples to coincidence of intuitionistic and classical (inhabited) logic in Propositions 3.1.16 and 3.1.17 apply also to the total inhabited formal systems.

For the total formal systems $Free\psi\mathrm{FS}+\mathrm{tot}$, there are weaker conservative extension results, namely:

- let L be the coherent fragment $\{\mathrm{Frag}(\bot \wedge \vee \exists)\} \Longrightarrow \mathrm{Frag}(\bot \wedge \vee \exists)$. If FS is either LPT or monLPT, then

$$FreeJ\mathrm{FS}+\mathrm{tot} \subset^c_{L\vdash L} FreeK\mathrm{FS}+\mathrm{tot}$$

- if FS is either $\lambda_p\beta\eta$ or mon$\lambda_p\beta\eta$, then (see [MMMS87, MM87]):
  - $FreeJ\mathrm{FS}+\mathrm{tot} \subset^c_{0\vdash E} FreeK\mathrm{FS}+\mathrm{tot}$, but
  - $FreeJ\mathrm{FS}+\mathrm{tot} \subset_{E\vdash E} FreeK\mathrm{FS}+\mathrm{tot}$
- if FS is either $\lambda_p\mathrm{Y}\beta\eta$ or $\lambda_p\mu\mathrm{Y}\beta\eta$, then Theorem 6.2.2 applies, because the formal system $Free\psi\mathrm{FS}+\mathrm{tot}$ is inhabited.

6.2.2 Conservative extension results

Given a partial algebra $\mathcal{A}$, there is a **trivial** way to turn it into a monotonic partial algebra, namely by interpreting $\leq$ as the equality. In a similar way, one can turn a model of the total lambda calculus into a model of the monotonic total lambda calculus into.

Lemma 6.2.4 Relative Interpretation

Let (triv.$\leq$) *be the axiom saying that "$\leq$ is the trivial partial order", i.e.*

triv. $\leq$ $x_1 \leq x_2 \Longleftrightarrow x_1 = x_2$

- $(\mathrm{mon_p})$ *is derivable from* (triv. $\leq$) *in* $FreeJ$LPT

- $(\xi.\lesssim)$ *is derivable from* (triv. $\leq$) *in the* $FreeJ\lambda_t\beta\eta$*-calculus*

Proof Straightforward ∎

Theorem 6.2.5 Conservative extension results

For any $\chi \in \{Free, Inhabited\}$ *and* $\psi \in \{J, K\}$ *(see Definition 6.1.3)*

$$\begin{array}{ccc}
\chi\psi\mathrm{LPT}+\mathrm{tot} & \subset^c & \chi\psi\mathrm{monLPT}+\mathrm{tot} \\
\cap^c & & \cap^c \\
\chi\psi\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} & \subset^c & \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} \\
\cap^c_{0\vdash E} & & \cap^c_{0\vdash Ein} \\
\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta+\mathrm{tot} & & \psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta+\mathrm{tot}
\end{array}$$

Proof By Lemma 1.1.4, the conservative extension results $\chi\psi\mathrm{LPT}+\mathrm{tot} \subset^c \chi\psi\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot}$ and $\chi\psi\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} \subset^c_{0\vdash E} \psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta+\mathrm{tot}$ are derivable from the others:

- $\chi\psi\mathrm{LPT}+\mathrm{tot} \subset^c \chi\psi\mathrm{monLPT}+\mathrm{tot}$ follows from $\mathrm{monLPT} \subseteq \mathrm{LPT}+\mathrm{triv.}\leq$ (see Lemma 6.2.4). In fact

  $$\begin{array}{ccc}
  \mathrm{LPT} & & \mathrm{LPT} \\
  |\cap & & \cap^c \\
  \mathrm{monLPT} & \subseteq & \mathrm{LPT}+\mathrm{triv.}\leq
  \end{array}$$

  therefore $\mathrm{LPT} \subset^c \mathrm{monLPT}$, by (convex) of Lemma 1.1.4

- $\chi\psi\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} \subset^c \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot}$ follows from $\mathrm{mon}\lambda_{\mathrm{t}}\beta\eta \subseteq \lambda_{\mathrm{t}}\beta\eta+\mathrm{triv.}\leq$ (see Lemma 6.2.4) as for the previous conservative extension result

- $\chi\psi\mathrm{monLPT}+\mathrm{tot} \subset^c \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot}$, because any monotonic algebra $\mathcal{A}$ is embedded in the *full monotonic type hierarchy* with base type the underlying poset of $\mathcal{A}$ (see Definition 5.1.21)

- $Inhabited K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} \subset^c_{0\vdash Ein} K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta+\mathrm{tot}$, because there is a $\Sigma_{\mathrm{mon}\lambda}$-partial homomorphism from the *full continuous type hierarchy* with base type the flat cpo $\mathbf{N}_\perp$ to the open term model of the $\lambda\beta\eta$-calculus with the trivial partial order (see Theorem 5.2.28 and [Plo82, AL85]).

  The general result $\chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta+\mathrm{tot} \subset^c_{0\vdash Ein} \psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta+\mathrm{tot}$ follows from $FreeJ\lambda_{\mathrm{t}}\beta\eta \subset^c_{0\vdash E} Inhabited K\lambda_{\mathrm{t}}\beta\eta$ (see [MMMS87, MM87]).

∎

In the previous theorem we have shown that some formal systems are an extension of others over a fragment rather than the entire language, e.g. $\chi\psi$LPT+ tot $\subset^c_{0\vdash E}$ $\psi\lambda_{\rm p}{\rm Y}\beta\eta$+tot. Next, we show that these extensions are not conservative over a slightly bigger fragment.

Theorem 6.2.6 Counterexamples to conservative extension results

For any $\chi \in \{Free, Inhabited\}$ and $\psi \in \{J, K\}$ (see Definition 6.1.3)

$$\begin{array}{ccc} \chi\psi\text{LPT} + \text{tot} & & \chi\psi\text{monLPT} + \text{tot} \\ \cap_{E\vdash E} & & \cap_{E\vdash E} \\ \psi\lambda_{\rm p}{\rm Y}\beta\eta + \text{tot} & \subset_{0\vdash E} & \psi\lambda_{\rm p}\mu{\rm Y}\beta\eta + \text{tot} \end{array}$$

Proof By Lemma 1.1.4, the counterexamples for the free formal systems $Free\psi$LPT+ tot and $Free\psi$monLPT + tot are derivable from those for the inhabited formal systems. The counterexample $Inhabited\psi$monLPT + tot $\subset_{E\vdash E}$ $\psi\lambda_{\rm p}\mu{\rm Y}\beta\eta$ + tot is derivable from the others, by Lemma 1.1.4 and Theorem 6.2.5. Moreover, by Theorem 6.2.2, it is enough to consider the classical inhabited formal systems $InhabitedK$FS + tot:

- $InhabitedK$LPT + tot $\subset_{E\vdash E}$ $K\lambda_{\rm p}{\rm Y}\beta\eta$ + tot, because there is a signature Σ for partial algebras, a set Ax_{int} of E-equations and an E-equation $M = N$, which is derivable from Ax_{int} in $K\lambda_{\rm p}{\rm Y}\beta\eta$ + tot, but not in $InhabitedK$LPT + tot. The counterexample is due to G. Plotkin and can be found in [BTM87]:

  Σ is the signature with one sort, I, and the following constants and operations

  $$\langle cond\colon I^3 \to I; -\colon I^2 \to I; succ\colon I \to I; 0, 1\colon I\rangle$$

  Ax_{int} is the set of equations

  - $cond(0, x, y) = x$
  - $cond(1, x, y) = y$
  - $x - x = 0$
  - $succ(x) - x = 1$

  $0 = 1$ is derivable from Ax_{int} in the $\lambda{\rm Y}\beta\eta$-calculus (i.e. $K\lambda_{\rm p}{\rm Y}\beta\eta$ + tot), but not in equational logic (i.e. $InhabitedK$LPT + tot)

- $K\lambda_{\rm p}{\rm Y}\beta\eta+{\rm tot} \subset_{0\vdash E} K\lambda_{\rm p}\mu{\rm Y}\beta\eta+{\rm tot}$, because for any sort τ the E-equation $(\mu f{:}\, \tau \rightharpoonup \tau.f) = \lambda x{:}\, \tau.(\mu y{:}\, \tau.y)$ is derivable in $K\lambda_{\rm p}\mu{\rm Y}\beta\eta + {\rm tot}$, but not in $K\lambda_{\rm p}{\rm Y}\beta\eta + {\rm tot}$.

  In fact, the l.h.s. and r.h.s. of this E-equation have different ${\rm Y}\beta\eta$-normal forms, therefore the E-equation is not derivable in the $\lambda{\rm Y}\beta\eta$-calculus (i.e. $K\lambda_{\rm p}{\rm Y}\beta\eta + {\rm tot}$)

∎

Remark 6.2.7 By Lemma 1.1.4 and the conservative extension results of Theorem 6.2.5, other counterexamples are derivable from the those of Theorem 6.2.6, e.g.

$$\begin{array}{ccc} \chi\psi\lambda_{\rm p}\beta\eta + {\rm tot} & & \chi\psi{\rm mon}\lambda_{\rm p}\beta\eta + {\rm tot} \\ \cap_{E\vdash E} & & \cap_{E\vdash E} \\ \psi\lambda_{\rm p}{\rm Y}\beta\eta + {\rm tot} & \subset_{0\vdash E} & \psi\lambda_{\rm p}\mu{\rm Y}\beta\eta + {\rm tot} \end{array}$$

6.3 The partial formal systems

In this section we study the relations among the **partial formal systems** $\chi\psi$FS.

The main differences between the partial and total lambda calculus are not due to **partial functions** in itself. In fact, the analogues of Lemma 6.2.2 and 6.2.4 fail because of **extensionality** for (monotonic) partial functions.

6.3.1 Intuitionistic versus classical logic

The intuitionistic and classical inhabited $\lambda_{\rm p}\beta\eta$-calculi do not coincide on L of Theorem 6.2.2, because extensionality for partial functions (ext. $\simeq$) (see Definition 4.2.9) is not in the fragment where intuitionistic and classical inhabited logics coincide.

Lemma 6.3.1 *There is an E-equation which is derivable in the Free$K\lambda_{\rm p}\beta$-calculus but not in the Inhabited$J\lambda_{\rm p}\mu{\rm Y}\beta\eta$-calculus.*

Proof Since ECE- and E-equations have the same expressive power in the $J\lambda_{\rm p}\beta$-calculus, it is enough to exhibit an ECE-equation with the required property.

Let $a{:}\, (\iota \rightharpoonup \iota) \rightharpoonup (\iota \rightharpoonup \iota) \rightharpoonup \iota$, $b{:}\, \iota$ and $f, g{:}\, \iota \rightharpoonup \iota$ be four *variables*, then

$$(a{\rm I}f)\downarrow, (a{\rm I}g)\downarrow, (af{\rm I})\downarrow, (ag{\rm I})\downarrow, (afg)\downarrow \Longrightarrow (agf)\downarrow \qquad (6.1)$$

is derivable from $x = b$ in the $K\lambda_{\mathrm{p}}\beta\eta$- but not in the $J\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus.

$x = b$ means that ι is the singleton $\{b\}$, then (classically) $\iota \rightharpoonup \iota$ may have at most two elements: the identity and the everywhere divergent function. A case analysis of the possible values for f and g shows that the sequent 6.1 is valid.

On the other hand, consider the Kripke full monotonic or full continuous partial type hierarchy over $\{\bot < \top\}$ with base type the one-element complete lattice $\{b\}$ (see Definitions 5.1.22 or 5.1.26), then $\iota \rightharpoonup \iota$ at stage $\bot$ has three elements $undef \leq inter \leq id$, where:

- $undef$ is the everywhere divergent function at $\bot$ and $\top$
- $inter$ is the everywhere divergent function at $\bot$ and becomes the identity at $\top$
- id is the identity at $\bot$ and $\top$

Let a be the Kripke partial function which at stage $\bot$ is s.t.

$$a(f)(g) \simeq \begin{cases} b & \text{if } inter \leq f \text{ or } id \leq g \\ undefined & \text{otherwise} \end{cases}$$

(at stage $\top$ we can take $a(f)(g)$ to be always b) and let $f = inter$ and $g = undef$, then the antecedent of the sequent 6.1 is valid, but the succedent is not.

The counterexample can be modified, so that we do not need the assumption $x = b$. More precisely, let P be the λ-term

$$P \triangleq (\lambda f\colon \iota \rightharpoonup \iota.(\lambda y\colon \iota.y \upharpoonright (fb)))$$

It is straightforward to show that the denotation of P is a retraction on $\iota \rightharpoonup \iota$, which (classically) has at most two fixed-points: the identity and the everywhere divergent function.

Then, the *substitution instance* of the sequent 6.1, obtained by replacing a with the λ-term $(\lambda f, g\colon \iota \rightharpoonup \iota.a(Pf)(Pg))$, is derivable from no assumptions in the $K\lambda_{\mathrm{p}}\beta$- but not in $J\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus ∎

Theorem 6.3.2 *Let L be the fragment* $\{\mathrm{Frag}(\wedge\forall\bot)\} \Longrightarrow \mathrm{Frag}(\bot)$. *If* FS *is either* LPT *or* monLPT, *then*

$$Inhabited J\mathrm{FS} \subset^{c}_{L\vdash L} Inhabited K\mathrm{FS}$$

If FS *is either* $\lambda_{\mathrm{p}}\beta\eta$, mon$\lambda_{\mathrm{p}}\beta\eta$, $\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$ *or* $\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$, *then*

$$Inhabited J\mathrm{FS} \subset_{0\vdash E} Inhabited K\mathrm{FS}$$

Proof The first statement is proved as in Theorem 6.2.2, while the second one follows immediately from Lemma 6.3.1 ∎

For the partial formal systems $Free\psi$FS + tot (when FS is either LPT or monLPT) there are weaker conservative extension results similar to those for the total formal systems, namely:

- let L be the coherent fragment $\{\mathrm{Frag}(\bot \wedge \vee\exists)\} \Longrightarrow \mathrm{Frag}(\bot \wedge \vee\exists)$. If FS is either LPT or monLPT, then

$$Free J\mathrm{FS} \subset^{c}_{L\vdash L} Free K\mathrm{FS}$$

- For the other free formal systems, the counterexample provided by Lemma 6.3.1 continues to hold.

6.3.2 Conservative extension results

The axiom $(\xi.\lesssim)$ is not derivable from (triv. $\leq$) (see Lemma 6.2.4) in the $Free J\lambda_{\mathrm{p}}\beta\eta$-calculus, because the axioms $(\xi.\simeq)$ and $(\xi.\lesssim)$ (see Definition 4.3.1) have a different structure. In fact, $M \simeq N$ is *symmetric* in M and N, while $M \lesssim N$ is not.

Lemma 6.3.3 *There is an E-equation which is derivable in the $Free\mathrm{mon}\lambda_{\mathrm{p}}\beta$-calculus but not in the $Inhabited K\lambda_{\mathrm{p}}\beta\eta$-calculus*

Proof Since ECE- and E-equations have the same expressive power in the $J\lambda_{\mathrm{p}}\beta$-calculus, it is enough to exhibit an ECE-equation with the required property.

Let $a\colon(\iota \rightharpoonup \iota) \rightharpoonup \iota$, $b\colon \iota \rightharpoonup \iota$ and $c\colon \iota$ be three *variables* and let $\mathrm{I}' \triangleq (\lambda x.y \restriction (bc))$, then

$$(a\mathrm{I}')\downarrow \Longrightarrow (a\mathrm{I})\downarrow \tag{6.2}$$

is derivable in the $J\mathrm{mon}\lambda_{\mathrm{p}}\beta$-calculus but not in the $K\lambda_{\mathrm{p}}\beta\eta$-calculus.

It is easy to show (in the $J\mathrm{mon}\lambda_{\mathrm{p}}\beta$-calculus) that $\mathrm{I}' \leq \mathrm{I}$, therefore by monotonicity $(a\mathrm{I}')\downarrow \Longrightarrow (a\mathrm{I}') \leq (a\mathrm{I})$.

On the other hand, consider the full partial type hierarchy whose base type is the one-element set $\{c\}$ (see Definition 5.1.14), let b be the everywhere divergent function and a be the function s.t.

$$a(f) \simeq \begin{cases} c & \text{if } f(c) \text{ diverges} \\ undefined & \text{otherwise} \end{cases}$$

Then $(a\mathrm{I}')$ is defined, because I' is the everywhere divergent function, while $(a\mathrm{I})$ is undefined ∎

Theorem 6.3.4 Conservative extension results

For any $\chi \in \{Free, Inhabited\}$ and $\psi \in \{J, K\}$ (see Definition 6.1.3)

$$\begin{array}{ccccc}
\chi\psi\mathrm{LPT} & \subset^c & \chi\psi\mathrm{monLPT} & \equiv & \chi\psi\mathrm{monLPT} \\
\cap^c & & \cap^c & & \cap^c \\
\chi\psi\lambda_{\mathrm{p}}\beta\eta & & \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta & \subset^c_{0\vdash Ein} & \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta \\
? & & & & \\
\chi\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta & & & &
\end{array}$$

Proof By Lemma 1.1.4, the conservative extension results $\chi\psi\mathrm{LPT} \subset^c \chi\psi\lambda_{\mathrm{p}}\beta\eta$ and $\chi\psi\mathrm{monLPT} \subset^c \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$ are derivable from the others:

- $\chi\psi\mathrm{LPT} \subset^c \chi\psi\mathrm{monLPT}$ follows from $\mathrm{monLPT} \subseteq \mathrm{LPT} + \text{triv.} \leq$ (see Lemma 6.2.4). The proof is similar to that of $\chi\psi\mathrm{LPT}+\mathrm{tot} \subset^c \chi\psi\mathrm{monLPT}+\mathrm{tot}$ in Theorem 6.2.5

- $\chi\psi\mathrm{monLPT} \subset^c \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$, because any monotonic partial algebra is embedded in a *model of the least fixed-point partial lambda calculus* (see Section 5.3). This result is based on a complicated technique, but $\chi\psi\mathrm{LPT} \subset^c \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ can be proved very easily. In fact, a partial algebra $\mathcal{B}$ is embedded in the *full continuous partial type hierarchy* with base type the flat cpo whose underlying set is $\iota^{\mathcal{B}}$ (see Definition 5.1.26)

- $\chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset^c_{0\vdash Ein} \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$, because for any typed extensional monotonic partial applicative structure $\mathcal{A}$ there is a $\Sigma_{\mathrm{mon}\lambda}$-partial homomorphism (see Definition 5.2.7) from the *full monotonic partial type hierarchy* with base type the complete lattice $(\mathcal{P}(\iota^{\mathcal{A}}), \subseteq)$ to $\mathcal{A}$ (see Theorem 5.2.26)

? $\chi J\lambda_{\mathrm{p}}\beta\eta \subset^c_{0\vdash E} \chi J\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$, because $\lambda\mathrm{Y}\beta\eta$p-equational logic is a conservative extension of $\lambda\beta\eta$p-equational logic over pλ-equations (see Definition 7.3.2). We can only give an informal argument, because $\lambda\mathrm{Y}\beta\eta$p-equational logic has not been defined. However, we conjecture that the results about $\lambda\beta\eta$p-equational logic extend smoothly to it.

  Let M and N be pλ-terms in β-normal form (since we consider typed terms without fixed-point operator, this is not a restriction), then:

- $M \simeq N$ is derivable in $\lambda \mathrm{Y}\beta\eta$p-equational logic iff (see Theorem 8.4.4)
- $M \simeq_{d\eta} N$ iff (since pλ-terms are closed w.r.t. $\simeq_{d\eta}$ for pλY-terms)
- $M \simeq N$ is derivable in $\lambda\beta\eta$p-equational logic

We conjecture that $\chi K \lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash E} \chi K \lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$, but the proof above cannot be modified, since there is no counterpart to $\lambda\beta\eta$p-equational logic for the classical partial lambda calculus

∎

Theorem 6.3.5 Counterexamples to conservative extension results

For any $\chi \in \{Free, Inhabited\}$ and $\psi \in \{J, K\}$ (see Definition 6.1.3)

$$\begin{array}{ccc} \chi\psi\lambda_{\mathrm{p}}\beta\eta & \subset_{0\vdash E} & \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \\ \cap_{E\vdash E} & & \cap_{E\vdash E} \\ \chi\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta & \subset_{0\vdash E} & \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta \end{array}$$

Proof

- $\chi\psi\lambda_{\mathrm{p}}\beta\eta \subset_{0\vdash E} \chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$ follows immediately from Lemma 6.3.3
- $\chi\psi\lambda_{\mathrm{p}}\beta\eta \subset_{E\vdash E} \chi\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$ follows immediately from the corresponding counterexample for the total formal systems $\chi\psi\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset_{E\vdash E} \psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta + \mathrm{tot}$ (see remarks after Theorem 6.2.6).
- $\chi\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset_{E\vdash E} \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ follows immediately from the corresponding counterexample for the total formal systems (as the previous counterexample)
- $\chi\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta \subset_{0\vdash E} \chi\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ follows immediately from the corresponding counterexample for the total formal systems $\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta+\mathrm{tot} \subset_{0\vdash E} \psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta+\mathrm{tot}$ (see proof of Theorem 6.2.6)

∎

6.4 The pure typed lambda-theories

In this section we restrict our analysis to the **pure typed lambda-theories** $S(\chi\psi\mathrm{FS})$, i.e. the set of S-equations, between pure typed λ-terms without fixed-point operator, derivable in $\chi\psi$FS (see notation after Definition 1.0.1). En passant we mention the situation for the *untyped pure lambda-theories*. We take into account also the pure typed lambda-theories of the $\lambda_{\mathrm{v}}\beta\eta$- and $\lambda\beta\eta$-calculus.

First, we prove that there are at most six pure typed lambda-theories. Then, we compare them on two fragments, namely the S-fragment and the TE-fragment.

Theorem 6.4.1 Coincidence of Free and Inhabited formal systems
For any $\psi \in \{J, K\}$ (see Definition 6.1.3)

- $Free\psi\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} Inhabited\psi\lambda_{\mathrm{p}}\beta\eta$
- $Free\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash Sin} Inhabited\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$
- $FreeJ\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset^{c}_{0\vdash S} K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$

? $Free\psi\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} Inhabited\psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$

Proof

- $Free\psi\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} Inhabited\psi\lambda_{\mathrm{p}}\beta\eta$.

  For any (Kripke) set X there is a partial retraction from X onto $X+1$, namely $In_{\iota}(a) = <0, a>$ and $H_{\iota}(<n, b>) = a \Longleftrightarrow n = 0 \wedge b = a$.

  Therefore, by Theorem 5.2.22, for any model $\mathcal{B}'$ of the $Free\lambda_{\mathrm{p}}\beta\eta$-calculus there exists a partial homomorphism from the full partial type hierarchy $\mathcal{B}$, whose base type is the set $\iota^{\mathcal{A}} + 1$, onto $\mathcal{B}'$.

  Since $\iota^{\mathcal{A}} + 1$ is inhabited, then $\mathcal{B}$ is a model of the $Inhabited\lambda_{\mathrm{p}}\beta\eta$-calculus (see Definitions 5.1.14 and 5.1.16). The conservative extension result follows from the preservation property of partial homomorphisms (see Theorem 5.2.12).

- $Free\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash Sin} Inhabited\psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ is proved like the previous conservative extension result.

  Since for any (Kripke) poset $(X, \leq)$ there is a partial retraction from $(X, \leq)$ onto $(\mathcal{P}(X), \subseteq)$, namely $In_{\iota}(a) = \{x \in X | x \leq a\} = \hat{a}$ and $H_{\iota}(A) = a \Longleftrightarrow A = \hat{a}$.

  Therefore, by Theorem 5.2.26, for any model $\mathcal{B}'$ of the $Free\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$-calculus there exists a partial homomorphism from the full monotonic partial type hierarchy $\mathcal{B}$, whose base type is the poset $(\mathcal{P}(\iota^{\mathcal{B}}), \subseteq)$, onto $\mathcal{B}'$.

Since $(\mathcal{P}(\iota^{\mathcal{B}}), \subseteq)$ is a complete lattice, then $\mathcal{B}$ is a model of the $Inhabited\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$-calculus (see Definitions 5.1.20 and 5.1.22). The conservative extension results follows from the preservation property of partial homomorphisms (see Theorem 5.2.12).

- $FreeJ\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset^{c}_{0\vdash S} K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$ is a consequence of a result due to Plotkin (see Theorem 5.2.28).

  Assume that $M \simeq N$ is derivable in $K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$, we have to show that it is derivable in $FreeJ\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot}$:

  - by soundness, the full continuous type hierarchy with base type $\mathbf{N}_{\perp}$ satisfies $M = N$
  - by Theorem 5.2.28, $M = N$ is derivable in the $\lambda\beta\eta$-calculus
  - since the inference rules of the $\lambda\beta\eta$-calculus are derivable in $FreeJ\lambda_{\mathrm{p}}\beta\eta+$ tot, then

    $M = N$ is derivable in $FreeJ\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot}$

? $FreeJ\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} InhabitedJ\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$, because $\lambda\mathrm{Y}\beta\eta$p-equational logic is a conservative extension of $\lambda\beta\eta$p-equational logic over pλ-equations. As in the proof of $\chi J\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash E} \chi J\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$, we can only give an informal argument, similar to that of in Theorem 6.3.4.

  We conjecture that $FreeK\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} InhabitedK\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$.

∎

Remark 6.4.2 Untyped λ-models are always inhabited, therefore there is no issue of free versus inhabited untyped lambda calculi and the analogue of Theorem 6.4.1 is:

- $\psi\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} \psi\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$, because there is a *pure untyped* λ-term Y satisfying the axiom for fixed-point operators, namely $\lambda F.\Delta\Delta$, where $\Delta \stackrel{\Delta}{\equiv} (\lambda f, x.F(ff)x)$

? $\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash Sin} \psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ we do not know if this is true.

? $J\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset^{c}_{0\vdash S} K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$ we do not know if this is true. However, $J\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset^{c}_{0\vdash S} K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot}$

We define six subsets of the S-equational fragment, i.e. the set of S-equations between typed λ-terms over Σ_λ:

Definition 6.4.3 Pure theories

$$\begin{array}{ccc}
S(\lambda_{\mathrm{v}}\beta\eta) & & \\
\mid\cap & & \\
S(J\lambda_{\mathrm{p}}\beta\eta) & \subseteq & S(K\lambda_{\mathrm{p}}\beta\eta) \\
\mid\cap & & \mid\cap \\
S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) & \subseteq & S(K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \\
 & & \mid\cap \\
 & & S(\lambda_{\mathrm{t}}\beta\eta)
\end{array}$$

- $(M \simeq N) \in S(\lambda_{\mathrm{v}}\beta\eta)$ *iff* $M \equiv N$ *is derivable in the typed* $\lambda_{\mathrm{ipe}}\beta\eta$*-calculus (see Definition 4.4.1).*

  Equivalently $(M \simeq N) \in S(\lambda_{\mathrm{v}}\beta\eta)$ *iff* $M = N$ *is derivable in the typed* $\lambda_{\mathrm{v}}\beta\eta$*-calculus (see Definition 4.4.10 and remark after Theorem 4.4.15)*

- $(M \simeq N) \in S(J\lambda_{\mathrm{p}}\beta\eta)$ *iff* $M \simeq N$ *is derivable in* $\mathit{Free}J\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta)$ *iff* $M \simeq N$ *is derivable in* $\mathit{Free}J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(K\lambda_{\mathrm{p}}\beta\eta)$ *iff* $M \simeq N$ *is derivable in* $\mathit{Free}K\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta)$ *iff* $M \simeq N$ *is derivable in* $\mathit{Free}K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(\lambda\beta\eta)$ *iff* $M \simeq N$ *is derivable in* $\mathit{Free}K\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot}$.

  Equivalently $(M \simeq N) \in S(\lambda\beta\eta)$ *iff* $M = N$ *is derivable in the typed* $\lambda\beta\eta$*-calculus*

Remark 6.4.4 According to Theorem 6.4.1 (apart from the unproved conjecture about $\chi K\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$) all the other pure typed lambda-theories coincide with one of those introduced in Definition 6.4.3. More precisely, for any $\chi \in \{\mathit{Free}, \mathit{Inhabited}\}$ and $\psi \in \{J, K\}$

- $(M \simeq N) \in S(J\lambda_{\mathrm{p}}\beta\eta)$ iff $M \simeq N$ is derivable in $\chi J\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta)$ iff $M \simeq N$ is derivable in $\chi J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta$ or $\chi J\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$

- $(M \simeq N) \in S(K\lambda_{\mathrm{p}}\beta\eta)$ iff $M \simeq N$ is derivable in $\chi K\lambda_{\mathrm{p}}\beta\eta$

- $(M \simeq N) \in S(K\text{mon}\lambda_{\rm p}\beta\eta)$ iff $M \simeq N$ is derivable in $\chi K\text{mon}\lambda_{\rm p}\beta\eta$ or $\chi K\lambda_{\rm p}\mu\mathrm{Y}\beta\eta$

- $(M \simeq N) \in S(\lambda\beta\eta)$ iff $M \simeq N$ is derivable in $\chi\psi\lambda_{\rm p}\beta\eta$+tot or $\chi\psi\text{mon}\lambda_{\rm p}\beta\eta$+ tot or $\psi\lambda_{\rm p}\mu\mathrm{Y}\beta\eta + \text{tot}$

Now we establish proper inclusions among the six pure typed lambda-theories:

Theorem 6.4.5 Relations on the S-fragment

$$
\begin{array}{ccc}
S(\lambda_{\rm v}\beta\eta) & & \\
\cap & & \\
S(J\lambda_{\rm p}\beta\eta) & \subset & S(K\lambda_{\rm p}\beta\eta) \\
\cap & & \cap \\
S(J\text{mon}\lambda_{\rm p}\beta\eta) & \subset & S(K\text{mon}\lambda_{\rm p}\beta\eta) \\
& & \cap \\
& & S(\lambda_{\rm t}\beta\eta)
\end{array}
$$

Proof

- $S(\lambda_{\rm v}\beta\eta) \subset S(J\lambda_{\rm p}\beta\eta)$, because $(\lambda z.z)(xy) \simeq (xy)$ is in $S(J\lambda_{\rm p}\beta\eta)$, but not in $S(\lambda_{\rm v}\beta\eta)$. In fact, the two side of the S-equations are different $\beta\eta$-normal forms w.r.t. the reduction for the $\lambda_{\rm v}\beta\eta$-calculus, therefore they cannot be equated in the $\lambda_{\rm v}\beta\eta$-calculus (see [Plo75]).

- $S(K\text{mon}\lambda_{\rm p}\beta\eta) \subset S(\lambda\beta\eta)$, because $(\lambda x, z.xz)(xy) \simeq (\lambda z.xyz)$ is in $S(\lambda\beta\eta)$, but not in $S(K\text{mon}\lambda_{\rm p}\beta\eta)$. In fact, in any full partial type hierarchy (see Definition 5.1.20), when x is the everywhere divergent function, the l.h.s. is undefined while the r.h.s. is defined.

Since the $TE \cup ECS$-fragment has the same expressive power as the S-fragment in the partial lambda calculus (see Theorem 4.2.8), the other proper inclusions are a consequence of the following counterexamples:

- $S(J\text{mon}\lambda_{\rm p}\beta\eta) \not\subseteq S(K\lambda_{\rm p}\beta\eta)$, as $x\mathrm{I}' \downarrow \Longrightarrow x\mathrm{I} \downarrow$ is derivable in $FreeJ\text{mon}\lambda_{\rm p}\beta\eta$ but not in $FreeK\lambda_{\rm p}\beta\eta$, where $\mathrm{I}' \stackrel{\Delta}{\equiv} (\lambda x.x\restriction bc)$ (see Lemma 6.3.3)

- $S(K\lambda_{\rm p}\beta\eta) \not\subseteq S(J\text{mon}\lambda_{\rm p}\beta\eta)$, as $O\mathrm{I}f \downarrow, Of\mathrm{I} \downarrow, O\mathrm{I}g \downarrow, Og\mathrm{I} \downarrow, Ofg \downarrow \Longrightarrow Ogf \downarrow$ is derivable in $FreeK\lambda_{\rm p}\beta\eta$ but not in $FreeJ\text{mon}\lambda_{\rm p}\beta\eta$, where P is the λ-term $(\lambda f{:}\,\iota \rightharpoonup \iota.\lambda x{:}\,\iota.x\restriction fb)$ and $O \stackrel{\Delta}{\equiv} (\lambda f, g{:}\,\iota \rightharpoonup \iota.a(Pf)(Pg))$ (see Lemma 6.3.1)

∎

Remark 6.4.6 For the untyped pure lambda-theories we conjecture the same proper inclusions. However, we do not know whether $S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \not\subseteq S(K\lambda_{\mathrm{p}}\beta\eta)$, because there is no model similar to the full partial type hierarchy for the untyped partial lambda calculus. Anyway, by using Theorem 8.4.4, it is possible to prove that $S(J\lambda_{\mathrm{p}}\beta\eta) \subset S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta)$.

Strictly speaking TE-statments are not S-equations. However, any TE-statement is provably equivalent to an S-equation. Therefore, we can identify the TE-statements derivable in one of the typed lambda calculi with a subset of its pure typed lambda-theory.

Proposition 6.4.7 TE-statements as S-equations

For any λ-term M:

- $\mathrm{E}(M) \Longleftrightarrow \mathrm{I} \equiv (\lambda x.\mathrm{I})M$ *is derivable in the $\lambda_{\mathrm{ipe}}\beta$-calculus*
- $M\downarrow \Longleftrightarrow \mathrm{I} \simeq (\lambda x.\mathrm{I})M$ *is derivable in the $\lambda_{\mathrm{p}}\beta$-calculus*

Proof Straightforward ∎

Definition 6.4.8 Pure TE-theories

$$\begin{array}{ccc}
TE(\lambda_{\mathrm{v}}\beta\eta) & & \\
|\cap & & \\
TE(J\lambda_{\mathrm{p}}\beta\eta) & \subseteq & TE(K\lambda_{\mathrm{p}}\beta\eta) \\
|\cap & & |\cap \\
TE(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) & \subseteq & TE(K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \\
 & & |\cap \\
 & & TE(\lambda_{\mathrm{t}}\beta\eta)
\end{array}$$

where $TE(\_)$ is the set $\{M\downarrow \,|(\mathrm{I} \simeq (\lambda x.\mathrm{I})M) \in S(\_)\}$ for any pure typed lambda theory $S(\_)$ in Definition 6.4.3.

Theorem 6.4.9 Relations on the TE-fragment

$$\begin{array}{ccc}
TE(\lambda_{\mathrm{v}}\beta\eta) & & \\
||| & & \\
TE(J\lambda_{\mathrm{p}}\beta\eta) & \equiv & TE(K\lambda_{\mathrm{p}}\beta\eta) \\
||| & & ||| \\
TE(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) & \equiv & TE(K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \\
 & & \cap \\
 & & TE(\lambda_{\mathrm{t}}\beta\eta)
\end{array}$$

Proof The coincidence of $TE(\lambda_{\mathrm{v}}\beta\eta)$ with the four partial lambda-theories is an immediate consequence of Corollary 5.4.11.

The inclusion $TE(K\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \subset TE(\lambda_{\mathrm{t}}\beta\eta)$ is obvious. ∎

Remark 6.4.10 The same relations hold among the untyped pure lambda-theories on the TE-fragment. However, the proof of coincidence relies on Corollary 5.4.14 instead of Corollary 5.4.11.

Chapter 7

Equational presentation

The formal systems, for partial applicative structures, introduced in Chapter 4 are based on the language of first order logic, hence they have little in common with the *equational presentation* of the lambda calculus used in the literature (see [Bar81]). Without a similar equational presentation, it is very unlikely that we can adapt the *techniques* developed for the lambda calculus to the study of the partial lambda calculus, unless they are *model-theoretic*.

The aim of this chapter is to develop such an equational presentation for the **intuitionistic** (monotonic) partial lambda calculus. It is unlikely that the **classical** partial lambda calculus *has an equational presentation*. In Chapter 6 we support this claim on the ground that the classical partial lambda calculus involves *some kind of case analysis even at the equational level*. In Section 7.4 we make this more precise by stating a conjecture (see Remark 7.4.3).

The basic idea is to *extend* (mon)p-equational logic (see Definition 4.1.9 and 4.1.27) to deal with λ-terms. For simplicity, we consider only the pure untyped (monotonic) partial lambda calculus, i.e. there is only one sort and the only *term-constructors* are λ-abstraction and application. However, the definitions and theorems (of this chapter) extend easily to signatures containing extra constants (and functions) and to the typed partial lambda calculus. We define the analogue of p-terms in the presence of λ-abstraction, called *pλ-terms*, and two formal systems on the set of pλ-equations and pλ-inequations (considered as ordered pairs of pλ-terms):

- $\lambda\beta\eta$p-equational logic (see Section 7.2), which is equivalent to the $J\lambda_{\mathrm{p}}\beta\eta$-calculus restricted to E-equations

- $\lambda\beta\eta$monp-equational logic (see Section 7.3), which is equivalent to the Jmon$\lambda_{\mathrm{p}}\beta\eta$-calculus restricted to Ein-equations

$\lambda\beta\eta$p- and $\lambda\beta\eta$monp-equational logic are very similar. Therefore, in Section 7.3 on $\lambda\beta\eta$monp-equational logic, we indicate only what are the *relevant* differences from $\lambda\beta\eta$p-equational logic.

7.1 Pλ-terms

We introduce a set of inference rules for generating pλ-terms. There are other sets of inference rules for pλ-terms (see Definitions 7.1.11 and 7.1.19), but they are not so *convenient* when one wants to prove, for instance, that a function defined (by induction) on *raw pλ-terms* (see below) satisfies a certain property, when applied to a pλ-term. In fact:

- the *substitutive* inference rule ($*$) (see Definition 7.1.11) is *too different* from the formation rules for raw λ-terms to exploit the inductive definition of the function on raw pλ-terms.
- the *alternative* inference rule (x') (see Definition 7.1.19) is *too cumbersome*, as it has an unbounded number of premisses and uses an auxiliary function d(_) in the side condition and the premisses.

However, they will be used for other purposes.

Definition 7.1.1 Structural inference rules for pλ-terms
The set pTerm *of* **pλ-terms** *is generated by the following inference rules:*

$$x\ \frac{}{x\restriction\emptyset\in\mathrm{pTerm}}$$

$$x.\mathrm{app}\ \frac{t_1\restriction D_1\in\mathrm{pTerm}\qquad t_2\restriction D_2\in\mathrm{pTerm}}{x\restriction\{t_1t_2\}\cup D_1\cup D_2\in\mathrm{pTerm}}$$

$$\mathrm{app}\ \frac{t_1\restriction D_1\in\mathrm{pTerm}\qquad t_2\restriction D_2\in\mathrm{pTerm}}{t_1t_2\restriction\{t_1t_2\}\cup D_1\cup D_2\in\mathrm{pTerm}}$$

$$\lambda\ \frac{t\restriction D'\cup D\in\mathrm{pTerm}\qquad D\in\mathrm{pTerm}}{(\lambda x.t\restriction D')\restriction D\in\mathrm{pTerm}}\ x\notin\mathrm{FV}(D)$$

D is a **dλ-term** *iff $y\restriction D$ is a pλ-term.*

Notation 7.1.2 We identify D with $y\restriction D$, where y is a *fixed* variable. For instance, $D \in \text{pTerm}$ means that $y\restriction D \in \text{pTerm}$.

Remark 7.1.3 The p-terms (as defined in Definition 4.1.9) are exactly those generated by the inference rules above, except (λ). If the inference rule (λ) were replaced by the simpler rule

$$\text{w-}\lambda \quad \frac{t\restriction D' \in \text{pTerm} \qquad D \in \text{pTerm}}{(\lambda x.t\restriction D')\restriction D \in \text{pTerm}}$$

then the set of pλ-terms would no longer be closed w.r.t. $*$-substitution.

We cannot give definitions by induction on the *structure* of pλ-terms, because they are defined by a set of rules which is not deterministic. To overcome this problem we introduce a superset of the pλ-terms, whose elements, called *raw pλ-terms*, are generated by formation rules.

Definition 7.1.4 Raw λ-terms, raw pλ-terms and raw dλ-terms

The set rTerm *of* **raw λ-terms** *is generated by the following formation rules:*

$$x \quad \frac{}{x \in \text{rTerm}}$$

$$\text{app} \quad \frac{t_1 \in \text{rTerm} \qquad t_2 \in \text{rTerm}}{t_1 t_2 \in \text{rTerm}}$$

$$\lambda \quad \frac{D \subseteq_{fin} \text{rTerm} \qquad t \in \text{rTerm}}{(\lambda x.t\restriction D) \in \text{rTerm}}$$

D is a **raw dλ-term** *iff* $D \subseteq \text{rTerm}$.

$t\restriction D$ is a **raw pλ-term** *iff t is a raw λ-term and D is a raw dλ-term*

The formation rules for raw λ-terms are like those for λ-terms, with the only exception of (λ). Moreover, there is no extra formation rules for restriction operators, as in the case of $\lambda\restriction$-terms.

Example 7.1.5 We illustrate with few examples the difference between $\lambda\restriction$-, raw pλ- and pλ-terms. Let $M\restriction\emptyset$, $N\restriction\emptyset$, $P\restriction\emptyset$ and $Q\restriction\emptyset$ be pλ-term.

- $(M\restriction\{P\})N$ is a $\lambda\restriction$-term, but not a raw λ-term.

  $MN\restriction\{P\}$ is a raw pλ-term.

- $MN\restriction\emptyset$, $M\restriction\{x\}$ and $M\restriction\{NPQ\}$ are raw pλ-terms, but not pλ-terms.

  $MN\restriction\{MN\}$, $M\restriction\emptyset$ and $M\restriction\{NPQ, NP\}$ are pλ-terms.

The restriction operator in (raw) pλ-terms is meant to take care of divergence *in the background* and should not be treated as an *ordinary* term constructor, therefore some notions (e.g. that of subterm) have to be changed accordingly.

Definition 7.1.6 Subterms

Given a raw λ-term M we define the set $\mathrm{subt}(M)$ *of its* **subterms** *and the set* $\mathrm{subd}(M)$ *of its* **d-subterms***:*

- $\mathrm{subt}(x) \triangleq \{x\}$

  $\mathrm{subt}(M_1M_2) \triangleq \{M_1M_2\} \cup \mathrm{subt}(M_1) \cup \mathrm{subt}(M_2)$

  $\mathrm{subt}(\lambda x.M\restriction D) \triangleq \{\lambda x.M\restriction D\} \cup \mathrm{subt}(M) \cup (\cup_{N\in D}\mathrm{subt}(N))$

- $\mathrm{subd}(x) \triangleq \emptyset$

  $\mathrm{subd}(M_1M_2) \triangleq \mathrm{subd}(M_1) \cup \mathrm{subd}(M_2)$

  $\mathrm{subd}(\lambda x.M\restriction D) \triangleq D \cup \mathrm{subd}(M) \cup (\cup_{N\in D}\mathrm{subd}(N))$

$\mathrm{subt}(\_)$ *and* $\mathrm{subd}(\_)$ *are extended to raw dλ- and pλ-terms as follows:*

- $\mathrm{subt}(D) \triangleq \cup_{N\in D}\mathrm{subt}(N)$

  $\mathrm{subt}(M\restriction D) \triangleq \mathrm{subt}(M) \cup \mathrm{subt}(D)$

- $\mathrm{subd}(D) \triangleq D \cup (\cup_{N\in D}\mathrm{subd}(N))$

  $\mathrm{subd}(M\restriction D) \triangleq \mathrm{subd}(M) \cup \mathrm{subd}(D)$

Notation 7.1.7 We say that a raw λ-term is a **value** if it is a variable or a λ-abstraction $(\lambda x.t\restriction D)$. We say that a raw λ-term (and similarly for dλ- and pλ-terms) M **has no redundancies** iff all $N \in \mathrm{subd}(M)$ are not values.

It is immediate to see that pλ-terms have no redundancies.

In the rest of this section we establish two *basic* properties of pλ-terms:

- pλ-terms are closed w.r.t. $*$-substitution (see Definition 4.1.9)

- it is decidable whether a raw pλ-term is a pλ-term

Each of the these two properties amounts to say that the set of pλ-terms is generated by a *suitable* set of inference rules.

7.1.1 Closure w.r.t. $*$-substitution

The set of pλ-terms is closed w.r.t. $*$-substitution iff the inference rule $(*)$ is admissible. The simplest way to prove admissibility would be to derive $(*)$ from the structural inference rules for pλ-terms. However, the inference rule $(*)$ is not derivable, and its admissibility must be proved by induction on the proof of $M\restriction D \in \mathrm{pTerm}$ (see Proposition 7.1.10). When M is a variable, we rely on the admissibility of two instances, (0-$*$) and (w-$*$), of the inference rule $(*)$.

Lemma 7.1.8 (0-$*$) is admissible

The following inference rule is admissible in the set of pλ-terms:

$$\text{0-}* \ \frac{M\restriction D \in \mathrm{pTerm}}{D \in \mathrm{pTerm}}$$

Proof By induction on the proof of $M\restriction D \in \mathrm{pTerm}$:

- $(x) \Longrightarrow$ immediate (compare with the case $(x.\mathrm{app})$)
- $(x.\mathrm{app}) \Longrightarrow$

  To derive $D \in \mathrm{pTerm}$, i.e. $y\restriction D \in \mathrm{pTerm}$, it is enough to apply the rule $(y.\mathrm{app})$ (instead of $(x.\mathrm{app})$) to the premisses of the last rule in the proof of $M\restriction D \in \mathrm{pTerm}$.
- $(\mathrm{app}) \Longrightarrow$ similar to the case $(x.\mathrm{app})$
- $(\lambda) \Longrightarrow$ trivial

  In fact, $D \in \mathrm{pTerm}$ is a premiss of the last rule in the proof of $M\restriction D \in \mathrm{pTerm}$.

∎

Lemma 7.1.9 (w-$*$) is admissible

The following inference rule is admissible in the set of pλ-terms:

$$\text{w-}* \ \frac{M\restriction D \in \mathrm{pTerm} \qquad E \in \mathrm{pTerm}}{M\restriction D \cup E \in \mathrm{pTerm}}$$

Proof By induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$:

- $(x) \Longrightarrow$ There are two subcases: (y), $(y.\mathrm{app})$. Depending on the last rule in the proof of $E \in \mathrm{pTerm}$ (i.e. $y{\restriction}E \in \mathrm{pTerm}$).
  - $(y) \Longrightarrow$ immediate (compare with the subcase $(y.\mathrm{app})$)
  - $(y.\mathrm{app}) \Longrightarrow$

    To derive $x{\restriction}E \in \mathrm{pTerm}$ it is enough to apply the rule $(x.\mathrm{app})$ (instead of $(y.\mathrm{app})$) to the premisses of the last rule in the proof of $E \in \mathrm{pTerm}$.

- $(x.\mathrm{app}) \Longrightarrow$

  Let $t_1{\restriction}D_1 \in \mathrm{pTerm}$ and $t_3{\restriction}D_3 \in \mathrm{pTerm}$ be the premisses of the last rule (i.e. $(x.\mathrm{app})$) in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

  By IH $t_1{\restriction}D_1 \cup E \in \mathrm{pTerm}$ and
  $t_3{\restriction}D_3 \cup E \in \mathrm{pTerm}$, therefore

  $M{\restriction}D \cup E \in \mathrm{pTerm}$ by $(x.\mathrm{app})$.

- $(\mathrm{app}) \Longrightarrow$ similar to the case $(x.\mathrm{app})$

- $(\lambda) \Longrightarrow$ similar to the case $(x.\mathrm{app})$

  The side condition $x \notin \mathrm{FV}(D \cup E)$ can be satisfied by renaming the bound variable x, so that it does not occur free in E.

∎

Proposition 7.1.10 $(*)$ is admissible
The following inference rule is admissible in the set of pλ-terms:

$$* \;\frac{M{\restriction}D \in \mathrm{pTerm} \qquad N{\restriction}E \in \mathrm{pTerm}}{(M{\restriction}D){*}[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}}$$

Proof By induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$:

- $(x) \Longrightarrow$ There are two subcases: $x \not\equiv y$ and $x \equiv y$.
  - $x \not\equiv y \Longrightarrow$ the proof uses $(0\text{-}*)$ and $(\mathrm{w}\text{-}*)$

    By the admissible rule $(0\text{-}*)$ $E \in \mathrm{pTerm}$, therefore

    $x{\restriction}E \in \mathrm{pTerm}$ by the admissible rule $(\mathrm{w}\text{-}*)$.

- $x \equiv y \Longrightarrow$ trivial

  In fact, the conclusion of $(*)$ is $N{\restriction}E \in \mathrm{pTerm}$.

- $(x.\mathrm{app}) \Longrightarrow$

  Let $t_1{\restriction}D_1 \in \mathrm{pTerm}$ and $t_3{\restriction}D_3 \in \mathrm{pTerm}$ be the premisses of the last rule (i.e. $(x.\mathrm{app})$) in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

  There are two subcases: $x \not\equiv y$ and $x \equiv y$.

  - $x \not\equiv y \Longrightarrow$ by IH (compare with the case (app))
  - $x \equiv y \Longrightarrow$

    by IH $(t_1{\restriction}D_1)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$ and
    $(t_3{\restriction}D_3)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$,
    by $(y.\mathrm{app})$ $(D)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$, therefore
    $N{\restriction}(D)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$ by the admissible rule (w-$*$).

- $(\mathrm{app}) \Longrightarrow$

  Let $t_1{\restriction}D_1 \in \mathrm{pTerm}$ and $t_3{\restriction}D_3 \in \mathrm{pTerm}$ be the premisses of the last rule (i.e. (app)) in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

  By IH $(t_1{\restriction}D_1)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$ and
  $(t_3{\restriction}D_3)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$, therefore

  $(M{\restriction}D)*[y{:=}\,N{\restriction}E] \in \mathrm{pTerm}$ by (app).

- $(\lambda) \Longrightarrow$ similar to the case (app)

  The side condition $x \notin \mathrm{FV}((D)*[y{:=}\,N{\restriction}E])$ can be satisfied by renaming the bound variable x, so that it does not occur free in $N{\restriction}E$.

∎

The admissibility result, above, suggests a different set of inference rules for pλ-terms, mainly based on $*$-substitution:

Definition 7.1.11 Substitutive inference rules for pλ-terms

$$x \quad \frac{}{x{\restriction}\emptyset \in \mathrm{pTerm}}$$

$$\text{app}\ \frac{}{x_1x_2\restriction\{x_1x_2\} \in \text{pTerm}}$$

$$*\ \frac{t_1\restriction D_1 \in \text{pTerm} \qquad t_2\restriction D_2 \in \text{pTerm}}{(t_1\restriction D_1)*[x:=t_2\restriction D_2] \in \text{pTerm}}$$

$$\lambda\ \frac{t\restriction D' \cup D \in \text{pTerm} \qquad D \in \text{pTerm}}{(\lambda x.t\restriction D')\restriction D \in \text{pTerm}}\ x \notin \text{FV}(D)$$

We claim that the substitutive inference rules generate the set of pλ-terms. The claim is a consequence of the following facts:

- the structural inference rules are derivable from the substitutive inference rules
- the substitutive inference rules, except $(*)$, are derivable from the structural inference rules
- the inference rule $(*)$ is admissible in the set of pλ-terms

The first two facts are easy to prove, the third one is Proposition 7.1.10.

7.1.2 Decidability and alternative inference rules

To prove decidability of being a pλ-term, we introduce a deterministic set of inference rules for pλ-terms (see Definition 7.1.19) s.t. the *size* of the premisses is strictly less than the *size* of the conclusion. Therefore, to prove that a raw pλ-term $M\restriction D$ is a pλ-term we can proceed backwards by *deterministically* reducing a goal to a *finite* set of smaller subgoals, according to the structure of the raw λ-term M (see Theorem 7.1.21).

Definition 7.1.12 The size size(_)

size(_) *takes a raw λ-term and returns its* **size**. size(_) *is defined by induction on the structure of raw λ-terms:*

- $\text{size}(x) \triangleq 0$
- $\text{size}(t_1t_2) \triangleq 1 + \text{size}(t_1) + \text{size}(t_2)$
- $\text{size}(\lambda x.t\restriction D) \triangleq 1 + \text{size}(t\restriction D)$

size(_) *is extended* additively *to raw pλ-terms and raw dλ-terms:*

- $\mathrm{size}(D) \triangleq \sum_{t \in D} \mathrm{size}(t)$
- $\mathrm{size}(t \restriction D) \triangleq \mathrm{size}(t) + \mathrm{size}(D)$

Remark 7.1.13 size(_) is consistent with the *subterm* relation on raw λ-terms, i.e. if t_1 is a proper *subterm* of t_2, then $\mathrm{size}(t_1) < \mathrm{size}(t_2)$

In order to define the set of alternative inference rules we need a function d(_) which takes a raw λ-term t and returns the *smallest* D s.t. $t \restriction D$ is a pλ-term, when such a D exists (see Proposition 7.1.23).

Definition 7.1.14 d(_)

d(_) *is a partial function defined by induction on the structure of raw λ-terms (if any of the expressions* d(_) *in the r.h.s. is undefined, then the whole r.h.s. is undefined):*

- $\mathrm{d}(x) \triangleq \emptyset$
- $\mathrm{d}(t_1 t_2) \stackrel{\Delta}{\simeq} \{t_1 t_2\} \cup \mathrm{d}(t_1) \cup \mathrm{d}(t_2)$
- $\mathrm{d}(\lambda x.t \restriction D) \stackrel{\Delta}{\simeq} \begin{cases} \cup\{\mathrm{d}(t') \mid t' \in \mathrm{d}(t \restriction D) - D\} & \text{if } x \notin \mathrm{FV}(t') \text{ for all } t' \in \mathrm{d}(t \restriction D) - D \\ undefined & otherwise \end{cases}$

d(_) *is extended* additively *to raw pλ-terms and raw dλ-terms:*

- $\mathrm{d}(D) \stackrel{\Delta}{\simeq} \cup_{t \in D} \mathrm{d}(t)$
- $\mathrm{d}(t \restriction D) \stackrel{\Delta}{\simeq} \mathrm{d}(t) \cup \mathrm{d}(D)$

It is obvious that for any raw λ-term t it is decidable whether $\mathrm{d}(t)$ *is defined and* $\mathrm{d}(t)$ *is* computed effectively *from t.*

Proposition 7.1.15 d(_) is monotonic

If $E \subseteq D$ and $\mathrm{d}(D)$ is defined, then $\mathrm{d}(E)$ is defined and $\mathrm{d}(E) \subseteq \mathrm{d}(D)$.

Remark 7.1.16 Note that d(_) is monotonic w.r.t. the inverse of the inclusion order on raw dλ-terms (compare with the axiom (incl) of monp-equational logic in Definition 4.1.27). In fact, this (not the inclusion order) is the *natural* partial order on raw dλ-terms (and raw pλ-terms), because the more elements a dλ-term D has the more difficult it is for all of them to denote.

Lemma 7.1.17 Properties of $\mathrm{d}(\_)$

if M is a raw λ-term, $\mathrm{d}(M)$ is defined and $N \in \mathrm{d}(M)$, then:

1. $\mathrm{size}(N) \leq \mathrm{size}(M)$
2. $\mathrm{FV}(N) \subseteq \mathrm{FV}(M)$
3. *N is a* subterm *of M*
4. *N is not a value*
5. $\mathrm{d}(N)$ *is defined and* $N \in \mathrm{d}(N) \subseteq \mathrm{d}(M)$

Proof all statements are proved by induction on the size of the raw λ-term M. We prove only the first one, since the others have similar proofs.

- $(x) \Longrightarrow M \equiv x$

  The statement is vacuously true, because $\mathrm{d}(x) = \emptyset$

- $(\mathrm{app}) \Longrightarrow M \equiv t_1 t_2$, the proof uses the IH .

  If $N \in \mathrm{d}(M) \equiv \{t_1 t_2\} \cup \mathrm{d}(t_1) \cup \mathrm{d}(t_2)$, then either $N \equiv t_1 t_2 \equiv M$, or $N \in \mathrm{d}(t_i)$ for i either 1 or 2. In the first case $\mathrm{size}(N) = \mathrm{size}(M)$, otherwise:

  - since $\mathrm{size}(t_i) < \mathrm{size}(t_1 t_2) = \mathrm{size}(M)$, by IH

    $\mathrm{size}(N) \leq \mathrm{size}(t_i)$
  - since $\mathrm{size}(t_i) < \mathrm{size}(M)$, then

    $\mathrm{size}(N) < \mathrm{size}(M)$

- $(\lambda) \Longrightarrow M \equiv (\lambda x.t \restriction D)$, the proof uses the IH .

  If $N \in \mathrm{d}(M) \equiv \cup\{\mathrm{d}(t') | t' \in \mathrm{d}(t \restriction D) - D\}$, then $N \in \mathrm{d}(t')$ for some $t'' \in \mathrm{d}(t \restriction D)$ and $t' \in \mathrm{d}(t'')$, therefore:

  - since $\mathrm{size}(t \restriction D) < \mathrm{size}(\lambda x.t \restriction D) = \mathrm{size}(M)$, by IH

    $\mathrm{size}(t'') \leq \mathrm{size}(t \restriction D)$
  - since $\mathrm{size}(t'') < \mathrm{size}(M)$, by IH

    $\mathrm{size}(t') \leq \mathrm{size}(t'')$
  - since $\mathrm{size}(t') < \mathrm{size}(M)$, by IH

    $\mathrm{size}(N) \leq \mathrm{size}(t')$

– since $\text{size}(t') < \text{size}(M)$, then

$\text{size}(N) < \text{size}(M)$

∎

Proposition 7.1.18 $\mathrm{d}(\_)$ is idempotent

If $\mathrm{d}(D)$ *is defined, then* $\mathrm{d}(\mathrm{d}(D)) = \mathrm{d}(D)$.

If $E \subseteq \mathrm{d}(D)$ *(for some* D*), then* $E \subseteq \mathrm{d}(E)$.

Proof The proof uses the fifth property of $\mathrm{d}(\_)$ (see Lemma 7.1.17). By definition, $\mathrm{d}(D) = \cup\{\mathrm{d}(M) | M \in D\}$ and $\mathrm{d}(\mathrm{d}(D)) \simeq \cup\{\mathrm{d}(N) | \exists M \in D. N \in \mathrm{d}(M)\}$, therefore:

- $\mathrm{d}(\mathrm{d}(D))$ is defined, because $\mathrm{d}(N)$ is defined for all $N \in \mathrm{d}(D)$ (by the fifth property).
- $\mathrm{d}(\mathrm{d}(D)) \subseteq \mathrm{d}(D)$, because $\mathrm{d}(N) \subseteq \mathrm{d}(M) (\subseteq \mathrm{d}(D))$ for all $M \in D$ and $N \in \mathrm{d}(M)$ (by the fifth property).
- $\mathrm{d}(D) \subseteq \mathrm{d}(\mathrm{d}(D))$, because $N \in \mathrm{d}(N)$ for all $N \in \mathrm{d}(D)$ (by the fifth property).

If $E \subseteq \mathrm{d}(D)$, then $E \subseteq \mathrm{d}(E)$ is implied by $N \in \mathrm{d}(N)$ for all $N \in E$.

- by $N \in E \subseteq \mathrm{d}(D)$, there exists $M \in D$ s.t.

  $N \in \mathrm{d}(M)$
- by the fifth property

  $N \in \mathrm{d}(N)$

∎

Definition 7.1.19 Alternative inference rules for pλ-terms

If any of the expressions $\mathrm{d}(\_)$ *in the premises and side conditions is undefined, then the whole inference rule is not applicable:*

$$x' \quad \frac{\{t_i{\restriction}\mathrm{d}(t_i) \in \mathrm{pTerm} | t_1 t_2 \in D \land i = 1, 2\}}{x{\restriction}D \in \mathrm{pTerm}} \quad D = \mathrm{d}(D)$$

$$\mathrm{app}' \quad \frac{t_1{\restriction}D \in \mathrm{pTerm} \qquad t_2{\restriction}D \in \mathrm{pTerm}}{t_1 t_2{\restriction}D \in \mathrm{pTerm}} \quad t_1 t_2 \in D$$

$$\lambda' \; \frac{t\restriction D' \cup D \in \mathrm{pTerm} \qquad D \in \mathrm{pTerm}}{(\lambda x.t\restriction D')\restriction D \in \mathrm{pTerm}} \; x \notin \mathrm{FV}(D)$$

To prove that, in the alternative inference rule (x'), the *size* of the premisses is strictly less than the *size* of the conclusion, we rely on the following property:

Lemma 7.1.20 Decreasing Property

If $\mathrm{d}(t_1 t_2)$ *is defined, then* $\mathrm{size}(t_i\restriction \mathrm{d}(t_i)) < \mathrm{size}(\mathrm{d}(t_1 t_2))$ *(for* $i = 1, 2$*)*

Proof

- by definition of d(_)

  $\mathrm{d}(t_1 t_2) \equiv \{t_1 t_2\} \cup \mathrm{d}(t_1) \cup \mathrm{d}(t_2)$

- since $t_1 t_2$ is not a subterm of t_i, by Lemma 7.1.17

  $t_1 t_2 \notin \mathrm{d}(t_i)$, therefore

  $\mathrm{size}(\mathrm{d}(t_1 t_2)) = \mathrm{size}(t_1 t_2) + \mathrm{size}(\mathrm{d}(t_1) \cup \mathrm{d}(t_2))$

- by definition of size(_)

  $\mathrm{size}(t_i\restriction \mathrm{d}(t_i)) < \mathrm{size}(t_1 t_2) + \mathrm{size}(\mathrm{d}(t_1) \cup \mathrm{d}(t_2)) = \mathrm{size}(\mathrm{d}(t_1 t_2))$

∎

Theorem 7.1.21 Decidability

It is decidable whether $M\restriction D \in \mathrm{pTerm}$ *is derivable from the alternative inference rules.*

Proof We prove decidability of $M\restriction D \in \mathrm{pTerm}$ by induction on the size of the raw pλ-term $M\restriction D$. There are three cases to consider, depending on the structure of the raw λ-term M:

- $(x) \Longrightarrow M \equiv x$

  The only applicable rule is (x'). Therefore, $x\restriction D \in \mathrm{pTerm}$ is derivable iff the side condition $D = \mathrm{d}(D)$ is true and the premisses $t_i\restriction \mathrm{d}(t_i) \in \mathrm{pTerm}$ (for all $t_1 t_2 \in D$ and $i = 1, 2$) are derivable.

The side condition is clearly decidable. To complete the proof of decidability we have to show that the premisses are decidable, whenever the side condition is true. By IH , it is enough to prove that $\text{size}(t_i\restriction \text{d}(t_i)) < \text{size}(x\restriction D)$, whenever the side condition is true.

$\text{size}(t_i\restriction \text{d}(t_i)) <$ by Lemma 7.1.20
$\text{size}(\text{d}(t_1t_2)) \leq$ by $t_1t_2 \in D$
$\text{size}(\text{d}(D)) =$ by the side condition
$\text{size}(D) = \text{size}(x\restriction D)$

- (app) $\Longrightarrow M \equiv t_1t_2$

  The only applicable rule is (app'). Therefore, $t_1t_2\restriction D \in \text{pTerm}$ is derivable iff the side condition $t_1t_2 \in D$ is true and the premisses $t_i\restriction D \in \text{pTerm}$ (for $i = 1, 2$) are derivable.

  The side condition is clearly decidable. To complete the proof of decidability we have to show that the premisses are decidable. By IH , it is enough to prove that $\text{size}(t_i\restriction D) < \text{size}(t_1t_2\restriction D)$.

  $\text{size}(t_i\restriction D) = \text{size}(t_i) + \text{size}(D) <$
  $1 + \text{size}(t_1) + \text{size}(t_2) + \text{size}(D) =$
  $\text{size}(t_1t_2) + \text{size}(D) = \text{size}(t_1t_2\restriction D)$

- $(\lambda) \Longrightarrow M \equiv (\lambda x.t\restriction D')$

  The only applicable rule is (λ'). Therefore, $(\lambda x.t\restriction D')\restriction D \in \text{pTerm}$ is derivable iff the side condition $x \notin \text{FV}(D)$ is true and the premisses $t\restriction D' \cup D \in \text{pTerm}$ and $D \in \text{pTerm}$ are derivable.

  The side condition is clearly decidable. To complete the proof of decidability we have to show that the premisses are decidable. By IH , it is enough to prove that $(\text{size}(D) \leq)\text{size}(t\restriction D' \cup D) < \text{size}((\lambda x.t\restriction D')\restriction D)$.

  $\text{size}(t\restriction D' \cup D) = \text{size}(t) + \text{size}(D' \cup D) \leq$
  $\text{size}(t) + \text{size}(D') + \text{size}(D) < \text{size}((\lambda x.t\restriction D')\restriction D)$

∎

We claim that the alternative inference rules (of Definition 7.1.19) generate the set of pλ-terms. The claim is a consequence of the following facts:

- the alternative inference rules are derivable from the substitutive inference rules (of Definition 7.1.11)
- the structural inference rules (x) and (λ) (of Definition 7.1.1) are derivable from the alternative inference rules
- the structural inference rules (x.app) and (app) are admissible in the set generated by the alternative inference rules

The first fact is Proposition 7.1.22, the second is trivial and the third one is Proposition 7.1.26 and 7.1.27. The structural inference rules (x.app) and (app) are not derivable. To prove their admissibility, we rely on the admissibility of two instances, (0-$*$) and (w-$*$), of the inference rule ($*$) (see Lemmas 7.1.24 and 7.1.25) and on the *characterization* of $\mathrm{d}(t)$ as the smallest D s.t. $t\upharpoonright D$ is a pλ-term, when such a D exists (see Proposition 7.1.23).

Proposition 7.1.22 Derivability of the alternative inference rules

The alternative inference rules are derivable from the substitutive inference rules.

Proof Derivability of (x') is the most difficult to prove and relies on the following property of $\mathrm{d}(\_)$:

Claim 7.1.22.1 $D = \cup_{N_1N_2\in D}(\{N_1N_2\} \cup \mathrm{d}(N_1) \cup \mathrm{d}(N_2))$, *if* $D = \mathrm{d}(D)$

Proof

- since $D = \mathrm{d}(D)$, by definition of $\mathrm{d}(\_)$

  for all $N \in D$ there exists $M \in D$ s.t. $N \in \mathrm{d}(M)$

- by the fourth property of $\mathrm{d}(\_)$ (see Lemma 7.1.17)

  N must be of the form N_1N_2, therefore

  $D = \mathrm{d}(D) = \cup_{N\in D}\mathrm{d}(N) =$

  $\cup_{N_1N_2\in D}\mathrm{d}(N_1N_2) = \cup_{N_1N_2\in D}(\{N_1N_2\} \cup \mathrm{d}(N_1) \cup \mathrm{d}(N_2))$.

∎

To prove derivability of an inference rule, we must derive the conclusion from the premisses by the substitutive inference rules (provided the side condition is satisfied).

- $(x') \Longrightarrow$

  Suppose that $D = \mathrm{d}(D)$, we have to derive $x \restriction D \in \mathrm{pTerm}$ from $N_i \restriction \mathrm{d}(N_i) \in \mathrm{pTerm}$ (for all $N_1 N_2 \in D$ and $i = 1, 2$).

  - by the substitutive inference rules (app) and $(*)$

    $N_1 N_2 \restriction \{N_1 N_2\} \cup \mathrm{d}(N_1) \cup \mathrm{d}(N_2) \in \mathrm{pTerm}$ (for all $N_1 N_2 \in D$)
  - by the substitutive inference rules (x) and $(*)$

    $x \restriction \cup_{N_1 N_2 \in D} (\{N_1 N_2\} \cup \mathrm{d}(N_1) \cup \mathrm{d}(N_2)) \in \mathrm{pTerm}$
  - by the claim

    $x \restriction D \in \mathrm{pTerm}$

- (app') $\Longrightarrow$

  Suppose that $N_1 N_2 \in D$, we have to derive $N_1 N_2 \restriction D \in \mathrm{pTerm}$ from $N_i \restriction D \in \mathrm{pTerm}$ (for $i = 1, 2$).

  - by the substitutive inference rules (app) and $(*)$

    $N_1 N_2 \restriction \{N_1 N_2\} \cup D \in \mathrm{pTerm}$
  - by $N_1 N_2 \in D$

    $N_1 N_2 \restriction D \in \mathrm{pTerm}$

- $(\lambda') \Longrightarrow$

  (λ') is trivially derivable, since (λ) and (λ') are the same rule

∎

Proposition 7.1.23 Characterization of d(_)

If $M \restriction D \in \mathrm{pTerm}$ is derivable by the alternative inference rules, then:

- *($\mathrm{d}(D)$ and $\mathrm{d}(t)$ are defined, and)* $\mathrm{d}(t) \subseteq \mathrm{d}(D) = D$
- $t \restriction \mathrm{d}(t) \cup \mathrm{d}(E) \in \mathrm{pTerm}$ *is derivable by the alternative inference rules, for all $E \subseteq D$*

Proof Both statements are proved by induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$. There are three cases to consider, depending on the last inference rule used.

The proof of the first statement is:

- (x') $\Longrightarrow M \equiv x$, the proof uses the side condition.

  $D = \mathrm{d}(D)$ by the side condition and $\mathrm{d}(x) \triangleq \emptyset \subseteq D$ trivially

- (app') $\Longrightarrow M \equiv t_1 t_2$, the proof uses the IH and the side condition.

  $D = \mathrm{d}(D)$ by IH for $t_i{\restriction}D$ (with i either 1 or 2).

  We have to show that $\mathrm{d}(t_1 t_2) \subseteq D$:

  - by IH for $t_i{\restriction}D$

    $\mathrm{d}(t_i) \subseteq D$ (for $i = 1, 2$), therefore

    $\mathrm{d}(t_1 t_2) = \{t_1 t_2\} \cup \mathrm{d}(t_1) \cup \mathrm{d}(t_2) \subseteq \{t_1 t_2\} \cup D$

  - by the side condition $t_1 t_2 \in D$

    $(\mathrm{d}(t_1 t_2) =)\{t_1 t_2\} \cup \mathrm{d}(t_1) \cup \mathrm{d}(t_2) \subseteq D$

- (λ') $\Longrightarrow M \equiv (\lambda x.t{\restriction}D')$, the proof uses the IH and the side condition.

  $D = \mathrm{d}(D)$ by IH for D.

  We have to show that $\mathrm{d}(\lambda x.t{\restriction}D') \subseteq D$:

  - by IH for $t{\restriction}D' \cup D$

    $\mathrm{d}(t) \subseteq \mathrm{d}(D' \cup D) = D' \cup D$, therefore

    $F \triangleq \mathrm{d}(t{\restriction}D') - D' \subseteq (D' \cup D) - D' \subseteq D$

  - by IH for D

    $(F \subseteq) D = \mathrm{d}(D)$

  - by *monotonicity* of $\mathrm{d}(\_)$ (see Proposition 7.1.15)

    $\mathrm{d}(F) \subseteq \mathrm{d}(D)(= D)$

  - by the side condition $x \notin \mathrm{FV}(D)$

    $x \notin \mathrm{FV}(F)$, therefore

    $\mathrm{d}(\lambda x.t{\restriction}D') = \mathrm{d}(F)(\subseteq D)$

The proof of the second statement is:

- (*x*') $\Longrightarrow M \equiv x$, the proof uses the side condition and the premisses.

  We have to show that $x \restriction E \in \mathrm{pTerm}$:

  - since $E \subseteq D = \mathrm{d}(D)$, by monotonicity of $\mathrm{d}(\_)$
    $\mathrm{d}(E) \subseteq \mathrm{d}(D)(= D)$
  - by *idempotency* of $\mathrm{d}(\_)$ (see Proposition 7.1.18)
    $\mathrm{d}(\mathrm{d}(E)) = \mathrm{d}(E) \subseteq D$
  - the following assertions are derivable, since they are in the premisses of the last rule (i.e. (*x*')) in the proof of $M \restriction D \in \mathrm{pTerm}$
    $\{t_i \restriction \mathrm{d}(t_i) \in \mathrm{pTerm} | t_1 t_2 \in \mathrm{d}(E) \wedge i = 1, 2\}$
  - by applying the inference rule (*x*')
    $x \restriction \mathrm{d}(E) \in \mathrm{pTerm}$

- (app') $\Longrightarrow M \equiv t_1 t_2$, the proof uses the IH and the side condition.

  - by $E \subseteq D$ and the side condition $t_1 t_2 \in D$
    $\{t_1 t_2\} \cup E \subseteq D$
  - by IH for $t_i \restriction D$
    $(t_i \restriction \mathrm{d}(t_1 t_2) \cup \mathrm{d}(E) \equiv) t_i \restriction \mathrm{d}(t_i) \cup \mathrm{d}(t_1 t_2) \cup \mathrm{d}(E) \in \mathrm{pTerm}$
  - since $t_1 t_2 \in \mathrm{d}(t_1 t_2)$, by the inference rule (app')
    $t_1 t_2 \restriction \mathrm{d}(t_1 t_2) \cup \mathrm{d}(E) \in \mathrm{pTerm}$

- (λ') $\Longrightarrow M \equiv (\lambda x.t \restriction D')$, the proof uses the IH and the side condition.

  By the first statement, $\mathrm{d}(\lambda x.t \restriction D') = \mathrm{d}(F) \subseteq D = \mathrm{d}(D)$, where $F = \mathrm{d}(t \restriction D') - D'$. In the proof we rely on the equality

  Claim 7.1.23.1 $D' \cup \mathrm{d}(F) = \mathrm{d}(t \restriction D') \cup \mathrm{d}(F)$

Proof The claim is a consequence of the following inclusions:

  - $D' \subseteq \mathrm{d}(t \restriction D')$, in fact
    by the first statement
    $(D' \subseteq) D' \cup D \subseteq \mathrm{d}(D' \cup D)$
    by Proposition 7.1.18
    $D' \subseteq \mathrm{d}(D') (\subseteq \mathrm{d}(t \restriction D'))$

- $\mathrm{d}(t\restriction D') \subseteq D' \cup F$ (by definition of F)
- $F \subseteq \mathrm{d}(F)$, in fact

  by the first statement (see case (λ'))

  $F \subseteq D = \mathrm{d}(D)$

  by Proposition 7.1.18

  $F \subseteq \mathrm{d}(F)$

∎

Now we prove that $(\lambda x.t\restriction D')\restriction\mathrm{d}(F) \cup \mathrm{d}(E) \in \mathrm{pTerm}$

- by $F \subseteq D$ (see case (λ') in the proof of the first statement) and $E \subseteq D$

  $D' \cup F \cup E \subseteq D' \cup D$ and $F \cup E \subseteq D$
- by IH for $t\restriction D' \cup D$

  $t\restriction\mathrm{d}(t) \cup \mathrm{d}(D') \cup \mathrm{d}(F) \cup \mathrm{d}(E) \in \mathrm{pTerm}$, therefore

  $t\restriction D' \cup \mathrm{d}(F) \cup \mathrm{d}(E) \in \mathrm{pTerm}$, by the claim
- by IH for D

  $\mathrm{d}(F) \cup \mathrm{d}(E) \in \mathrm{pTerm}$
- since $F \cup E \subseteq D = \mathrm{d}(D)$, by monotonicity of $\mathrm{d}(\_)$ (see Proposition 7.1.15)

  $\mathrm{d}(F \cup E) \subseteq D$, therefore

  $x \notin \mathrm{FV}(\mathrm{d}(F \cup E))$, by the side condition $x \notin \mathrm{FV}(D)$
- by the inference rule (λ')

  $(\lambda x.t\restriction D')\restriction\mathrm{d}(F) \cup \mathrm{d}(E) \in \mathrm{pTerm}$

∎

Lemma 7.1.24 (0-$*$) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$0\text{-}*\ \frac{M\restriction D \in \mathrm{pTerm}}{D \in \mathrm{pTerm}}$$

Proof By induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$:

- (x') $\Longrightarrow$

  To derive $D \in \mathrm{pTerm}$, i.e. $y{\restriction}D \in \mathrm{pTerm}$, it is enough to apply the rule (y') (instead of (x')) to the premisses of the last rule in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

- (app') $\Longrightarrow$

  by IH for any of the two premisses of the last rule in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

- (λ') $\Longrightarrow$ trivial

  In fact, $D \in \mathrm{pTerm}$ is a premiss of the last rule in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

∎

Lemma 7.1.25 (w-$*$) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$\text{w-}*\ \frac{M{\restriction}D \in \mathrm{pTerm} \qquad E \in \mathrm{pTerm}}{M{\restriction}D \cup E \in \mathrm{pTerm}}$$

Proof By induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$:

- (x') $\Longrightarrow M \equiv x$, the proof uses the side condition the premisses and $E \in \mathrm{pTerm}$.

  We have to show that $x{\restriction}D \cup E \in \mathrm{pTerm}$. We rely on the fact that the last rule in the proof of $E \in \mathrm{pTerm}$ (i.e. $y{\restriction}E \in \mathrm{pTerm}$) must necessarily be ($y$').

  - by the side conditions $D = \mathrm{d}(D)$ and $E = \mathrm{d}(E)$
    $\mathrm{d}(D \cup E) \equiv \mathrm{d}(D) \cup \mathrm{d}(E) = D \cup E$
  - the following assertions are derivable, since they are the premisses of the last rules (i.e. (x') and (y')) in the proofs of $M{\restriction}D \in \mathrm{pTerm}$ and $E \in \mathrm{pTerm}$
    $\{t_i{\restriction}\mathrm{d}(t_i) \in \mathrm{pTerm} | t_1 t_2 \in D \cup E \wedge i = 1, 2\}$

- by applying the inference rule (x')

  $x \restriction D \cup E \in \mathrm{pTerm}$

- (app') $\Longrightarrow M \equiv t_1 t_2$, the proof uses the IH and the side condition.

  - by IH for $t_i \restriction D$

    $t_i \restriction D \cup E \in \mathrm{pTerm}$

  - by the side condition $t_1 t_2 \in D$

    $t_1 t_2 \in D \cup E$

  - by the inference rule (app')

    $t_1 t_2 \restriction D \cup E \in \mathrm{pTerm}$

- (λ') $\Longrightarrow M \equiv (\lambda x.t \restriction D')$, the proof uses the IH and the side condition.

  - by IH for $t \restriction D' \cup D$ and D

    $t \restriction D' \cup D \cup E \in \mathrm{pTerm}$ and

    $D \cup E \in \mathrm{pTerm}$

  - by the side condition $x \notin \mathrm{FV}(D)$ and by renaming the bound variable x so that it does not occur free in E

    $x \notin \mathrm{FV}(D \cup E)$

  - by the inference rule (λ')

    $(\lambda x.t \restriction D') \restriction D \cup E \in \mathrm{pTerm}$

∎

Proposition 7.1.26 (x.app) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$x.\mathrm{app}\ \frac{M \restriction D \in \mathrm{pTerm} \qquad N \restriction E \in \mathrm{pTerm}}{x \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}}$$

Proof Suppose that both $M \restriction D \in \mathrm{pTerm}$ and $N \restriction E \in \mathrm{pTerm}$ are derivable by the alternative inference rules. If we show that $\{MN\} \cup D \cup E = \mathrm{d}(\{MN\} \cup D \cup E)$ and $t_i \restriction \mathrm{d}(t_i) \in \mathrm{pTerm}$ (for all $t_1 t_2 \in \{MN\} \cup D \cup E$ and $i = 1, 2$) are derivable, then $x \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ is derivable by the inference rule (x').

- $\{MN\} \cup D \cup E = \mathrm{d}(\{MN\} \cup D \cup E)$. In fact,

  by Proposition 7.1.23

  $\mathrm{d}(M) \subseteq D = \mathrm{d}(D)$ and $\mathrm{d}(N) \subseteq E = \mathrm{d}(E)$, therefore

  $\mathrm{d}(\{MN\} \cup D \cup E) \equiv$

  $(\{MN\} \cup \mathrm{d}(M) \cup \mathrm{d}(N)) \cup \mathrm{d}(D) \cup \mathrm{d}(E) =$

  $\{MN\} \cup D \cup E$

- $t_i \restriction \mathrm{d}(t_i) \in \mathrm{pTerm}$ (for all $t_1 t_2 \in D$ and $i = 1, 2$) and similarly for E. In fact,

  by the admissible inference rule (0-$*$)

  $D \in \mathrm{pTerm}$ is derivable, but the last inference rule in its derivation must necessarily be (y'), therefore

  $t_i \restriction \mathrm{d}(t_i) \in \mathrm{pTerm}$ is derivable (for all $t_1 t_2 \in D$ and $i = 1, 2$)

- $M \restriction \mathrm{d}(M) \in \mathrm{pTerm}$ follows immediately by Proposition 7.1.23, and similarly for $N \restriction \mathrm{d}(N) \in \mathrm{pTerm}$

∎

Proposition 7.1.27 (app) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$\text{app} \ \frac{M \restriction D \in \mathrm{pTerm} \qquad N \restriction E \in \mathrm{pTerm}}{MN \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}}$$

Proof Suppose that both $M \restriction D \in \mathrm{pTerm}$ and $N \restriction E \in \mathrm{pTerm}$ are derivable by the alternative inference rules. If we show that $M \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ and $N \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ are derivable, then $MN \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ is derivable by the inference rule (app').

- $M \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ and similarly for N. In fact,

  by the admissible inference rule (y.app)

  $\{MN\} \cup D \cup E \in \mathrm{pTerm}$ is derivable

  by the admissible inference rule (w-$*$)

  $M \restriction \{MN\} \cup D \cup E \in \mathrm{pTerm}$ is derivable

∎

7.1.3 Discussion on pλ-terms

Why consider the restriction operator?

In the $J\lambda_{\mathrm{p}}\beta$-calculus the restriction operator is *redundant* (see Theorem 4.2.8), but in the $J\lambda_{\mathrm{p}}$-calculus (and the $J\lambda_{\mathrm{p}}\eta$-calculus) it does actually increase the expressive power of equations.

Remark 7.1.28 In $J\lambda_{\mathrm{p}}+\restriction$, any $\lambda\restriction$-term is provably equivalent to a pλ-term (see Proposition 7.2.13), but $(\lambda x.x\restriction\{yz\})$ is not equivalent to any λ-term.

For studying β-reduction (see Chapter 8) we can identify terms up to provable equivalence in the $J\lambda_{\mathrm{p}}$-calculus (or the $J\lambda_{\mathrm{p}}\eta$-calculus) at most, and the choice between λ-terms and $\lambda\restriction$-terms is an issue. The natural counterpart of β-reduction for the $J\lambda_{\mathrm{p}}\beta$-calculus $(\lambda x.M)N = M[x := N]$ is a conditional reduction

$$N\downarrow \Longrightarrow (\lambda x.M)N \simeq M[x := N]$$

We claim that the restriction operator is *necessary* in order to define β-reduction as a binary relation. In fact, there are two ways of turning this conditional reduction into a reduction:

- $(\lambda x.M)N \simeq (\lambda x.M[x := N])N$
- $(\lambda x.M)N \simeq M[x := N]\restriction N$

The first option is quite unsatisfactory, because it modifies the β-redex without removing it. However, it is the only option available, when we do not want the restriction operator.

Remark 7.1.29 The second option does not make sense for pλ-terms, because $(\lambda x.M)N$ and $M[x := N]\restriction N$ are not pλ-terms. However, there is an alternative reduction with the same underlying idea:

$$(\lambda x.M\restriction D)N\restriction\{(\lambda x.M\restriction D)N\}\cup E \simeq (M\restriction D)*[x := N\restriction E]$$

Note that for pλ-terms we do not have to keep *track* of N in the r.h.s., because this is done in the background. In fact, when N is not a value $N \in E$, by definition of pλ-term.

Now we explain why other notions of β-reduction proposed in the literature are not suitable for the $J\lambda_{\mathrm{p}}\beta$-calculus (see also Theorem 6.4.5):

- $(\lambda x.M)N \simeq M[x:=N]$ the usual β-reduction is not correct. For instance, if $M \equiv (\lambda y.x)$ and $N \equiv yz$, then the l.h.s. is undefined when N is undefined, while the r.h.s. is always defined.

- $(\lambda x.M)N \simeq M[x:=N]$ (if N is a *value*) the β_v-reduction (see [Plo75]) is too weak. For instance, if $M \equiv (\lambda x.x)$ and $N \equiv yz$, then the equation above is not provable in the $\lambda_{\mathrm{v}}\beta$-calculus, but it is provable in the $J\lambda_{\mathrm{p}}\beta$-calculus.

Are pλ-terms canonical?

In section 4.1.2 we have shown that every term with restriction operators is equivalent (in $\mathrm{LPT}+\restriction$ as well as $\mathrm{monLPT}+\restriction$) to *exactly* one p-term. Therefore p-terms can be regarded as *canonical forms* and provable equivalence between them is just syntactic equality. We show that pλ-terms are not *canonical forms* for $\lambda\restriction$-terms and that there are difficulties in trying to achieve this.

Definition 7.1.30 d-Conversion

Let M and N be two $\lambda\restriction$-terms.

- *$M \simeq_d N$ iff $M \simeq N$ is derivable in $J\lambda_{\mathrm{p}}+\restriction$ (**d-conversion**)*

- *$M \simeq_{md} N$ iff $M \simeq N$ is derivable in $J\mathrm{mon}\lambda_{\mathrm{p}}+\restriction$ (**md-conversion**)*

These two equivalence relations are different on $\lambda\restriction$-terms (see Theorem 6.3.3), therefore no set of $\lambda\restriction$-terms is canonical for both of them. These two equivalence relations are syntactic equality on p-terms (by Theorem 6.3.4) and α-conversion on λ-terms (by an open term model construction).

Example 7.1.31 The equivalence relations $\simeq_d$ and $\simeq_{md}$ are not trivial and do not coincide on pλ-terms, more precisely:

- syntactic equality is properly included in $\simeq_d$. In fact,

  $(\lambda x.x\restriction\{yz\})\restriction\{yz\} \simeq_d (\lambda x.x\restriction\emptyset)\restriction\{yz\}$.

  One can find more elaborated examples by following the same general idea, i.e. $(\lambda x.M\restriction D)\restriction E\cup F \simeq_d (\lambda x.M\restriction D\cup E)\restriction E\cup F$ (if $x \notin \mathrm{FV}(E)$).

- $\simeq_d$ is properly included in $\simeq_{md}$.
  Let $\mathrm{I} \equiv (\lambda x.x)$ and $\mathrm{I}' \equiv (\lambda x.x\restriction\{yz\})$, then
  $x\restriction\{x\mathrm{I}, x\mathrm{I}'\} \simeq_{md} x\restriction\{x\mathrm{I}'\}$, but $x\restriction\{x\mathrm{I}, x\mathrm{I}'\} \not\simeq_d x\restriction\{x\mathrm{I}'\}$.

A p-term $N\restriction E$ is characterized as the **biggest** raw p-term with **no redundancies** (see Notation 7.1.7) in its equivalence class w.r.t. d-conversion, or equivalently:

- $\{t\downarrow | t \in E\} \Longrightarrow N\downarrow$. i.e. all information about divergence is in E
- $\{t\downarrow | t \in E\} \Longrightarrow t_1t_2\downarrow$ implies $t_1t_2 \in E$, i.e. E is closed w.r.t. derivability

We want to show that for certain $\lambda\restriction$-terms there is no biggest raw pλ-term d-convertible to it, unless we allow infinite terms. Therefore, we cannot make pλ-terms canonical by requiring them to be **as big as possible**.

Proposition 7.1.32 *Let M_n be defined by induction on n:*

- $M_0 \stackrel{\Delta}{\equiv} yz$
- $M_{n+1} \stackrel{\Delta}{\equiv} (\lambda x.x\restriction\{M_n\})(\lambda x.x\restriction\{M_n\})$

then the sequent $M_0\downarrow, M_1\downarrow \Longrightarrow M_1 = M_{n+1}$ is derivable in $J\lambda_{\mathrm{p}} + \restriction$ for all n, but $M_n\downarrow \Longleftrightarrow M_m\downarrow$ is not derivable in $J\mathrm{mon}\lambda_{\mathrm{p}} + \restriction$ unless $n = m$.

Therefore, if there exists the biggest dλ-term D d-convertible to $\{M_0, M_1\}$, then it has to contain (something d-convertible to) M_n for all n, hence it has to be infinite.

Proof It is easy to derive $M_0\downarrow, M_1\downarrow \Longrightarrow M_1 = M_{n+1}$. For the other claim we exhibit a counterexample, namely a Kripke λ-structure (see Definition 4.2.25) which is a model of the $J\mathrm{mon}\lambda_{\mathrm{p}}$-calculus.

- K is ω with the inverse of the standard linear order $\leq$.
- The interpretation of ι is the lifting of 1 in the category of Kripke posets over K and monotonic partial functions, i.e.
  $\iota^{\mathcal{B}(\alpha)}$ is the poset $\{n \in \omega | n \leq \alpha + 1\}$ with the standard order and
  $\mathcal{B}(f)(n) = min(n, \alpha' + 1)$ for all $f\colon \alpha \to \alpha'$ (i.e. $\alpha' \leq \alpha$) and $n \in \iota^{\mathcal{B}(\alpha)}$ (i.e. $n \leq \alpha + 1$).

- $\mathrm{app}^{\mathcal{B}(\alpha)}$ is the monotonic partial function s.t.

$$\mathrm{app}^{\mathcal{B}(\alpha)}(n,m) \stackrel{\Delta}{\simeq} \begin{cases} \alpha+1 & \text{if } \alpha \leq n \leq \alpha+1 \\ \mathit{undefined} & \text{otherwise} \end{cases}$$

- Instead of defining $\Psi^{\mathcal{A}}$ we give only the clause for λ-abstraction:

$$[\![(\lambda x.t)]\!]^{\mathcal{B},\alpha}_{\rho} \stackrel{\Delta}{=} \begin{cases} n+1 & \text{if } n \text{ is the biggest } \alpha' \leq \alpha \text{ s.t.} \\ & [\![t]\!]^{\mathcal{B},\alpha'}_{\rho\restriction f\lfloor x:=\alpha'+1} \text{ is defined} \\ 0 & \text{otherwise} \end{cases}$$

We claim that $\mathcal{B}$ is a model of $J\mathrm{mon}\lambda_{\mathrm{p}} + \restriction$. Moreover, if ρ is the environment that maps every variable to 0, then for all stages $\alpha \in \mathrm{K}$

$$[\![M_n]\!]^{\mathcal{B},\alpha}_{\rho} \simeq \begin{cases} \alpha+1 & \text{if } \alpha \leq n \\ \mathit{undefined} & \text{otherwise} \end{cases}$$

$$[\![(\lambda x.x\restriction M_n)]\!]^{\mathcal{B},\alpha}_{\rho} = \begin{cases} \alpha+1 & \text{if } \alpha \leq n \\ n & \text{otherwise} \end{cases}$$

∎

7.2 $\lambda\beta\eta$p-Equational Logic

In this section we introduce an equational presentation of the $J\lambda_{\mathrm{p}}\beta\eta$-calculus, $\lambda\beta\eta$p-equational logic, which *consists* of p-equational logic *extended* with one inference rule ((ξ)) and two axioms ((β) and (η)). We prove that $\lambda\beta\eta$p-equational logic is $J\lambda_{\mathrm{p}}\beta\eta + \restriction$ restricted to the fragment of *pλ-equations*, by *extending* the proof given in Section 4.1.2 for p-equational logic.

The similarity between the pure inference rules (see Definition 7.2.3), that generate the smallest *$\lambda\beta\eta$p-theory*, and those for the $\lambda\beta\eta$-calculus (see Definition 3.2.3 of [Bar81]) makes it possible to (give definitions and) prove results for the partial lambda calculus by *analogy* with the lambda calculus. With this approach we will (define and) prove the analogues of some *familiar* (definitions and) results from the literature on the lambda calculus:

- Jacopini's lemma and the relation between λ-models and λ-algebras
- β-reduction and the Church-Rosser property

Definition 7.2.1 $\lambda\beta\eta$p-equational logic $\lambda\beta\eta$pEQL

A set T of pλ-equations (considered as ordered pairs of pλ-terms) is a $\lambda\beta\eta$**p-theory** *iff it is closed w.r.t. the inference rules*

$$\text{refl} \;\; t\restriction D \simeq t\restriction D$$

$$\text{symm} \;\; \frac{t_1\restriction D_1 \simeq t_2\restriction D_2}{t_2\restriction D_2 \simeq t_1\restriction D_1}$$

$$\text{trans} \;\; \frac{t_1\restriction D_1 \simeq t_2\restriction D_2 \qquad t_2\restriction D_2 \simeq t_3\restriction D_3}{t_1\restriction D_1 \simeq t_3\restriction D_3}$$

$$*\text{subst} \;\; \frac{t_1\restriction D_1 \simeq t_2\restriction D_2 \qquad t_3\restriction D_3 \simeq t_4\restriction D_4}{(t_1\restriction D_1)*[x{:=}\, t_3\restriction D_3] \simeq (t_2\restriction D_2)*[x{:=}\, t_4\restriction D_4]}$$

$$\xi \;\; \frac{t_1\restriction D_1' \cup D_1 \simeq t_2\restriction D_2' \cup D_2 \qquad D_1 \simeq D_2}{(\lambda x.t_1\restriction D_1')\restriction D_1 \simeq (\lambda x.t_2\restriction D_2')\restriction D_2} \;\; x \notin \mathrm{FV}(D_1 \cup D_2)$$

$$\beta \;\; (\lambda y.t\restriction D)y\restriction\{(\lambda y.t\restriction D)y\} \simeq t\restriction D \qquad t\restriction D \in \mathrm{pTerm}$$

$$\eta \;\; (\lambda y.xy\restriction\{xy\})\restriction\emptyset \simeq x\restriction\emptyset \qquad y \notin \mathrm{FV}(x)$$

$T \vdash M\restriction D \simeq N\restriction E$ in $\lambda\beta\eta$p-equational logic ($\lambda\beta\eta$pEQL) $\overset{\Delta}{\Longleftrightarrow}$ $M\restriction D \simeq N\restriction E$ is in any $\lambda\beta\eta$p-theory which contains T

Remark 7.2.2 There are three *natural* variants of $\lambda\beta\eta$p-equational logic, obtained by dropping either (β) or (η) or both: $\lambda\eta$p-, $\lambda\beta$p- and λp-equational logics. Each of them gives an equational presentation of the *corresponding* variant of the $J\lambda_{\mathrm{p}}\beta\eta$-calculus.

7.2.1 Pure $\lambda\beta\eta$p-equational Logic

To generate the smallest $\lambda\beta\eta$p-theory, we introduce a set of (derivable) inference rules, similar to that for the lambda calculus, which is partitioned into three groups:

- the structural inference rules, that are similar to the structural inference rules for pλ-terms (see Definition 7.1.1)
- the inference rules for *one-step parallel* β- and η-reduction

- symmetry and transitivity

Definition 7.2.3 Pure Inference Rules

The **pure inference rules** *for $\lambda\beta\eta p$-equational logic is the following set of rules on $p\lambda$-equations:*

$$x \quad x{\restriction}\emptyset \simeq x{\restriction}\emptyset$$

$$x.\mathrm{app}\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2 \qquad t_3{\restriction}D_3 \simeq t_4{\restriction}D_4}{x{\restriction}\{t_1t_3\} \cup D_1 \cup D_3 \simeq x{\restriction}\{t_2t_4\} \cup D_2 \cup D_4}$$

$$\mathrm{app}\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2 \qquad t_3{\restriction}D_3 \simeq t_4{\restriction}D_4}{(t_1t_3){\restriction}\{t_1t_3\} \cup D_1 \cup D_3 \simeq (t_2t_4){\restriction}\{t_2t_4\} \cup D_2 \cup D_4}$$

$$\xi\ \frac{t_1{\restriction}D_1' \cup D_1 \simeq t_2{\restriction}D_2' \cup D_2 \qquad D_1 \simeq D_2}{(\lambda x.t_1{\restriction}D_1'){\restriction}D_1 \simeq (\lambda x.t_2{\restriction}D_2'){\restriction}D_2}\ x \notin \mathrm{FV}(D_1 \cup D_2)$$

$$x.\beta\ \frac{t_1{\restriction}D_1 \cup D_2 \simeq t_3{\restriction}D_3 \qquad t_2{\restriction}D_2 \simeq t_4{\restriction}D_4}{x{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2 \simeq x{\restriction}(D_3){*}[y{:}= t_4{\restriction}D_4]}\ y \notin \mathrm{FV}(D_2)$$

$$\beta\ \frac{t_1{\restriction}D_1 \cup D_2 \simeq t_3{\restriction}D_3 \qquad t_2{\restriction}D_2 \simeq t_4{\restriction}D_4}{(\lambda y.t_1{\restriction}D_1)t_2{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2 \simeq (t_3{\restriction}D_3){*}[y{:}= t_4{\restriction}D_4]}\ y \notin \mathrm{FV}(D_2)$$

$$\eta\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2}{(\lambda y.t_1y{\restriction}\{t_1y\}){\restriction}D_1 \simeq t_2{\restriction}D_2}\ y \notin \mathrm{FV}(t_1)$$

$$\mathrm{symm}\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2}{t_2{\restriction}D_2 \simeq t_1{\restriction}D_1}$$

$$\mathrm{trans}\ \frac{t_1{\restriction}D_1 \simeq t_2{\restriction}D_2 \qquad t_2{\restriction}D_2 \simeq t_3{\restriction}D_3}{t_1{\restriction}D_1 \simeq t_3{\restriction}D_3}$$

Remark 7.2.4 One can easily extend the pure inference rules to generate the smallest λp-theory containing a set T of pλ-equations, by introducing two pure inference rules (schemas) $(x.T)$ and (T). This would make the similarities between pure and revisited inference rules (see Definition 3.2.9) more apparent.

We claim that the pure inference rules generate **the smallest $\lambda\beta\eta$p-theory** $pS(\lambda\beta\eta\mathrm{pEQL})$. Since they are derivable in $\lambda\beta\eta$p-equational logic, it is enough to show that any pλ-equation in $pS(\lambda\beta\eta\mathrm{pEQL})$ is derivable, or equivalently that the inference rules of $\lambda\beta\eta$p-equational logic are admissible in the set generated by the pure inference rules. We prove only admissibility of (refl) and ($*$subst), since the derivability of the other inference rules follows easily (e.g. the axiom (β) is derived by using the pure inference rule (β) and reflexivity). Reflexivity is derivable from the structural inference rules, while the inference rule ($*$subst) is not derivable and its admissibility is proved by induction on the proof of $M{\restriction}D \simeq N{\restriction}E$ (see Proposition 7.2.9). There are strong similarities with the proof that ($*$) is admissible in the set generated by the structural inference rules for pλ-terms (see Proposition 7.1.10). In particular, when the last rule in the proof of $M{\restriction}D \simeq N{\restriction}E$ is (x) or (x.app), we rely on the admissibility of two instances, (0-$*$subst) and (w-$*$subst), of the inference rule ($*$subst).

Remark 7.2.5 If we remove any of the non-structural pure inference rules (but ($x.\beta$) and (β) should be treated as one rule), then the inference rules (refl) and ($*$-subst) are still admissible. This is apparent by looking at the proofs of admissibility for (0-$*$subst), (w-$*$subst) and ($*$subst).

Proposition 7.2.6 (refl) is derivable

For any pλ-term $M{\restriction}D$ the pλ-equation $M{\restriction}D \simeq M{\restriction}D$ is derivable from the structural inference rules (of Definition 7.2.3)

Proof By induction on the proof of $M{\restriction}D \in \mathrm{pTerm}$ by using the structural inference rules (see Definition 7.1.1):

- (x) $\Longrightarrow$

  $x{\restriction}\emptyset \simeq x{\restriction}\emptyset$ by the axiom (x)

- (x.app) $\Longrightarrow$

  Let $t_1{\restriction}D_1 \in \mathrm{pTerm}$ and $t_3{\restriction}D_3 \in \mathrm{pTerm}$ be the premisses of the last rule (i.e. (x.app)) in the proof of $M{\restriction}D \in \mathrm{pTerm}$.

  By IH $t_1{\restriction}D_1 \simeq t_1{\restriction}D_1$ and $t_2{\restriction}D_2 \simeq t_2{\restriction}D_2$, therefore

  $M{\restriction}D \simeq M{\restriction}D$ by the inference rule (x.app)

- (app) $\Longrightarrow$ similar to the case (x.app)
- (λ) $\Longrightarrow$

  Let $t\restriction D' \cup D \in$ pTerm and $D \in$ pTerm be the premisses of the last rule (i.e. (λ)) in the proof of $M\restriction D \in$ pTerm.

  By IH $t\restriction D' \cup D \simeq t\restriction D' \cup D$ and $D \simeq D$,

  $x \notin \mathrm{FV}(D)$, because it is the side condition of (λ), therefore

  $M\restriction D \simeq M\restriction D$ by the inference rule (ξ)

∎

Lemma 7.2.7 (0-$*$subst) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$\text{0-}{*}\text{subst} \quad \frac{M\restriction D \simeq N\restriction E}{D \simeq E}$$

Proof By induction on the proof of $M\restriction D \simeq N\restriction E$:

- (x) $\Longrightarrow$ immediate (compare with the case (x.app))
- (x.app) $\Longrightarrow$

  To derive $D \simeq E$, i.e. $y\restriction D \simeq y\restriction E$, it is enough to apply the rule (y.app) (instead of (x.app)) to the premisses of the last rule in the proof of $M\restriction D \simeq N\restriction E$.
- (app) $\Longrightarrow$ similar to the case (x.app)
- (ξ) $\Longrightarrow$ trivial

  In fact, $D \simeq E$ is a premiss of the last rule in the proof of $M\restriction D \simeq N\restriction E$.
- ($x.\beta$) $\Longrightarrow$ similar to the case (x.app)
- (β) $\Longrightarrow$ similar to the case (x.app)
- (η) $\Longrightarrow$

  $D \simeq E$ by IH

- (symm) $\Longrightarrow$

  by IH $E \simeq D$, therefore $D \simeq E$ by (symm).

- (trans) $\Longrightarrow$ similar to the case (symm)

∎

Lemma 7.2.8 (w-$*$subst) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$\text{w-}{*}\text{subst}\ \frac{M{\restriction}D \simeq N{\restriction}E \qquad F \simeq G}{M{\restriction}D \cup F \simeq N{\restriction}E \cup G}$$

Proof By induction on the proof of $M{\restriction}D \simeq N{\restriction}E$:

- $(x) \Longrightarrow$ There are three subcases: (y), $(y.\text{app})$ and $(y.\beta)$. Depending on the last rule in the proof of $F \simeq G$ (i.e. $y{\restriction}F \simeq y{\restriction}G$).

  - $(y) \Longrightarrow$ immediate (compare with the subcase $(y.\text{app})$)
  - $(y.\text{app}) \Longrightarrow$

    To derive $x{\restriction}F \simeq x{\restriction}G$ it is enough to apply the rule $(x.\text{app})$ (instead of $(y.\text{app})$) to the premisses of the last rule in the proof of $F \simeq G$.
  - $(y.\beta) \Longrightarrow$ similar to the subcase $(y.\text{app})$

- $(x.\text{app}) \Longrightarrow$

  Let $t_1{\restriction}D_1 \simeq t_2{\restriction}D_2$ and $t_3{\restriction}D_3 \simeq t_4{\restriction}D_4$ be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M{\restriction}D \simeq N{\restriction}E$.

  By IH $t_1{\restriction}D_1 \cup F \simeq t_2{\restriction}D_2 \cup G$ and
  $t_3{\restriction}D_3 \cup F \simeq t_4{\restriction}D_4 \cup G$, therefore

  $M{\restriction}D \cup F \simeq N{\restriction}E \cup G$ by $(x.\text{app})$.

- (app) $\Longrightarrow$ similar to the case $(x.\text{app})$

- $(\xi) \Longrightarrow$ similar to the case $(x.\text{app})$

  The side condition $x \notin \mathrm{FV}(D \cup E \cup F \cup G)$ can be satisfied by renaming the bound variable x, so that it does not occur free either in F or in G.

- $(x.\beta) \Longrightarrow$ similar to the case $(x.\text{app})$
- $(\beta) \Longrightarrow$ similar to the case $(x.\text{app})$
- $(\eta) \Longrightarrow$ similar to the case $(x.\text{app})$
- (symm) $\Longrightarrow$ similar to the case $(x.\text{app})$
- (trans) $\Longrightarrow$ similar to the case $(x.\text{app})$

∎

Proposition 7.2.9 ($*$subst) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$*\text{subst}\ \frac{M{\upharpoonright}D \simeq N{\upharpoonright}E \qquad P{\upharpoonright}F \simeq Q{\upharpoonright}G}{(M{\upharpoonright}D)*[y := P{\upharpoonright}F] \simeq (N{\upharpoonright}E)*[y := Q{\upharpoonright}G]}$$

Proof By induction on the proof of $M{\upharpoonright}D \simeq N{\upharpoonright}E$:

- $(x) \Longrightarrow$ There are two subcases: $x \not\equiv y$ and $x \equiv y$.
  - $x \not\equiv y \Longrightarrow$ the proof uses (0-$*$subst) and (w-$*$subst)

    By the admissible rule (0-$*$subst) $F \simeq G$, therefore

    $x{\upharpoonright}F \simeq x{\upharpoonright}G$ by the admissible rule (w-$*$subst).
  - $x \equiv y \Longrightarrow$ trivial

    In fact, the conclusion of ($*$subst) is $P{\upharpoonright}F \simeq Q{\upharpoonright}G$.
- $(x.\text{app}) \Longrightarrow$

  Let $t_1{\upharpoonright}D_1 \simeq t_2{\upharpoonright}D_2$ and $t_3{\upharpoonright}D_3 \simeq t_4{\upharpoonright}D_4$ be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M{\upharpoonright}D \simeq N{\upharpoonright}E$.

  There are two subcases: $x \not\equiv y$ and $x \equiv y$.
  - $x \not\equiv y \Longrightarrow$ by IH (compare with the case (app))

- $x \equiv y \Longrightarrow$

  by IH $(t_1 \restriction D_1)*[y := P \restriction F] \simeq (t_2 \restriction D_2)*[y := Q \restriction G]$ and $(t_3 \restriction D_3)*[y := P \restriction F] \simeq (t_4 \restriction D_4)*[y := Q \restriction G]$,

  by (y.app) $(D)*[y := P \restriction F] \simeq (E)*[y := Q \restriction G]$, therefore

  $P \restriction (D)*[y := P \restriction F] \simeq Q \restriction (E)*[y := Q \restriction G]$ by the admissible rule (w-$*$subst).

- (app) $\Longrightarrow$

  Let $t_1 \restriction D_1 \simeq t_2 \restriction D_2$ and $t_3 \restriction D_3 \simeq t_4 \restriction D_4$ be the premisses of the last rule (i.e. (app)) in the proof of $M \restriction D \simeq N \restriction E$.

  By IH $(t_1 \restriction D_1)*[y := P \restriction F] \simeq (t_2 \restriction D_2)*[y := Q \restriction G]$ and $(t_3 \restriction D_3)*[y := P \restriction F] \simeq (t_4 \restriction D_4)*[y := Q \restriction G]$, therefore

  $(M \restriction D)*[y := P \restriction F] \simeq (N \restriction E)*[y := Q \restriction G]$ by (app).

- $(\xi) \Longrightarrow$ similar to the case (app)

  The side condition $x \notin \mathrm{FV}((D)*[y := P \restriction F] \cup (E)*[y := Q \restriction G])$ can be satisfied by renaming the bound variable x, so that it does not occur free either in $P \restriction F$ or in $Q \restriction G$.

- $(x.\beta) \Longrightarrow$ similar to the case (x.app)

- $(\beta) \Longrightarrow$ similar to the case (app)

- $(\eta) \Longrightarrow$ similar to the case (ξ)

- (symm) $\Longrightarrow$

  By (symm) $Q \restriction G \simeq P \restriction F$,

  by IH $(N \restriction E)*[y := Q \restriction G] \simeq (M \restriction D)*[y := P \restriction F]$, therefore

  $(M \restriction D)*[y := P \restriction F] \simeq (N \restriction E)*[y := Q \restriction G]$ by (symm).

- (trans) $\Longrightarrow$ the proof uses reflexivity

  Let $M \restriction D \simeq R \restriction H$ and $R \restriction H \simeq N \restriction E$ be the premisses of the last rule (i.e. (trans)) in the proof of $M \restriction D \simeq N \restriction E$.

  By reflexivity $P \restriction F \simeq P \restriction F$, By IH $(M{\restriction}D)*[y{:=}P{\restriction}F] \simeq (R{\restriction}H)*[y{:=}P{\restriction}F]$ and $(R{\restriction}H)*[y{:=}P{\restriction}F] \simeq (N{\restriction}E)*[y{:=}Q{\restriction}G]$, therefore $(M{\restriction}D)*[y{:=}P{\restriction}F] \simeq (N{\restriction}E)*[y{:=}Q{\restriction}G]$ by (trans).

∎

7.2.2 Equivalence of $\lambda\beta\eta$pEQL and $J\lambda_{\mathrm{p}}\beta\eta+{\restriction}$

In this section we establish the correspondence between $\lambda\beta\eta$p-equational logic and $J\lambda_{\mathrm{p}}\beta\eta+{\restriction}$, by extending the results of Section 4.1.2 as follows:

- in $J\lambda_{\mathrm{p}}+{\restriction}$ every $\lambda{\restriction}$-term t is provably equivalent to a pλ-term $\mathrm{s}(t)$. Therefore, pλ-equations have the same expressive power as S-equations in $J\lambda_{\mathrm{p}}\beta\eta+{\restriction}$. As pointed out in 7.1.3 there can be more than one pλ-term d-convertible to t.

- $\lambda\beta\eta$p-equational logic is $J\lambda_{\mathrm{p}}\beta\eta+{\restriction}$ restricted to the fragment of pλ-equations.

Remark 7.2.10 Similar results hold for the formal systems without either (β) or (η) or both. However, to prove the analogue of Proposition 7.2.17 for the $J\lambda_{\mathrm{p}}$- and $J\lambda_{\mathrm{p}}\eta$-calculus, we have to use a more general definition of model (see Definition 4.2.25).

We extend Definition 4.1.14 of saturation to $\lambda{\restriction}$-terms, so that $\mathrm{s}(t)$ is a pλ-term provably equivalent to t:

Definition 7.2.11 Saturation

The **saturation** $\mathrm{s}(t) \equiv \mathrm{st}(t){\restriction}\mathrm{sd}(t)$ *of* t *is defined by induction on the structure of the* $\lambda{\restriction}$*-term* t*:*

- $\mathrm{s}(x) \stackrel{\Delta}{\equiv} x{\restriction}\emptyset$

- $\mathrm{s}(t_1t_2) \stackrel{\Delta}{\equiv} \mathrm{st}(t_1)\mathrm{st}(t_2){\restriction}\{\mathrm{st}(t_1)\mathrm{st}(t_2)\} \cup \mathrm{sd}(t_1) \cup \mathrm{sd}(t_2)$,

- $\mathrm{s}(t_1{\restriction}t_2) \stackrel{\Delta}{\equiv} \mathrm{st}(t_1){\restriction}\mathrm{sd}(t_1) \cup \mathrm{sd}(t_2)$,

- $\mathrm{s}(\lambda x.t) \stackrel{\Delta}{\equiv} (\lambda x.\mathrm{s}(t)){\restriction}\emptyset$

The definition of $\mathrm{s}(\_)$ *is extended in the obvious way to sets of* $\lambda{\restriction}$*-terms:*

- $\mathrm{sd}(D) \stackrel{\Delta}{\equiv} \cup_{t\in D}\mathrm{sd}(t)$

- $\mathrm{s}(t\restriction D) \stackrel{\Delta}{\equiv} \mathrm{st}(t)\restriction \mathrm{sd}(t) \cup \mathrm{sd}(D)$

Example 7.2.12 pλ-terms are not necessarily invariant w.r.t. saturation, e.g. $\mathrm{s}((\lambda x.yz\restriction\emptyset)\restriction\{yz\}) \equiv (\lambda x.yz\restriction\{yz\})\restriction\{yz\}$.

Proposition 7.2.13 *If t is a $\lambda\restriction$-term , then $\mathrm{s}(t)$ is a pλ-term and $t \simeq \mathrm{s}(t)$ is derivable in $J\lambda_{\mathrm{p}} + \restriction$*

Proof By induction on the structure of the $\lambda\restriction$-term t. ∎

Proposition 7.2.14 *If $t\restriction D$ is a pλ-term , then $t\restriction D \simeq \mathrm{s}(t\restriction D)$ is derivable in λp-equational logic*

Proof By induction on the proof of $t\restriction D \in \mathrm{pTerm}$ by the structural inference rules. ∎

We prove that $\lambda\beta\eta$p-equational logic and $J\lambda_{\mathrm{p}}\beta\eta + \restriction$ are the same formal system on pλ-equations, by showing that any $\lambda\beta\eta$p-theory T is the set of pλ-equations valid in a Kripke model $\mathcal{B}_T$ of $J\lambda_{\mathrm{p}}\beta\eta + \restriction$.

Definition 7.2.15 The Kripke Term Model $\mathcal{B}_T$

Given a $\lambda\beta\eta$p-theory T, the Kripke partial applicative structure $\mathcal{B}_T$ over K_T is defined by (see also Definition 4.1.19):

- *K_T is the set of dλ-terms ordered by inclusion*
- *if $D \in \mathrm{K}_T$, then $\mathcal{A} \equiv \mathcal{B}_T(D)$ is the partial applicative structure s.t.:*
  - $\iota^{\mathcal{A}} \stackrel{\Delta}{\equiv} \{[t_1\restriction D_1]_D | (D \simeq D_1 \cup D) \in T\}$, *where*
    $[t_1\restriction D_1]_D \stackrel{\Delta}{\equiv} \{t_2\restriction D_2 | (t_1\restriction D_1 \cup D \simeq t_2\restriction D_2 \cup D) \in T\}$
  - $\mathrm{app}^{\mathcal{A}}([t_1\restriction D_1]_D, [t_1\restriction D_1]_D) \stackrel{\Delta}{\simeq} \begin{cases} [t_1t_2\restriction\{t_1t_2\} \cup (D_1 \cup D_2)]_D & \textit{if it is in } \iota^{\mathcal{A}} \\ \textit{undefined} & \textit{otherwise} \end{cases}$

  Moreover, for all type environments η, $\lambda\restriction$-terms $t \in \mathrm{Term}_{\iota}^{\Sigma_{\lambda\restriction}}(\eta)$, environments $\rho \in \eta^{\mathcal{A}}$ and substitutions $\sigma \in \prod \rho$

  $$[\![t]\!]_{\rho}^{\mathcal{B}_T,D} \simeq \begin{cases} [(\mathrm{s}(t))*[\sigma]]_D & \textit{if it is in } \iota^{\mathcal{A}} \\ \textit{undefined} & \textit{otherwise} \end{cases}$$

- *if $f\colon D_1 \to D_2$ (i.e. $D_1 \subseteq D_2$), then*
  $\mathcal{B}_T(f)([t\restriction D]_{D_1}) \stackrel{\Delta}{\equiv} [t\restriction D]_{D_2}$

Proposition 7.2.16 *$\mathcal{B}_T$ is a model of $J\lambda_{\mathrm{p}}\beta\eta + \restriction$.*

Proof We prove only that $\mathcal{B}_T$ is an extensional partial applicative structure, i.e. it satisfies (ext. $\simeq$).

Let $\mathcal{B} \equiv \mathcal{B}_T$, $\alpha \equiv D$, $d_i = [M_i \restriction D_i]_D \in \iota^{\mathcal{B}(\alpha)}$ (for $i = 1,2$). We have to show that $d_1 = d_2$, whenever they have the same applicative behaviour, i.e. $\mathrm{app}^{\mathcal{B}(\alpha')}(d_{1\restriction f}, a) \simeq \mathrm{app}^{\mathcal{B}(\alpha')}(d_{2\restriction f}, a)$ for all $f\colon \alpha \to \alpha'$ and $a \in \iota^{\mathcal{B}(\alpha')}$.

We consider the applications $\mathrm{app}^{\mathcal{B}(\alpha')}(d_{i\restriction f}, a)$ when α' is $E_j \equiv \{M_j x\} \cup D_j \cup D$ and a is $a_j = [x \restriction \emptyset]_{E_j}$ (for $j = 1,2$).

1. $[M_1 x \restriction \{M_1 x\} \cup D_1]_{E_1} =$ by definition of $\mathrm{app}^{\mathcal{A}}$
   $\mathrm{app}^{\mathcal{B}(E_1)}(d_1, a_1) =$ by the assumption
   $\mathrm{app}^{\mathcal{B}(E_1)}(d_2, a_1) =$ by definition of $\mathrm{app}^{\mathcal{A}}$
   $[M_2 x \restriction \{M_2 x\} \cup D_2]_{E_1}$

2. $[M_1 x \restriction \{M_1 x\} \cup D_1]_{E_2} =$ by definition of $\mathrm{app}^{\mathcal{A}}$
   $\mathrm{app}^{\mathcal{B}(E_2)}(d_1, a_2) =$ by the assumption
   $\mathrm{app}^{\mathcal{B}(E_2)}(d_2, a_2) =$ by definition of $\mathrm{app}^{\mathcal{A}}$
   $[M_2 x \restriction \{M_2 x\} \cup D_2]_{E_2}$

Now we have to derive the pλ-equation $M_1 \restriction D_1 \cup D \simeq M_2 \restriction D_2 \cup D$:

- by the first identity above

  $M_1 x \restriction E_1 \simeq M_2 x \restriction E_1 \cup E_2$

- from the second identity above, by (0-$*$subst)

  $E_1 \cup E_2 \simeq E_2$

- by reflexivity

  $M_2 x \restriction E_2 \simeq M_2 x \restriction E_2$

- by applying (w-$*$subst)

  $M_2 x \restriction E_1 \cup E_2 \simeq M_2 x \restriction E_2$

- by transitivity $M_1 x \restriction E_1 \simeq M_2 x \restriction E_2$, i.e.

  $M_1 x \restriction \{M_1 x\} \cup D_1 \cup D \simeq M_2 x \restriction \{M_2 x\} \cup D_2 \cup D$

- since x can be taken so that $x \notin \mathrm{FV}(M_i \restriction D_i \cup D)$ (for $i = 1, 2$), by (ξ)

  $(\lambda x.M_1 x \restriction \{M_1 x\}) \restriction D_1 \cup D \simeq (\lambda x.M_2 x \restriction \{M_2 x\}) \restriction D_2 \cup D$

- by (η)

  $(\lambda x.M_i x \restriction \{M_i x\}) \restriction D_i \cup D \simeq M_i \restriction D_i \cup D$ (for $i = 1, 2$)

- by transitivity

  $M_1 \restriction D_1 \cup D \simeq M_2 \restriction D_2 \cup D$, therefore

  $d_1 = [M_1 \restriction D_1]_D = [M_2 \restriction D_2]_D = d_2$

∎

Proposition 7.2.17 Characterization of $\mathcal{B}_T$

If T is a $\lambda\beta\eta$p-theory, then $\mathcal{B}_T \Vdash t_1 \simeq t_2$ iff $(\mathrm{s}(t_1) \simeq \mathrm{s}(t_2)) \in T$ for any pair of $\lambda\restriction$-terms

Proof The implication from right to left follows immediately from the way $\lambda\restriction$-terms are interpreted (see Definition 7.2.15).

The proof of the other implication is like that in Proposition 4.1.20. ∎

Theorem 7.2.18 *$\lambda\beta\eta$p-equational logic is $J\lambda_{\mathrm{p}}\beta\eta + \restriction$ restricted to pλ-equations, i.e. $T \vdash M \restriction D \simeq N \restriction E$ in $\lambda\beta\eta$p-equational logic iff $T \vdash M \restriction D \simeq N \restriction E$ in $J\lambda_{\mathrm{p}}\beta\eta + \restriction$*

Proof The implication from left to right is easy, in fact it is enough to prove that the inference rules for $\lambda\beta\eta$p-equational logic are derivable in $J\lambda_{\mathrm{p}}\beta\eta + \restriction$.

The proof of the other implication uses the characterization of $\mathcal{B}_T$ (see Lemma 7.2.17). by Proposition 7.2.14, a pλ-equation $t_1 \restriction D_1 \simeq t_2 \restriction D_2$ is equivalent to $\mathrm{s}(t_1 \restriction D_1) \simeq \mathrm{s}(t_2 \restriction D_2)$, in $\lambda\beta\eta$p-equational logic. Therefore, Lemma 7.2.17 implies that for any pair of pλ-terms $\mathcal{B}_T \Vdash t_1 \simeq t_2$ iff $t_1 \simeq t_2 \in T$.

We prove that $M \restriction D \simeq N \restriction E \in T$, provided $M \restriction D \simeq N \restriction E$ is derivable from T' in $J\lambda_{\mathrm{p}}\beta\eta + \restriction$ and T is a $\lambda\beta\eta$p-theory containing T':

- since T' is a set of pλ-equations, by Lemma 7.2.17

  $\mathcal{B}_T \Vdash T'$

- since $\mathcal{B}_T$ is a model of $J\lambda_{\rm p}\beta\eta$, by soundness

  $\mathcal{B}_T \Vdash M{\restriction}D \simeq N{\restriction}E$

- since $M{\restriction}D \simeq N{\restriction}E$ is a pλ-equation, by Lemma 7.2.17

  $(M{\restriction}D \simeq N{\restriction}E) \in T$

∎

7.3 $\lambda\beta\eta$monp-equational Logic

In this section we introduce an inequational presentation of the Jmon$\lambda_{\rm p}\beta\eta$-calculus, $\lambda\beta\eta$monp-equational logic, which *consists* of monp-equational logic *extended* with one inference rule ((ξ)) and four axioms ((β), (η), and their *duals*).

Notation 7.3.1 $t_2{\restriction}D_2 \lesssim t_1{\restriction}D_1$ is called the **dual** of $t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2$. The dual of an inference rule is the inference rule obtained by replacing premisses and conclusion with their duals.

In $\lambda\beta\eta$monp-equational logic a pλ-equation $t_1{\restriction}D_1 \simeq t_2{\restriction}D_2$ stands for the pair of pλ-inequations $t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2$ and its dual $t_2{\restriction}D_2 \lesssim t_1{\restriction}D_1$.

Definition 7.3.2 $\lambda\beta\eta$monp-Equational Logic $\lambda\beta\eta$monpEQL

A set T of pλ-inequations (considered as ordered pairs of pλ-terms) is a $\lambda\beta\eta$mon**p-theory** *iff it is closed w.r.t. the inference rules*

$$\text{incl}\quad t{\restriction}D \cup E \lesssim t{\restriction}D$$

$$\text{trans}\quad \frac{t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2 \qquad t_2{\restriction}D_2 \lesssim t_3{\restriction}D_3}{t_1{\restriction}D_1 \lesssim t_3{\restriction}D_3}$$

$$*\text{subst}\quad \frac{t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2 \qquad t_3{\restriction}D_3 \lesssim t_4{\restriction}D_4}{(t_1{\restriction}D_1)*[x:=t_3{\restriction}D_3] \lesssim (t_2{\restriction}D_2)*[x:=t_4{\restriction}D_4]}$$

$$\xi\quad \frac{t_1{\restriction}D_1' \cup D_1 \lesssim t_2{\restriction}D_2' \cup D_2 \qquad D_1 \lesssim D_2}{(\lambda x.t_1{\restriction}D_1'){\restriction}D_1 \lesssim (\lambda x.t_2{\restriction}D_2'){\restriction}D_2}\quad x \notin \mathrm{FV}(D_1 \cup D_2)$$

$$\beta\quad (\lambda y.t{\restriction}D)y{\restriction}\{(\lambda y.t{\restriction}D)y\} \simeq t{\restriction}D \qquad t{\restriction}D \in \mathrm{pTerm}$$

$$\eta\quad (\lambda y.xy{\restriction}\{xy\}){\restriction}\emptyset \simeq x{\restriction}\emptyset \qquad y \notin \mathrm{FV}(x)$$

$T \vdash M \restriction D \lesssim N \restriction E$ in $\lambda\beta\eta$monp-equational logic ($\lambda\beta\eta$monpEQL) $\stackrel{\Delta}{\Longleftrightarrow} M \restriction D \lesssim N \restriction E$ is in any $\lambda\beta\eta$monp-theory which contains T

The differences between $\lambda\beta\eta$pEQL and $\lambda\beta\eta$monpEQL are smaller than the inference rules suggest. In fact, both of them are obtained by extending the corresponding p-equational logic with the inference rule (ξ) and the axioms (β), (η), and their *duals*. However, in $\lambda\beta\eta$pEQL the *duals* are redundant, because they can be derived by symmetry. Most of the results about $\lambda\beta\eta$pEQL (see Section 7.2) hold *mutatis mutandis* for $\lambda\beta\eta$monpEQL. So we will give the definitions and state the main results, but we will skip all the proofs and technical lemmas, unless there are relevant differences from those for $\lambda\beta\eta$p-equational logic.

Remark 7.3.3 By interpreting a pλ-equation as a pair of pλ-inequations we get a relative interpretation of $\lambda\beta\eta$pEQL in $\lambda\beta\eta$monpEQL. More precisely, if T is a $\lambda\beta\eta$monp-theory, then the set of pλ-equations $\{t_1 \restriction D_1 \simeq t_2 \restriction D_2 | (t_1 \restriction D_1 \lesssim t_2 \restriction D_2) \in T \wedge (t_2 \restriction D_2 \lesssim t_1 \restriction D_1) \in T\}$ is a $\lambda\beta\eta$p-theory.

7.3.1 Pure $\lambda\beta\eta$monp-equational Logic

To generate the smallest $\lambda\beta\eta$monp-theory, we introduce a set of (derivable) inference rules.

Definition 7.3.4 Pure Inference Rules

The **pure inference rules** *for $\lambda\beta\eta$monp-equational logic are the following set of rules on pλ-inequations:*

$$x \quad x \restriction D \lesssim x \restriction \emptyset \qquad D \in \mathrm{pTerm}$$

$$x.\mathrm{app} \ \frac{t_1 \restriction D_1 \lesssim t_2 \restriction D_2 \qquad t_3 \restriction D_3 \lesssim t_4 \restriction D_4}{x \restriction \{t_1 t_3\} \cup D_1 \cup D_3 \lesssim x \restriction \{t_2 t_4\} \cup D_2 \cup D_4}$$

$$\mathrm{app} \ \frac{t_1 \restriction D_1 \lesssim t_2 \restriction D_2 \qquad t_3 \restriction D_3 \lesssim t_4 \restriction D_4}{(t_1 t_3) \restriction \{t_1 t_3\} \cup D_1 \cup D_3 \lesssim (t_2 t_4) \restriction \{t_2 t_4\} \cup D_2 \cup D_4}$$

$$\xi \ \frac{t_1 \restriction D'_1 \cup D_1 \lesssim t_2 \restriction D'_2 \cup D_2 \qquad D_1 \lesssim D_2}{(\lambda x.t_1 \restriction D'_1) \restriction D_1 \lesssim (\lambda x.t_2 \restriction D'_2) \restriction D_2} \quad x \notin \mathrm{FV}(D_1 \cup D_2)$$

$$x.\beta \ \frac{t_1{\restriction}D_1 \cup D_2 {\lesssim} t_3{\restriction}D_3 \qquad t_2{\restriction}D_2{\lesssim}t_4{\restriction}D_4}{x{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2{\lesssim}x{\restriction}(D_3)*[y:=t_4{\restriction}D_4]} \ y \notin \mathrm{FV}(D_2)$$

$$\beta \ \frac{t_1{\restriction}D_1 \cup D_2{\lesssim}t_3{\restriction}D_3 \qquad t_2{\restriction}D_2{\lesssim}t_4{\restriction}D_4}{(\lambda y.t_1{\restriction}D_1)t_2{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2{\lesssim}(t_3{\restriction}D_3)*[y:=t_4{\restriction}D_4]} \ y \notin \mathrm{FV}(D_2)$$

$$\eta \ \frac{t_1{\restriction}D_1{\lesssim}t_2{\restriction}D_2}{(\lambda y.t_1 y{\restriction}\{t_1 y\}){\restriction}D_1{\lesssim}t_2{\restriction}D_2} \ y \notin \mathrm{FV}(t_1)$$

$$x.\beta^{op} \ \frac{t_3{\restriction}D_3{\lesssim}t_1{\restriction}D_1 \cup D_2 \qquad t_4{\restriction}D_4{\lesssim}t_2{\restriction}D_2}{x{\restriction}(D_3)*[y:=t_4{\restriction}D_4]{\lesssim}x{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2} \ y \notin \mathrm{FV}(D_2)$$

$$\beta^{op} \ \frac{t_3{\restriction}D_3{\lesssim}t_1{\restriction}D_1 \cup D_2 \qquad t_4{\restriction}D_4{\lesssim}t_2{\restriction}D_2}{(t_3{\restriction}D_3)*[y:=t_4{\restriction}D_4]{\lesssim}(\lambda y.t_1{\restriction}D_1)t_2{\restriction}\{(\lambda y.t_1{\restriction}D_1)t_2\} \cup D_2} \ y \notin \mathrm{FV}(D_2)$$

$$\eta^{op} \ \frac{t_2{\restriction}D_2{\lesssim}t_1{\restriction}D_1}{t_2{\restriction}D_2{\lesssim}(\lambda y.t_1 y{\restriction}\{t_1 y\}){\restriction}D_1} \ y \notin \mathrm{FV}(t_1)$$

$$\text{trans} \ \frac{t_1{\restriction}D_1{\lesssim}t_2{\restriction}D_2 \qquad t_2{\restriction}D_2{\lesssim}t_3{\restriction}D_3}{t_1{\restriction}D_1{\lesssim}t_3{\restriction}D_3}$$

The difference between the pure inference rules for $\lambda\beta\eta$monpEQL and those for $\lambda\beta\eta$pEQL (see Definition 7.2.3) can be reduced to just one, namely the axiom (x). In fact, the inference rules of Definition 7.2.3, with (symm) replaced by the *duals* of the inference rules $(x.\beta)$, (β) and (η), generate the smallest $\lambda\beta\eta$p-theory.

We claim that the pure inference rules generate **the smallest** $\lambda\beta\eta$mon**p-theory** $pSin(\lambda\beta\eta\text{monpEQL})$. The proof is similar to that given in Section 7.2.1 to show that the pure inference rules for $\lambda\beta\eta$p-equational logic generate the smallest $\lambda\beta\eta$p-theory, therefore we mention only those steps where there is a difference, namely the case (x) in the proof of admissibility for (w-$*$subst) and ($*$subst).

Remark 7.3.5 If we remove any of the non-structural pure inference rules (but $(x.\beta)$ and (β) should be treated as one rule and similarly for $(x.\beta^{op})$ and (β^{op})), then the inference rules (incl) and ($*$-subst) are still admissible. This is apparent by looking at the proofs of admissibility for ($*$incl), (0-$*$subst), (w-$*$subst) and ($*$subst).

Proposition 7.3.6 (incl) is derivable

For any pλ-term $M\restriction D$ and dλ-term E the pλ-inequation $M\restriction D\cup E \lesssim M\restriction D$ is derivable from the structural inference rules (of Definition 7.3.4)

Proof Similar to the proof of Proposition 7.2.6 ∎

Corollary 7.3.7 ($*$incl) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$*\text{incl}\ \frac{M\restriction D \lesssim N\restriction E}{M\restriction D\cup F \lesssim N\restriction E}\ \ F\in \text{pTerm}$$

Proof The inference rule ($*$incl) is actually derivable from (incl) and (trans). However, we give a proof by induction on the derivation of $M\restriction D \lesssim N\restriction E$, which is correct even when (trans) is removed (see $\lesssim_{1d\eta}$ in Definition 8.3.2).

- $(x) \Longrightarrow$ immediate.

  In fact, $x\restriction D\cup F \lesssim x\restriction\emptyset$ is derivable by (x), since $D\cup F$ is a dλ-term.

- $(x.\text{app}) \Longrightarrow$

  Let $t_1\restriction D_1 \lesssim t_2\restriction D_2$ and $t_3\restriction D_3 \lesssim t_4\restriction D_4$ be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M\restriction D \lesssim N\restriction E$.

  By IH $t_1\restriction D_1\cup F \lesssim t_2\restriction D_2$ and
  $t_3\restriction D_3\cup F \lesssim t_4\restriction D_4$, therefore

  $M\restriction D\cup F \lesssim N\restriction E$ by $(x.\text{app})$.

- $(\text{app}) \Longrightarrow$ similar to the case $(x.\text{app})$

- $(\xi) \Longrightarrow$ similar to the case $(x.\text{app})$

  The side condition $x \notin \text{FV}(D \cup E \cup F)$ can be satisfied by renaming the bound variable x, so that it does not occur free in F.

- $(x.\beta) \Longrightarrow$ similar to the case $(x.\text{app})$

- $(\beta) \Longrightarrow$ similar to the case $(x.\text{app})$

- $(\eta) \Longrightarrow$ similar to the case $(x.\text{app})$

- $(\text{trans}) \Longrightarrow$

  Let $M{\restriction}D{\lesssim}R{\restriction}H$ and $R{\restriction}H{\lesssim}N{\restriction}E$ be the premisses of the last rule (i.e. (trans)) in the proof of $M{\restriction}D{\lesssim}N{\restriction}E$.

  By IH $M{\restriction}D \cup F{\lesssim}R{\restriction}H$ therefore $M{\restriction}D \cup F{\lesssim}N{\restriction}E$ by (trans).

∎

Lemma 7.3.8 (w-$*$subst) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$\text{w-}{*}\text{subst}\ \frac{M{\restriction}D{\lesssim}N{\restriction}E \qquad F{\lesssim}G}{M{\restriction}D \cup F{\lesssim}N{\restriction}E \cup G}$$

Proof By induction on the proof of $M{\restriction}D{\lesssim}N{\restriction}E$:

- $(x) \Longrightarrow$ There are three subcases: (y), $(y.\text{app})$ and $(y.\beta)$. Depending on the last rule in the proof of $F{\lesssim}G$ (i.e. $y{\restriction}F{\lesssim}y{\restriction}G$).

  – $(y) \Longrightarrow$ (compare with the subcase $(y.\text{app})$)

  – $(y.\text{app}) \Longrightarrow$

    $x{\restriction}F{\lesssim}x{\restriction}G$, by applying the rule $(x.\text{app})$ (instead of $(y.\text{app})$) to the premisses of the last rule in the proof of $F{\lesssim}G$.

    Therefore, $x{\restriction}F \cup D{\lesssim}x{\restriction}G$ by the admissible inference rule ($*$incl).

  – $(y.\beta) \Longrightarrow$ similar to the subcase $(y.\text{app})$

- the other cases are like Lemma 7.2.8

∎

Proposition 7.3.9 (∗subst) is admissible

The following inference rule is admissible in the set generated by the pure inference rules:

$$*\text{subst}\ \frac{M{\restriction}D{\lesssim}N{\restriction}E \qquad P{\restriction}F{\lesssim}Q{\restriction}G}{(M{\restriction}D)*[y{:=}P{\restriction}F]{\lesssim}(N{\restriction}E)*[y{:=}Q{\restriction}G]}$$

Proof By induction on the proof of $M{\restriction}D{\lesssim}N{\restriction}E$:

- $(x) \Longrightarrow$ the proof uses (0-∗subst) and (w-∗subst).

  There are two subcases: $x \not\equiv y$ and $x \equiv y$.

  - $x \not\equiv y \Longrightarrow$

    By the admissible rule (0-∗subst) $F{\lesssim}G$, therefore $x{\restriction}D \cup F{\lesssim}x{\restriction}G$ by the admissible rule (w-∗subst).

  - $x \equiv y \Longrightarrow$

    By the admissible rule (0-∗subst) $D{\lesssim}\emptyset$, therefore $P{\restriction}D \cup F{\lesssim}Q{\restriction}G$ by the admissible rule (w-∗subst).

- the other cases are like Proposition 7.2.9

∎

7.3.2 Equivalence of $\lambda\beta\eta$monpEQL and $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \restriction$

In this section we establish the correspondence between $\lambda\beta\eta$monp-equational logic and $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \restriction$, namely:

- $\lambda\beta\eta$monp-equational logic is $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \restriction$ restricted to the fragment of pλ-inequations.

We prove that $\lambda\beta\eta$monp-equational logic and $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \restriction$ are the same formal system on pλ-inequations, by showing that any $\lambda\beta\eta$monp-theory T is the set of pλ-inequations valid in a Kripke model $\mathcal{B}_T$ of $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + \restriction$.

Definition 7.3.10 The Kripke Term Model $\mathcal{B}_T$

Given a $\lambda\beta\eta$monp*-theory* T*, the Kripke monotonic partial applicative structure* $\mathcal{B}_T$ *over* K_T *is like that in Definition 7.2.15, where* $(t_1{\upharpoonright}D_1 \simeq t_2{\upharpoonright}D_2) \in T$ *means that* $(t_1{\upharpoonright}D_1{\lesssim}t_2{\upharpoonright}D_2) \in T$ *and* $(t_2{\upharpoonright}D_2{\lesssim}t_1{\upharpoonright}D_1) \in T$*, moreover*

$$[t_1{\upharpoonright}D_1]_D \leq^{\mathcal{A}} [t_2{\upharpoonright}D_2]_D \stackrel{\Delta}{\Longleftrightarrow} (t_1{\upharpoonright}D_1 \cup D{\lesssim}t_2{\upharpoonright}D_2 \cup D) \in T$$

for all $D \in \mathrm{K}_T$*, and* $\mathcal{A} \equiv \mathcal{B}_T(D)$

Proposition 7.3.11 $\mathcal{B}_T$ *is a model of* $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$*.*

Proposition 7.3.12 Characterization of $\mathcal{B}_T$

If T *is a* $\lambda\beta\eta$monp*-theory, then* $\mathcal{B}_T \Vdash t_1{\lesssim}t_2$ *iff* $(\mathrm{s}(t_1){\lesssim}\mathrm{s}(t_2)) \in T$ *for any pair of* $\lambda{\upharpoonright}$*-terms*

Theorem 7.3.13 $\lambda\beta\eta$monp*-equational logic is* $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$ *restricted to* pλ*-inequations, i.e.* $T \vdash M{\upharpoonright}D{\lesssim}N{\upharpoonright}E$ *in* $\lambda\beta\eta$monp*-equational logic iff* $T \vdash M{\upharpoonright}D{\lesssim}N{\upharpoonright}E$ *in* $J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$

7.3.3 Decidability of the d-preorder

In this section we show that it is decidable whether $t_1{\upharpoonright}D_1{\lesssim}t_2{\upharpoonright}D_2$ is derivable in λmonp-equational logic. This decidability result is particularly useful in connection with the Church-Rosser property for β-reduction to prove other decidability results (see remarks after Theorem 8.4.4).

Remark 7.3.14 We claim decidability also for $\lambda\eta$monp-equational logic, but the proof is far more complicated and we preferred not to give it.

We do not know whether λp- and $\lambda\eta$p-equational logic are decidable.

> The *idea* to prove decidability is to remove the rule (trans) (but making sure that it is admissible) and use structural rules s.t. the *size* of the r.h.s. in the premisses is strictly less than the size of the r.h.s. in the conclusion. However, the *size* of the l.h.s. may increase, therefore the argument cannot be carried over to $\simeq$, which is symmetric.

Some of the tools used to prove decidability of λmonp-equational logic have already been developed in Section 7.1.2.

Notation 7.3.15 If D and E are dλ-terms, then $D \sqsubseteq E$ is a shorthand for

$$\forall(t_1t_2) \in E.\exists(t'_1t'_2) \in D.t'_1\restriction D\lesssim t_1\restriction \mathrm{d}(t_1) \wedge t'_2\restriction D\lesssim t_2\restriction \mathrm{d}(t_2)$$

Definition 7.3.16 Alternative Inference Rules

The **alternative inference rules** *for* λmon*p-equational logic are the following set of rules on pλ-inequations:*

$$x\text{'}\quad \frac{D_1 \sqsubseteq D_2}{x\restriction D_1\lesssim x\restriction D_2}$$

$$\text{app'}\quad \frac{t_1\restriction D_1\lesssim t_2\restriction D_2 \qquad t_3\restriction D_1\lesssim t_4\restriction D_2}{(t_1t_3)\restriction D_1\lesssim(t_2t_4)\restriction D_2}$$

$$\xi\text{'}\quad \frac{t_1\restriction D_1 \cup D'_1\lesssim t_2\restriction D_2 \cup D'_2 \qquad D'_1\lesssim D'_2}{(\lambda x.t_1\restriction D_1)\restriction D'_1\lesssim(\lambda x.t_2\restriction D_2)\restriction D'_2}\quad x \notin \mathrm{FV}(D'_1 \cup D'_2)$$

Remark 7.3.17 The alternative inference rules for pλ-terms (see Definition 7.1.19) can be obtained from the inference rules above by *replacing* $t_1\restriction D_1\lesssim t_2\restriction D_2$ with $t_2\restriction D_2 \in \mathrm{pTerm}$.

Theorem 7.3.18 Decidability

It is decidable whether $t_1\restriction D_1\lesssim t_2\restriction D_2$ *is derivable from the inference rules* (x'), (ξ') *and* (app').

Proof By Theorem 7.1.21, it is decidable whether $t_1\restriction D_1$ and $t_2\restriction D_2$ are pλ-terms, therefore we do not worry about that. We proceed by induction on the derivation of $t_2\restriction D_2 \in \mathrm{pTerm}$ by the alternative inference rules (see Definition 7.1.19). There are three cases to consider, depending on the structure of t_2:

1. $(x) \Longrightarrow$

   $t_1\restriction D_1\lesssim t_2\restriction D_2$ is derivable iff $t_1 \equiv x$ and $D_1 \sqsubseteq D_2$

   but $D_1 \sqsubseteq D_2$ is decidable, in fact the two quantifications are over finite sets and the quantified formula is decidable, by IH .

2. $(\lambda) \Longrightarrow t_2 \equiv (\lambda x.t_4\restriction D_4)$

   $t_1\restriction D_1\lesssim t_2\restriction D_2$ is derivable iff $t_1 \equiv (\lambda x.t_3\restriction D_3)$ (for some $t_3\restriction D_3$), $t_3\restriction D_3 \cup D_1\lesssim t_4\restriction D_4 \cup D_2$ and $D_1\lesssim D_2$

   but $t_3\restriction D_3 \cup D_1\lesssim t_4\restriction D_4 \cup D_2$ and $D_1\lesssim D_2$ are decidable, by IH .

3. (app) $\Longrightarrow t_2 \equiv (t_2' t_2'')$

   $t_1 \restriction D_1 \lesssim t_2 \restriction D_2$ is derivable iff $t_2 \equiv t_2' t_2''$ (for some t_1' and t_1''), $t_1' \restriction D_1 \lesssim t_2' \restriction D_2$ and $t_1'' \restriction D_1 \lesssim t_2'' \restriction D_2$

   but $t_1' \restriction D_1 \lesssim t_2' \restriction D_2$ (and similarly for the other pλ-inequation) is decidable, by IH .

∎

We claim that the alternative inference rules (of Definition 7.3.16) generate the smallest λmonp-theory. The claim is a consequence of the following facts:

- the alternative inference rules are derivable in λmonp-equational logic (see Definition 7.3.2)
- the pure inference rules (x) and (ξ) (of Definition 7.3.4) are derivable from the alternative inference rules
- the pure inference rules (trans), $(x.\text{app})$ and (app) are admissible in the set generated by the alternative inference rules

The first fact is Proposition 7.3.19, the second is trivial and the third is Propositions 7.3.21, 7.3.23 and 7.3.24. The inference rule (trans) is not derivable. To prove its admissibility we rely on the admissibility of ($*$incl) and (w-$*$subst) (see Lemma 7.3.20). To prove admissibility of $(x.\text{app})$ and (app) we rely also on the admissibility of (0-$*$subst) (see Lemma 7.3.20) and the derivability of (incl) (see Lemma 7.3.22).

Proposition 7.3.19 Derivability of the alternative inference rules

The inference rules (x'), (app') *and* (ξ') *are derivable in* λmon-*equational logic.*

Proof The derivability of (app') and (ξ') is obvious. In fact, (app') is an instance of (app), while (ξ') and (ξ) are the same rule. We show that (x') is derivable in λmon-equational logic.

If $D_2 \equiv \emptyset$ the conclusion is an instance of (incl), otherwise

- let D be the dλ-term $\{x_i y_i | 1 \leq i \leq |D_2|\}$

- from $x\restriction D \lesssim x\restriction D$ and $D_1 \sqsubseteq D_2$, by repeatedly applying ($*$subst) $D_1 \lesssim D_2 \cup (\cup\{\mathrm{d}(t_1) \cup \mathrm{d}(t_2) | (t_1 t_2) \in D_2\})$

- but the r.h.s. is D_2, because $t_i\restriction D_2$ is a pλ-term and (by Lemma 7.1.23) $\mathrm{d}(t_i) \subseteq D_2$

∎

Lemma 7.3.20 Some admissible rules

The following inference rules are admissible in the set generated by the alternative inference rules:

$$\text{0-}*\text{subst} \quad \frac{M\restriction D \lesssim N\restriction E}{D \lesssim E}$$

$$*\text{incl} \quad \frac{M\restriction D \lesssim N\restriction E}{M\restriction D \cup F \lesssim N\restriction E}$$

$$\text{w-}*\text{subst} \quad \frac{M\restriction D \lesssim N\restriction E \qquad F \lesssim G}{M\restriction D \cup F \lesssim N\restriction E \cup G}$$

Proof The three statements are proved by induction on the proof of $M\restriction D \lesssim N\restriction E$ (using the alternative inference rules).

Admissibility of (0-$*$subst):

- (x') $\Longrightarrow$ immediate.

  To derive $D \lesssim E$, i.e. $y\restriction D \lesssim y\restriction E$, it is enough to apply the rule (y') (instead of (x')) to the premisses of the last rule in the proof of $M\restriction D \lesssim N\restriction E$.

- (app') $\Longrightarrow$

  By IH for any of the two premisses of the last rule in the proof of $M\restriction D \lesssim N\restriction E$.

- (ξ') $\Longrightarrow$ trivial

  In fact, $D \lesssim E$ is a premiss of the last rule in the proof of $M\restriction D \lesssim N\restriction E$.

Admissibility of ($*$incl):

- (x') $\Longrightarrow$ by IH and (x').

- by IH applied to the premisses $D \sqsubseteq E$ of the last inference rule (i.e. (x'))

  $D \cup F \sqsubseteq E$

- by the inference rule (x')

  $x{\restriction}D \cup F {\lesssim} x{\restriction}E$

- (app') $\Longrightarrow$ by IH and (app').

  Similar to the case (x').

- (ξ') $\Longrightarrow$ by IH and (ξ').

  Similar to the case (x'), but it may be necessary to rename the bound variable.

Admissibility of (w-$*$subst):

- (x') $\Longrightarrow$ by ($*$incl) and (x').

  - the premisses of the last inference rule in the proof of $M{\restriction}D{\lesssim}N{\restriction}E$ and $F{\lesssim}G$ must be

    $D \sqsubseteq E$ and $F \sqsubseteq G$

  - by the admissible inference rule ($*$incl)

    $D \cup F \sqsubseteq E \cup G$

  - by the inference rule (x')

    $x{\restriction}D \cup F{\lesssim}x{\restriction}E \cup G$

- (app') $\Longrightarrow$ by IH and (app').

  Let $t_1{\restriction}D{\lesssim}t_2{\restriction}E$ and $t_3{\restriction}D{\lesssim}t_4{\restriction}E$ be the premisses of the last rule (i.e. (app')) in the proof of $M{\restriction}D{\lesssim}N{\restriction}E$.

  - by IH

    $t_1{\restriction}D \cup F{\lesssim}t_2{\restriction}E \cup G$ and $t_3{\restriction}D \cup F{\lesssim}t_4{\restriction}E \cup G$

  - by the inference rule (app')

    $M{\restriction}D \cup F{\lesssim}N{\restriction}E \cup G$

- (ξ') $\Longrightarrow$ by IH and (ξ').

  Similar to the case (app'), but it may be necessary to rename the bound variable.

∎

Proposition 7.3.21 (trans) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$\text{trans}\ \frac{P{\restriction}F{\lesssim}M{\restriction}D \qquad M{\restriction}D{\lesssim}N{\restriction}E}{P{\restriction}F{\lesssim}N{\restriction}E}$$

Proof By induction on the proof of $M{\restriction}D{\lesssim}N{\restriction}E$ (using the alternative inference rules):

- (x') $\Longrightarrow$ by IH , (w-$*$subst) and (x').

  Since $M \equiv x$, the last inference rule in the proof of $P{\restriction}F{\lesssim}M{\restriction}D$ must also be (x').

  Let $F \sqsubseteq D \sqsubseteq E$ be the premisses of the last rules (i.e. (x')) in the proof of $P{\restriction}F{\lesssim}M{\restriction}D{\lesssim}N{\restriction}E$. First we prove $F \sqsubseteq E$, i.e. for all $(t_1t_2) \in E$ we find $(t_1''t_2'') \in F$ s.t. $t_i''{\restriction}F{\lesssim}t_i{\restriction}\mathrm{d}(t_i)$ is derivable:

  - since $D \sqsubseteq E$ are the premisses of the last inference rule (i.e. (x')) in the proof of $M{\restriction}D{\lesssim}N{\restriction}E$, then
    there exists $(t_1't_2') \in D$ s.t. $t_i'{\restriction}D{\lesssim}t_i{\restriction}\mathrm{d}(t_i)$ is a premiss of (x')
  - since $F \sqsubseteq D$ are the premisses of the last inference rule (i.e. (x')) in the proof of $P{\restriction}F{\lesssim}M{\restriction}D$, then
    there exists $(t_1''t_2'') \in F$ s.t. $t_i''{\restriction}F{\lesssim}t_i'{\restriction}\mathrm{d}(t_i')$ is derivable and
    $F{\lesssim}D$ is derivable (by (y'))
  - since $\mathrm{d}(t_i') \subseteq D$ (see Proposition 7.1.23), by the admissible inference rule (w-$*$subst)
    $t_i''{\restriction}F{\lesssim}t_i'{\restriction}D$
  - by IH for $t_i'{\restriction}D{\lesssim}t_i{\restriction}\mathrm{d}(t_i)$
    $t_i''{\restriction}F{\lesssim}t_i{\restriction}\mathrm{d}(t_i)$

Then, we derive $P{\restriction}F \equiv x{\restriction}F \lesssim x{\restriction}E \equiv N{\restriction}E$ from $F \sqsubseteq E$ (by (x')).

- (app') $\Longrightarrow$ by IH and (app').

  Since M is an application, the last inference rule in the proof of $P{\restriction}F \lesssim M{\restriction}D$ must also be (app').

  Let $P_1{\restriction}F \lesssim M_1{\restriction}D \lesssim N_1{\restriction}E$ and $P_2{\restriction}F \lesssim M_2{\restriction}D \lesssim N_2{\restriction}E$ be the premisses of the last rules (i.e. (app')) in the proof of $P{\restriction}F \lesssim M{\restriction}D \lesssim N{\restriction}E$.

  - by IH for $M_1{\restriction}D \lesssim N_1{\restriction}E$ and $M_2{\restriction}D \lesssim N_2{\restriction}E$

    $P_1{\restriction}F \lesssim N_1{\restriction}E$ and $P_2{\restriction}F \lesssim N_2{\restriction}E$
  - by (app')

    $P{\restriction}F \equiv P_1P_2{\restriction}F \lesssim N_1N_2{\restriction}E \equiv N{\restriction}E$

- (ξ') $\Longrightarrow$ by IH and (ξ').

  Since M is a λ-abstraction, the last inference rule in the proof of $P{\restriction}F \lesssim M{\restriction}D$ must also be (app').

  Let $P'{\restriction}F' \cup F \lesssim M'{\restriction}D' \cup D \lesssim N'{\restriction}E' \cup E$ and $F \lesssim D \lesssim E$ be the premisses of the last rules (i.e. (ξ')) in the proof of $P{\restriction}F \lesssim M{\restriction}D \lesssim N{\restriction}E$.

  - by IH for $M'{\restriction}D' \cup D \lesssim N'{\restriction}E' \cup E$ and $D \lesssim E$

    $P'{\restriction}F' \cup F \lesssim N'{\restriction}E' \cup E$ and $F \lesssim E$
  - $x \notin \mathrm{FV}(F \cup E)$ (by the side conditions for (ξ')), therefore by (ξ')

    $P{\restriction}F \equiv (\lambda x.P'{\restriction}F'){\restriction}F \lesssim (\lambda x.N'{\restriction}E'){\restriction}E \equiv N{\restriction}E$

∎

Lemma 7.3.22 (incl) is derivable

For any pλ-term $M{\restriction}D$ and dλ-term E the pλ-inequation $M{\restriction}D \cup E \lesssim M{\restriction}D$ is derivable from the alternative inference rules (see Definition 7.3.16).

Proof By induction on the proof of $M{\restriction}D \in$ pTerm (using the alternative rules of Definition 7.1.19):

- (x') $\Longrightarrow$ by IH and (x').

  Let $t_i{\restriction}\mathrm{d}(t_i) \in$ pTerm (for all $t_1t_2 \in D$ and $i = 1, 2$) be the premisses of the last inference rule (i.e. (x')) in the proof of $M{\restriction}D \in$ pTerm.

- since $\mathrm{d}(t_i) \subseteq D \cup E$, by IH for the premisses

  $t_i \restriction D \cup E \lesssim t_i \restriction \mathrm{d}(t_i)$ (for all $t_1 t_2 \in D$ and $i = 1, 2$)
- by the inference rule (x')

  $M \restriction D \cup E \equiv x \restriction D \cup E \lesssim x \restriction D \equiv M \restriction D$

- (app') $\Longrightarrow$ by IH and (app').

  Let $M_i \restriction D \in$ pTerm (for $i = 1, 2$) be the premisses of the last inference rule (i.e. (app')) in the proof of $M \restriction D \in$ pTerm.

  - by IH for the premisses

    $M_i \restriction D \cup E \lesssim M_i \restriction D$ (for $i = 1, 2$)
  - by the inference rule (app')

    $M \restriction D \cup E \equiv M_1 M_2 \restriction D \cup E \lesssim M_1 M_2 \restriction D \equiv M \restriction D$

- (λ') $\Longrightarrow$ by IH and (ξ').

  Similar to the case (app'), but it may be necessary to rename the bound variable.

∎

Proposition 7.3.23 (x.app) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$x.\mathrm{app} \ \frac{M_1 \restriction D_1 \lesssim N_1 \restriction E_1 \qquad M_2 \restriction D_2 \lesssim N_2 \restriction E_2}{x \restriction \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim x \restriction \{N_1 N_2\} \cup E_1 \cup E_2}$$

Proof Suppose that both $M_1 \restriction D_1 \lesssim N_1 \restriction E_1$ and $M_2 \restriction D_2 \lesssim N_2 \restriction E_2$ are derivable by the alternative inference rules. If we show that for all $(t_1 t_2)$ in $\{N_1 N_2\} \cup E_1 \cup E_2$ there exists $(t'_1 t'_2)$ in $\{M_1 M_2\} \cup D_1 \cup D_2$ s.t.

$$t'_i \restriction \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim t_i \restriction \mathrm{d}(t_i) \text{ (for } i = 1, 2) \text{ are derivable}$$

then, $x \restriction \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim x \restriction \{N_1 N_2\} \cup E_1 \cup E_2$ is derivable by the inference rule (x'). There are three cases to consider, depending on which subset of $\{N_1 N_2\} \cup E_1 \cup E_2$ the term $(t_1 t_2)$ is in:

- $t_1 t_2 \equiv N_1 N_2 \Longrightarrow$

  We take $t'_1 t'_2$ to be $M_1 M_2$ and derive $M_i \upharpoonright \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim N_i \upharpoonright \mathrm{d}(N_i)$:

  - from the assumption $M_i \upharpoonright D_i \lesssim N_i \upharpoonright E_i$, by the admissible inference rule ($*$incl)

    $M_i \upharpoonright \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim N_i \upharpoonright E_i$
  - since $\mathrm{d}(N_i) \subseteq E_i$, by (incl)

    $N_i \upharpoonright E_i \lesssim N_i \upharpoonright \mathrm{d}(N_i)$
  - by the admissible inference rule (trans)

    $M_i \upharpoonright \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim N_i \upharpoonright \mathrm{d}(N_i)$

- $t_1 t_2 \in E_1 \Longrightarrow$

  - from the assumption $M_1 \upharpoonright D_1 \lesssim N_1 \upharpoonright E_1$, by the admissible inference rule (0-$*$subst), $D_1 \sqsubseteq E_1$. Therefore

    there exists $(t'_1 t'_2) \in D_1$ s.t. $t'_i \upharpoonright D_1 \lesssim t_i \upharpoonright \mathrm{d}(t_i)$
  - by the admissible inference rule ($*$incl)

    $t'_i \upharpoonright \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim t_i \upharpoonright \mathrm{d}(t_i)$

- $t_1 t_2 \in E_2 \Longrightarrow$ similar to the case $t_1 t_2 \in E_1$.

∎

Proposition 7.3.24 (app) is admissible

The following inference rule is admissible in the set generated by the alternative inference rules:

$$\text{app} \quad \frac{M_1 \upharpoonright D_1 \lesssim N_1 \upharpoonright E_1 \qquad M_2 \upharpoonright D_2 \lesssim N_2 \upharpoonright E_2}{M_1 M_2 \upharpoonright \{M_1 M_2\} \cup D_1 \cup D_2 \lesssim N_1 N_2 \upharpoonright \{N_1 N_2\} \cup E_1 \cup E_2}$$

Proof Suppose that both $M_1 \upharpoonright D_1 \lesssim N_1 \upharpoonright E_1$ and $M_2 \upharpoonright D_2 \lesssim N_2 \upharpoonright E_2$ are derivable by the alternative inference rules.

- from the assumptions, by the admissible inference rule (y.app)

  $\{M_1 M_2\} \cup D_1 \cup D_2 \lesssim \{N_1 N_2\} \cup E_1 \cup E_2$

- from the assumption $M_i\restriction D_i \lesssim N_i\restriction E_i$ (for $i = 1, 2$), by the admissible inference rule (w-$*$subst)

  $M_i\restriction\{M_1M_2\} \cup D_1 \cup D_2 \lesssim N_i\restriction\{N_1N_2\} \cup E_1 \cup E_2$ (for $i = 1, 2$)

- by the inference rule (app')

  $M_1M_2\restriction\{M_1M_2\} \cup D_1 \cup D_2 \lesssim N_1N_2\restriction\{N_1N_2\} \cup E_1 \cup E_2$

∎

7.4 λ_{p}-algebras

Extensional combinatory logic is not *axiomatizable* by equational axioms only, otherwise the class of extensional combinatory algebras (or equivalently λ-*models*) would be closed under substructure. However, one can consider the *variety* of λ-algebras, i.e. the (total) algebras that satisfy the equations $E(\mathrm{CL} + \mathrm{ext.} =)$ provable in extensional combinatory logic. There is an alternative characterization of λ-algebras, namely they are the substructures of extensional combinatory algebras (see [Bar82]). Therefore, an equation $t_1 = t_2$ is valid in all extensional combinatory algebras satisfying a set T of **closed** equations iff $t_1 = t_2$ is valid in all λ-algebras satisfying T. In other words, we can use equational logic with axioms $E(\mathrm{CL} + \mathrm{ext.} =)$, instead of first order logic with the axioms $\mathrm{CL} + \mathrm{ext.} =$, for deriving equations from a set T of closed equations.

Remark 7.4.1 For λ-algebras there is also a **finite** axiomatization, namely the axioms CL of combinatory logic plus *Curry's axioms* $A_{\beta\eta}$ (see 7.3.15 of [Bar81]). We have not found a finite axiomatization for λ_{p}-algebras.

Definition 7.4.2 (Monotonic) λ_{p}-algebra

- *a* λ_{p}**-algebra** *is a classical model of* $S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$*, i.e. the set of* S*-equations derivable in intuitionistic extensional partial combinatory logic (see Definition 4.2.9)*

- *a* **monotonic** λ_{p}**-algebra** *is a classical model of* $Sin(J\mathrm{CL_p}{+}\mathrm{mon_p}{+}\mathrm{ext.}{\lesssim})$*, the set of* Sin*-equations derivable in intuitionistic extensional monotonic partial combinatory logic (see Remark 4.3.2)*

Remark 7.4.3 There is a more *restrictive* definition of λ_{p}-algebra, namely a model of $S(K\mathrm{CL_p} + \mathrm{ext.} \simeq)$. However, we do not expect that such algebras have an alternative **model-theoretic** characterization in terms of **classical** extensional partial combinatory algebras. Moreover, we conjecture that Theorem 7.4.9 below does not hold if we **replace** $J\mathrm{CL_p} + \mathrm{ext.} \simeq$ with $K\mathrm{CL_p} + \mathrm{ext.} \simeq$.

In this section we characterize λ_{p}-algebras in terms of Kripke extensional partial combinatory algebras. This characterization (see Corollary 7.4.11) is the *model-theoretic equivalent* of the connection (see Theorem 7.4.9) between intuitionistic extensional partial combinatory algebra $J\mathrm{CL_p}+\mathrm{ext.} \simeq$ and the *equational* formal system for λ_{p}-algebras $K\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$.

We consider also *monotonic* λ_{p}*-algebras*. There are only minor differences between the monotonic and non-monotonic cases, therefore Section 7.4.2 states only the main results and points out the relevant differences in the proofs.

7.4.1 The non-monotonic case

In this section we put to good use the equational presentation to prove the analogue of *Jacopini's Lemma* for $\lambda\beta$p-equational logic (see [Jac75, Sta85a]). This lemma is more informative than Theorem 7.4.9, although we do not consider other applications of it. The main applications of Jacopini's lemma are to solve consistency questions (see [Jac75]). For instance, R. Statman has applied it to solve a consistency problem for the second-order lambda calculus (see [Sta86]).

Lemma 7.4.4 Jacopini-like Lemma

If T is a set of closed pλ-equations of the form $u{\restriction}\emptyset \simeq v{\restriction}\emptyset$, then for all $t_1{\restriction}D_1 \simeq t_2{\restriction}D_2$ derivable in $\lambda\beta$p-equational logic from T and all pairs of distinct *variables $y, z \notin \mathrm{FV}(t_1{\restriction}D_1, t_2{\restriction}D_2)$ there exist $n \in \omega$, pλ-terms $M_i{\restriction}E_i$ and* axioms *$u_i{\restriction}\emptyset \simeq v_i{\restriction}\emptyset \in T$ (for $1 \leq i \leq n$) s.t. for all $0 \leq i \leq n$*

$$(M_i{\restriction}E_i)*[y:=v_i\lfloor z:=u_i] \simeq (M_{i+1}{\restriction}E_{i+1})*[y:=u_{i+1}\lfloor z:=v_{i+1}] \in pS(\lambda\beta\mathrm{pEQL})$$

where $M_0{\restriction}E_0 \stackrel{\Delta}{\equiv} t_1{\restriction}D_1$, $M_{n+1}{\restriction}E_{n+1} \stackrel{\Delta}{\equiv} t_2{\restriction}D_2$ and $pS(\lambda\beta\mathrm{pEQL})$ is the set of pλ-equations derivable in $\lambda\beta$p-equational logic

Proof By induction on the derivation of $t_1{\restriction}D_1 \simeq t_2{\restriction}D_2$ from T in $\lambda\beta$p-equational logic. In all cases the proof relies on the admissibility of the inference rules for $\lambda\beta$pEQL in $pS(\lambda\beta\mathrm{pEQL})$.

Notation 7.4.5 For convenience in the proof we write X for the finite set $\mathrm{FV}(t_1{\upharpoonright}D_1, t_2{\upharpoonright}D_2) \cup \{y, z\}$ and σ_i and σ'_i $(1 \leq i \leq n)$ for the substitutions $(y{:=}\ u_i \lfloor z{:=}\ v_i)$ and $(y{:=}\ v_i \lfloor z{:=}\ u_i)$.

- $t_1{\upharpoonright}D_1 \simeq t_2{\upharpoonright}D_2 \in T \Longrightarrow$ immediate. In fact
  - $n \triangleq 1$
  - $M_1{\upharpoonright}E_1 \triangleq y{\upharpoonright}\emptyset$
  - $u_1{\upharpoonright}\emptyset \simeq v_1{\upharpoonright}\emptyset \equiv t_1{\upharpoonright}D_1 \simeq t_2{\upharpoonright}D_2 \in T$

  satisfy the required properties (by (refl)).

- (refl) $\Longrightarrow$ immediate. In fact
  - $n \triangleq 0$

  satisfy the required properties (by (refl)).

- $(\beta) \Longrightarrow$ immediate. In fact
  - $n \triangleq 0$

  satisfy the required properties (by (β)).

- (symm) $\Longrightarrow$

  By IH there exist
  - $n \in \omega$, pλ-terms $M'_i{\upharpoonright}E'_i$ and axioms $u'_i{\upharpoonright}\emptyset \simeq v'_i{\upharpoonright}\emptyset$ (for $1 \leq i \leq n$) satisfying the required properties for the pλ-equation $t_2{\upharpoonright}D_2 \simeq t_1{\upharpoonright}D_1$ and the variables y and z

  It is immediate to check, using the IH , that the "reverse sequence of pλ-terms and axioms"
  - n
  - $M_i{\upharpoonright}E_i \triangleq M'_{n-i+1}{\upharpoonright}E'_{n-i+1}$ (for $1 \leq i \leq n$)
  - $u_i{\upharpoonright}\emptyset \simeq v_i{\upharpoonright}\emptyset \equiv u'_{n-i+1}{\upharpoonright}\emptyset \simeq v'_{n-i+1}{\upharpoonright}\emptyset \in T$ (for $1 \leq i \leq n$)

  satisfy the required properties for the pλ-equation $t_1{\upharpoonright}D_1 \simeq t_2{\upharpoonright}D_2$ and the variables y and z.

- (trans) $\Longrightarrow$

  Let $t_1\lceil D_1 \simeq t\lceil D$ and $t\lceil D \simeq t_2\lceil D_2$ be the premisses of the last rule (i.e. (trans)) in the proof of $t_1\lceil D_1 \simeq t_2\lceil D_2$.

  By IH there exist

  - $n' \in \omega$, pλ-terms $M'_i\lceil E'_i$ and axioms $u'_i\lceil\emptyset \simeq v'_i\lceil\emptyset$ (for $1 \leq i \leq n'$) satisfying the required properties for the pλ-equation $t_1\lceil D_1 \simeq t\lceil D$ and the variables y and z
  - $n'' \in \omega$, pλ-terms $M''_i\lceil E''_i$ and axioms $u''_i\lceil\emptyset \simeq v''_i\lceil\emptyset$ (for $1 \leq i \leq n''$) satisfying the required properties for the pλ-equation $t\lceil D \simeq t_2\lceil D_2$ and the variables y and z

  It is immediate to check, using the IH (and (trans)), that the "concatenation of the two sequences of pλ-terms and axioms"

  - $n \triangleq n' + n''$
  - $M_i\lceil E_i \triangleq M'_i\lceil E'_i$ (for $1 \leq i \leq n'$) and
    $M_{n'+i}\lceil E_{n'+i} \triangleq M''_i\lceil E''_i$ (for $1 \leq i \leq n''$)
  - $u_i\lceil\emptyset \simeq v_i\lceil\emptyset \equiv u'_i\lceil\emptyset \simeq v'_i\lceil\emptyset \in T$ (for $1 \leq i \leq n'$) and
    $u_{n'+i}\lceil\emptyset \simeq v_{n'+i}\lceil\emptyset \equiv u''_i\lceil\emptyset \simeq v''_i\lceil\emptyset \in T$ (for $1 \leq i \leq n''$)

  satisfy the required properties for the pλ-equation $t_1\lceil D_1 \simeq t_2\lceil D_2$ and the variables y and z. The admissible inference rule (trans) is used only for deriving $(M_{n'}\lceil E_{n'})*[\sigma'_{n'}] \simeq (M_{n'+1}\lceil E_{n'+1})*[\sigma_{n'+1}]$.

- $(\xi) \Longrightarrow$

  Let $t'_1\lceil D'_1 \cup D_1 \simeq t'_2\lceil D'_2 \cup D_2$ and $D_1 \simeq D_2$ be the premisses of the last rule (i.e. (ξ)) in the proof of $t_1\lceil D_1 \simeq t_2\lceil D_2$.

  We assume that the bound variable x is not in X (otherwise we can rename it). By IH there exist

  - $n' \in \omega$, pλ-terms $M'_i\lceil E'_i$ and axioms $u'_i\lceil\emptyset \simeq v'_i\lceil\emptyset$ (for $1 \leq i \leq n'$) satisfying the required properties for the pλ-equation $t'_1\lceil D'_1 \cup D_1 \simeq t'_2\lceil D'_2 \cup D_2$ and the variables y and z

- $n'' \in \omega$, pλ-terms $M''_i \upharpoonright E''_i$ and $u''_i \upharpoonright \emptyset \simeq v''_i \upharpoonright \emptyset$ (for $1 \leq i \leq n''$) satisfying the required properties for the pλ-equation $D_1 \simeq D_2$ and the variables y and z

we claim that

- $n \triangleq n' + n''$
- $M_i \upharpoonright E_i \triangleq (\lambda x. M'_i \upharpoonright E'_i) \upharpoonright D_1$ (for $1 \leq i \leq n'$) and
  $M_{n'+i} \upharpoonright E_{n'+i} \triangleq (\lambda x. t'_2 \upharpoonright D'_2 \cup D_2) \upharpoonright E''_i$ (for $1 \leq i \leq n''$)
- $u_i \upharpoonright \emptyset \simeq v_i \upharpoonright \emptyset \equiv u'_i \upharpoonright \emptyset \simeq v'_i \upharpoonright \emptyset \in T$ (for $1 \leq i \leq n'$) and
  $u_{n'+i} \upharpoonright \emptyset \simeq v_{n'+i} \upharpoonright \emptyset \equiv u''_i \upharpoonright \emptyset \simeq v''_i \upharpoonright \emptyset \in T$ (for $1 \leq i \leq n''$)

satisfy the required properties for the pλ-equation $t_1 \upharpoonright D_1 \simeq t_2 \upharpoonright D_2$ and the variables y and z.

We have to distinguish five cases for $0 \leq i \leq n$:

$i = 0$ $t_1 \upharpoonright D_1 \simeq (M_1 \upharpoonright E_1)*[\sigma_1] \in pS(\lambda\beta\text{pEQL})$, in fact
$t'_1 \upharpoonright D'_1 \cup D_1 \simeq (M'_1 \upharpoonright E'_1)*[\sigma_1]$ (by IH for the first premiss)
$D_1 \simeq D_1$ (by (refl))
$t'_1 \upharpoonright D'_1 \cup D_1 \simeq (M'_1 \upharpoonright E'_1 \cup D_1)*[\sigma_1]$ (by (w-$*$subst) and $x \notin X$)
$(\lambda x. t'_1 \upharpoonright D'_1) \upharpoonright D_1 \simeq (\lambda x.(M'_1 \upharpoonright E'_1)*[\sigma_1]) \upharpoonright D_1$ (by (ξ))
but (by definition of $M_1 \upharpoonright E_1$ and $x \notin X$)
$(\lambda x.(M'_1 \upharpoonright E'_1)*[\sigma_1]) \upharpoonright D_1 \equiv (M_1 \upharpoonright E_1)*[\sigma_1]$

$0 < i < n'$ $(M_i \upharpoonright E_i)*[\sigma'_i] \simeq (M_{i+1} \upharpoonright E_{i+1})*[\sigma_{i+1}] \in pS(\lambda\beta\text{pEQL})$ is proved as in the case $i = 0$

$i = n'$ $(M_{n'} \upharpoonright E_{n'})*[\sigma'_{n'}] \simeq (M_{n'+1} \upharpoonright E_{n'+1})*[\sigma_{n'+1}] \in pS(\lambda\beta\text{pEQL})$ follows by (trans) from
$(M_{n'} \upharpoonright E_{n'})*[\sigma'_{n'}] \simeq (\lambda x. t'_2 \upharpoonright D'_2 \cup D_2) \upharpoonright D_1$ and
$(\lambda x. t'_2 \upharpoonright D'_2 \cup D_2) \upharpoonright D_1 \simeq (M_{n'+1} \upharpoonright E_{n'+1})*[\sigma_{n'+1}]$.
The first pλ-equation is proved as in the case $i = 0$, so we prove only the second.
$t'_2 \upharpoonright D'_2 \cup D_2 \simeq t'_2 \upharpoonright D'_2 \cup D_2$ (by (refl))
$D_1 \simeq (E''_1)*[\sigma_{n'+1}]$ (by IH for the second premiss)

$t_2'\lceil D_2' \cup D_2 \cup D_1 \simeq (t_2'\lceil D_2' \cup D_2 \cup E_1'')*[\sigma_{n'+1}]$ (by (w-$*$subst) and $x \notin X$)

$(\lambda x.t_2'\lceil D_2' \cup D_2)\lceil D_1 \simeq (\lambda x.t_2'\lceil D_2' \cup D_2)\lceil(E_1'')*[\sigma_{n'+1}]$ (by (ξ) and $x \notin X$)

but (by definition of $M_{n'+1}\lceil E_{n'+1}$ and $x \notin X$)

$(\lambda x.t_2'\lceil D_2' \cup D_2)\lceil(E_1'')*[\sigma_{n'+1}] \equiv (M_{n'+1}\lceil E_{n'+1})*[\sigma_{n'+1}]$

$n' < i < n$ $(M_i\lceil E_i)*[\sigma_i'] \simeq (M_{i+1}\lceil E_{i+1})*[\sigma_{i+1}] \in pS(\lambda\beta\text{pEQL})$ is proved as the second pλ-equation of the case $i = n'$

$i = n$ $(M_n\lceil E_n)*[\sigma_n'] \simeq t_2\lceil D_2 \in pS(\lambda\beta\text{pEQL})$ is proved as the case $n' < i < n$

- ($*$subst) $\Longrightarrow$ similar to the case (ξ).

  Let $t_1'\lceil D_1' \simeq t_2'\lceil D_2'$ and $t_1''\lceil D_1'' \simeq t_2''\lceil D_2''$ be the premisses of the last rule (i.e. ($*$subst)) in the proof of $t_1\lceil D_1 \simeq t_2\lceil D_2$.

  We assume that the substituted variable x is not in X (otherwise we can rename it). By IH there exist

  - $n' \in \omega$, pλ-terms $M_i'\lceil E_i'$ and axioms $u_i'\lceil\emptyset \simeq v_i'\lceil\emptyset$ (for $1 \leq i \leq n'$) satisfying the required properties for the pλ-equation $t_1'\lceil D_1' \simeq t_2'\lceil D_2'$ and the variables y and z
  - $n'' \in \omega$, pλ-terms $M_i''\lceil E_i''$ and axioms $u_i''\lceil\emptyset \simeq v_i''\lceil\emptyset$ (for $1 \leq i \leq n''$) satisfying the required properties for the pλ-equation $t_1''\lceil D_1'' \simeq t_2''\lceil D_2''$ and the variables y and z

  we claim that

  - $n \triangleq n' + n''$
  - $M_i\lceil E_i \triangleq (M_i'\lceil E_i')*[x := t_1''\lceil D_1'']$ (for $1 \leq i \leq n'$) and $M_{n'+i}\lceil E_{n'+i} \triangleq (t_2'\lceil D_2')*[x := M_i''\lceil E_i'']$ (for $1 \leq i \leq n''$)
  - $u_i\lceil\emptyset \simeq v_i\lceil\emptyset \equiv u_i'\lceil\emptyset \simeq v_i'\lceil\emptyset \in T$ (for $1 \leq i \leq n'$) and $u_{n'+i}\lceil\emptyset \simeq v_{n'+i}\lceil\emptyset \equiv u_i''\lceil\emptyset \simeq v_i''\lceil\emptyset \in T$ (for $1 \leq i \leq n''$)

  satisfy the required properties for the pλ-equation $t_1\lceil D_1 \simeq t_2\lceil D_2$ and the variables y and z.

∎

Remark 7.4.6 In Lemma 7.4.4 we can add the requirement that $\mathrm{FV}(M_i{\restriction}E_i) \subseteq X = \mathrm{FV}(t_1{\restriction}D_1, t_2{\restriction}D_2) \cup \{y, z\}$ (for $1 \leq i \leq n$). In fact, whenever $x \notin X$, it can be replace with the closed term I in every $M_i{\restriction}E_i$ without invalidating the required properties.

Lemma 7.4.4 holds also for λpEQL ($\lambda\eta$pEQL and $\lambda\beta\eta$pEQL). Since in λpEQL the axiom (β) is not equivalent to any set of closed pλ-equations, the lemma for $\lambda\beta$pEQL is not an immediate *corollary* of the lemma for λpEQL.

We do not prove Theorem 7.4.9 directly, because the relevant steps in the proof would be mixed with *irrelevant* translations from CL-terms to pλ-terms and conversely. So we prove the theorem using $\lambda{\restriction}$-terms (see Lemma 7.4.7) and then show that we can safely replace $\lambda{\restriction}$-terms with CL-terms (Lemma 7.4.8).

Lemma 7.4.7 Equational axiomatization of $J\lambda_{\mathrm{p}}\beta\eta + {\restriction}$

If T_0 is a set of closed E-equations, then every S-equation $t_1 \simeq t_2$ derivable in $J\lambda_{\mathrm{p}}\beta\eta + {\restriction}$ from T_0 is derivable in $J\lambda_{\mathrm{p}} + S(J\lambda_{\mathrm{p}}\beta\eta + {\restriction})$ from T_0 **without** *using* $(\xi.\simeq)$.

Proof We take a set T of closed S-equations s.t.

1. T_0 is derivable from T in $J\lambda_{\mathrm{p}}\beta\eta + {\restriction}$
2. $\mathrm{s}(T)$ satisfies the conditions of Lemma 7.4.4
3. T is derivable from T_0 in $J\lambda_{\mathrm{p}} + S(J\lambda_{\mathrm{p}}\beta\eta + {\restriction})$ without using $(\xi.\simeq)$

To show that T (with the above properties) exists, we assume w.l.o.g. that T_0 is a singleton, say $\{u = v\}$. In the general case T can be defined as the union of the Ts corresponding to $\{u = v\}$ for $(u = v) \in T_0$. Let $T \equiv \{(\lambda x.u) \simeq (\lambda x.v), (\lambda x.x{\restriction}u) \simeq (\lambda x.x)\}$. The first and second properties are easy to verify. Therefore we prove only the third property.

$(\lambda x.u) \simeq (\lambda x.v)$ is derivable from $u = v$, in fact:

- since $u = v$, by strictness of $=$

  $u\downarrow$ and $v\downarrow$

- if M is a closed term and $M\downarrow$, then

  by $((\lambda x.y) \simeq (\lambda y, x.y)y) \in S(J\lambda_{\mathrm{p}}\beta\eta + {\restriction})$ and (subst)

$(\lambda x.M) \simeq (\lambda y, x.y)M$, in particular

$(\lambda x.u) \simeq (\lambda y, x.y)u$ and $(\lambda x.v) \simeq (\lambda y, x.y)v$

- by congruence of app and $u = v$

  $(\lambda y, x.y)u \simeq (\lambda y, x.y)v$

- by transitivity

  $(\lambda x.u) \simeq (\lambda x.v)$

$(\lambda x.x{\upharpoonright}u) \simeq (\lambda x.x)$ is derivable from $u = v$, in fact:

- since $u = v$, by strictness of $=$

  $u \downarrow$

- if M is a closed term and $M \downarrow$, then

  by $((\lambda x.x{\upharpoonright}y) \simeq (\lambda x.x)) \in S(J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright})$, and (subst)

  $(\lambda x.x{\upharpoonright}M) \simeq (\lambda x.x)$, in particular

  $(\lambda x.x{\upharpoonright}u) \simeq (\lambda x.x)$

To prove the lemma we argue as follows:

- by assumption

  $t_1 \simeq t_2$ is derivable from T_0 in $J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$

- since T_0 is derivable from T in $J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$, then

  $t_1 \simeq t_2$ is derivable from T in $J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$

- by Proposition 7.2.13

  $\mathrm{s}(t_1) \simeq \mathrm{s}(t_2)$ is derivable from $\mathrm{s}(T)$ in $J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright}$

- by Lemma 7.2.18

  $\mathrm{s}(t_1) \simeq \mathrm{s}(t_2)$ is derivable from $\mathrm{s}(T)$ in $\lambda\beta\eta$p-equational logic

- since $pS(\lambda\beta\mathrm{pEQL}) \subseteq S(J\lambda_{\mathrm{p}}\beta\eta + {\upharpoonright})$, by Lemma 7.4.4 for $\mathrm{s}(t_1) \simeq \mathrm{s}(t_2)$

  there exist $n \in \omega$, pλ-terms $M_i{\upharpoonright}E_i$ and *axioms* $\mathrm{s}(u_i) \simeq \mathrm{s}(v_i) \in \mathrm{s}(T)$ (for $1 \leq i \leq n$) s.t. for all $0 \leq i \leq n$

  $(M_i{\upharpoonright}E_i)\Join[y\!:= v_i\lfloor z\!:= u_i] \simeq (M_{i+1}{\upharpoonright}E_{i+1})\Join[y\!:= u_{i+1}\lfloor z\!:= v_{i+1}] \in S(J\lambda_{\mathrm{p}}\beta\eta+{\upharpoonright})$

  where $M_0{\upharpoonright}E_0 \stackrel{\Delta}{\equiv} \mathrm{s}(t_1)$ and $M_{n+1}{\upharpoonright}E_{n+1} \stackrel{\Delta}{\equiv} \mathrm{s}(t_2)$

- since $\mathrm{s}(u_i)\downarrow$ and $\mathrm{s}(v_i)\downarrow$ (for $1 \le i \le n$) and $t \simeq \mathrm{s}(t)$ are derivable in $J\lambda_\mathrm{p}\beta\eta + \upharpoonleft$, then the following S-equations are in $S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$:
  - $t_i \simeq \mathrm{s}(t_i)$
  - $(M_i\upharpoonleft E_i)*[y := \mathrm{s}(u_i)\lfloor z := \mathrm{s}(v_i)] \simeq (\lambda y, z.M_i\upharpoonleft E_i)u_i v_i$
  - $(M_i\upharpoonleft E_i)*[y := \mathrm{s}(v_i)\lfloor z := \mathrm{s}(u_i)] \simeq (\lambda y, z.M_i\upharpoonleft E_i)v_i u_i$
- therefore, by transitivity

  $t_1 \simeq t_2$ is derivable from T in $J\lambda_\mathrm{p} + S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$ without using $(\xi.\simeq)$
- since T is derivable from T_0 in $J\lambda_\mathrm{p} + S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$ without using $(\xi.\simeq)$, then

  $t_1 \simeq t_2$ is derivable from T_0 in $J\lambda_\mathrm{p} + S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$ without using $(\xi.\simeq)$

∎

Lemma 7.4.8 Equivalence of equational presentations

$J\lambda_\mathrm{p} + S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$ **without** *the axiom* $(\xi.\simeq)$ *and* $J\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.}\simeq)$ *are equivalent at the level of terms.*

Proof The equivalence is proved as the equivalence of the $\lambda_\mathrm{p}\beta\eta$-calculus and $\mathrm{CL_p} + \mathrm{ext.}\simeq$ (see Section 4.2.1). The translation from CL-terms to $\lambda\upharpoonleft$-terms is $\_^{\lambda}$ (see Definition 4.2.14), the translation $\_^{CL}$ in the opposite direction extends $(\_)^{CL}$ of Definition 4.2.14 with an extra clause for the restriction operator:

- $(t_1\upharpoonleft t_2)^{CL} \stackrel{\Delta}{\equiv} \mathrm{K}{t_1}^{CL}{t_2}^{CL}$

∎

Theorem 7.4.9 Equational axiomatization of $J\mathrm{CL_p} + \mathrm{ext.}\simeq$

If T is a set of closed E-equations, then every S-equation $t_1 \simeq t_2$ derivable from T in $J\mathrm{CL_p} + \mathrm{ext.}\simeq$ is also derivable from T in $J\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.}\simeq)$.

Proof The claim is an immediate consequence of Lemma 7.4.7. In fact:

- by Theorem 4.2.19 we can replace $J\lambda_\mathrm{p}\beta\eta + \upharpoonleft$ with $J\mathrm{CL_p} + \mathrm{ext.}\simeq$
- by Lemma 7.4.8 we can replace $J\lambda_\mathrm{p} + S(J\lambda_\mathrm{p}\beta\eta + \upharpoonleft)$ without the axiom $(\xi.\simeq)$ with $J\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.}\simeq)$.

∎

Remark 7.4.10 In the theorem above we can replace $S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$ with the set of axioms $\mathrm{CL_p} + E_0(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$, as they are provably equivalent in JLPT. Moreover, the formal system $J\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$ can be replaced with $K\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$, since intuitionistic and classical logic coincide on the S-fragment (see Theorem 3.1.5).

We conjecture that there is a direct proof of Theorem 7.4.9 based on the revisited inference rules of $J\mathrm{CL_p} + \mathrm{ext.} \simeq$ restricted to the ECE-fragment.

Corollary 7.4.11 Characterization of λ_{p}-algebras

The λ_{p}-algebras are exactly the partial algebras $\mathcal{B}(\alpha)$, where $\mathcal{B}$ is a Kripke extensional partial combinatory algebra over some K *and $\alpha \in$* K *is a stage of knowledge.*

Proof It is easy to prove that $\mathcal{B}(\alpha)$ is a λ_{p}-algebra. In fact, $\mathcal{B}$ is a Kripke model of $S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$ (which is provably equivalent to a set of coherent sequents), therefore $\mathcal{B}(\alpha)$ is a classical model of $S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$, by Proposition 3.1.4.

For the other inclusion, we show that any λ_{p}-algebra $\mathcal{A}$, considered as a $\Sigma_{\mathcal{A}}$-structure, is isomorphic to $\mathcal{A}' = \mathcal{B}^{\perp}_{(\Sigma_{\mathcal{A}},T)}(\perp)$ (see Definition 3.1.8), where $\Sigma_{\mathcal{A}}$ is Σ extended with one constant symbol c_a for each element a of $\iota^{\mathcal{A}}$ and T is any set of Harrop sequents equivalent to the set of axioms $\mathrm{CL_p} + \mathrm{ext.} \simeq$ plus the set $E_0(\mathcal{A})$ of **closed E-equations valid in $\mathcal{A}$**.

Let H be the set of closed formulae intuitionistically derivable from T, then there is only one $\Sigma_{\mathcal{A}}$-morphism from $\mathcal{A}$ to $\mathcal{A}'$, which maps $a \in \iota^{\mathcal{A}}$ to $[c_a]_H$. To prove that it is an isomorphism, we have to show that $t_1 = t_2 \in H$ iff $t_1 = t_2 \in E_0(\mathcal{A})$. The implication from right to left is immediate, since $E_0(\mathcal{A}) \subseteq T$.

For the other implication we use Theorem 7.4.9:

- by definition of H and T

  $t_1 = t_2$ is derivable from $E_0(\mathcal{A})$ in $J\mathrm{CL_p} + \mathrm{ext.} \simeq$

- by Theorem 7.4.9

  $t_1 = t_2$ is derivable from $E_0(\mathcal{A})$ in $J\mathrm{LPT} + S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$

- since $\mathcal{A}$ is a model of $S(J\mathrm{CL_p} + \mathrm{ext.} \simeq)$ and $E_0(\mathcal{A})$, then

  $\mathcal{A}$ satisfies $t_1 = t_2$

- therefore, by definition of $E_0(\mathcal{A})$

  $t_1 = t_2 \in E_0(\mathcal{A})$

∎

In JCL_p, unlike $JCL_p + \text{ext.} \simeq$, S-equations have more expressive power than E-equations, therefore we could try to improve Theorem 7.4.9 by allowing T to be a set of (closed) S-equations, instead of E-equations. However, such an improvement is impossible.

Proposition 7.4.12 *Let Σ_{CL} be the signature for partial combinatory algebras with three constants a, b and c, then there is a λ_p-algebra $\mathcal{A}$ satisfying the S-equation $ac \simeq bc$, but not $([x]ac) = ([x]bc)$.*

Proof Let $\mathcal{B}^{\perp}_{(\Sigma_{\mathrm{CL}},T)}$ be the initial Kripke extensional partial combinatory algebra over Σ_{CL} (see Definition 3.1.8), where T is any set of Harrop sequents equivalent to $\mathrm{CL}_p + \text{ext.} \simeq$. We show that the λ_p-algebra $\mathcal{A} = \mathcal{B}^{\perp}_{(\Sigma_{\mathrm{CL}},T)}(\perp)$ has the required properties:

- since $ac\downarrow$ and $bc\downarrow$ are not derivable in $JCL_p + \text{ext.} \simeq$, then they are both undefined at stage $\perp$, therefore

  $\mathcal{A}$ satisfies $ac \simeq bc$

- since $([x]ac) = ([x]bc)$ is not derivable in $JCL_p + \text{ext.} \simeq$, then

  it is not satisfied at stage $\perp$, therefore

  $\mathcal{A}$ does not satisfy $([x]ac) = ([x]bc)$

∎

7.4.2 The monotonic case

In this section we extend the result of Section 7.4.1 to $\lambda\beta$monp-equational logic. The main difference is in the statement of Jacopini's Lemma. In fact, $u{\restriction}\emptyset \lesssim v{\restriction}\emptyset \in T$ does not implies $T \vdash v{\restriction}\emptyset \lesssim u{\restriction}\emptyset$, as the relation $\lesssim$ (unlike $\simeq$) is not required to be symmetric.

Lemma 7.4.13 Monotonic Jacopini-like Lemma

*If T is a set of closed pλ-inequations of the form $u\restriction\emptyset \lesssim v\restriction\emptyset$, then for all $t_1\restriction D_1 \lesssim t_2\restriction D_2$ derivable in $\lambda\beta$*monp*-equational logic from T and all variables $y \notin$ FV$(t_1\restriction D_1, t_2\restriction D_2)$ there exist $n \in \omega$, pλ-terms $M_i\restriction E_i$ and* axioms $u_i\restriction\emptyset \lesssim v_i\restriction\emptyset \in T$ *(for $1 \leq i \leq n$) s.t. for all $0 \leq i \leq n$*

$$(M_i\restriction E_i)*[y{:}=v_i] \lesssim (M_{i+1}\restriction E_{i+1})*[y{:}=u_{i+1}] \in pSin(\lambda\beta\mathrm{monpEQL})$$

*where $M_0\restriction E_0 \stackrel{\Delta}{\equiv} t_1\restriction D_1$, $M_{n+1}\restriction E_{n+1} \stackrel{\Delta}{\equiv} t_2\restriction D_2$ and $pSin(\lambda\beta\mathrm{monpEQL})$ is the set of pλ-equations derivable in $\lambda\beta$*monp*-equational logic*

Proof By induction on the derivation of $t_1\restriction D_1 \lesssim t_2\restriction D_2$ from T in $\lambda\beta$monp-equational logic. The proof is similar to that of Lemma 7.4.4:

- $t_1\restriction D_1 \lesssim t_2\restriction D_2 \in T \Longrightarrow$ immediate. In fact
  - $n \stackrel{\Delta}{\equiv} 1$
  - $M_1\restriction E_1 \stackrel{\Delta}{\equiv} y\restriction\emptyset$
  - $u_1\restriction\emptyset \lesssim v_1\restriction\emptyset \equiv t_1\restriction D_1 \lesssim t_2\restriction D_2 \in T$

  satisfy the required properties (by (incl)).
- (incl) $\Longrightarrow$ immediate. In fact
  - $n \stackrel{\Delta}{\equiv} 0$

  satisfy the required properties (by (incl)).
- $(\beta) \Longrightarrow$ similar to the case (β) of Lemma 7.4.4
- (trans) $\Longrightarrow$ similar to the case (trans) of Lemma 7.4.4
- $(\xi) \Longrightarrow$ similar to the case (ξ) of Lemma 7.4.4
- $(*\mathrm{subst}) \Longrightarrow$ similar to the case $(*\mathrm{subst})$ of Lemma 7.4.4

∎

Theorem 7.4.14 Inequational axiomatization of $J\mathrm{CL_p} + \mathrm{mon_p} + \mathrm{ext.}\lesssim$

If T is a set of closed Ein-equations, then every Sin-equation $t_1 \lesssim t_2$ derivable from T in $J\mathrm{CL_p} + \mathrm{mon_p} + \mathrm{ext.}\lesssim$ is also derivable from T in $J\mathrm{monLPT} + Sin(J\mathrm{CL_p} + \mathrm{mon_p} + \mathrm{ext.}\lesssim)$

Chapter 8

Reduction

In this chapter we define a relation $\rightarrow_1$ on pλ-terms (see Definition 8.2.17) similar to the *one-step parallel* reduction for the $\lambda\beta$-calculus (see Chapter 3 and 11 of [Bar81]). We have not defined a *one-step* reduction, because pλ-terms are not closed w.r.t. the simple minded one-step reduction. The relation $\rightarrow_1$ satisfies the *strong diamond property* (see Corollary 8.2.18), but its symmetric and transitive closure is not the equivalence relation $\simeq$ of $\lambda_p\beta$pEQL[1], because $\rightarrow_1$ does not satisfy the inference rule (ξ).

We study only how β-reduction is related to $\lambda\beta\eta$monp-equational logic (see Section 8.3), because this *p-equational logic* is more relevant than others for applications to computer science and foremost because in the literature it is unusual to study the relations between a reduction and a preorder. However, there are similar results for the other p-equational logics. We do not consider η-reduction, and the reason for this is explained in Section 8.3; instead we treat η-conversion as part of a (decidable) preorder relation. We prove that $\lesssim$ for β-normal forms is decidable in $\lambda\beta$monp-equational logic and claim that it is decidable also in $\lambda\beta\eta$monp-equational logic (see Section 8.4). We conjecture that a similar decidability result holds for $\lambda\beta\eta$p-equational logic, but we have not been able to prove it.

The crucial step for characterizing $\lesssim$ of $\lambda\beta\eta$monpEQL is to establish that $\rightarrow_1$ has **the diamond property up to** the dη-preorder $\lesssim_{d\eta}$ (see Corollary 8.3.8). In general $\lesssim_{d\eta}$ cannot be postponed after $\rightarrow_1$ (see Example 8.3.11), therefore the diamond property up to $\lesssim_{d\eta}$ is not an easy consequence of the diamond

[1] the equivalence relation $\simeq$ of $\lambda_p\beta$pEQL is the symmetric and transitive closure of $\rightarrow_1 \cup \simeq_d$, where $\simeq_d$ is d-conversion on pλ-terms

property for $\rightarrow_1$ (as in [Sha84]). If we had been able to define pλ-terms so that d-conversion on them is just α-conversion (see Counterexample 7.1.31), then the relation between reduction for pλ-terms and equivalence $\simeq$ of $\lambda\beta$pEQL would have been much easier and *familiar*.

8.1 Basic definitions and facts

In this section we define *commutativity up to* a binary relation (see Definition 8.1.2) and related concepts and the notions of *w*- and *s-compatible* relation on pλ-terms. Moreover, we establish some of their basic properties (see Propositions 8.1.4 and 8.1.6 and Lemma 8.1.9), that are applied in subsequent sections.

Lemma 8.1.1 General facts

1. *If R is a reflexive relation on X and $m \leq n$, then $R^m \subseteq R^n$.*
2. *If R and S are symmetric relations on X, then $R \cup S$, $R \cap S$ and R^* are symmetric.*
3. *If R and S are reflexive relations on X, then $(R \cup S)^* = (RS)^*$.*

Proof Straightforward. ∎

Definition 8.1.2 Commutativity

Let R, S and T be three binary relations on a set X.

- R **commutes** *with* S **up to** T $\overset{\Delta}{\iff}$

  if $x_1 \, T \, y_1 \wedge x_1 \, R \, x_2 \wedge y_1 \, S \, y_2$, *then*

  there exist $x_3, y_3 \in X$ *s.t.* $x_2 \, S \, x_3 \wedge y_2 \, R \, y_3 \wedge x_3 \, T \, y_3$

$$\begin{tikzcd}
 & x_1 \arrow[r, "T"] \arrow[dl, "R"'] & y_1 \arrow[dr, "S"] & \\
x_2 \arrow[dr, "S"'] & & & y_2 \arrow[dl, "R"] \\
 & x_3 \arrow[r, "T"] & y_3 &
\end{tikzcd}$$

- *R commutes* **strongly** *with S up to T* $\overset{\Delta}{\Longleftrightarrow}$

  there exists $f\colon X \to X$ s.t. if $x_1 \, T \, y_1 \wedge x_1 \, R \, x_2 \wedge y_1 \, S \, y_2$, then

  $x_2 \, S \, f(x_1) \wedge y_2 \, R \, f(y_1) \wedge f(x_1) \, T \, f(y_1)$

$$\begin{tikzcd}
& x_1 \arrow[r, "T"] \arrow[dl, "R"'] & y_1 \arrow[dr, "S"] & \\
x_2 \arrow[dr, "S"'] & & & y_2 \arrow[dl, "R"] \\
& f(x_1) \arrow[r, "T"] & f(y_1) &
\end{tikzcd}$$

We introduce also the following derived definitions:

- *R commutes (strongly) with S iff R commutes (strongly) with S up to Δ_X .*

- *R has the (strong)* **diamond** *property (up to T) iff R commutes (strongly) with R (up to T).*

- *R can be* **postponed** *after S iff R^{op} commutes with S*

$$\begin{tikzcd}
x_1 \arrow[r, "R"] \arrow[d, "S"'] & y_1 \arrow[d, "S"] \\
x_2 \arrow[r, "R"] & y_2
\end{tikzcd}$$

Remark 8.1.3 Only the definition of “_ commutes (strongly) with _ up to _” is new, the others are standard (see [Bar81]). If R commutes with S up to T, then S commutes with R up to T^{op}, but in general not up to T (unless T is symmetric).

Proposition 8.1.4 Postponement of R^{op} after R

If R and T are reflexive relations on X and R has the diamond property up to T, then

$$(R \cup R^{op} \cup T)^* = (R \cup T)^*(R^{op} \cup T)^*$$

Proof First we establish the following fact:

Claim 8.1.4.1 *R commutes with $(R \cup T)^*$.*

Proof We prove, by induction on n, that if $x_1 \, R \, x_2$ and $x_1 \, (R \cup T)^n \, x_3$, then there exists x_4 s.t. $x_3 \, R \, x_4$ and $x_2 \, (R \cup T)^* \, x_4$. The basic case 0 is trivial, therefore suppose that $x_1 \, (R \cup T)^{n+1} \, x_3$. There are two cases to distinguish, depending on the first step:

- $R \implies x_1 \, R \, x_1' \, (R \cup T)^n \, x_3$.
  - by the diamond property up to T and reflexivity of T there exists x_2' s.t. $x_2 \, R \, T x_2'$ and $x_1' \, R \, x_2'$
  - by IH for $x_1' \, (R \cup T)^n \, x_3$ there exists x_4 s.t. $x_3 \, R \, x_4$ and $x_2' \, (R \cup T)^* \, x_4$
  - by composing $x_2 \, RT \, x_2'$ and $x_2' \, (R \cup T)^* \, x_4$ $x_2 \, (R \cup T)^* \, x_4$
- $T \implies x_1 \, T \, x_1' \, (R \cup T)^n \, x_3$.
  - by the diamond property up to T and reflexivity of R there exists x_2' s.t. $x_2 \, R \, T x_2'$ and $x_1' \, R \, x_2'$. Therefore, we can proceed as in the case R.

∎

We prove, by induction on n, that if $x \, (R \cup R^{op} \cup T)^n \, z$, then there exists y s.t. $x \, (R \cup T)^* \, y \, (R^{op} \cup T)^* \, z$. The basic case 0 is trivial, therefore suppose that $x \, (R \cup R^{op} \cup T)^{n+1} \, z$. There are three cases to distinguish, depending on the first step:

- $R \implies x \, R \, x' \, (R \cup R^{op} \cup T)^n \, z$.
  - by IH , there exists y s.t. $x' \, (R \cup T)^* \, y$ and $y \, (R^{op} \cup T)^* \, z$.
  - by composing $x \, R \, x'$ and $x' \, (R \cup T)^* \, y$ $x \, (R \cup T)^* \, y$

- $T \implies x\,T\,x'\,(R \cup R^{op} \cup T)^n\,z$. Similar to the case R.
- $R^{op} \implies x\,R^{op}\,x'\,(R \cup R^{op} \cup T)^n\,z$.
  - by IH , there exists y' s.t.
    $x'\,(R \cup T)^*\,y'$ and $y'\,(R^{op} \cup T)^*\,z$
  - since $x'\,R\,x$ and $x'\,(R \cup T)^*\,y'$, by the claim there exists y s.t.
    $x\,(R \cup T)^*\,y$ and $y'\,R\,y$
  - by composing $y\,R^{op}\,y'$ and $y'\,(R^{op} \cup T)^*\,z$
    $(x\,(R \cup T)^*\,)y\,(R^{op} \cup T)^*\,z$

∎

Remark 8.1.5 If T can be postponed after R, then Proposition 8.1.4 can be improved, i.e. $(R \cup R^{op} \cup T)^* = R^*T^*(R^{op})^*$. However, some of the Rs and Ts considered do not satisfy this additional requirement (see Theorem 8.3.10 and Example 8.3.11).

Proposition 8.1.6 Erasing of R

If R and T are two relations on X, $\mathrm{NF} \subseteq X$ and $f: X \to X$ are s.t.

1. $x\,R\,f(x)$

   if $x \in \mathrm{NF}$ and $x\,R\,y$, then $y = x$

2. *if $x\,T\,y$, then $f(x)\,T\,f(y)$*

   if $x \in \mathrm{NF}$ and $x\,T\,y$, then $f(y) \in \mathrm{NF}$

3. *if $x\,R\,y$, then $f(x)\,R\,f(y)$*

   if $x\,R\,y$ and $y \in \mathrm{NF}$, then $f(x) = y$

then for all x and z in NF

$$x\,(R \cup R^{op} \cup T)^*\,z \iff x\,T^*\,z$$

The first property means that NF is a set of fix-points for R and f is a reduction strategy for R, therefore we can define a partial function $nf(x) = y \overset{\Delta}{\iff} \exists n.f^n(x) = y \in \mathrm{NF}$. The second property means that if $x\,T\,y$ and $nf(x)$ exists, then $nf(x)\,T\,nf(y)$. The third property means that $nf(x) = y$ iff $x\,R^*\,y \in \mathrm{NF}$. The claim above implies that if $x\,(R \cup R^{op} \cup T)^*\,z$ and $nf(x)$ exists, then $nf(z)$ exists and $nf(x)\,T^*\,nf(z)$, in particular $x\,R^* T^* R^{op*}\,z$.

Proof We prove, by induction on n, that if $x, z \in \mathrm{NF}$ and $x\,(R \cup R^{op} \cup T)^n\,z$, then $x\,T^*\,z$. The basic case 0 is trivial, therefore suppose that $x\,(R \cup R^{op} \cup T)^{n+1}\,z$. There are three cases to distinguish, depending on the first step:

- $R \implies x\,R\,y\,(R \cup R^{op} \cup T)^n\,z$.
  - By the first property, $x = y$
  - By IH for $x = y\,(R \cup R^{op} \cup T)^n\,z$
    $x\,T^*\,z$
- $R^{op} \implies x\,R^{op}\,y\,(R \cup R^{op} \cup T)^n\,z$.
  - By the third and first property
    $x = f(y)$ and $f(z) = z$
  - by the second and third property
    $x = f(y)\,(R \cup R^{op} \cup T)^n\,f(z) = z$
  - by IH for $x\,(R \cup R^{op} \cup T)^n\,z$
    $x\,T^*\,z$
- $T \implies x\,T\,y\,(R \cup R^{op} \cup T)^n\,z$.
  - by the first property
    $x = f(x)$ and $f(z) = z$
  - by the second and third property
    $x = f(x)\,T\,f(y)\,(R \cup R^{op} \cup T)^n\,f(z) = z$
  - since $x \in \mathrm{NF}$, by the second property
    $f(y) \in \mathrm{NF}$
  - by IH for $f(y)\,(R \cup R^{op} \cup T)^n\,z$
    $(x\,T\,)f(y)\,T^*\,z$

∎

Remark 8.1.7 If NF is closed w.r.t. T (i.e. $x \in$ NF and $x\,T\,y$ imply $y \in$ NF) and R has the diamond property up to T, then $x\,(R \cup R^{op} \cup T)^*\,z \iff x\,T^*\,z$ for all x and z in NF. However, some of the NF and T considered do not satisfy these requirements (see Theorem 8.4.4).

Definition 8.1.8 Compatible relations

Let R be a binary relation on pλ-terms **modulo α-conversion***:*

- *R is* **w-compatible** *iff it is closed w.r.t. the inference rules* (refl)*,* ($*$subst) *and* (w-ξ)

$$\text{w-}\xi\ \frac{t_1\!\upharpoonright\! D_1\; R\, t_2\!\upharpoonright\! D_2 \qquad D_1'\; R\; D_2'}{(\lambda x.t_1\!\upharpoonright\! D_1)\!\upharpoonright\! D_1'\; R\, (\lambda x.t_2\!\upharpoonright\! D_2)\!\upharpoonright\! D_2'}$$

- *R is* **s-compatible** *iff it is closed w.r.t. the inference rules* (refl)*,* ($*$subst) *and* (ξ)

Lemma 8.1.9 Basic Facts

1. *If R is s-compatible, then it is also w-compatible.*
2. *If R and S are (w-compatible) s-compatible, then $R \cap S$ is (w-compatible) s-compatible.*
3. *If R and S are (w-compatible) s-compatible, then RS is (w-compatible) s-compatible and $(R \cup S)^*$ is (w-compatible) s-compatible.*
4. *If S is w-compatible and R and T are s-compatible, then RST is s-compatible and $(R \cup S \cup T)^*$ is s-compatible.*

Proof First we establish some simple properties of compatible relations.

Claim 8.1.9.1 *If R is a relation on pλ-terms (modulo α-conversion) satisfying* (refl) *and* ($*$subst)*, then R satisfies also* (0-$*$subst)*,* (w-$*$subst) *and is closed w.r.t.* renaming *of free variables.*

Proof We prove only closure w.r.t. renaming of free variables, as the other properties are proved similarly.

Given two variables x and z, we derive $(M_1\!\upharpoonright\! D_1)*[x := z]\; R\, (M_2\!\upharpoonright\! D_2)*[x := z]$ from the assumption $M_1\!\upharpoonright\! D_1\; R\, M_2\!\upharpoonright\! D_2$:

- by (refl)
  $z\restriction\emptyset\, R\, z\restriction\emptyset$
- by (*subst) applied to the assumption
  $(M_1\restriction D_1)*[x{:=}\,z]\, R\, (M_2\restriction D_2)*[x{:=}\,z]$

∎

1. We show that R satisfies (w-ξ). Suppose that

   - $M_1\restriction D_1\, R\, M_2\restriction D_2$
   - $y\restriction E_1\, R\, y\restriction E_2$

   we have to derive

   $$(\lambda x.M_1\restriction D_1)\restriction E_1\, R\, (\lambda x.M_2\restriction D_2)\restriction E_2$$

   We assume that $x \notin \mathrm{FV}(E_1 \cup E_2)$. In fact, we can always replace x by a new variable $z \notin \mathrm{FV}(M_i\restriction D_i \cup E_i)$ and:

   - rename the bound variable x with z without changing the conclusion, because of α-conversion
   - derive the first premiss with x replaced by z, because R is closed w.r.t. renaming of free variables

   Then, we proceed as follows:

   - by (w-*subst)
     $M_1\restriction D_1 \cup E_1\, R\, M_2\restriction D_2 \cup E_2$
   - since $x \notin \mathrm{FV}(E_1 \cup E_2)$, by ($\xi$)
     $(\lambda x.M_1\restriction D_1)\restriction E_1\, R\, (\lambda x.M_2\restriction D_2)\restriction E_2$

2. easy.

3. (refl) is immediate.

   (*subst) is straightforward. Suppose that

   - $M_1\restriction D_1\, R\, M_2\restriction D_2\, S\, M_3\restriction D_3$

- $N_1 \restriction E_1 \, R \, N_2 \restriction E_2 \, S \, N_3 \restriction E_3$

then by ($*$subst) applied componentwise

$(M_1 \restriction D_1)*[x := N_1 \restriction E_1] \, R \, (M_2 \restriction D_2)*[x := N_2 \restriction E_2] \, S \, (M_3 \restriction D_3)*[x := N_3 \restriction E_3]$

((w-ξ) and) (ξ) requires some extra care. Suppose that

- $M_1 \restriction D_1 \cup E_1 \, R \, M_2 \restriction D_2 \, S \, M_3 \restriction D_3 \cup E_3$
- $y \restriction E_1 \, R \, N_2 \restriction E_2 \, S \, y \restriction E_3$
- $x \notin \mathrm{FV}(E_1 \cup E_3)$

we have to derive

$$(\lambda x.M_1 \restriction D_1) \restriction E_1 \, R \, (\lambda x.M_2 \restriction D_2) \restriction E_2 \, S \, (\lambda x.M_3 \restriction D_3) \restriction E_3$$

By arguing as in the proof of the first statement, we can assume that $x \notin \mathrm{FV}(E_2)$:

- by (0-$*$subst) applied componentwise to the second assumption $y \restriction E_1 \, R \, y \restriction E_2 \, S \, y \restriction E_3$
- by (w-$*$subst) applied componentwise the first assumption $M_1 \restriction D_1 \cup E_1 \, R \, M_2 \restriction D_2 \cup E_2 \, S \, M_3 \restriction D_3 \cup E_3$
- since $x \notin \mathrm{FV}(E_1 \cup E_2 \cup E_3)$, by ($\xi$) applied componentwise $(\lambda x.M_1 \restriction D_1) \restriction E_1 \, R \, (\lambda x.M_2 \restriction D_2) \restriction E_2 \, S \, (\lambda x.M_3 \restriction D_3) \restriction E_3$

Now we prove that $(R \cup S)^*$ is (w-compatible) s-compatible:

- by the third statement of Lemma 8.1.1 $(R \cup S)^* = (RS)^*$
- since RS is s-compatible, by repeating the argument above $(RS)^n$ is s-compatible for every $n \geq 1$. Therefore, its *limit* $(RS)^*$ has to be s-compatible

4. By the first and third statements, RST is w-compatible. Therefore we have to establish only (ξ). Suppose that

    - $M_1 \restriction D_1 \cup E_1 \, R \, M_2 \restriction D_2 \, S \, M_3 \restriction D_3 \, T \, M_4 \restriction D_4 \cup E_4$

- $y\restriction E_1 \, R \, N_2\restriction E_2 \, S \, N_3\restriction E_3 \, T \, y\restriction E_4$
- $x \notin \mathrm{FV}(E_1 \cup E_4)$

we have to derive

$(\lambda x.M_1\restriction D_1)\restriction E_1 \, R \, (\lambda x.M_2\restriction D_2)\restriction E_2 \, S (\lambda x.M_3\restriction D_3)\restriction E_3 \, T \, (\lambda x.M_4\restriction D_4)\restriction E_4$

By arguing as in the proof of the third statement, we can assume that $N_2, N_3 \equiv y$ and $x \notin \mathrm{FV}(E_2 \cup E_3)$:

- by (w-$*$subst) for R and T
  $M_1\restriction D_1 \cup E_1 \, R \, M_2\restriction D_2 \cup E_2$ and
  $M_3\restriction D_3 \cup E_3 \, T \, M_4\restriction D_4 \cup E_4$
- since $x \notin \mathrm{FV}(E_1 \cup E_2 \cup E_3 \cup E_4)$, by ($\xi$) for R and T
  $(\lambda x.M_1\restriction D_1)\restriction E_1 \, R \, (\lambda x.M_2\restriction D_2)\restriction E_2$ and
  $(\lambda x.M_3\restriction D_3)\restriction E_3 \, T \, (\lambda x.M_4\restriction D_4)\restriction E_4$
- by (w-ξ) for S
  $(\lambda x.M_2\restriction D_2)\restriction E_2 \, S \, (\lambda x.M_3\restriction D_3)\restriction E_3$ and by composing with the other two relations
  $(\lambda x.M_1\restriction D_1)\restriction E_1 \, R \, (\lambda x.M_2\restriction D_2)\restriction E_2 \, S$
  $(\lambda x.M_3\restriction D_3)\restriction E_3 \, T \, (\lambda x.M_4\restriction D_4)\restriction E_4$

Now we prove that $(R \cup S \cup T)^*$ is s-compatible:

- by the third statement of Lemma 8.1.1
  $(R \cup S \cup T)^* = (RST)^*$
- since RST is s-compatible, by the third statement
  $(RST)^n$ is s-compatible for every $n \geq 1$. Therefore, its *limit* $(RST)^*$ has to be s-compatible

∎

8.2 β-reduction

We define pλl-terms and raw λl-terms, roughly speaking, they are pλ-terms and raw λ-terms with some of the *β-redexes labelled*. In order to define the *one-step parallel* reduction $\rightarrow_1$ on pλ-terms, we introduce two functions from pλl-terms to pλ-terms: $|\_|$ and $\phi(\_)$ (see Section 11.1 of [Bar81]). The function $|\_|$

erases the labels, while $\phi(\_)$ *contracts* all the labelled β-redexes (from the inside to the outside). Then the *one-step parallel* reduction can be defined as (see Definition 8.2.17):

- $t_1 {\upharpoonright} D_1 \to_1 t_2 {\upharpoonright} D_2 \overset{\Delta}{\Longleftrightarrow}$ there exists a pλl-term $t {\upharpoonright} D$ s.t.

  $t_1 {\upharpoonright} D_1 \equiv |t {\upharpoonright} D|$ and $t_2 {\upharpoonright} D_2 \equiv \phi(t {\upharpoonright} D)$

In other words, $t_1 {\upharpoonright} D_1 \to_1 t_2 {\upharpoonright} D_2$ holds iff $t_2 {\upharpoonright} D_2$ can be obtained from $t_1 {\upharpoonright} D_1$ by replacing (in a *consistent* way) some β-redexes with their contracta. We introduce also a function $\phi_m(\_)$, the *maximum one-step parallel* (see Definition 8.2.10), and prove that $\to_1$ has the strong diamond property (see Corollary 8.2.18).

Remark 8.2.1 For the $\lambda\beta$-calculus there is an equivalent definition of the one-step parallel reduction, given by a set of inference rules (see Definition 3.2.3 of [Bar81]). If one tries the same for the $\lambda_p\beta$-calculus (e.g. by taking the pure inference rules without (η), (symm) and (trans)), then the resulting relation not only replaces some β-redexes with their contracta, but it may also move some terms in or out of the body of a λ-abstraction (because of (ξ)). Moreover, the diamond property for such a relation (which we expect to hold) cannot be proved as in Lemma 3.2.6 of [Bar81], because the inference rules (x.app) and ($x.\beta$) escape any *simple* case analysis.

The notion of β-normal form cannot be defined as in Chapter 3 of [Bar81], because we do not have a *one-step* β-reduction. However, they can be defined directly as pλ-terms without β-redexes.

Definition 8.2.2 β-redexes and -normal forms

- *a* β**-redex** *is a raw* λ*-term of the form* $(\lambda x.t_1 {\upharpoonright} D_1)t_2$

- *a* β**-normal form** *(shortly* β*-NF) is a raw* λ*-term, p*λ*-term or d*λ*-term that does not contain* β*-redexes*

Remark 8.2.3 It is possible to consider a notion of reduction for the fixed-point partial lambda calculus (see Definition 4.3.5), namely:

- a Y**-redex** is a *raw* λY*-term* of the form $\mathrm{Y}t_2$

- a Y**-normal form** (shortly Y-NF) is a *raw* λY*-term*, *p*λY*-term* or *d*λY*-term* that does not contain Y-redexes

We claim that the strong diamond property for $\rightarrow_1$ (see Corollary 8.2.18) holds for a wide class of reductions, and that this can be proved by techniques similar to those developed in [Klo80] for establishing the Church-Rosser property of certain combinatory reduction systems.

Definition 8.2.4 Structural inference rules for pλl-terms

The set plTerm *of* **pλl-terms** *is generated by the structural inference rules for pλ-terms and two additional rules,* $(x.\lambda')$ *and* (λ')*, i.e.:*

$$x \quad \frac{}{x\restriction\emptyset \in \text{plTerm}}$$

$$x.\text{app} \quad \frac{t_1\restriction D_1 \in \text{plTerm} \qquad t_2\restriction D_2 \in \text{plTerm}}{x\restriction\{t_1t_2\} \cup D_1 \cup D_2 \in \text{plTerm}}$$

$$\text{app} \quad \frac{t_1\restriction D_1 \in \text{plTerm} \qquad t_2\restriction D_2 \in \text{plTerm}}{t_1t_2\restriction\{t_1t_2\} \cup D_1 \cup D_2 \in \text{plTerm}}$$

$$x.\lambda' \quad \frac{t_1\restriction D_1 \cup D_2 \in \text{plTerm} \qquad t_2\restriction D_2 \in \text{plTerm}}{\{(\lambda' y.t_1\restriction D_1)t_2\} \cup D_2 \in \text{plTerm}} \quad y \notin \text{FV}(D_2)$$

$$\lambda' \quad \frac{t_1\restriction D_1 \cup D_2 \in \text{plTerm} \qquad t_2\restriction D_2 \in \text{plTerm}}{(\lambda' y.t_1\restriction D_1)t_2\restriction\{(\lambda' y.t_1\restriction D_1)t_2\} \cup D_2 \in \text{plTerm}} \quad y \notin \text{FV}(D_2)$$

$$\lambda \quad \frac{t\restriction D' \cup D \in \text{plTerm} \qquad D \in \text{plTerm}}{(\lambda x.t\restriction D')\restriction D \in \text{plTerm}} \quad x \notin \text{FV}(D)$$

Definition 8.2.5 Raw λl-terms, raw pλl-terms and raw dλl-terms

The set rlTerm *of* **raw λl-terms** *is generated by the formation rules for raw λ-terms plus the formation rule* (λ')*, i.e.:*

$$x \quad \frac{}{x \in \text{rlTerm}}$$

$$\text{app} \quad \frac{t_1 \in \text{rlTerm} \qquad t_2 \in \text{rlTerm}}{t_1t_2 \in \text{rlTerm}}$$

$$\lambda' \quad \frac{t_1 \in \text{rlTerm} \qquad D_1 \subseteq_{fin} \text{rlTerm} \qquad t_2 \in \text{rlTerm}}{(\lambda' x.t_1\restriction D_1)t_2 \in \text{rlTerm}}$$

$$\lambda \quad \frac{D \subseteq_{fin} \text{rlTerm} \qquad t \in \text{rlTerm}}{(\lambda x.t\restriction D) \in \text{rlTerm}}$$

D is a **raw dλl-term** *iff $D \subseteq$ rlTerm.*

$t\restriction D$ is a **raw pλl-term** *iff t is a raw λl-term and D is a raw dλl-terms*

Remark 8.2.6 Also the other definitions (and results) of Section 7.1, on pλ-terms, extend smoothly to pλl-terms. Although their extension to pλl-terms is not stated explicitly, they will be used in the sequel.

The *erasing* $|\_|$ and the *parallel reduction* $\phi(\_)$ will be defined as functions from raw pλl-terms to raw pλ-terms, and only subsequently do we prove that they can be restricted to pλl-terms (see Lemma 8.2.16). Similarly, the *maximum one-step parallel* $\phi_m(\_)$ will be defined as an endofunction on raw pλ-terms, and only subsequently do we prove that it can be restricted to pλ-terms. We introduce also an auxiliary endofunction r$(_)$ on raw pλl-terms, which is essential to prove Corollary 8.2.18.

Definition 8.2.7 Erase $|\_|$

$|\_|$ is defined by induction on the structure of raw λl-terms:

- $|x| \triangleq x$
- $|t_1 t_2| \triangleq \begin{cases} (\lambda x.|t\restriction D|)|t_2| & \text{if } t_1 \equiv (\lambda' x.t\restriction D) \\ |t_1||t_2| & \text{otherwise} \end{cases}$
- $|\lambda x.t\restriction D| \triangleq (\lambda x.|t\restriction D|)$

$|\_|$ is extended additively *to raw pλl-terms and raw dλl-terms:*

- $|D| \overset{\triangle}{\simeq} \cup_{t\in D}\{|t|\}$
- $|t\restriction D| \overset{\triangle}{\simeq} |t|\restriction|D|$

Remark 8.2.8 It is immediate from its definition, that $|\_|$ is additive (i.e. $|D \cup D'| = |D| \cup |D'|$), preserves free variables (i.e. FV$(|t|) =$ FV(t)) and is the identity on raw pλ-terms (i.e. $|t\restriction D| \equiv t\restriction D$ when $t\restriction D \in$ rlTerm).

Moreover, $|\_|$ commutes with substitution, i.e. $|M[x := N]| \equiv |M|[x := |N|]$ for all raw λl-term M and N.

Definition 8.2.9 Parallel Reduction $\phi(\_)$

$\phi(\_)$ and $\phi_d(\_)$ are defined by combined induction on the structure of raw λl-terms:

- $\phi(x) \triangleq x$, $\phi_d(x) \triangleq \emptyset$

- $\phi(t_1t_2) \triangleq \begin{cases} \phi(t)[x\!:=\phi(t_2)] & \textit{if } t_1 \equiv (\lambda' x.t\restriction D) \\ \phi(t_1)\phi(t_2) & \textit{otherwise} \end{cases}$

  $\phi_d(t_1t_2) \triangleq \begin{cases} \phi(D)[x\!:=\phi(t_2)] & \textit{if } t_1 \equiv (\lambda' x.t\restriction D) \\ \{\phi(t_1)\phi(t_2)\} & \textit{otherwise} \end{cases}$

- $\phi(\lambda x.t\restriction D) \triangleq (\lambda x.\phi(t\restriction D))$, $\phi_d(\lambda x.t\restriction D) \triangleq \emptyset$

$\phi(\_)$ is extended additively *to raw $p\lambda l$-terms and raw $d\lambda l$-terms:*

- $\phi(D) \overset{\Delta}{\simeq} \cup_{t\in D}\phi_d(t)$

- $\phi(t\restriction D) \overset{\Delta}{\simeq} \phi(t)\restriction\phi(D)$

Definition 8.2.10 Maximum Reduction $\phi_m(\_)$

$\phi_m(\_)$ and $\phi_{md}(\_)$ are defined by combined induction on the structure of raw λ-terms:

- $\phi_m(x) \triangleq x$, $\phi_{md}(x) \triangleq \emptyset$

- $\phi_m(t_1t_2) \triangleq \begin{cases} \phi_m(t)[x\!:=\phi_m(t_2)] & \textit{if } t_1 \equiv (\lambda x.t\restriction D) \\ \phi_m(t_1)\phi_m(t_2) & \textit{otherwise} \end{cases}$

  $\phi_{md}(t_1t_2) \triangleq \begin{cases} \phi_m(D)[x\!:=\phi_m(t_2)] & \textit{if } t_1 \equiv (\lambda x.t\restriction D) \\ \{\phi_m(t_1)\phi_m(t_2)\} & \textit{otherwise} \end{cases}$

- $\phi_m(\lambda x.t\restriction D) \triangleq (\lambda x.\phi_m(t\restriction D))$, $\phi_{md}(\lambda x.t\restriction D) \triangleq \emptyset$

$\phi_m(\_)$ is extended additively *to raw $p\lambda$-terms and raw $d\lambda$-terms:*

- $\phi_m(D) \overset{\Delta}{\simeq} \cup_{t\in D}\phi_{md}(t)$

- $\phi_m(t\restriction D) \overset{\Delta}{\simeq} \phi_m(t)\restriction\phi_m(D)$

Definition 8.2.11 Residual Reduction $\mathrm{r}(\_)$

$\mathrm{r}(\_)$ and $\mathrm{r}_d(\_)$ are defined by combined induction on the structure of raw λl-terms:

- $\mathrm{r}(x) \triangleq x$, $\mathrm{r}_d(x) \triangleq \emptyset$

- $\mathrm{r}(t_1t_2) \triangleq \begin{cases} \mathrm{r}(t)[x\!:=\mathrm{r}(t_2)] & \textit{if } t_1 \equiv (\lambda' x.t\restriction D) \\ (\lambda' x.\mathrm{r}(t\restriction D))\mathrm{r}(t_2) & \textit{if } t_1 \equiv (\lambda x.t\restriction D) \\ \mathrm{r}(t_1)\mathrm{r}(t_2) & \textit{otherwise} \end{cases}$

  $\mathrm{r}_d(t_1t_2) \triangleq \begin{cases} \mathrm{r}(D)[x\!:=\mathrm{r}(t_2)] & \textit{if } t_1 \equiv (\lambda' x.t\restriction D) \\ \{(\lambda' x.\mathrm{r}(t\restriction D))\mathrm{r}(t_2)\} & \textit{if } t_1 \equiv (\lambda x.t\restriction D) \\ \{\mathrm{r}(t_1)\mathrm{r}(t_2)\} & \textit{otherwise} \end{cases}$

- $\mathrm{r}(\lambda x.t\upharpoonright D) \triangleq (\lambda x.\mathrm{r}(t\upharpoonright D))$, $\mathrm{r}_d(\lambda x.t\upharpoonright D) \triangleq \emptyset$

$\mathrm{r}(\_)$ *is extended* additively *to raw pλl-terms and raw dλl-terms:*

- $\mathrm{r}(D) \overset{\Delta}{\simeq} \cup_{t\in D}\mathrm{r}_d(t)$
- $\mathrm{r}(t\upharpoonright D) \overset{\Delta}{\simeq} \mathrm{r}(t)\upharpoonright\mathrm{r}(D)$

Remark 8.2.12 It is immediate from their definition, that $\phi(\_)$, $\phi_m(\_)$ and $\mathrm{r}(\_)$ are additive (i.e. $\phi(D\cup D') = \phi(D)\cup\phi(D')$), do not introduce new free variables (i.e. $\mathrm{FV}(\phi(t)) \subseteq \mathrm{FV}(t)$) and remove *redundancies* (i.e. $\phi(t)$ *has no redundancies*, see Notation 7.1.7).

Moreover, if a raw pλ-term $t\upharpoonright D$ has no redundancies, then $\phi(t\upharpoonright D) \equiv t\upharpoonright D$.

Lemma 8.2.13 $\phi(\_)$ commutes with substitution

For all raw λl-terms M and N s.t. M has no redundancies

- $\phi(M[y\!:= N]) \equiv \phi(M)[y\!:= \phi(N)]$
- $\phi_d(M[y\!:= N]) \equiv \phi_d(M)[y\!:= \phi(N)]$, *if M is not a value.*

In particular, $\phi((M\upharpoonright D)[y\!:= N]) \equiv \phi(M\upharpoonright D)[y\!:= \phi(N)]$ for any raw pλl-term $M\upharpoonright D$ without redundancies.

Proof By induction on the structure of the raw λl-term M.

- $(x) \Longrightarrow M \equiv x$ There are two subcases: $x \not\equiv y$ and $x \equiv y$. In both subcases the proof is immediate.
- (app) $\Longrightarrow M \equiv t_1t_2$. As M has no redundancies, then t_1 and t_2 have no redundancies.

  $\phi(M[y\!:= N]) \equiv \phi(t_1[y\!:= N])\phi(t_2[y\!:= N]) \equiv$
  by IH for t_1 and t_2
  $(\phi(t_1)[y\!:= \phi(N)])(\phi(t_2)[y\!:= \phi(N)]) \equiv \phi(M)[y\!:= \phi(N)]$.

  $\phi_d(M[y\!:= N]) \equiv \{\phi(t_1[y\!:= N])\phi(t_2[y\!:= N])\} \equiv$
  by IH for t_1 and t_2
  $\{(\phi(t_1)[y\!:= \phi(N)])(\phi(t_2)[y\!:= \phi(N)])\} \equiv \phi_d(M)[y\!:= \phi(N)]$.

- $(\lambda') \implies M \equiv (\lambda' x.t_1 \restriction D_1)t_2$. W.l.o.g. we can assume that x does not clash with y or N. As M has no redundancies, then t_1 and t_2 have no redundancies and all $t \in D_1$ are not values and have no redundancies.

  $\phi(M[y:=N]) \equiv (\phi(t_1[y:=N]))[x:=\phi(t_2[y:=N])] \equiv$
  by IH for t_1 and t_2
  $(\phi(t_1)[y:=\phi(N)])[x:=\phi(t_2)[y:=\phi(N)]] \equiv$
  since $\phi(\_)$ does not introduce new variables, by the assumptions on x
  $(\phi(t_1)[x:=\phi(t_2)])[y:=N] \equiv \phi(M)[y:=\phi(N)]$.

  $\phi_d(M[y:=N]) \equiv \phi(D_1[y:=N])[x:=\phi(t_2[y:=N])] \equiv$
  by additivity and IH for $t \in D_1$ and t_2
  $(\phi(D_1)[y:=\phi(N)])[x:=\phi(t_2)[y:=\phi(N)]] \equiv$
  since $\phi(\_)$ does not introduce new variables, by the assumptions on x
  $(\phi(D_1)[x:=\phi(t_2)])[y:=N] \equiv \phi_d(M)[y:=\phi(N)]$.

- $(\lambda) \implies M \equiv (\lambda x.t_1 \restriction D_1)$ W.l.o.g. we can assume that x does not clash with y or N. As M has no redundancies, then t_1 has no redundancies and all $t \in D_1$ are not values and have no redundancies.

  $\phi(M[y:=N]) \equiv (\lambda x.\phi(t_1 \restriction D_1[y:=N])) \equiv$
  by additivity and IH for t_1 and $t \in D_1$
  $(\lambda x.\phi(t_1 \restriction D_1)[y:=\phi(N)]) \equiv \phi(M)[y:=\phi(N)]$.

  $\phi_d(M[y:=N]) \equiv \emptyset \equiv \phi_d(M)[y:=\phi(N)]$.

∎

Example 8.2.14 $\phi_m(\_)$ and $\mathrm{r}(\_)$ do not commute with substitution, e.g. let $M \stackrel{\Delta}{\equiv} xy$ and $N \stackrel{\Delta}{\equiv} \mathrm{I}$, then $\phi_m(M[x:=N]) \equiv y$ while $\phi_m(M)[x:=\phi(N)] \equiv \mathrm{I}y$.

Lemma 8.2.15 *for any raw λl-term M:*

- $\phi(M) \equiv |\mathrm{r}(M)|$ *and* $\phi_d(M) \equiv |\mathrm{r}_d(M)|$
- $\phi_m(|M|) \equiv \phi(\mathrm{r}(M))$ *and* $\phi_{md}(|M|) \equiv \phi(\mathrm{r}_d(M))$

In particular, $\phi(M \restriction D) \equiv |\mathrm{r}(M \restriction D)|$ *and* $\phi_m(|M \restriction D|) \equiv \phi(\mathrm{r}(M \restriction D))$ *for any raw $p\lambda l$-term $M \restriction D$.*

Proof Both statements are proved by induction on the structure of the raw λl-term M (see Definition 8.2.4). We carry out in details the proof of $\phi_m(|M|) \equiv \phi(\mathrm{r}(M))$ and $\phi_{md}(|M|) \equiv \phi(\mathrm{r}_d(M))$.

- $(x) \Longrightarrow M \equiv x$ immediate.

  $\phi_m(|M|), \phi(\mathrm{r}(M)) \equiv x$ and $\phi_{md}(|M|), \phi(\mathrm{r}_d(M)) \equiv \{x\}$.

- (app) $\Longrightarrow M \equiv t_1 t_2$ by IH .

  There are two subcases: (λ), otherwise. Depending on the structure of the raw λ-term t_1.

  - $(\lambda) \Longrightarrow t_1 \equiv (\lambda x.t_3 \restriction D_3)$.

    $\phi_m(|M|) \equiv \phi_m(|t_3|)[x := \phi_m(|t_2|)] \equiv$
    by IH for t_3 and t_2
    $\phi(\mathrm{r}(t_3))[x := \phi(\mathrm{r}(t_2))] \equiv$
    $\phi((\lambda' x.\mathrm{r}(t_3 \restriction D_3))\mathrm{r}(t_2)) \equiv \phi(\mathrm{r}(M))$.

    $\phi_{md}(|M|) \equiv \phi_m(|D_3|)[x := \phi_m(|t_2|)] \equiv$
    by additivity and IH for $t \in D_3$ and t_2
    $\phi(\mathrm{r}(D_3))[x := \phi(\mathrm{r}(t_2))] \equiv$
    $\phi(\{(\lambda' x.\mathrm{r}(t_3 \restriction D_3))\mathrm{r}(t_2)\}) \equiv \phi(\mathrm{r}_d(M))$.

  - otherwise $\Longrightarrow$

    $\phi_m(|M|) \equiv \phi_m(|t_1|)\phi_m(|t_2|) \equiv$
    by IH for t_1 and t_2
    $\phi(\mathrm{r}(t_1))\phi(\mathrm{r}(t_2)) \equiv \phi(\mathrm{r}(M))$.

    $\phi_{md}(|M|) \equiv \{\phi_m(|t_1|)\phi_m(|t_2|)\} \equiv$
    by IH for t_1 and t_2
    $\{\phi(\mathrm{r}(t_1))\phi(\mathrm{r}(t_2))\} \equiv \phi(\mathrm{r}_d(M))$.

- $(\lambda') \Longrightarrow M \equiv (\lambda' x.t_1 \restriction D_1)t_2$ by IH and Lemma 8.2.13.

  $\phi_m(|M|) \equiv \phi_m(|t_1|)[x := \phi_m(|t_2|)] \equiv$
  by IH for t_1 and t_2
  $\phi(\mathrm{r}(t_1))[x := \phi(\mathrm{r}(t_2))] \equiv$
  since $\mathrm{r}(\_)$ removes redundancies, by Lemma 8.2.13
  $\phi(\mathrm{r}(t_1)[x := \mathrm{r}(t_2)]) \equiv \phi(\mathrm{r}(M))$.

$\phi_{md}(|M|) \equiv \phi_m(|D_1|)[x := \phi_m(|t_2|)] \equiv$
by additivity and IH for $t \in D_1$ and t_2
$\phi(\mathrm{r}(D_1))[x := \phi(\mathrm{r}(t_2))] \equiv$
since r(_) removes redundancies, by Lemma 8.2.13
$\phi(\mathrm{r}(D_1)[x := \mathrm{r}(t_2)]) \equiv \phi(\mathrm{r}_d(M))$.

- $(\lambda) \Longrightarrow M \equiv (\lambda x.t_1 \upharpoonright D_1)$ by IH .

  $\phi_m(|M|) \equiv (\lambda x.\phi_m(|t_1 \upharpoonright D_1|)) \equiv$
  by additivity and IH for t_1 and $t \in D_1$
  $(\lambda x.\phi(\mathrm{r}(t_1 \upharpoonright D_1))) \equiv \phi(\mathrm{r}(M))$.

  Similarly for $\phi_{md}(|M|) \equiv \phi(\mathrm{r}_d(M))$.

∎

Lemma 8.2.16 Preservation of Saturation
The functions $|\_|$, $\phi(\_)$, $\phi_m(\_)$ *and* r(_) *preserve* pλl*-terms, i.e.*

- *if* $t \upharpoonright D \in \mathrm{plTerm}$, *then* $|t \upharpoonright D| \in \mathrm{pTerm}$
- *if* $t \upharpoonright D \in \mathrm{plTerm}$, *then* $\phi(t \upharpoonright D) \in \mathrm{pTerm}$
- *if* $t \upharpoonright D \in \mathrm{pTerm}$, *then* $\phi_m(t \upharpoonright D) \in \mathrm{pTerm}$
- *if* $t \upharpoonright D \in \mathrm{plTerm}$, *then* $\mathrm{r}(t \upharpoonright D) \in \mathrm{plTerm}$

Proof The claim for $|\_|$ can be proved by a straightforward induction on the proof of $t \upharpoonright D \in \mathrm{plTerm}$. The claim for $\phi(\_)$ follows from that for $|\_|$ and r(_), because $\phi(t \upharpoonright D) \equiv |\mathrm{r}(t \upharpoonright D)|$ (see Lemma 8.2.15). The claim for $\phi_m(\_)$ follows from that for $\phi(\_)$ and r(_), because if $t \upharpoonright D$ is a pλ-term, then it is also a pλl-term, $t \upharpoonright D \equiv |t \upharpoonright D|$ and $\phi_m(t \upharpoonright D) \equiv \phi(\mathrm{r}(t \upharpoonright D))$ (see Lemma 8.2.15).

The claim for r(_) is proved by induction on the proof of $M \upharpoonright D \in \mathrm{plTerm}$ (see Definition 8.2.4).

- $(x) \Longrightarrow$ immediate.

  $\mathrm{r}(M \upharpoonright D) \equiv x \upharpoonright \emptyset$

- $(x.\mathrm{app}) \Longrightarrow$

Let $t_1 \restriction D_1 \in$ plTerm and $t_2 \restriction D_2 \in$ plTerm be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M \restriction D \in$ plTerm.

By IH $\mathrm{r}(t_1 \restriction D_1) \in$ plTerm and $\mathrm{r}(t_2 \restriction D_2) \in$ plTerm.

We have to prove that $\mathrm{r}(M \restriction D)$ is a pλl-term. There are two subcases: (λ), otherwise. Depending on the structure of the raw λ-term t_1.

- $(\lambda) \Longrightarrow t_1 \equiv (\lambda y.t_3 \restriction D_3)$.
  $\mathrm{r}(M \restriction D) \equiv x \restriction \{(\lambda' y.\mathrm{r}(t_3) \restriction \mathrm{r}(D_3))\mathrm{r}(t_2)\} \cup \mathrm{r}(D_1) \cup \mathrm{r}(D_2)$:
  * since $(\lambda y.\mathrm{r}(t_3) \restriction \mathrm{r}(D_3)) \restriction \mathrm{r}(D_1) \equiv \mathrm{r}(t_1 \restriction D_1) \in$ plTerm, then
    $\mathrm{r}(t_3) \restriction \mathrm{r}(D_3) \cup \mathrm{r}(D_1) \in$ plTerm
  * by (0-$*$) and (w-$*$) for pλl-terms
    $\mathrm{r}(t_3) \restriction \mathrm{r}(D_3) \cup \mathrm{r}(D_1) \cup \mathrm{r}(D_2) \in$ plTerm and
    $\mathrm{r}(t_2) \restriction \mathrm{r}(D_1) \cup \mathrm{r}(D_2) \in$ plTerm
  * By $(x.\lambda')$
    $x \restriction \{(\lambda' y.\mathrm{r}(t_3) \restriction \mathrm{r}(D_3))\mathrm{r}(t_2)\} \cup \mathrm{r}(D_1) \cup \mathrm{r}(D_2) \in$ plTerm
- otherwise $\Longrightarrow$
  $\mathrm{r}(M \restriction D) \equiv x \restriction \{\mathrm{r}(t_1)\mathrm{r}(t_2)\} \cup \mathrm{r}(D_1) \cup \mathrm{r}(D_2)$:
  * By $(x.\text{app})$
    $x \restriction \{\mathrm{r}(t_1)\mathrm{r}(t_2)\} \cup \mathrm{r}(D_1) \cup \mathrm{r}(D_2) \in$ plTerm

- (app) $\Longrightarrow$ similar to the case $(x.\text{app})$
- $(x.\lambda') \Longrightarrow$

  Let $t_1 \restriction D_1 \cup D_2 \in$ plTerm and $t_2 \restriction D_2 \in$ plTerm be the premisses of the last rule (i.e. $(x.\lambda')$) in the proof of $M \restriction D \in$ plTerm.

  We have to prove that $\mathrm{r}(M \restriction D) \equiv x \restriction (\mathrm{r}(D_1)) * [y := \mathrm{r}(t_2)] \cup \mathrm{r}(D_2)$ is a pλl-term.

  - By IH
    $\mathrm{r}(t_1 \restriction D_1 \cup D_2) \in$ plTerm and $\mathrm{r}(t_2 \restriction D_2) \in$ plTerm
  - by (0-$*$) for pλl-terms
    $\mathrm{r}(D_1) \cup \mathrm{r}(D_2)$ is a dλl-term

- since $y \notin \mathrm{FV}(D_2)$, by $(*)$ for pλl-terms
  $\mathrm{r}(D) \equiv (\mathrm{r}(D_1) \cup \mathrm{r}(D_2))*[y := \mathrm{r}(t_2 \upharpoonright D_2)]$ is a dλl-term, therefore $x \upharpoonright D$ is a pλl-term

- $(\lambda') \Longrightarrow$ similar to the case $(x.\lambda')$

- $(\lambda) \Longrightarrow$ trivial

  Let $t \upharpoonright D' \cup D \in$ plTerm and $D \in$ plTerm be the premisses of the last rule (i.e. (λ)) in the proof of $M \upharpoonright D \in$ plTerm.

  - By IH
    $\mathrm{r}(t \upharpoonright D' \cup D) \in$ plTerm and $\mathrm{r}(D) \in$ plTerm
  - since $\mathrm{r}(\_)$ does not introduce new free variables, by (λ)
    $\mathrm{r}(M \upharpoonright D) \equiv (\lambda x.\mathrm{r}(t) \upharpoonright \mathrm{r}(D')) \upharpoonright \mathrm{r}(D) \in$ plTerm

∎

Definition 8.2.17 One-step parallel reduction
The **one-step parallel reduction** $\rightarrow_1$ *is the relation* $t_1 \upharpoonright D_1 \rightarrow_1 t_2 \upharpoonright D_2 \overset{\Delta}{\Longleftrightarrow}$ *there exists a pλl-term* $t \upharpoonright D$ *s.t.* $t_1 \upharpoonright D_1 \equiv |t \upharpoonright D|$ *and* $t_2 \upharpoonright D_2 \equiv \phi(t \upharpoonright D)$. *By Lemma 8.2.16,* $\rightarrow_1$ *is actually a relation on pλ-terms.*

Corollary 8.2.18 Strong diamond property
If $t_1 \upharpoonright D_1 \rightarrow_1 t_2 \upharpoonright D_2$*, then* $t_2 \upharpoonright D_2 \rightarrow_1 \phi_m(t_1 \upharpoonright D_1)$.

Proof By definition of $\rightarrow_1$ there exists a pλl-term $t \upharpoonright D$ s.t. $t_1 \upharpoonright D_1 \equiv |t \upharpoonright D|$ and $t_2 \upharpoonright D_2 \equiv \phi(t \upharpoonright D)$:

- by Lemma 8.2.16
  $\mathrm{r}(t \upharpoonright D)$ is also a pλl-term

- by Lemma 8.2.15
  $t_2 \upharpoonright D_2 \equiv |\mathrm{r}(t \upharpoonright D)|$ and $\phi_m(t_1 \upharpoonright D_1) \equiv \phi(\mathrm{r}(t \upharpoonright D))$

- by definition of $\rightarrow_1$
  $t_2 \upharpoonright D_2 \rightarrow_1 \phi_m(t_1 \upharpoonright D_1)$

∎

Lemma 8.2.19 Soundness of $\rightarrow_1$

If $t_1 \restriction D_1 \rightarrow_1 t_2 \restriction D_2$*, then* $t_1 \restriction D_1 \simeq t_2 \restriction D_2$ *is derivable in* $\lambda\beta$pEQL.

Proof We show that $|t \restriction D| \simeq \phi(t \restriction D)$ is derivable in $\lambda\beta$pEQL by induction on the proof of $t \restriction D \in$ plTerm. The only interesting case is $(x.\lambda')$ (and (λ')).

Let $t_1 \restriction D_1 \cup D_2 \in$ plTerm and $t_2 \restriction D_2 \in$ plTerm be the premisses of the last rule (i.e. $(x.\lambda')$) in the proof of $t \restriction D \in$ plTerm. We have to prove that

$$x \restriction \{(\lambda y.t_1 \restriction D_1)t_2\} \cup D_2 \simeq x \restriction \phi(D_1)[y\!:= \phi(t_2)] \cup \phi(D_2)$$

- by IH

  $t_1 \restriction D_1 \cup D_2 \simeq \phi(t_1) \restriction \phi(D_1) \cup \phi(D_2)$ and

  $t_2 \restriction D_2 \simeq \phi(t_2) \restriction \phi(D_2)$

- by the pure inference rule $(x.\beta)$ (see Definition 7.2.3)

  $x \restriction \{(\lambda y.t_1 \restriction D_1)t_2\} \cup D_2 \simeq x \restriction (\phi(D_1) \cup \phi(D_2)) * [y\!:= \phi(t_2 \restriction D_2)] \equiv$
  since $y \notin \mathrm{FV}(D_2)$ and $\phi(\_)$ does not introduce new free variables
  $x \restriction \phi(D_1)[y\!:= \phi(t_2)] \cup \phi(D_2)$

∎

Proposition 8.2.20 Admissible Inference Rules for $\rightarrow_1$

$\rightarrow_1$ *is a w-compatible relation and satisfies the axiom* (β) *of* $\lambda\beta$*p-equational logic (see Definition 7.2.1).*

Proof We have to prove that $\rightarrow_1$ satisfies (refl), (β), (w-ξ) and ($*$subst).

- (refl) $\Longrightarrow$ If $t \restriction D$ is a pλ-term, then $t \restriction D$ is a pλl-term, $|t \restriction D| \equiv t \restriction D$ and $\phi(t \restriction D) \equiv t \restriction D$ (as $t \restriction D$ has no redundancies).

- (β) $\Longrightarrow$ If $t \restriction D$ is a pλ-term, then $M \equiv (\lambda' x.t \restriction D)x \restriction \{(\lambda' x.t \restriction D)x\}$ is a pλl-term, $|M| \equiv (\lambda x.t \restriction D)x \restriction \{(\lambda x.t \restriction D)x\}$ and $\phi(M) \equiv t \restriction D$.

- (w-ξ) $\Longrightarrow$ If $t \restriction D$ and D' are pλl-terms, then $M \equiv (\lambda y.t \restriction D) \restriction D'$ is a pλl-term, $|M| \equiv (\lambda x.|t \restriction D|) \restriction |D'|$ and $\phi(M) \equiv (\lambda x.\phi(t \restriction D)) \restriction \phi(D')$.

- ($*$subst) $\Longrightarrow$ If $t_1 \restriction D_1$ and $t_2 \restriction D_2$ are pλl-terms, then $M \equiv (t_1 \restriction D_1) * [y\!:= t_2 \restriction D_2]$ is a pλl-term, $|M| \equiv (|t_1 \restriction D_1|) * [y\!:= |t_2 \restriction D_2|]$ and $\phi(M) \equiv (\phi(t_1 \restriction D_1)) * [y\!:= \phi(t_2 \restriction D_2)]$ (by Lemma 8.2.13, as $t \restriction D$ has no redundancies).

∎

Remark 8.2.21 The relation $\rightarrow_1$ is not s-compatible and its symmetric and transitive closure is properly included in the set of pλ-equations provable in $\lambda\beta$p-equational logic.

8.3 β-reduction and $\lambda\beta\eta$monp-equational logic

In this section we want to relate the one-step parallel reduction $\rightarrow_1$ to the preorder $\lesssim$ of $\lambda\beta\eta$monp-equational logic (see Theorem 8.3.10). In order to do this, we introduce the *one-step dη-preorder* $\lesssim_{1d\eta}$, whose transitive closure is the preorder $\lesssim$ of $\lambda\eta$monp-equational logic (see Lemma 8.3.5).

Remark 8.3.1 One may wonder why we did not define $\beta\eta$-reduction and relate it to $\lesssim$ of $\lambda\beta\eta$monp-equational logic via the *d-preorder*. It is straightforward to define a notion of η-reduction on pλ-terms

> $(\lambda x.Mx{\restriction}Mx){\restriction}D \rightarrow_\eta M{\restriction}D$, provided $x \notin \mathrm{FV}(M)$. There is another restriction on $M{\restriction}D$ (namely $M \in D$ if M is an application) to enforce the correctness of η-reduction, but it is automatically satisfied, when $M{\restriction}D$ is a pλ-terms.

Then we can proceed as in Section 8.2 to prove a strong diamond property (see Corollary 8.2.18) for η-reduction. However, *one-step parallel η-reduction* does not have the diamond property up to the d-preorder $\lesssim_d$ (see Example 8.3.9), therefore we have decided to take care of η in the dη-preorder.

We think that the problems with η-reduction are *due* to its *non-operational nature*, in the sense that an interpreter for a functional programming language does not perform η-reductions.

Definition 8.3.2 (One-step) dη-Preorder

*The (***one-step***)* **dη-preorder** $\lesssim_{d\eta}$ *($\lesssim_{1d\eta}$) is the binary relation on pλ-terms generated by the pure inference rules, of Definition 7.3.4, for $\lambda\eta$*mon*p-equational logic (except* (trans)*).*

Remark 8.3.3 The results stated in this section for the dη-preorder hold also for the *d-preorder* $\lesssim_d$ (i.e. $\lesssim$ of λmonp-equational logic), *d-conversion* $\simeq_d$ (i.e. $\simeq$ of λp-equational logic) and *dη-conversion* $\simeq_{d\eta}$ (i.e. $\simeq$ of $\lambda\eta$p-equational logic). The proofs (and definitions) are actually simpler.

Proposition 8.3.4 Admissible Inference Rules for $\lesssim_{1d\eta}$

The relation $\lesssim_{1d\eta}$ *is s-compatible and satisfies the axioms* (incl) *and* (η) *of* $\lambda\beta\eta$monp*-equational logic (see Definition 7.3.2).*

Proof The rule (ξ) is admissible, by definition of $\lesssim_{1d\eta}$.

The proof that (incl) and ($*$subst) are admissible in the set generated by the pure inference rules for $\lambda\beta\eta$monp-equational logic (see Propositions 7.3.6 and 7.3.9) can be *reused* to prove the admissibility in $\lesssim_{1d\eta}$.

The axioms (refl) and (η) follow easily from (incl) (and the inference rules (η) and (η^{op})). ∎

Lemma 8.3.5 Postponement of (trans) in $\lambda\eta$monpEQL

$\lesssim_{d\eta} = \lesssim_{1d\eta}^*$.

Proof Since $\lesssim_{1d\eta} \subseteq \lesssim_{d\eta}$ and $\lesssim_{d\eta}$ is transitive, then $\lesssim_{1d\eta}^* \subseteq \lesssim_{d\eta}$. To establish the other inclusion, it is enough to prove that the axioms and inference rules of $\lambda\eta$monp-equational logic are admissible in $\lesssim_{1d\eta}^*$, because $\lesssim_{d\eta}$ is the smallest $\lambda\eta$monp-theory.

The satisfaction of the axioms (incl) and (η) follows from Proposition 8.3.4. (trans) is immediate. Since $\lesssim_{1d\eta}$ is s-compatible (by Proposition 8.3.4), $\lesssim_{1d\eta}^*$ is also s-compatible (by Lemma 8.1.9). Therefore, $\lesssim_{1d\eta}^*$ satisfies ($*$subst) and (ξ). ∎

The lemma below is not interesting in itself, but it is used several times in the proof of the *main lemmas*.

Lemma 8.3.6

- *If* $y \notin \mathrm{FV}(M)$*, then*

  $\phi_m((\lambda y.My\restriction\{My\})N) \equiv \phi_m(MN)$ *and*

  $\phi_{md}((\lambda y.My\restriction\{My\})N) \equiv \phi_{md}(MN)$.

- *If* $y \notin \mathrm{FV}(M)$ *and* M *is a* λ*-abstraction, then*

  $\phi_m(\lambda y.My\restriction\{My\}) \equiv \phi_m(M)$

Proof We prove that $\phi_m((\lambda y.My\restriction\{My\})N) \equiv \phi_m(MN)$. There are two cases: (λ), otherwise. Depending on the structure of the raw λ-term M.

- $(\lambda) \Longrightarrow$ since $y \notin \mathrm{FV}(M)$, we can assume that $M \equiv (\lambda y.P{\restriction}E)$.

  $\phi_m((\lambda y.My{\restriction}\{My\})N) \equiv \phi_m((\lambda y.P{\restriction}E)y)[y{:=}\,\phi_m(N)] \equiv$
  $\phi_m(P)[y{:=}\,\phi_m(N)] \equiv \phi_m(MN)$.

- otherwise $\Longrightarrow$ neither My nor MN are β-redexes.

  $\phi_m((\lambda y.My{\restriction}\{My\})N) \equiv \phi_m(M)y[y{:=}\,\phi_m(N)] \equiv$
  since $y \notin \mathrm{FV}(M)$ and $\phi_m(\_)$ does not introduce new free variables
  $\phi_m(M)\phi_m(N) \equiv \phi_m(MN)$.

$\phi_{md}((\lambda y.My{\restriction}\{My\})N) \equiv \phi_{md}(MN)$ is proved similarly.

We prove that $\phi_m(\lambda y.My{\restriction}\{My\}) \equiv \phi_m(M)$. Since $y \notin \mathrm{FV}(M)$, we can assume that $M \equiv (\lambda y.N{\restriction}E)$.

- by assumption

  $(\lambda y.My{\restriction}\{My\}) \equiv (\lambda y.(\lambda y.N{\restriction}E)y{\restriction}\{(\lambda y.N{\restriction}E)y\})$

- by definition of $\phi_m(\_)$

  $\phi_m(\lambda y.My{\restriction}\{My\}) \equiv (\lambda y.\phi_m(N)[y{:=}\,y]{\restriction}\phi_m(E)[y{:=}\,y]) \equiv \phi_m(M)$

∎

Lemma 8.3.7 First Main Lemma for $\lessapprox_{d\eta}$
If $M{\restriction}D \lessapprox_{1d\eta} N{\restriction}E$, then $\phi_m(M{\restriction}D) \lessapprox_{1d\eta} \phi_m(N{\restriction}E)$.

Proof We prove that $\phi_m(M{\restriction}D) \lessapprox_{1d\eta} \phi_m(N{\restriction}E)$ by induction on the number of inference rules used in the proof of $M{\restriction}D \lessapprox_{1d\eta} N{\restriction}E$, using the pure inference rules for $\lessapprox_{1d\eta}$ (see Definition 8.3.2).

The basic case 0 is vacuously true, because a proof of $M{\restriction}D \lessapprox_{1d\eta} N{\restriction}E$ contains at least one inference rule (as no assumptions are allowed).

Suppose that there is a proof of $M{\restriction}D \lessapprox_{1d\eta} N{\restriction}E$ using $n+1$ inference rules and that the claim of the lemma holds for all pλ-inequations provable using at most n inference rules. There are six cases, depending on the last rule in the proof of $M{\restriction}D \lessapprox_{1d\eta} N{\restriction}E$.

- $(x) \Longrightarrow$ immediate.

- since D is a dλ-term, by Lemma 8.2.16
  $\phi_m(D)$ is a dλ-term, therefore by (x)
  $x \upharpoonright \phi_m(D) \lesssim x \upharpoonright \emptyset$

- $(x.\text{app}) \Longrightarrow$

  Let $t_1 \upharpoonright D_1 \lesssim t_2 \upharpoonright D_2$ and $t_3 \upharpoonright D_3 \lesssim t_4 \upharpoonright D_4$ be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M \upharpoonright D \lesssim N \upharpoonright E$.

  There are four subcases: (ξ), (η), (η^{op}) and *otherwise*. Depending on the last rule in the proof of $t_1 \upharpoonright D_1 \lesssim t_2 \upharpoonright D_2$:

  - $(\xi) \Longrightarrow$ by IH and $(*\text{subst})$.
    Let $t'_1 \upharpoonright D'_1 \cup D_1 \lesssim t'_2 \upharpoonright D'_2 \cup D_2$ and $D_1 \lesssim D_2$ be the premisses of the last rule (i.e. (ξ)) in the proof of $t_1 \upharpoonright D_1 \lesssim t_2 \upharpoonright D_2$.
    First we prove that $\phi_m(D) \equiv (\phi_m(D'_1 \cup D_1)) * [x := \phi_m(t_3 \upharpoonright D_3)]$ (and similarly $\phi_m(E) \equiv (\phi_m(D'_2 \cup D_2)) * [x := \phi_m(t_4 \upharpoonright D_4)]$)

    * by the inference rules
      $D \equiv \{(\lambda x.t'_1 \upharpoonright D'_1)t_3\} \cup D_1 \cup D_3$
    * by definition of $\phi_m(\_)$
      $\phi_m(D) \equiv \phi_m(D'_1)[x := \phi_m(t_3)] \cup \phi_m(D_1) \cup \phi_m(D_3)$
    * $x \notin \text{FV}(\phi_m(D_1))$, because $\phi_m(\_)$ does not introduce new free variables and $x \notin \text{FV}(D_1)$ (by the side condition of (ξ)), therefore
      $\phi_m(D) \equiv (\phi_m(D'_1 \cup D_1)) * [x := \phi_m(t_3 \upharpoonright D_3)]$

    Then we show that $\phi_m(D) \lesssim \phi_m(E)$:

    * by IH applied to the first premiss of (ξ)
      $\phi_m(t'_1 \upharpoonright D'_1 \cup D_1) \lesssim \phi_m(t'_2 \upharpoonright D'_2 \cup D_2)$
    * by the admissible rule $(0\text{-}*\text{subst})$
      $\phi_m(D'_1 \cup D_1) \lesssim \phi_m(D'_2 \cup D_2)$
    * by IH applied to the second premiss of $(x.\text{app})$
      $\phi_m(t_3 \upharpoonright D_3) \lesssim \phi_m(t_4 \upharpoonright D_4)$
    * by the admissible rule $(*\text{subst})$
      $\phi_m(D) \equiv (\phi_m(D'_1 \cup D_1)) * [x := \phi_m(t_3 \upharpoonright D_3)] \lesssim$
      $(\phi_m(D'_2 \cup D_2)) * [x := \phi_m(t_4 \upharpoonright D_4)] \equiv \phi_m(E)$

- $(\eta) \Longrightarrow$ by IH .

  Let $t_1'\lceil D_1 \lesssim t_2 \lceil D_2$ be the premiss of the last rule (i.e. (η)) in the proof of $t_1 \lceil D_1 \lesssim t_2 \lceil D_2$.

  First we prove that $\phi_m(D) \equiv \phi_m(\{t_1' t_3\} \cup D_1 \cup D_3)$

  * by the inference rules
    $D \equiv \{(\lambda y.t_1' y \lceil \{t_1' y\})t_3\} \cup D_1 \cup D_3$
  * $y \notin \mathrm{FV}(t_1')$, by the side condition of (η). Therefore, by Lemma 8.3.6
    $\phi_m(D) \equiv \phi_m(\{t_1' t_3\} \cup D_1 \cup D_3)$

  Then we show that $\phi_m(D) \lesssim \phi_m(E)$:

  * by applying $(x.\mathrm{app})$ to the premiss of (η) and the second premiss of $(x.\mathrm{app})$
    $\{t_1' t_3\} \cup D_1 \cup D_3 \lesssim \{t_2 t_4\} \cup D_2 \cup D_4$ can be proved by using n inference rules
  * by IH
    $\phi_m(D) \equiv \phi_m(\{t_1' t_3\} \cup D_1 \cup D_3) \lesssim$
    $\phi_m(\{t_2 t_4\} \cup D_2 \cup D_4) \equiv \phi_m(E)$

- $(\eta^{op}) \Longrightarrow$ is *dual* to the subcase (η).
- *otherwise* $\Longrightarrow$ by IH and $(x.\mathrm{app})$.

  $\phi_m(D) \equiv \{\phi_m(t_1)\phi_m(t_3)\} \cup \phi_m(D_1) \cup \phi_m(D_3)$ (and $\phi_m(E) \equiv \{\phi_m(t_2)\phi_m(t_4)\} \cup \phi_m(D_2) \cup \phi_m(D_4)$), because t_1 (t_2) is not a λ-abstraction.

  * by IH applied to the premisses of $(x.\mathrm{app})$
    $\phi_m(t_1 \lceil D_1) \lesssim \phi_m(t_2 \lceil D_2)$ and $\phi_m(t_3 \lceil D_3) \lesssim \phi_m(t_4 \lceil D_4)$
  * by the inference rule $(x.\mathrm{app})$
    $\phi_m(D) \equiv \{\phi_m(t_1)\phi_m(t_3)\} \cup \phi_m(D_1) \cup \phi_m(D_3) \lesssim$
    $\{\phi_m(t_2)\phi_m(t_4)\} \cup \phi_m(D_2) \cup \phi_m(D_4) \equiv \phi_m(E)$

• (app) $\Longrightarrow$ similar to the case $(x.\mathrm{app})$.

• $(\xi) \Longrightarrow$ by IH and (ξ).

Let $t_1' \lceil D_1' \cup D \lesssim t_2' \lceil D_2' \cup E$ and $D \lesssim E$ be the premisses of the last rule (i.e. (ξ)) in the proof of $M \lceil D \lesssim N \lceil E$.

- by IH applied to the premisses of (ξ)
  $\phi_m(t_1' \lceil D_1' \cup D) \lesssim \phi_m(t_2' \lceil D_2' \cup E)$ and $\phi_m(D) \lesssim \phi_m(E)$

- since $\phi_m(\_)$ does not introduce new free variables and $x \notin \mathrm{FV}(D \cup E)$ (by the side condition of (ξ)), then $x \notin \mathrm{FV}(\phi_m(D \cup E))$. Therefore, by the inference rule (ξ)
  $\phi_m(M{\restriction}D) \equiv (\lambda x.\phi_m(t_1'){\restriction}\phi_m(D_1')){\restriction}\phi_m(D) \lesssim$
  $(\lambda x.\phi_m(t_2'){\restriction}\phi_m(D_2')){\restriction}\phi_m(E) \equiv \phi_m(N{\restriction}E)$

- (η) $\Longrightarrow$

  Let $t_1{\restriction}D \lesssim N{\restriction}E$ be the premiss of the last rule (i.e. (η)) in the proof of $M{\restriction}D \lesssim N{\restriction}E$.

  There are two subcases: (λ) and *otherwise*. Depending on the structure of the raw λ-term t_1:

  - (λ) $\Longrightarrow$ t_1 is a λ-abstraction, by IH .

    First we prove that $\phi_m(M{\restriction}D) \equiv \phi_m(t_1{\restriction}D)$.

    * by the inference rules
      $M{\restriction}D \equiv (\lambda y.t_1 y{\restriction}\{t_1 y\}){\restriction}D$
    * $y \notin \mathrm{FV}(t_1)$, by the side condition of (η). Therefore, by Lemma 8.3.6
      $M{\restriction}D \equiv \phi_m(t_1{\restriction}D)$

    Then we show that $\phi_m(D) \lesssim \phi_m(E)$:

    * by IH applied to the premiss of (η)
      $\phi_m(M{\restriction}D) \equiv \phi_m(t_1{\restriction}D) \lesssim \phi_m(N{\restriction}E)$

  - *otherwise* $\Longrightarrow$ by IH and (η).

    $\phi_m(M{\restriction}D) \equiv (\lambda y.\phi_m(t_1)y{\restriction}\{\phi_m(t_1)y\}){\restriction}\phi_m(D)$, because t_1 is not a λ-abstraction.

    * by IH applied to the premiss of (η)
      $\phi_m(t_1{\restriction}D) \lesssim \phi_m(N{\restriction}E)$
    * since $\phi_m(\_)$ does not introduce new free variables and $y \notin \mathrm{FV}(D)$ (by the side condition of (η)), then $y \notin \mathrm{FV}(\phi_m(D))$. Therefore, by the inference rule (η)
      $\phi_m(M{\restriction}D) \equiv (\lambda y.\phi_m(t_1)y{\restriction}\{\phi_m(t_1)y\}){\restriction}\phi_m(D) \lesssim \phi_m(N{\restriction}E)$

- (η^{op}) $\Longrightarrow$ *dual* to the case (η).

∎

Corollary 8.3.8 Strong diamond property up to $\lesssim_{1d\eta}$

$\to_1$ *has the strong diamond property up to* $\lesssim_{1d\eta}$, *i.e.*
if $t_1{\restriction}D_1 \lesssim_{1d\eta} t_2{\restriction}D_2$, $t_1{\restriction}D_1 \to_1 t_3{\restriction}D_3$ *and* $t_2{\restriction}D_2 \to_1 t_4{\restriction}D_4$, *then*
$t_3{\restriction}D_3 \to_1 \phi_m(t_1{\restriction}D_1)$, $t_4{\restriction}D_4 \to_1 \phi_m(t_2{\restriction}D_2)$ *and* $\phi_m(t_1{\restriction}D_1) \lesssim_{1d\eta} \phi_m(t_2{\restriction}D_2)$

Proof Immediate by Corollary 8.2.18 and Lemma 8.3.7. ∎

Example 8.3.9 We give a counterexample to show that *η-reduction*, i.e. replacing an η-redex $(\lambda x.Mx{\restriction}\{Mx\})$ with its η-contractum M, does not have the diamond property up to $\lesssim_d$ (or $\simeq_d$).

Consider the following pλ-terms:

- $t_1{\restriction}D_1 \stackrel{\Delta}{\equiv} (\lambda x.yx{\restriction}\{yx, yz\}){\restriction}\{yz\}$
- $t_2{\restriction}D_2 \stackrel{\Delta}{\equiv} (\lambda x.yx{\restriction}\{yx\}){\restriction}\{yz\}$
- $t_3{\restriction}D_3 \stackrel{\Delta}{\equiv} y{\restriction}\{yz\}$

It is immediate to see that $t_1{\restriction}D_1 \simeq_d t_2{\restriction}D_2 \to_\eta t_3{\restriction}D_3$, $t_1{\restriction}D_1$ and $t_3{\restriction}D_3$ are *η-normal forms*, but $t_1{\restriction}D_1 \simeq t_3{\restriction}D_3$ is not derivable in λmon-equational logic.

Theorem 8.3.10 Characterization of $\lesssim$

$t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2$ *in* $\lambda\beta\eta$monpEQL *iff* $t_1{\restriction}D_1(\to_1 \lesssim_{1d\eta})^*(\to_1{}^{op} \lesssim_{1d\eta})^* t_2{\restriction}D_2$.

Proof By Proposition 8.1.4 and Corollary 8.3.8 it is enough to prove that $\lesssim$ for $\lambda\beta\eta$monp-equational logic is $(\to_1 \cup \to_1{}^{op} \cup \lesssim_{1d\eta})^*$.

$(\to_1 \cup \to_1{}^{op} \cup \lesssim_{1d\eta})^*$ is included in $\lesssim$, because $\lesssim$ is reflexive, transitive and

- by Lemma 8.2.19

  $\to_1$ and $\to_1{}^{op}$ are included $\lesssim$
- by definition

  $\lesssim_{1d\eta}$ is included in $\lesssim$

For the other inclusion it is enough to show that the inference rules for $\lambda\beta\eta$monp-equational logic (that generate $\lesssim$) are admissible in $(\to_1 \cup \to_1{}^{op} \cup \lesssim_{1d\eta})^*$:

- (trans) is immediate

- since $\lesssim_{1d\eta}$ is s-compatible and $\to_1$ (and therefore ${\to_1}^{op}$) is w-compatible, by Lemma 8.1.9

  $(\to_1 \cup {\to_1}^{op} \cup \lesssim_{1d\eta})^*$ is s-compatible. Therefore, ($*$subst) and (ξ) are admissible.

- by Lemma 8.3.4 $\lesssim_{1d\eta}$ satisfies (incl) and (η), therefore

  $(\to_1 \cup {\to_1}^{op} \cup \lesssim_{1d\eta})^*$ satisfies (incl), (η).

- by Lemma 8.2.20 $\to_1$ satisfies (β), therefore

  $(\to_1 \cup {\to_1}^{op} \cup \lesssim_{1d\eta})^*$ satisfies (β).

∎

Example 8.3.11 In general $\lesssim_{1d\eta}$ cannot be postponed after $\to_1$, and the same holds for the relations $\lesssim_d$, $\simeq_d$ and $\simeq_{d\eta}$. We give a counterexample that applies to all of them.

Let $\mathrm{I} \stackrel{\Delta}{\equiv} (\lambda x.x{\restriction}\emptyset)$ and $N \stackrel{\Delta}{\equiv} (\lambda x.\mathrm{I}{\restriction}\emptyset)\mathrm{I}$. Consider the following pλ-terms:

- $t_1{\restriction}D_1 \stackrel{\Delta}{\equiv} (\lambda x.(\lambda y.N{\restriction}\emptyset){\restriction}\emptyset){\restriction}\{N\}$
- $t_2{\restriction}D_2 \stackrel{\Delta}{\equiv} (\lambda x.(\lambda y.N{\restriction}\emptyset){\restriction}\{N\}){\restriction}\{N\}$
- $t_3{\restriction}D_3 \stackrel{\Delta}{\equiv} (\lambda x.(\lambda y.N{\restriction}\emptyset){\restriction}\{N\}){\restriction}\emptyset$

It is immediate to see that $t_1{\restriction}D_1 \simeq_d t_2{\restriction}D_2 \to_1 t_3{\restriction}D_3$ and for all $t{\restriction}D$ s.t. $t_1{\restriction}D_1 \to_1^* t{\restriction}D$, i.e.

- $(\lambda x.(\lambda y.N{\restriction}\emptyset){\restriction}\emptyset){\restriction}\{N\}$
- $(\lambda x.(\lambda y.\mathrm{I}{\restriction}\emptyset){\restriction}\emptyset){\restriction}\{N\}$
- $(\lambda x.(\lambda y.\mathrm{I}{\restriction}\emptyset){\restriction}\emptyset){\restriction}\emptyset$

$t{\restriction}D \simeq t_3{\restriction}D_3$ is not derivable in mon$\lambda_{\mathrm{p}}\eta + {\restriction}$. If we allow arbitrary rewriting of β-redexes, then there is a fourth possibility for $t{\restriction}D$, namely $(\lambda x.(\lambda y.N{\restriction}\emptyset){\restriction}\emptyset){\restriction}\emptyset$, but also in this case $t{\restriction}D \simeq t_3{\restriction}D_3$ is not derivable in mon$\lambda_{\mathrm{p}}\eta + {\restriction}$.

8.4 β-normal forms

In this section we improve the characterization of $t_1{\restriction}D_1{\lesssim}t_2{\restriction}D_2$ (see Theorem 8.3.10) under the assumption that both pλ-terms are β-NFs (see Theorem 8.4.4), namely $\lesssim$ and $\lesssim_{d\eta}$ coincide on β-NF. In particular, decidability of $\lesssim_{d\eta}$ implies decidability of $\lesssim$ on the pλ-terms with a β-NF.

Lemma 8.4.1 *If $M{\restriction}D{\lesssim_{1d\eta}}N{\restriction}E$, M is a β-NF and N is an application (variable), then $\phi_m(N)$ is an application (variable).*

Probably, this lemma holds under more general circumstances, e.g. when all β-redexes in M contract to themselves, like $(\lambda x.xx{\restriction}xx)(\lambda x.xx{\restriction}xx)$. This would yield a similar improvement in Theorem 8.4.4.

Proof If N is a variable, then $\phi_m(N) \equiv N$. If N is an application, then we prove that $\phi_m(N)$ is an application by induction on the number of inference rules used in the proof of $M{\restriction}D{\lesssim_{1d\eta}}N{\restriction}E$, using the pure inference rules for $\lesssim_{1d\eta}$ (see Definition 8.3.2).

The basic case 0 is vacuously true, because a proof of $M{\restriction}D{\lesssim_{1d\eta}}N{\restriction}E$ contains at least one inference rule (as no assumptions are allowed).

Suppose that there is a proof of $M{\restriction}D{\lesssim_{1d\eta}}N{\restriction}E$ using $n+1$ inference rules and that the claim holds for all pλ-inequations provable using at most n inference rules. There are six cases, depending on the last rule in the proof of $M{\restriction}D{\lesssim_{1d\eta}}N{\restriction}E$.

- $(x) \Longrightarrow$ impossible, because N is an application.
- $(x.\mathrm{app}) \Longrightarrow$ impossible, because N is an application.
- $(\mathrm{app}) \Longrightarrow$

  Let $t_1{\restriction}D_1{\lesssim}t_2{\restriction}D_2$ and $t_3{\restriction}D_3{\lesssim}t_4{\restriction}D_4$ be the premisses of the last rule (i.e. (app)) in the proof of $M{\restriction}D{\lesssim}N{\restriction}E$.

  There are two subcases: (λ) and *otherwise*. Depending on the structure of the raw λ-term t_2:

  - $(\lambda) \Longrightarrow t_2$ is a λ-abstraction.

t_1 cannot be a λ-abstraction, otherwise $M \equiv t_1 t_3$ would be a β-redex, contradicting the assumption that M is a β-NF. Therefore, the last rule in the proof of $t_1 \restriction D_1 \lesssim t_2 \restriction D_2$ must be (η^{op}).

Let $t_1 \restriction D_1 \lesssim t_2' \restriction D_2$ be the premiss of the last rule (i.e. (η^{op})) in the proof of $t_1 \restriction D_1 \lesssim t_2 \restriction D_2$.

First we prove that $\phi_m(N) \equiv \phi_m(t_2' t_4)$

* by the inference rules
  $N \equiv (\lambda y.t_2' y \restriction \{t_2' y\}) t_4$
* $y \notin \mathrm{FV}(t_2')$, by the side condition of (η^{op}). Therefore, by Lemma 8.3.6
  $\phi_m(N) \equiv \phi_m(t_2' t_4)$

Then we show that $\phi_m(N)$ is an application:

* by applying (app) to the premiss of (η^{op}) and the second premiss of (app)
  $M \restriction D \equiv t_1 t_3 \restriction \{t_1 t_3\} \cup D_1 \cup D_3 \lesssim t_2' t_4 \restriction \{t_2' t_4\} \cup D_2 \cup D_4$ can be proved by using n inference rules
* by IH
  $\phi_m(N) \equiv \phi_m(t_2' t_4)$ is an application

– *otherwise* $\Longrightarrow$ immediate.
  $\phi_m(N) \equiv \phi_m(t_2)\phi_m(t_4)$, because t_2 is not a λ-abstraction.

- $(\xi) \Longrightarrow$ impossible, because N is an application.
- $(\eta) \Longrightarrow$ by IH .

Let $t_1 \restriction D \lesssim N \restriction E$ be the premiss of the last rule (i.e. (η)) in the proof of $M \restriction D \lesssim N \restriction E$.

– since $M \equiv (\lambda y.t_1 y \restriction \{t_1 y\})$ is a β-NF, then
  t_1 is a β-NF

– by IH applied to the premiss of (η)
  $\phi_m(N)$ is an application

- $(\eta^{op}) \Longrightarrow$ impossible, because N is an application.

∎

Lemma 8.4.2 Second Main Lemma for $\lesssim_{d\eta}$

If $M\restriction D$ is a β-normal form and $M\restriction D\lesssim_{1d\eta}N\restriction E$, then $\phi_m(N\restriction E)$ is a β-normal form.

Proof We prove that $\phi_m(N\restriction E)$ is a β-NF by induction on the number of inference rules used in the proof of $M\restriction D\lesssim_{1d\eta}N\restriction E$, using the pure inference rules for $\lesssim_{1d\eta}$ (see Definition 8.3.2).

The basic case 0 is vacuously true, because a proof of $M\restriction D\lesssim_{1d\eta}N\restriction E$ contains at least one inference rule (as no assumptions are allowed).

Suppose that there is a proof of $M\restriction D\lesssim_{1d\eta}N\restriction E$ using $n+1$ inference rules and that the claim of the lemma holds for all pλ-inequations provable using at most n inference rules. There are six cases, depending on the last rule in the proof of $M\restriction D\lesssim_{1d\eta}N\restriction E$.

- $(x) \Longrightarrow$ trivial, because $N\restriction E \equiv x\restriction\emptyset$ is already a β-NF.
- $(x.\text{app}) \Longrightarrow$

  Let $t_1\restriction D_1\lesssim t_2\restriction D_2$ and $t_3\restriction D_3\lesssim t_4\restriction D_4$ be the premisses of the last rule (i.e. $(x.\text{app})$) in the proof of $M\restriction D\lesssim N\restriction E$.

  By IH for the premisses of $(x.\text{app})$, both $\phi_m(D_2)$ and $\phi_m(D_4)$ are β-NFs. Therefore, to show that $\phi_m(E)\equiv\phi_{md}(t_2t_4)\cup\phi_m(D_2)\cup\phi_m(D_4)$ is a β-NF, it is enough to prove that $\phi_{md}(t_2t_4)$ is a β-NF.

  There are two subcases: (λ) and *otherwise*. Depending on the structure of the raw λ-term t_2:

  - $(\lambda) \Longrightarrow t_2$ is a λ-abstraction.

    t_1 cannot be a λ-abstraction, otherwise $t_1t_3\in D$ would be a β-redex, contradicting the assumption that $M\restriction D$ is a β-NF. Therefore, the last rule in the proof of $t_1\restriction D_1\lesssim t_2\restriction D_2$ must be (η^{op}).

    Let $t_1\restriction D_1\lesssim t_2'\restriction D_2$ be the premiss of the last rule (i.e. (η^{op})) in the proof of $t_1\restriction D_1\lesssim t_2\restriction D_2$.

    First we prove that $\phi_{md}(t_2t_4)\equiv\phi_{md}(t_2't_4)$

    * by the inference rules

      $t_2t_4\equiv(\lambda y.t_2'y\restriction\{t_2'y\})t_4$

  * $y \notin \mathrm{FV}(t'_2)$, by the side condition of (η^{op}). Therefore, by Lemma 8.3.6 $\phi_{md}(t_2 t_4) \equiv \phi_{md}(t'_2 t_4)$

  Then we show that $\phi_{md}(t_2 t_4)$ is a β-NF:

  * by applying $(x.\mathrm{app})$ to the premiss of (η^{op}) and the second premiss of $(x.\mathrm{app})$
    $D\restriction \equiv \{t_1 t_3\} \cup D_1 \cup D_3 \lesssim \{t'_2 t_4\} \cup D_2 \cup D_4$ can be proved by using n inference rules
  * by IH , $\phi_m(\{t'_2 t_4\} \cup D_2 \cup D_4)$ is a β-NF. Therefore $\phi_{md}(t_2 t_4) \equiv \phi_{md}(t'_2 t_4)$ is a β-NF

- *otherwise* $\Longrightarrow$ by IH and Lemma 8.4.1.

  $\phi_{md}(t_2 t_4) \equiv \{\phi_m(t_2)\phi_m(t_4)\}$, because t_2 is not a λ-abstraction. By IH for the premisses of $(x.\mathrm{app})$, both $\phi_m(t_2)$ and $\phi_m(t_4)$ are β-NFs. Therefore, we have only to show that $\phi_m(t_2)\phi_m(t_4)$ is not a β-redex:

  * by the first premiss of $(x.\mathrm{app})$ and the assumptions
    $t_1\restriction D_1 \lesssim t_2\restriction D_2$, t_1 is a β-NF and t_2 is not a λ-abstraction
  * by Lemma 8.4.1 $\phi_m(t_2)$ is not a λ-abstraction, therefore $\phi_m(t_2)\phi_m(t_4)$ is not a β-redex

- (app) $\Longrightarrow$ similar to the case $(x.\mathrm{app})$.

- $(\xi) \Longrightarrow$ by IH .

  Let $t_1\restriction D_1 \cup D \lesssim t_2\restriction D_2 \cup E$ and $D \lesssim E$ be the premisses of the last rule (i.e. (ξ)) in the proof of $M\restriction D \lesssim N\restriction E$.

  By IH for the premisses of (ξ), $\phi_m(t_2)$, $\phi_m(D_2)$ and $\phi_m(E)$ are β-NFs. Therefore, $\phi_m(N\restriction E) \equiv (\lambda x.\phi_m(t_2)\restriction\phi_m(D_2))\restriction\phi_m(E)$ is a β-NF.

- $(\eta) \Longrightarrow$

  Let $t_1\restriction D \lesssim N\restriction E$ be the premiss of the last rule (i.e. (η)) in the proof of $M\restriction D \lesssim N\restriction E$.

  By IH for the premiss of (η), $\phi_m(N\restriction E)$ is a β-NF.

- $(\eta^{op}) \Longrightarrow$

  Let $M\restriction D \lesssim t_2\restriction E$ be the premiss of the last rule (i.e. (η^{op})) in the proof of $M\restriction D \lesssim N\restriction E$.

By IH for the premiss of (η^{op}), $\phi_m(E)$ is a β-NF. Therefore, to show that $\phi_m(N\restriction E) \equiv \phi_m((\lambda y.t_2 y\restriction\{t_2 y\}))\restriction\phi_m(E)$ is a β-NF, it is enough to prove that $\phi_m((\lambda y.t_2 y\restriction\{t_2 y\}))$ is.

There are two subcases: (λ) and *otherwise*. Depending on the structure of the raw λ-term t_2:

- $(\lambda) \Longrightarrow t_2$ is a λ-abstraction, by IH .

  First we prove that $\phi_m(N\restriction E) \equiv \phi_m(t_2\restriction E)$.

  * by the inference rules
    $N\restriction E \equiv (\lambda y.t_2 y\restriction\{t_2 y\})\restriction E$
  * $y \notin \mathrm{FV}(t_2)$, by the side condition of (η^{op}). Therefore, by Lemma 8.3.6
    $N\restriction E \equiv \phi_m(t_2\restriction E)$

  by IH applied to the premiss of (η^{op}) $\phi_m(N) \equiv \phi_m(t_2)$ is a β-NF.

- *otherwise* $\Longrightarrow$ by IH and Lemma 8.4.1.

  $\phi_m((\lambda y.t_2 y\restriction\{t_2 y\})) \equiv (\lambda y.\phi_m(t_2)y\restriction\{\phi_m(t_2)y\})$, because t_2 is not a λ-abstraction. By IH for the premisses of (η^{op}), $\phi_m(t_2)$ is β-NFs. Therefore, we have only to show that $\phi_m(t_2)y$ is not a β-redex:

  * by the premiss of (η^{op}) and the assumptions
    $M\restriction D \lesssim t_2\restriction E$, M is a β-NF and t_2 is not a λ-abstraction
  * by Lemma 8.4.1 $\phi_m(t_2)$ is not a λ-abstraction, therefore
    $\phi_m(t_2)y$ is not a β-redex

∎

Example 8.4.3 Lemma 8.4.2 does not hold for ${\lesssim_{1d\eta}}^{op}$ (and ${\lesssim_{1d}}^{op}$). More precisely, there are pλ-terms $t_1\restriction D_1$ and $t_2\restriction D_2$ s.t. $t_2\restriction D_2$ is a β-NF and $t_1\restriction D_1 \lesssim_d t_2\restriction D_2$, but $t_1\restriction D_1$ has no β-NF, for instance

- $t_1\restriction D_1 \stackrel{\Delta}{\equiv} x\restriction\{\Omega\}$, where $\Omega \equiv (\lambda x.xx\restriction\{xx\})(\lambda x.xx\restriction\{xx\})$
- $t_2\restriction D_2 \stackrel{\Delta}{\equiv} x\restriction\emptyset$

Theorem 8.4.4 Characterization of $\lesssim$ for β-NFs

$t_1\restriction D_1 \lesssim t_2\restriction D_2$ in $\lambda\beta\eta$monpEQL iff $t_1\restriction D_1 \lesssim_{d\eta} t_2\restriction D_2$, when $t_1\restriction D_1$ and $t_2\restriction D_2$ are β-NFs.

Proof Since $\lesssim\ = (\rightarrow_1 \cup \rightarrow_1{}^{op} \cup \lesssim_{1d\eta})^*$ (see Theorem 8.3.10) and $\lesssim_{d\eta}\ = \lesssim^*_{1d\eta}$ (by Lemma 8.3.5), it is enough to show that the conditions in Proposition 8.1.6 hold, when R is $\rightarrow_1$, T is $\lesssim_{1d\eta}$, NF is the set of β-NFs and f is $\phi_m(\_)$.

1. $t{\restriction}D \rightarrow_1 \phi_m(t{\restriction}D)$, by the strong diamond property (see Corollary 8.2.18) and reflexivity of $\rightarrow_1$ (see Lemma 8.2.20).

   If $t_1{\restriction}D_1 \in$ NF and $t_1{\restriction}D_1 \rightarrow_1 t_2{\restriction}D_2$, then $t_2{\restriction}D_2 \equiv t_2{\restriction}D_2$. In fact, for any p$\lambda$l-term $t{\restriction}D$ if $|t{\restriction}D| \equiv t_1{\restriction}D_1$, then $t{\restriction}D \equiv t_1{\restriction}D_1$ and therefore $\phi(t{\restriction}D) \equiv t_1{\restriction}D_1$.

2. If $t_1{\restriction}D_1 \lesssim_{1d\eta} t_2{\restriction}D_2$, then $\phi_m(t_1{\restriction}D_1) \lesssim_{1d\eta} \phi_m(t_2{\restriction}D_2)$ is the First Main Lemma for $\lesssim_{d\eta}$ (see Lemma 8.3.7).

   If $t_1{\restriction}D_1 \in$ NF and $t_1{\restriction}D_1 \lesssim_{1d\eta} t_2{\restriction}D_2$, then $\phi_m(t_2{\restriction}D_2) \in$ NF is the Second Main Lemma for $\lesssim_{d\eta}$ (see Lemma 8.4.2).

3. If $t_1{\restriction}D_1 \rightarrow_1 t_2{\restriction}D_2$, then $\phi_m(t_2{\restriction}D_2) \rightarrow_1 \phi_m(t_1{\restriction}D_1)$, by the strong diamond property. In fact, $t_2{\restriction}D_2 \rightarrow_1 \phi_m(t_1{\restriction}D_1)$ and $\phi_m(t_2{\restriction}D_2) \rightarrow_1 \phi_m(t_1{\restriction}D_1)$.

   If $t_1{\restriction}D_1 \rightarrow_1 t_2{\restriction}D_2$ and $t_2{\restriction}D_2 \in$ NF, then $\phi_m(t_1{\restriction}D_1) \equiv t_2{\restriction}D_2$, by the strong diamond property. In fact, $t_2{\restriction}D_2 \rightarrow_1 \phi_m(t_1{\restriction}D_1)$ and therefore $\phi_m(t_1{\restriction}D_1) \equiv t_2{\restriction}D_2$ (by the first property above).

∎

Remark 8.4.5 Since (we claim that) $\lesssim_{d\eta}$ is decidable (see Section 7.1.2), Theorem 8.4.4 and the remarks after Proposition 8.1.6 imply that $t_1{\restriction}D_1 \lesssim t_2{\restriction}D_2$ is decidable, when both pλ-terms have β-NF (in particular when they are *typable*).

8.5 Discussion on alternatives to pλ-terms

At this point we want to discuss the other alternatives to pλ-terms. We have already said why we need $\restriction$, so the alternatives at hand are: $\lambda{\restriction}$-terms and raw pλ-terms. Both of them are easier to handle than pλ-terms, since they are defined by formation rules. For each alternative we consider the problems that arise in $\lambda\beta\eta$p-equational logic and β-reduction. In summary, β-reduction for $\lambda{\restriction}$- and raw pλ-terms is not Church-Rosser; however we conjecture that it is Church-Rosser up to d-conversion.

$\lambda\restriction$-terms. $\lambda\beta\eta$p-equational logic has to be modified:

- the inference rule ($*$-subst) is not sound. This can be fixed by changing the definition of $*$-substitution

$$(M)*[x:=N] \stackrel{\Delta}{\equiv} M[x:=N]\restriction N$$

  but $*$-substitution is no longer associative, which is really annoying

- some axioms (for $\restriction$) need to be added for retaining completeness (see [CO87])

For β-normal forms there are complications, if they ought to be invariant w.r.t. d-conversion.

Example 8.5.1 $((\lambda x.M)N)\restriction P \simeq_d ((\lambda x.M)\restriction P)N$, the l.h.s. contains a clear β-redex $(\lambda x.M)N$, which contracts to $(M)*[x:=N]$, while it is arguable whether r.h.s. has a redex. We can elaborate this example and *separate* $(\lambda x.M)$ and N as much as we like.

Any kind of Church-Rosser property is problematic to prove, because $*$-substitution is not associative.

raw pλ-terms. $\lambda\beta\eta$p-equational logic has to be modified:

- the inference rule ($*$-subst) is not sound. This can be fixed by changing the definition of $*$-substitution

$$(M\restriction D)*[x:=N\restriction E] \stackrel{\Delta}{\equiv} M[x:=N]\restriction D[x:=N] \cup \{N\} \cup E$$

- some axioms (for $\restriction$) need to be added for retaining completeness

The definition of β-redex (and β-normal form) is straightforward: $(\lambda x.M\restriction D)N$. The problem is what to do with its contractum $P\restriction E \equiv (M\restriction D)*[x:=N]$, which is a raw p$\lambda$-term. We cannot simply replace the redex with the contractum, because the first is, unlike the second, a raw λ-term. However, we can replace the redex with P and *add* E to the right of the innermost $\restriction$ containing the redex, since E may contain information about divergence **not available anywhere else**. This β-reduction is not Church-Rosser:

Example 8.5.2 $\mathrm{K}(\mathrm{I}x)$ has two β-normal forms:

- $(\lambda y.x){\restriction}x$ (if we reduce $\mathrm{I}x$ first)
- $(\lambda y.x{\restriction}x){\restriction}x$ (otherwise)

Conclusion and further research

In this thesis we have introduced many formal systems for reasoning about partial functions. All of them have a *model-theoretic justification*, i.e. they are sound and complete w.r.t. a *natural* class of models. We felt that an *operational justification*, i.e. correctness w.r.t. some operational equivalence (see [Plo75]), is too weak. In fact, the operational criterion applies only to *equational* theories (rather than formal systems) and it does not identify a unique formal system.

Most of the time we have used *first order* rather than *equational* languages, because they are more expressive and allow a natural formalization. Moreover, the equational formal systems considered in this thesis (the $\lambda_v\beta$-calculus, $\lambda\beta\eta$p- and p-equational logic) have been shown to be the restriction of a first order formal to a suitable *equational* fragment.

We have reject the view of the (partial) lambda calculus as a *logic free* calculus, instead we have produced four partial lambda calculi correct w.r.t. the call-by-value operational equivalence, but with different *pure equational theories*. We have considered both classical and intuitionistic logic, because of the remarks in [Sco79] and the categorical approach followed by Rosolini (see [Ros86]), and we have paid more attention to monotonic partial functions (as done in [Plo85]), because they are more appropriate for applications to computer science.

After studying conservative extension results among the formal systems axiomatized in first order logic, we have investigated equational presentation and reduction for the intuitionistic partial lambda calculi, in particular the $J\mathrm{mon}\lambda_p\beta\eta$-calculus.

There are various areas for further research that we briefly review.

Powerdomains and partial functions. Gordon Plotkin has reformulated most of domain theory in terms of continuous partial functions and formalized it in the $J\mathrm{mon}\lambda_p\beta\eta$-calculus (see [Plo85]): what remains to be done is to reformulate powerdomains and formalize them in intuitionistic logic. Carl Gunter has already done some work in this direction by extending the relation between operational termination and denotational existence (see [Plo85] and Theorem 5.4.8) to the Smith's powerdomain.

A promising approach would be to give a categorical characterization of powerdomains in p-categories (see [Ros86]).

Nondeterministic lambda calculus. The definition of nondeterministic lambda calculus in [Sha84] is inspired by operational considerations and it may have weaknesses, in proving equivalences of terms, similar to those pointed out for the call-by-value lambda calculus (see Theorem 6.4.5).

We suggest adding nondeterminism to the partial lambda calculus and considering the syntactic aspects of such a calculus, in particular reusing the techniques developed in the last chapter for proving the Church-Rosser property of a suitable *β-reduction* for nondeterministic partial functions (up to a suitable preorder).

It would be desirable to *justify* such a calculus model-theoretically, as done for $\lambda\beta\eta$p-equational logic, and a formalization of powerdomains in intuitionistic logic would be of great help in this.

Conjectures. There are various conjecture that we were unable to solve, we give a list of them together with references for more details:

1. we have been unable to prove completeness of the classical monotonic partial lambda calculus w.r.t. the continuous partial type hierarchy and we have made a precise conjecture about the intuitionistic monotonic partial lambda calculus (see Theorem 5.2.28 and following remark)

2. $\chi K\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash E} \chi K\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$ (see Theorem 6.3.4)

3. $FreeK\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash S} InhabitedK\lambda_{\mathrm{p}}\mathrm{Y}\beta\eta$ (see Theorem 6.4.1)

4. $\psi\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta \subset^{c}_{0\vdash Sin} \psi\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta$ and

   $J\lambda_{\mathrm{p}}\beta\eta + \mathrm{tot} \subset^{c}_{0\vdash S} K\lambda_{\mathrm{p}}\mu\mathrm{Y}\beta\eta + \mathrm{tot}$ (see remark after Theorem 6.4.1)

5. $S(J\mathrm{mon}\lambda_{\mathrm{p}}\beta\eta) \not\subseteq S(K\lambda_{\mathrm{p}}\beta\eta)$ (see remark after Theorem 6.4.5)

6. we conjecture that there is no equational presentation for the classical partial lambda calculi and extensional partial combinatory logic (see Remark 7.4.3 and Theorem 7.4.9)

7. we conjecture that λp- and $\lambda\eta$p-equational logic are decidable (see Theorem 7.3.18 and the introduction of Chapter 8). Ramos Pino Perez has recently proved this conjecture (see [Per88])

Other open problems are: an equational, rather than inequational, presentation of monotonic λ_p-algebras and finite axiomatizability results, e.g. a finite set of equational axioms for λ_p-algebras.

Revisited formal systems. In Section 3.2 we have given a cut free formal system for first order theories (either classical or intuitionistic) axiomatized by negative formulas. Related work on cut free formal systems has been carried out by David Pym for the Edinburgh Logical Framework (see [HHP87, Plo87]). It would be interesting to extend these results beyond the negative fragment (see discussion at the end of Section 3.2).

General concepts and notations

The purpose of this section is to serve as a concise reference for basic concepts and notations widely used in this thesis. The reader is supposed to be familiar with elementary concepts in Set Theory (see [Mon69]).

$\emptyset$	is the empty set
n	is the set $\{0, \ldots, n-1\}$
$\aleph_0$	is the smallest infinite cardinal
ω	is the ordinal $\{0, \ldots, n, \ldots\}$
$\mathbf{N}$	is the set of natural numbers $\{0, \ldots, n, \ldots\}$
$\mathcal{P}(X)$	is the powerset of X, i.e. the set of subsets of X
$X \subseteq_{fin} Y$	means that X is a finite subset of Y
$\mathcal{P}_{fin}(X)$	is the set of finite subsets of X
$\lvert X\rvert$	is the cardinality of X. When X is finite, $\lvert X\rvert$ is a natural number
$A - B$	is the set $\{x \in A \mid x \notin B\}$

Relations A **binary relation** R **on** X is a subset of $X \times X$. We write "$x\,R\,y$" for $\langle x, y\rangle \in R$. Moreover, when X_1 and X_2 are subsets of X, then "$X_1\,R\,X_2$" means that $x_1\,R\,x_2$ for all $x_1 \in X_1$ and $x_2 \in X_2$.

Δ_X	is the diagonal relation on X, i.e. $\{\langle x, x\rangle \mid x \in X\}$
R^{op}	is the opposite of R, i.e. the relation $\{\langle y, x\rangle \mid x\,R\,y\}$
RS	is the composition of R and S, i.e. the relation $\{\langle x, z\rangle \mid \exists y. x\,R\,y \wedge y\,S\,z\}$
R^n	is the nth iterate of R
R^*	is the reflexive and transitive closure of R
$\mathrm{dom}(R)$	is the domain of R, i.e. the set $\{x \mid \exists y. x\,R\,y\}$
$\mathrm{cod}(R)$	is the codomain of R, i.e. the set $\{y \mid \exists x. x\,R\,y\}$

Functions A **function** is a relation f s.t. if $\langle x, y\rangle \in f$ and $\langle x, z\rangle \in f$, then $y = z$. We write "$f(x)$" or "fx" for the unique y (if it exists) s.t. $\langle x, y\rangle \in f$. Moreover, when X is a subset of $\mathrm{dom}(f)$, then "$f(X)$" stands for the **image** of X, i.e. the set $\{y \mid \exists x \in X. \langle x, y\rangle \in f\}$.

Y^X or $X \to Y$	is the set of (total) functions from X to Y
$X \rightharpoonup Y$	is the set of partial functions from X to Y
$x := y$	is the function $\{\langle x, y\rangle\}$
$f \lfloor g$	is the overwriting of f by g, i.e. the function s.t. $(f \lfloor g)x$ is gx when $x \in \mathrm{dom}(g)$ and fx otherwise
id_X	is the identity on X, i.e. the diagonal relation on X
$g \circ f$	is the composition of f and g, i.e. the function s.t. $(g \circ f)x = g(fx)$

Families An I-indexed **family** is a function with domain the *index set* I. The codomain of a family is called its set of *elements*. The *cardinality* of a family is the cardinality of its index set. An I-indexed family x is also written as "$\langle x_i \mid i \in I\rangle$" where "$x_i$" stands for $x(i)$.

If X is an I-indexed family of sets, then

$\prod X = \prod_{i \in I} X_i$	is the set $\{\langle x_i \mid i \in I\rangle \mid \forall i \in I. x_i \in X_i\}$
$\cup X = \cup_{i \in I} X_i$	is the set $\{x \mid \exists i \in I. x \in X_i\}$

moreover, if f is a function with codomain I, then

X_f	is the set $\prod(X \circ f)$

If X and Y are I-indexed families of sets, then

$X \subseteq Y$	means that $X_i \subseteq Y_i$ for all $i \in I$
$F: X \to Y$	means that $F = \langle F_i \mid i \in I\rangle$ is an I-indexed family of functions s.t. $F_i: X_i \to Y_i$ for all $i \in I$
$F: X \rightharpoonup Y$	means that F is an I-indexed family s.t. $F_i: X_i \rightharpoonup Y_i$ for all $i \in I$

If F is an I-indexed family of functions from X to Y, then

$\prod F = \prod_{i \in I} F_i$	is the function from $\prod X$ to $\prod Y$ s.t. $(\prod F)\langle x_i \mid i \in I\rangle \triangleq \langle F_i(x_i) \mid i \in I\rangle$

moreover, if f is a function with codomain I, then

F_f	is the function $\prod(F \circ f)$ from X_f to Y_f

If J is a set, $R \subseteq \prod_{i \in I} X_i$ and $F\colon (\prod_{i \in I} X_i) \to Y$, then

R^J	is the subset $\{\langle f_i \mid i \in I\rangle \mid \forall j \in J. \langle f_i(j) \mid i \in I\rangle \in R\}$ of $\prod_{i \in I} X_i^J$
F^J	is the function from $\prod_{i \in I} X_i^J$ to Y^J s.t. $F^J(\langle f_i \mid i \in I\rangle) = \langle F(\langle f_i(j) \mid i \in I\rangle) \mid j \in J\rangle$

When J is clear from the context, e.g. when R or F are applied to an I-indexed family of J-indexed families, then the superscript J is dropped.

Conventions for expressions that may or may not denote The notation "fx" for function application denotes a element only when x is in the domain of f. The same problem occurs in other circumstances, e.g. when defining an element as the unique element satisfying a certain property, i.e. by **description** (see [Sco79]). The conventions adopted here for partial functions and non denoting expressions are take from [Kle52] and Chapter VI of [Bee85]. Formulae, unlike expressions for denoting elements, do always denote truth values, even when they contain expressions that may not denote.

$e \downarrow$	means that e denotes (is defined)
$e = e'$	means that both e and e' denote the same element
$e \in e'$	means that e denotes x, e' denotes X and $x \in X$
$e \simeq e'$	means that if $e \downarrow$ or $e' \downarrow$ then $e = e'$
$e(e')$	denotes y iff e denotes a function f, e' denotes an element x in the domain of f and $f(x) = y$
$\langle e_i \mid i \in I\rangle$	denotes the I-indexed family x iff e_i denotes x_i for all $i \in I$

if $e[x]$ is an expression with a *variable* x, then

$(\lambda x \in X.e[x])$	denotes the function f s.t. $\langle x, y\rangle \in f$ iff $x \in X$ and $e[x]$ denotes y

Sequences A **sequence** is a family whose index set is a natural number. The *length* of a sequence is the cardinality of its index set. a sequence $\overline{x}$ of length n is also written as "$\langle \overline{x}_1, \ldots, \overline{x}_n \rangle$".

X^*	is the set $\cup_{n \in \omega} X^n$ of sequences of elements of X
$\overline{x}\ \overline{y}$	is the concatenation of $\overline{x}$ and $\overline{y}$
$\langle x_1, \ldots, x_n \rangle$	is the sequence $\langle x_{i+1} \vert i \in n \rangle$
$X_1 \times \ldots \times X_n$	is the product $\prod \langle X_1, \ldots, X_n \rangle$
$X_1 \cup \ldots \cup X_n$	is the union $\cup \langle X_1, \ldots, X_n \rangle$

For n-ary relation we write $R(x_1, \ldots, x_n)$" for $\langle x_1, \ldots, x_n \rangle \in R$ and similarly for n-ary (partial) functions we write "$f(x_1, \ldots, x_n)$" for $f(\langle x_1, \ldots, x_n \rangle)$.

Metavariables The following conventions for metavariables are widely used in this thesis, although sometimes we may adopt different symbols. An **assignment** is a function whose domain is a finite set of *variables*[2].

		sequence	(finite) set	assignment
sort	τ	$\overline{\tau}$		η
variable	x	$\overline{x}$		
constant sym.	c	$\overline{c}$	C	γ
term	t	$\overline{t}$	D	σ
formula	A		Γ	
individual	a	$\overline{a}$		ρ
struct.	$\mathcal{A}$			
Kripke struct.	$\mathcal{B}$			

[2]the set $\mathbf{N}$ of natural numbers or any other infinite (countable) set can be taken as the set of variables, as far as it is disjoint from the other sets of symbols in a *signature*

Indexes

The **index of mathematical symbols** is hierarchically structured. At the top level there are very generic entries, e.g. "greek symbols". At the next level entries with syntactically similar form are gathered together, and they are ordered alphabetically, when it is sensible to do so. The place holder "_" may be used to indicate a missing *argument* or *parameter*.

The **index of rules and axioms**, as well as that of **formal systems**, is not hierarchical and similar symbols are gathered together.

The **index of notions** is alphabetically ordered and hierarchically structured. The place holder "_" stands for the entry immediately above in the hierarchy.

Index

Index

Index

Index

Bibliography

[Abr87] S. Abramsky. *Domain Theory and the Logic of Observable Properties.* PhD thesis, University of London, 1987.

[Acz68] P. Aczel. Saturated intuitionistic theories. In H.A. Schmidt, K. Schütte, and H.J. Thiele, editors, *Contributions to Mathematical Logic.* North Holland, 1968.

[Acz77] P. Aczel. An introduction to inductive definitions. In J. Barwise, editor, *Handbook of Mathematical Logic.* North Holland, 1977.

[AL85] A. Asperti and G. Longo. Categories of partial morphisms and the relation between type-structures. Technical Report S-7-85, Dip. di Informatica, Univ. di Pisa, 1985.

[Bar74] J. Barwise. Axioms for abstract model theory. *Annals of Mathematical Logic*, 7, 1974.

[Bar81] H.P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* North Holland, 1981.

[Bar82] H.P. Barendregt. The lambda calculus and its models. In G. Lolli, G. Longo, and A. Marcja, editors, *Logic Colloquium.* North Holland, 1982.

[Bar84] H.P. Barendregt. *The Lambda Calculus: Its Syntax and Semantics.* North Holland, 1984. revised edition.

[BBTW81] J.A. Bergstra, M. Broy, J.V. Tucker, and M. Wirsing. On the power of algebraic specification. In *Math. Found. of Comp. Sci.*, volume 118 of *Lecture Notes in Computer Science.* Springer Verlag, 1981.

[Bee85] M.J. Beeson. *Foundations of Constructive Mathematics*. Springer Verlag, 1985.

[BTC87] V. Breazu-Tannen and T. Coquand. Extensional models for polymorphism. In H. Ehrig and al., editors, *TAPSOFT '87*, volume 250 of *Lecture Notes in Computer Science*. Springer Verlag, 1987.

[BTM87] V. Breazu-Tannen and A. Meyer. Computable values can be classical. In *14th Symp. on Principles of Prog. Lang.* ACM, 1987.

[Bur82] P. Burmeister. Partial algebras - survey of an unifying approach towards a two-valued model theory for partial algebras. *Algebra Universalis*, 15, 1982.

[CO87] P-L. Curien and A. Obtulowicz. Partiality, cartesian closedness and toposes. Ecole Normale Sup., preprint, 1987.

[dPH86] R. di Paola and A. Heller. Dominical categories. *Journal of Symbolic Logic*, 1986.

[Flo67] R. Floyd. Assigning meaning to programs. In Schwartz, editor, *Proc. Symp. in Applied Math.*, 1967.

[Fou77] M.P. Fourman. The logic of topoi. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic*. North Holland, 1977.

[Fri75] H. Friedman. Equality between functionals. In R. Parikh, editor, *Logic Colloquium*, volume 453 of *Lecture Notes in Mathematics*. Springer Verlag, 1975.

[Gen69] G. Gentzen. *The Collected Papers of Gerhard Gentzen*. North Holland, 1969.

[GMW79] M.J.C. Gordon, R. Milner, and C.P. Wadsworth. *Edinburgh LCF: A Mechanized Logic of Computation*, volume 78 of *Lecture Notes in Computer Science*. Springer Verlag, 1979.

[Hen49] L. Henkin. The completeness of first-order functional calculus. *Journal of Symbolic Logic*, 14, 1949.

[HHP87] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. In *2nd Conf. on Logic in Computer Science*. IEEE, 1987.

[HN88] S. Hayashi and H. Nakano. **PX***: A Computational Logic*. MIT press, 1988.

[Hoa69] C.A.R. Hoare. An axiomatic basis for computer programming. *C. ACM*, 12, 1969.

[Hyl82] J.M.E. Hyland. The effective topos. In A.S. Troelstra and D. van Dalen, editors, *The L.E.J. Brouwer Centenary Symposium*. North Holland, 1982.

[Jac75] G. Jacopini. A condition for identifying two elements of whatever model of combinatory logic. In C. Böhm, editor, *Proceedings of the Symposium on λ-calculus and Computer Science Theory, Rome 1975*, volume 37 of *Lecture Notes in Computer Science*. Springer Verlag, 1975.

[Kle52] S.C. Kleene. *Introduction to Methamathematics*. van Nostrand, 1952.

[Klo80] J.W. Klop. *Combinatory Reduction Systems*. PhD thesis, Mathematical Center Amsterdam, 1980. Tracts 129.

[KR77] A. Kock and G.E. Reyes. Doctrines in categorical logic. In J. Barwise, editor, *Handbook of Mathematical Logic*, volume 90 of *Studies in Logic*. North Holland, 1977.

[Lam80] J. Lambek. From λ-calculus to cartesian closed categories. In R. Hindley and J. Seldin, editors, *To H.B. Curry: essays in Combinarory Logic, lambda calculus and Formalisms*. Academic Press, 1980.

[Lan64] P.J. Landin. The mechanical evaluation of expressions. *Computer Journal*, 6(4), 1964.

[LM84] G. Longo and E. Moggi. Cartesian closed categories of enumerations for effective type-structures. In G. Kahn, D. MacQueen, and G.D. Plotkin, editors, *Symp. on Semantics of Data Types*, volume 173 of *Lecture Notes in Computer Science*. Springer Verlag, 1984.

[LS86] J. Lambek and P.J. Scott. *Introduction to Higher-Order Categorical Logic*, volume 7 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, 1986.

[Mac71] S. MacLane. *Categories for the Working Mathematician*. Springer Verlag, 1971.

[McC62] J. McCarthy. *The LISP 1.5 Programmers' Manual*, 1962.

[Mey82] A. Meyer. What is a model of the lambda calculus? *Information and Computation*, 52, 1982.

[Mil77] R. Milner. Fully abstract models of typed lambda calculus. *Theoretical Computer Science*, 4, 1977.

[ML83] P. Martin-Löf. The domain interpretation of type theory. In P. Dybjer, B. Nordström, K. Petersson, and J. Smith, editors, *Workshop on the Semantics of Programming Languages*. Chalmers University of Technology, Göteborg, Sweden, 1983.

[MM87] J. Mitchell and E. Moggi. Kripke-style models for typed lambda calculus. In *2nd Conf. on Logic in Computer Science*. IEEE, 1987.

[MMMS87] A. Meyer, J. Mitchell, E. Moggi, and R. Statman. Empty types in polymorphic lambda calculus. In *14th Symp. on Principles of Prog. Lang.* ACM, 1987.

[MNS87] D. Miller, G. Nadathur, and A. Scedrov. Hereditary harrop formulas and uniform proof systems. In *2nd Conf. on Logic in Computer Science*. IEEE, 1987.

[Mog86] E. Moggi. Categories of partial morphisms and the partial lambda-calculus. In *Proceedings Workshop on Category Theory and Computer Programming, Guildford 1985*, volume 240 of *Lecture Notes in Computer Science*. Springer Verlag, 1986.

[Mon69] J.D. Monk. *Introduction to Set Theory*. Mc Graw-Hill, 1969.

[MR77] M. Makkai and G. Reyes. *First Order Categorical Logic*. Springer Verlag, 1977.

[Obt86] A. Obtulowicz. The logic of categories of partial functions and its applications, 1982 thesis. *Dissertationes Mathematicae*, 241, 1986.

[Ong88] C.-H.L. Ong. *Lazy Lambda Calculus: An Investigation into the Foundations of Functional Programming*. PhD thesis, University of London, 1988. Draft March 5, 1988.

[Pau85] L.C. Paulson. Interactive theorem proving with cambridge lcf: a user's manual. Technical Report 80, Univ. of Cambridge, Computer Laboratory, 1985.

[Per88] R.P. Perez. Decidability of the restriction equational theory in the partial lambda calculus. presented at the workshop on the typed lambda calculus, Turin, Italy, 15-20 May 1988, 1988.

[Plo75] G.D. Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1, 1975.

[Plo77] G.D. Plotkin. LCF as a programming language. *Theoretical Computer Science*, 5, 1977.

[Plo80] G.D. Plotkin. Lambda definability in the full type hierarchy. In R. Hindley and J. Seldin, editors, *To H.B. Curry: essays in Combinarory Logic, lambda calculus and Formalisms*. Academic Press, 1980.

[Plo81] G.D. Plotkin. Postgraduate lecture notes in domain theory (incorporating the "pisa notes"). Edinburgh Univ., Dept. of Comp. Sci., 1981.

[Plo82] G.D. Plotkin. Notes on completeness of the full continuous type hierarchy. Manuscript, Lab. of Comp. Sci., M.I.T., 1982.

[Plo85] G.D. Plotkin. Denotational semantics with partial functions. Lecture at C.S.L.I. Summer School, 1985.

[Plo87] G.D. Plotkin. Towards search spaces for the edinburgh logical framework. Presented at the workshop on general logic, Edinburgh, UK , 23-27 Feb. 1987, 1987.

[Pra65] D. Prawitz. *Natural Deduction.* Almquist and Wiksell, Stockholm, 1965.

[Rei87] H. Reichel. *Initial Computability, Algebraic Specifications, and Partial Algebras.* Oxford University Press, 1987.

[Ros86] G. Rosolini. *Continuity and Effectiveness in Topoi.* PhD thesis, University of Oxford, 1986.

[RR86] E. Robinson and G. Rosolini. Categories of partial maps. Quaderno 18, Dip. di Matematica, Universita' di Parma, 1986.

[Sch85] O. Schoett. Behavioural correctness of data representation. Technical Report CSR-85-185, Edinburgh Univ., Dept. of Comp. Sci., 1985.

[Sch87] O. Schoett. *Data Abstraction and the Correctness of Modular Programming.* PhD thesis, University of Edinburgh, 1987.

[Sco67] D.S. Scott. Existence and description in formal logic. In Schoenmann, editor, *Bertrand Russell: Philosopher of the Century.* Allen and Unwin, 1967.

[Sco69] D.S. Scott. A type-theoretic alternative to CUCH, ISWIM, OWHY. Oxford notes, 1969.

[Sco70] D.S. Scott. Outline of a mathematical theory of computation. Technical Report PRG-2, Oxford Univ. Computing Lab., 1970.

[Sco76] D.S. Scott. Data types as lattices. *SIAM Journal of Computing*, 5, 1976.

[Sco79] D.S. Scott. Identity and existence in intuitionistic logic. In M.P. Fourman, C.J. Mulvey, and D.S. Scott, editors, *Applications of Sheaves*, volume 753 of *Lecture Notes in Mathematics.* Springer Verlag, 1979.

[Sco80] D.S. Scott. Relating theories of the λ-calculus. In R. Hindley and J. Seldin, editors, *To H.B. Curry: essays in Combinarory Logic, lambda calculus and Formalisms.* Academic Press, 1980.

[Sco82] D.S. Scott. Domains for denotational sementics. In M. Nielsen and E.M. Schimdt, editors, *Automata, Languages and Programming, Ninth Colloquium*, volume 140 of *Lecture Notes in Computer Science*. Springer Verlag, 1982.

[SH84] P Schröder-Heister. A natual extension of natural deduction. *Journal of Symbolic Logic*, 49, 1984.

[Sha84] K. Sharma. Syntactic aspects of the non-deterministic lambda calculus. Master's thesis, Washington State University, September 1984. available as internal report CS-84-127 of the comp. sci. dept.

[SS71] D.S. Scott and C. Strachey. Toward a mathematical semantics for computer languages. Technical Report PRG-6, Oxford Univ. Computing Lab., 1971.

[Sta83] R. Statman. Equality between funcionals revisited, 1983.

[Sta85a] R. Statman. Lecture notes on the simple typed λ-calculus. CMU, 1985.

[Sta85b] R. Statman. Logical relations and the typed lambda calculus. *Information and Computation*, 65, 1985.

[Sta86] R. Statman. Horizontal and vertical polymorphic equations. Private communication whose main result was announced in TYPES mailing list on Jan the 19th 1987 "Maximal theories in polymorphic lambda calculus", 1986.

[Sto86] A. Stoughton. *Fully Abstract Models of Programming Languages*. PhD thesis, University of Edinburgh, 1986.

[Sto88] A. Stoughton. *Fully Abstract Models of Programming Languages*. Research Notes in Theoretical Computer Science. Pitman, London, 1988.

[Ten78] N.W. Tennent. *Natural Logic*. Edinburgh University Press, 1978.

[Tro73] A.S. Troelstra, editor. *Metamathematical Investigations of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Computer Science*. Springer Verlag, 1973.

[vD86] D. van Dalen. Intuitionistic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*, volume III. D. Reidel, Dordrecht, 1986.

[Wad76] C. Wadsworth. The relation between computational and denotational properties for scott's D_∞-models of the lambda calculus. *SIAM Journal of Computing*, 5, 1976.

[WL84] G. Winskel and K. Larsen. Using information systems to solve recursive domain equations effectively. In G. Kahn, D. MacQueen, and G.D. Plotkin, editors, *Symp. on Semantics of Data Types*, volume 173 of *Lecture Notes in Computer Science*. Springer Verlag, 1984.