A Generalisation of Pre-Logical Predicates and Its Applications

Shin-ya Katsumata

Doctor of Philosophy

Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh

2004

Abstract

This thesis proposes a generalisation of pre-logical predicates to simply typed formal systems and their categorical models. We analyse the three elements involved in pre-logical predicates — syntax, semantics and predicates — within a categorical framework for typed binding syntax and semantics. We then formulate generalised pre-logical predicates and show two distinguishing properties: a) equivalence with the basic lemma and b) closure of binary pre-logical relations under relational composition.

To test the adequacy of this generalisation, we derive pre-logical predicates for various calculi and their categorical models including variations of lambda calculi and non-lambda calculi such as many-sorted algebras as well as first-order logic. We then apply generalised pre-logical predicates to characterising behavioural equivalence. Examples of constructive data refinement of typed formal systems are shown, where behavioural equivalence plays a crucial role in achieving data abstraction.

Acknowledgements

First of all, I thank my supervisor Don Sannella for all aspects of my Ph.D. study. This thesis would not be here without his continuous support and encouragement. His careful reading and comments are always valuable for improving my thesis.

I thank Atsushi Ohori who encouraged me to study abroad, and Masahito Hasegawa who helped me to start a new life in Edinburgh.

During my Ph.D. study in Edinburgh, I had a lot of opportunities to discuss various topics in computer science. Many thanks to John Longley, Furio Honsell, John Power, Ian Stark, Alex Simpson, Miki Tanaka, Misao Nagayama and Samuel Lindley for intellectual stimulation and guidance. Daniel Turi's insightful comments contributed to improve this thesis.

I would like to thank Martin Dicks and Nanako Dicks for the best friendship in Edinburgh. Particularly, I am very grateful to them and Jon Cook for offering temporary accomodaion when I had a difficulty in finding a new flat after leaving Mylnes Court. Finally, special thanks to Louise for her support and endurance during the final stages of writing.

My Ph.D. study was funded for three years by LFCS studentship and Nihon Ikueikai studentship.

To my parents

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. The material in chapter 3 and 4 are the extended version of [Kat04] published by the author.

(*author*)

Table of Contents

Chapter 1

Introduction

1.1 Overview

In the study of the semantics of programming languages, constructing submodels and relational models is a widely acknowledged method to show properties and compare models of a language. For example, in the field of algebraic specification, the equivalence of models ”up to observation” can be captured well by relations between two algebras, rather than homomorphisms. That is, there are algebras $\mathcal{A}$ and $\mathcal{B}$ representing sets of integers that have the same externally visible behaviour where there is neither a homomorphism $\mathcal{A} \to \mathcal{B}$ nor $\mathcal{B} \to \mathcal{A}$; however there is a homomorphic *relation* called a correspondence [Sho83, Sch90] which witnesses their similarity. In the simply typed lambda calculus, there is a construction method called *logical predicates*, which constructs a part of a model by induction on types. Thanks to its simplicity, logical predicates are widely applied to show various properties of the lambda calculus, such as the strong normalisation theorem of the simply typed lambda calculus, characterisation of $\beta\eta$-equality, computational adequacy result, etc.

Once you construct a part of a model, you would like to show that it is a submodel. However this is sometimes difficult, depending on the way it is constructed. Therefore it is desirable to have an equivalent characterisation of submodels. *Pre-logical predicates* serve this purpose; they provide an equivalent characterisation of submodels of the set-theoretic models of the simply typed lambda calculus.

There are two differences between logical predicates and pre-logical predicates. First, logical predicates are more like a concrete method to construct a submodel, while pre-logical predicates capture the whole class of submodels. Thus logical predicates are pre-logical predicates, but not vice versa. Second, it is known that binary logical relations are not closed under relational composition, while binary pre-logical relations are (here a binary logical relation between two models is just a logical predicate on their product model, and similarly for binary pre-logical relations). By this property, binary pre-logical relations can characterise the equivalence of models of the lambda calculus up to observation, in the same way as we can characterise such equivalence for algebras. Furthermore, pre-logical predicates for certain class of models of the simply typed lambda calculus have a simple algebraic characterisation by means of combinators. This provides a convenient method to construct pre-logical predicates.

The goal of this thesis is to propose a generalisation of pre-logical predicates, not only to the extensions of the simply typed lambda calculus, but also for *any* typed calculus with variable binding — we call them *typed formal systems*.

Typed formal systems are a generic name for type systems whose typing rules match the following scheme:

$$\frac{\Gamma, \overrightarrow{x_1 : \tau_1} \vdash M_1 : \tau_1 \quad \cdots \quad \Gamma, \overrightarrow{x_n : \tau_n} \vdash M_n : \tau_n}{\Gamma \vdash o(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n) : \tau}$$

This scheme expresses an inference rule of a term construct o which behaves as a binder of $\overrightarrow{x_i}$ in M_i for each $1 \leq i \leq n$. Examples of typing rules which match the above scheme are ordinary operators in many-sorted signatures (where no binding takes place), lambda abstraction in the simply typed lambda calculus, the case syntax and let binding in extensions of the lambda calculus, universal and existential quantifiers in first-order logic, the name restriction operator in pi calculus, and more.

Therefore our generalisation covers all of the above calculi, that is, the extensions of the simply typed lambda calculus and the non-lambda calculi such as logics and process calculi. The key observation of this generalisation is that the notion of submodel is not special to the simply typed lambda calculus; we can talk about submodels of logics, programming languages, process calculi, etc. Our generalisation stems from this observation.

We move to looking in more detail at several key topics: logical predicates, pre-logical predicates and behavioural equivalence.

Logical Predicates

The origin of logical predicates is the technique used by Tait to prove the strong normalisation theorem [Tai67]. The definition of the strongly normalising terms is easy, but it is not obvious to prove that all the terms are strongly normalising, since naive induction on the structure of terms does not work. To overcome this difficulty, Tait constructed a subset of the set of simply typed lambda terms by induction on the structure of types. Instead of his proof, in the following we show a reformulated and sketchy version of his proof [HS86, GLT88]. Tait essentially constructed the following type-indexed family of subsets of typed terms:

$$\begin{aligned} R^b &= \{M \mid \exists \Gamma \,.\, \Gamma \vdash M : b \wedge M \text{ is strongly normalising}\} \quad (b \in B) \\ R^{\tau \Rightarrow \tau'} &= \{M \mid \exists \Gamma \,.\, \Gamma \vdash M : \tau \Rightarrow \tau' \wedge \forall N \in R^{\tau} \,.\, MN \in R^{\tau'}\}. \end{aligned}$$

where B is the set of base types. The core idea of this construction is that $R^{\tau \Rightarrow \tau'}$ is determined by R^{τ} and $R^{\tau'}$. Tait's proof continues as follows; the two milestones of Tait's proof are then the following.

- He showed that for any type τ, R^{τ} is included in the set of strongly normalising terms of type τ. This shows that the definition of R successfully captures a subset of the set of strongly normalising terms. We note that variables of type τ are also included in R^{τ}.

- He then showed that R is closed under term substitution: for any terms $M_1 \in R^{\tau_1}, \cdots, M_n \in R^{\tau_n}$ and term $x_1 : \tau_1, \cdots, x_n : \tau_n \vdash M : \tau$, we have

$$M[M_1/x_1, \cdots, M_n/x_n] \in R^{\tau}.$$

The theorem is a corollary of the second statement, by simply letting $M_1, \cdots, M_n$ be variables.

His construction was considered in a semantic framework in [Plo80, Sta85]. Below we introduce logical predicates for set-theoretic models of the simply typed lambda

calculus. We write $\mathbf{Typ}^{\Rightarrow}(B)$ for the set of types defined by the BNF $\tau ::= b \mid \tau \Rightarrow \tau$ where b ranges over B. A set-theoretic model $\mathcal{A}$ of the simply typed lambda calculus consists of a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of carrier sets $\{A^\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$, application operators $\bullet^{\tau,\tau'} : A^{\tau \Rightarrow \tau'} \times A^\tau \to A^{\tau'}$ for each $\tau, \tau' \in \mathbf{Typ}^{\Rightarrow}(B)$ and a meaning function $\mathcal{A}[\![-]\!]$, which maps a well-formed lambda term $\Gamma \vdash M : \tau$ and a Γ-environment ρ to an element $\mathcal{A}[\![M]\!]\rho$ in A^τ, where a Γ-environment is an assignment of a value to each variable in Γ that respects types. A *logical predicate* in general form is a *family of subsets* $\{P^\tau \subseteq A^\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$ such that the subset at a higher type is given by *exponentiation* with respect to the application operator:

$$P^{\tau \Rightarrow \tau'} = \{f \in A^{\tau \Rightarrow \tau'} \mid \forall x \in P^\tau \,.\, f \bullet^{\tau,\tau'} x \in P^{\tau'}\}. \tag{1.1}$$

Another way to view logical predicates is that they provide a method to *extend* a family of subsets $\{P^b \subseteq A^b\}_{b \in B}$ for base types to those for higher-order types.

Any logical predicate is large enough to interpret the simply typed lambda calculus. This proposition is referred to as the *basic lemma* or the *fundamental lemma of logical predicates*. [1] Formally, it is the following proposition:

$$\begin{aligned} &\forall x_1 : \tau_1, \cdots, x_n : \tau_n \vdash M : \tau \,. \\ &\quad \forall v_1 \in P^{\tau_1}, \cdots, v_n \in P^{\tau_n} \,. \\ &\qquad \mathcal{A}[\![M]\!]\{x_1 \mapsto v_1, \cdots, x_n \mapsto v_n\} \in P^\tau \end{aligned} \tag{1.2}$$

Statement 1.2 is saying nothing other than that the logical predicate forms a *submodel* of the simply typed lambda calculus.

Since Tait's proof, logical predicates have been extensively applied to the study of properties of the simply typed lambda calculus. Here is an incomplete list of applications of logical predicates:

- Friedman showed that $\beta\eta$-equality is characterised by the full type hierarchy over an infinite set for a base type [Fri73].
- Sieber's definability result up to rank 2 [Sie92].

[1] When the lambda calculus has (possibly higher-order) constants, to show the basic lemma, for each constant of type τ, we need to show that its meaning is included in P^τ.

- Mitchell's representation independence result [Mit86].
- Plotkin's computational adequacy result for PCF [Plo77].

Besides the application of logical predicates, there have been several researches to understand logical predicates in a category theoretic way. In the categorical semantics of the simply typed lambda calculus, the carrier sets are replaced with objects. However, the notion of predicate, which was just subsets in set-theoretic models, is more subtle. The pioneering works on categorical generalisation of logical predicates are [MS93],[MR92] and [Her93].

Before we move to reviewing these works, we quickly recall the categorical semantics of the simply typed lambda calculus. In category theoretic terms, an interpretation of the simply typed lambda calculus in a Cartesian closed category (CCC) $\mathbb{C}$ is expressed by a functor M from the free CCC $\mathbf{L}$ generated from a set of base types B to a CCC $\mathbb{C}$ preserving finite products and exponentials strictly (see e.g. [Cro94]).

In [MS93], Mitchell and Scedrov formulated predicates as subsets of global elements of objects:

$$P \text{ is a predicate over an object } C \text{ in } \mathbb{C} \iff P \subseteq \mathbb{C}(1, C).$$

This is a reasonable generalisation of the notion of predicate, since when $\mathbb{C} = \mathbf{Sets}$, the notion of predicate coincides with subsets. They then constructed the category $\tilde{\mathbb{C}}$ of predicates over $\mathbb{C}$ which they called *sconing*, together with a projection functor $p : \tilde{\mathbb{C}} \to \mathbb{C}$. They showed that $\tilde{\mathbb{C}}$ is a CCC, exponential objects in $\tilde{\mathbb{C}}$ corresponds to the exponentiation (1.1) of predicates and p strictly preserves the CCC structure. They then showed that giving a logical predicate over an interpretation $M : \mathbf{L} \to \mathbb{C}$ is equivalent to giving a functor $P : \mathbf{L} \to \tilde{\mathbb{C}}$ such that $P \circ p = M$.

In [MR92], Ma and Reynolds gave a more general formulation of the notion of predicate. They considered two CCCs $\mathbb{C}$ and $\mathbb{D}$ linked by a finite product preserving functor $G : \mathbb{C} \to \mathbb{D}$, then formulated a predicate by the following mono from some object X:

$$P \text{ is a predicate over } C \iff P : X \rightarrowtail GC.$$

Mitchell and Scedrov's formulation is the special case where $\mathbb{D} = \mathbf{Sets}$ and $G = \mathbb{C}(1, -)$. Ma and Reynolds constructed the category of predicates $Rel(\mathbb{C}, \mathbb{D}, G)$ with

a projection functor $p : Rel(\mathbb{C}, \mathbb{D}, G) \to \mathbb{C}$, and showed that $Rel(\mathbb{C}, \mathbb{D}, G)$ is a CCC and p preserves finite products and exponentials when $\mathbb{D}$ has pullbacks. Following Mitchell and Scedrov, they showed that giving a logical predicate over an interpretation $M : \mathbf{L} \to \mathbb{C}$ is equivalent to giving a functor $P : \mathbf{L} \to \mathbf{Rel}(\mathbb{C}, \mathbb{D}, G)$ such that $P \circ p = M$.

In [Her93], Hermida further generalised Ma and Reynolds' work using fibrations. A fibration is a functor $p : \mathbb{E} \to \mathbb{C}$ having the so-called *Cartesian lifting property*. One way to view a fibration is that it expresses a situation where a category $\mathbb{C}$ is equipped with a category $\mathbb{E}$ of predicates on the objects in $\mathbb{C}$. The notion of predicate in this setting is:

$$P \text{ is a predicate over } C \iff pP = C.$$

Hermida showed that certain structures over p and $\mathbb{C}$ yield a CCC structure over $\mathbb{E}$ which is strictly preserved by p. Exponential objects are expressed by exponentiation (1.1) in the internal logic of fibrations. Ma and Reynolds' formulation is then an instance of Hermida's theory for a fibration $p : Rel(\mathbb{C}, \mathbb{D}, G) \to \mathbb{C}$.

The benefit of these categorical generalisations is that we can consider logical predicates in more abstract settings. For example, Kripke semantics of the simply typed lambda calculus [MM91] has a natural description in terms of category theory. By applying the above categorical formulations, we obtain a natural construction of logical predicates over the Kripke semantics of the simply typed lambda calculus. This provides fruitful results [JT93, PR00].

Another direction of the evolution of logical predicates concerns the extension of the simply typed lambda calculus with various type constructors. In [FS99], Fiore and Simpson considered Grothendieck logical predicates over the simply typed lambda calculus with stable sums, and showed the strong normalisation theorem. Girard's strong normalisation theorem of System F uses an extension of logical predicates to System F (see e.g. [GLT88]). Binary logical relations for System F play a crucial role in define the concept of parametricity advocated by Reynolds [MR92]. Logical relations have also been considered in Moggi's computational metalanguage [Mog91]; see [GLLN02] and [LS05, Lin04]. There are of course many other applications and extensions of logical predicates which are not listed here.

Pre-Logical Predicates

In [HS02], Honsell and Sannella proposed *pre-logical predicates* as a weakening of logical predicates. A pre-logical predicate is a type-indexed family of subsets $\{P^\tau \subseteq A^\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$ satisfyiing

- for each $x \in P^{\tau \Rightarrow \tau'}$ and $y \in P^\tau$, $x \bullet y \in P^{\tau'}$ holds and
- for each term $x_1 : \tau_1, \cdots, x_n : \tau_n, x : \tau \vdash M : \tau'$ and values $v_1 \in P^{\tau_1}, \cdots, v_n \in P^{\tau_n}$,

  $$\forall v \in P^\tau \,.\, \mathcal{A}[\![M]\!]\{x_1 \mapsto v_1, \cdots, x_n \mapsto v_n, x \mapsto v\} \in P^{\tau'}$$

  implies

  $$\mathcal{A}[\![\lambda x^\tau \,.\, M]\!]\{x_1 \mapsto v_1, \cdots, x_n \mapsto v_n\} \in P^{\tau \Rightarrow \tau'}.$$

The first condition is equivalent to the left-to-right inclusion in (1.1). The right-to-left inclusion is weakened to the second condition, which is necessary to show that any pre-logical predicate satisfies (1.2) (the basic lemma of pre-logical predicates). The converse is also true; any family of subsets $\{P^\tau \subseteq A^\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$ satisfying (1.2) is pre-logical. This equivalence is unique to pre-logical predicates, since logical predicates in general imply (1.2) but not the other way around.

The type-wise relational composition of two binary pre-logical relations is again a binary pre-logical relation. This property, which does not hold for logical relations in general, is appropriate for characterising observational equivalence [2] and data refinement. In [HS02] Honsell and Sannella showed that pre-logical predicates can characterise observational equivalence in terms of the existence of a binary pre-logical relation which is a partial injection at observable types.

The construction of logical predicates are type-directed. On the other hand, there is a syntax-directed construction of the *least* pre-logical predicates extending a given type-indexed family of subsets of the carrier sets. In particular, the least pre-logical extension of the empty predicate yields the predicate which consists of values that are definable in a model.

[2] There is a possible terminological confusion: some people use observational equivalence to refer to contextual equivalence, while in this context we mean the equivalence relation between two models.

Behavioural Equivalence and Data Refinement

Behavioural equivalence arose in the study of *abstract data types*. An abstract data type is a type whose internal representation is hidden from programmers. Programmers can only access values of the abstract data type via operators associated with it, but cannot inspect their internal representation. The opposite of abstract data types is *observable data types*; programmers can freely touch, create and inspect the values of observable data types. Abstract data types are now widely recognised as an important concept for modular programming of large systems.

This distinction of types restricts programmers' knowledge about programming environments to the observable parts. Thus there are implementations in which programs over observable types show the same behaviour, despite the fact that they realise abstract data types in different ways. From the programmer's viewpoint, such "observationally" equivalent implementations realise the same abstract data types.

The equivalence relation between models "up to observation" was formulated in the field of universal algebra [ST87], where it is called *observational equivalence* or *behavioural equivalence*. Schoett characterised behavioural equivalence in terms of the existence of a correspondence, which is a homomorphic relation between two algebras [Sch85, Sch90]. Mitchell used a similar idea to show representation independence in the simply typed lambda calculus in [Mit86]. He showed that certain binary logical relations can characterise observational equivalence, provided that the underlying signature for the abstract data types in question have at most rank 1 (that is, they cannot take functions as arguments). This restriction on rank is later removed by [HS02], in which Honsell and Sannella used binary pre-logical relations instead of binary logical relations.

1.2 Structure of this Thesis

The goal of this thesis is to propose an extension of pre-logical predicates from the simply typed lambda calculi and their set-theoretic models to a class of simple type systems (called *simply typed formal systems*) and their categorical models. We then re-construct the behavioural theory of models and formulate data refinement for sim-

ply typed formal systems by means of pre-logical relations. This reconstruction is a strict generalisation of the traditional many-sorted case. Toward this goal, this thesis is organised as follows.

The development of our generalisation of pre-logical predicates relies on fibred category theory and the internal logic of fibrations. We devote chapter 2 to preliminary on fibred category theory. This chapter does not contain anything new, and a reader who is familiar with this topic can skip it.

We propose the generalisation of pre-logical predicates in chapter 3. There are three underlying elements on which pre-logical predicates are defined: syntax (the simply typed lambda calculus), semantics (set-theoretic environmental models) and predicates (as subsets of carrier sets). These three elements are generalised, and are expressed in the category of *presentation models* of [MS03]. We then formulate what it means for a predicate to satisfy the basic lemma and define pre-logical predicates, and show their equivalence. We then show that binary pre-logical relations are closed under (a categorical generalisation of) relational composition, and that the least pre-logical extension of a given predicate can be explicitly constructed, provided that the semantic category satisfies certain conditions.

In the following chapter, we examine our generalisation of pre-logical predicates by means of several examples, such as the case of traditional many-sorted algebras, the formal equivalence between lax logical predicates and pre-logical predicates for the simply typed lambda calculus with finite products, Moggi's computational meta-language [Mog91], and first-order logic.

We then move to an application of pre-logical predicates (chapter 5). First we show that pre-logical predicates can characterise behavioural equivalence. Next, we introduce the concept of *indistinguishability*, which is another approach to achieving data abstraction. We show that behavioural equivalence is factorisable by the indistinguishability relation, that is, it characterises behavioural equivalence in terms of the isomorphism between models quotiented by the indistinguishability relation.

Behavioural equivalence plays an essential role in data refinement. We apply our characterisation theorem of behavioural equivalence by means of binary pre-logical relations to develop a theory of constructive data refinement for typed formal systems,

called *pre-logical refinement*. We see two examples of pre-logical refinements; one is the standard example of implementing finite sets of elements by finite lists, and the other is implementing the lambda calculus by a combinatory algebra.

1.3 Notational Conventions

The vector notation $\overrightarrow{\text{expression}}$ represents a finite sequence of the expression whose metavariables are indexed by $1, 2, 3, \cdots$. For example, when x and τ are reserved for metavariables ranging over object variables and types of a language, $\overrightarrow{x : \tau}$ represents the sequence $x_1 : \tau_1, x_2 : \tau_2, \cdots, x_n : \tau_n$ for some length n, which is referred to by $|\overrightarrow{x : \tau}|$. The meaning of this sequence depends on the context; for example, $\lambda \overrightarrow{x} \,.\, M$ expands to $\lambda x_1, \cdots, x_n \,.\, M$, which usually means the lambda term $\lambda x_1 \,.\, \cdots \lambda x_n \,.\, M$, while $(\overrightarrow{M})$ means a tuple $(M_1, \cdots, M_n)$.

For a finite set A, by $\#A$ we mean the number of elements contained in A.

Chapter 2

Preliminaries

2.1 Contexts

Throughout this thesis we fix a countably infinite set of variables $\mathbf{V}$ ranged over by x, y, z, w. Let T be the set of simple types (ranged over by Greek letters τ and σ). A *T-context* Γ is a function from a finite subset of $\mathbf{V}$ to T. When T is obvious from the context, we just say "a context Γ". We assume an enumeration of variables given by a bijection $\mathbf{v} : \mathbf{N} \to \mathbf{V}$ and linear ordering $\leq_{\mathbf{v}}$ over $\mathbf{V}$ defined by $x \leq_{\mathbf{v}} y \iff \mathbf{v}^{-1}(x) \leq \mathbf{v}^{-1}(y)$. We just write $\mathbf{v}_i$ instead of $\mathbf{v}(i)$. The purpose of introducing this enumeration is to give a precise interpretation of contexts. Later in this thesis we assign to a context Γ an interpretation by the product object $\prod_{x_i \in \mathrm{dom}(\Gamma)} A(\Gamma(x_i))$ in a Cartesian category. When taking this product object, we need to specify an order for the variables in $\mathrm{dom}(\Gamma)$. To give such an order for a context Γ, we define an indexing function (which will be written by the small letter of the context) $\gamma : \mathrm{dom}(\Gamma) \to \{1, \cdots, \#\mathrm{dom}(\Gamma)\}$ by $\gamma(x) = i$ if x is the i-th smallest variable in $\mathrm{dom}(\Gamma)$ by $\leq_{\mathbf{v}}$. With this indexing function, we take variables in $\mathrm{dom}(\Gamma)$ from $\gamma^{-1}(1)$ to $\gamma^{-1}(\#\Gamma)$ to obtain the above product object. We note that this is just a design choice in this thesis to resolve the above problem. Another solution is to use de-Bruijn indexing, but we prefer to use variables for convenience and readability. We identify a sequence of types $\tau_1, \cdots, \tau_n$ and a context $\{\mathbf{v}_1 \mapsto \tau_1, \cdots, \mathbf{v}_n \mapsto \tau_n\}$.

2.2 Category Theory

We assume that readers have basic knowledge of category theory. Good references are [Mac71, LS86, Cro94, Bor94] and many other textbooks.

We use letters $\mathbb{C}, \mathbb{D}$ etc. to range over categories. The categories we consider in this thesis are at most locally small. We identify a set and its discrete category. When a category has certain structures, such as limits, colimits, exponentials, etc, we always talk about *specified* structure. We adopt the following notations:

$\mathbf{Obj}(\mathbb{C}), \mathbb{C}(X, Y)$	The collection of objects in $\mathbb{C}$ and the hom-set
$\mathbf{Sets}$	The category of sets
$\mathrm{dom}(f), \mathrm{cod}(f)$	The domain and codomain of a morphism f
$X_1 \times \cdots \times X_n, \prod_{i \in I} X_i$	Products
$\pi, \pi', \{\pi_i\}_{i \in I}, \langle -, - \rangle$	Projections and tupling
$X_1 + \cdots + X_n, \coprod_{i \in I} X_i$	Coproducts
$\iota, \iota', \{\iota_i\}_{i \in I}, [-, -]$	Injections and cotupling
$X \Rightarrow Y$	An exponential object
$\lambda(f), @$	A currying of a morphism f and an evaluation map in a CCC

2.3 Fibred Category Theory

Fibration

We use fibrations as a categorical formulation of the situation when a category is equipped with a notion of predicates. The material in this section and the next section is mainly taken from [Jac99]. Let $\mathbb{E}, \mathbb{B}$ be categories and $p : \mathbb{E} \to \mathbb{B}$ be a functor.

Definition 2.3.1 An object X in $\mathbb{E}$ is *above* an object I in $\mathbb{B}$ if $pX = I$. A morphism $u : X \to Y$ in $\mathbb{E}$ is *above* $f : I \to J$ in $\mathbb{B}$ if $pu = f$. A morphism $u : X \to Y$ is *vertical* if it is above an identity morphism. For objects X and Y in $\mathbb{E}$ and a morphism $f : pX \to pY$ in $\mathbb{B}$, we define $\mathbb{E}_f(X, Y) = \{u \in \mathbb{E}(X, Y) \mid pu = f\}$. □

Definition 2.3.2 We define a *fibre category* $\mathbb{E}_I$ over an object I in $\mathbb{B}$ by the following data:

An object in $\mathbb{E}_I$ is an object X in $\mathbb{E}$ above I.

A morphism in $\mathbb{E}_I$ is a morphism $f \in \mathbb{E}_{\mathrm{id}_I}(X, Y)$. □

Definition 2.3.3 ([Jac99], definition 1.1.3 and definition 1.4.3) A morphism $u : X \to Y$ in $\mathbb{E}$ is *Cartesian* above a morphism $f : I \to J$ in $\mathbb{B}$ if $pu = f$ and for any morphisms $v : Z \to Y$ in $\mathbb{E}$ and $g : pZ \to I$ in $\mathbb{B}$ such that $f \circ g = pv$, there exists a unique morphism $w : Z \to X$ above g such that $u \circ w = v$.

We say that a functor $p : \mathbb{E} \to \mathbb{B}$ is a *fibration* if for each pair of an object X in $\mathbb{E}$ and a morphism $f : I \to pX$ in $\mathbb{B}$, there exists an object Y in $\mathbb{E}$ and a Cartesian morphism $u : Y \to X$ above f called the *Cartesian lifting* of (X, f). $\mathbb{E}$ is called the *total category* and $\mathbb{B}$ is called the *base category* of fibration p.

A choice of a Cartesian lifting for each (X, f) is called a *cleavage* (on p), and the choice is denoted by $\overline{f}X$. A fibration with a cleavage is called a *cloven fibration*. The mapping $f^* : X \mapsto \mathrm{dom}(\overline{f}X)$ induces a *reindexing functor* $f^* : \mathbb{E}_J \to \mathbb{E}_I$; it sends a morphism $u : X \to Y$ in $\mathbb{E}_J$ to the morphism $v : f^*X \to f^*Y$ obtained by the universal property of the Cartesian morphism $\overline{f}Y$:

$$\begin{array}{ccc} f^*X & \xrightarrow{\overline{f}X} & X \\ {\scriptstyle v}\vdots & & \downarrow{\scriptstyle u} \\ f^*Y & \xrightarrow[\overline{f}Y]{} & Y \end{array}$$

For any morphisms $f : I \to J$ and $g : J \to K$ in $\mathbb{B}$, we have natural isomorphisms $(g \circ f)^* \cong f^* \circ g^*$ and $\mathrm{id}_I^* \cong \mathrm{Id}_{\mathbb{E}_I}$ satisfying certain coherence conditions (see [Jac99]). When they are identities, the cleavage is called a *splitting* and a fibration with a splitting is called a *split fibration*.

A fibration p is *preordered* (*partially ordered*) if each fibre category is a preorder (partial order). A fibration p is preordered if and only if p is faithful. Let $p : \mathbb{E} \to \mathbb{B}$ be a preordered fibration, X, Y be objects in $\mathbb{E}$ and $f : pX \to pY$ be a morphism in $\mathbb{B}$. We write $f : X \to Y$ if there exists a unique morphism from X to Y above f. Note that a partially ordered fibration is always split (see [Jac99], exercise 1.4.5). □

Proposition 2.3.4 *Let $p : \mathbb{E} \to \mathbb{B}$ be a cloven fibration, $f : I \to J$ and $g : J \to K$ be morphisms in $\mathbb{B}$, X and Y be objects in $\mathbb{E}$ above I and K. We have an isomorphism:*

$$h : \mathbb{E}_{g \circ f}(X, Y) \cong \mathbb{E}_f(X, g^* Y).$$

(thus $\mathbb{E}_f(X, Y) \cong \mathbb{E}_I(X, f^* Y)$ for any morphism $f : I \to J$ in $\mathbb{B}$). □

PROOF From the universal property of Cartesian lifting, for a morphism $u : X \to Y$ in $\mathbb{E}$ above $g \circ f$, there exists a unique morphism $v : X \to f^* Y$ in $\mathbb{E}$ above g such that $\overline{f}Y \circ v = u$. Thus we define $h(u) = v$. On the other hand, for a morphism $v : X \to f^* Y$ in $\mathbb{E}$, we define $h^{-1}(v)$ by $h^{-1}(v) = \overline{f}Y \circ v$. It is easy to see that they form an isomorphism. ■

We can create a new fibration by taking a pullback of a fibration along some functor. This is called a *change-of-base* of the fibration, and provides a convenient way to construct a fibration over some category.

Theorem 2.3.5 ([Jac99], lemma 1.5.1) *Let $p : \mathbb{E} \to \mathbb{B}$ be a (cloven, split, preordered, partially ordered) fibration and $G : \mathbb{C} \to \mathbb{B}$ be a functor. Then we take the pullback of p along G as follows:*

$$\begin{array}{ccc} G^*\mathbb{E} & \longrightarrow & \mathbb{E} \\ {\scriptstyle G^*p}\downarrow & \lrcorner & \downarrow{\scriptstyle p} \\ \mathbb{C} & \xrightarrow[G]{} & \mathbb{B} \end{array}$$

The functor G^*p obtained by the pullback is again a (cloven, split, preordered, partially ordered) fibration. □

PROOF The category $G^*\mathbb{E}$ is defined by the following data.

An object in $G^*\mathbb{E}$ is a pair (X, C) such that X is an object in $\mathbb{E}$ above GC.

A morphism in $G^*\mathbb{E}$ from (X, C) to (Y, D) is a pair (u, f) such that $u : X \to Y$ is a morphism in $\mathbb{E}$ and $f : C \to D$ is a morphism in $\mathbb{C}$ and u is above Gf.

Let $f : C \to D$ be a morphism in $\mathbb{C}$ and (X, D) be an object in $G^*\mathbb{E}$. We define the Cartesian lifting of f by $(\overline{Gf}X, f) : ((Gf)^* X, C) \to (X, D)$. We leave readers to check that this is indeed a Cartesian morphism. ■

Op-Fibration

We introduce the dual of Cartesian morphisms and fibration.

Definition 2.3.6 ([Jac99], definition 9.1.1) A morphism $u : X \to Y$ in $\mathbb{E}$ above a morphism $f : I \to J$ in $\mathbb{B}$ is *op-Cartesian* if u is Cartesian above f for $p : \mathbb{E}^{op} \to \mathbb{B}^{op}$. In other words, $pu = f$ and for any morphism $v : X \to Z$ in $\mathbb{E}$ and $g : J \to pZ$ in $\mathbb{B}$ such that $g \circ f = pv$, there exists a unique morphism $w : Y \to Z$ above g such that $w \circ u = v$.

A functor $p : \mathbb{E} \to \mathbb{B}$ is an *op-fibration* if $p : \mathbb{E}^{op} \to \mathbb{B}^{op}$ is a fibration. That is, for an object X in $\mathbb{E}$ and a morphism $f : pX \to I$ in $\mathbb{B}$, there exists an op-Cartesian morphism (called op-Cartesian lifting) $u : X \to \bullet$ in $\mathbb{E}$ above f. An op-fibration p is *cloven* if it comes with a cleavage on $p : \mathbb{E}^{op} \to \mathbb{B}^{op}$. We denote the op-Cartesian lifting of (X, f) by $\underline{f}X$. For a morphism $f : I \to J$ in $\mathbb{B}$, cleavage induces an *op-reindexing functor* $f_* : \mathbb{E}_I \to \mathbb{E}_J$ which sends an object X in $\mathbb{E}_I$ to $\mathrm{dom}(\underline{f}X)$ in $\mathbb{E}_J$.

A functor $p : \mathbb{E} \to \mathbb{B}$ is a (cloven) *bifibration* if it is both a (cloven) fibration and a (cloven) op-fibration. □

The following is the dual of proposition 2.3.4.

Proposition 2.3.7 *Let $p : \mathbb{E} \to \mathbb{B}$ be a cloven op-fibration, $f : I \to J$ and $g : J \to K$ be morphisms in $\mathbb{B}$, and X and Y be objects in $\mathbb{E}$ above I and K. We have an isomorphism:*

$$h : \mathbb{E}_{g \circ f}(X, Y) \cong \mathbb{E}_g(f_* X, Y).$$

Proposition 2.3.8 *Let $p : \mathbb{E} \to \mathbb{B}$ be a cloven fibration. Then p is a bifibration if and only if for each morphism $f : I \to J$ in $\mathbb{B}$, $f^* : \mathbb{E}_J \to \mathbb{E}_I$ has a left adjoint.* □

PROOF See [Jac99], lemma 9.1.2.

Subobject Fibration

The fibration we use most in this thesis is *subobject fibration*. First we define the category of subobjects.

Definition 2.3.9 Let $\mathbb{C}$ be a category. We define an equivalence relation $=_m$ over monomorphisms in $\mathbb{C}$ by

$$f =_m g \iff \mathrm{cod}(f) = \mathrm{cod}(g) \\ \wedge \quad \exists h : \mathrm{dom}(f) \to \mathrm{dom}(g) \,.\, h \text{ is an isomorphism} \wedge g \circ h = f.$$

The category $\mathbf{Sub}(\mathbb{C})$ of subobjects is defined by the following data.

An object in $\mathbf{Sub}(\mathbb{C})$ is an equivalence class P of monomorphisms in $\mathbb{C}$ by $=_m$. We write $p_{\mathbb{C}}P$ for the common codomain of monomorphisms in P.

A morphism in $\mathbf{Sub}(\mathbb{C})$ from P to P' is a morphism $f : p_{\mathbb{C}}P \to p_{\mathbb{C}}P'$ in $\mathbb{C}$ such that there exists morphisms $m \in P, m' \in P'$ and a (necessarily unique) morphism $h : \mathrm{dom}(m) \to \mathrm{dom}(m')$ in $\mathbb{C}$ such that $m' \circ h = f \circ m$. □

The above assignment $p_{\mathbb{C}} : \mathbf{Obj}(\mathbf{Sub}(\mathbb{C})) \to \mathbf{Obj}(\mathbb{C})$ of objects can be extended to a functor $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$.

We note that for each object C in $\mathbb{C}$, the fibre category $\mathbf{Sub}(\mathbb{C})_C$ is a partial order, since when two objects P and P' in $\mathbf{Sub}(\mathbb{C})_C$ are isomorphic, they should be the same equivalence class of monos.

In general $p_{\mathbb{C}}$ might not be a fibration, but when $\mathbb{C}$ has pullbacks, it is a fibration.

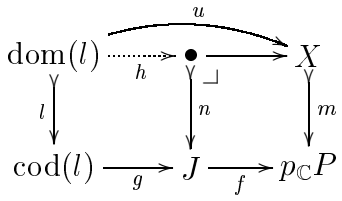
Proposition 2.3.10 *For any category $\mathbb{C}$ with pullbacks, $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ is a partially ordered fibration.* □

PROOF Let P be an object in $\mathbf{Sub}(\mathbb{C})$ and $f : J \to p_{\mathbb{C}}P$ be a morphism in $\mathbb{C}$. We take a monomorphism $m : X \rightarrowtail p_{\mathbb{C}}P$ from P and consider its pullback along f.

$$\begin{array}{ccc} \bullet & \longrightarrow & X \\ {\scriptstyle n}\downarrow & \lrcorner & \downarrow{\scriptstyle m} \\ J & \xrightarrow[f]{} & p_{\mathbb{C}}P \end{array}$$

The morphism n is mono because pullback of a monomorphism yields a monomorphism. It is easy to see that $[n]_{=_m}$ does not depend on the choice of $m \in P$ but is determined only by f and P, so we write f^*P for $[n]_{=_m}$. From the definition of morphisms in $\mathbf{Sub}(\mathbb{C})$, f is a morphism from f^*P to P. We take f itself as the Cartesian lifting of f.

To see that f is Cartesian, let Q be a subobject, $l \in Q$ be a monomorphism, $g : \mathrm{cod}(l) \to J$ be a morphism in $\mathbb{C}$ and assume that there exists a morphism $u : \mathrm{dom}(l) \to X$ such that $m \circ u = f \circ g \circ l$. Since the previous square is a pullback, we have the mediating morphism $h : \mathrm{dom}(l) \to \bullet$ satisfying $n \circ h = g \circ l$.



Thus we showed that g is a morphism from Q to $f^*(P)$ in $\mathbf{Sub}(\mathbb{C})$. Its uniqueness is obvious. ∎

It is often tedious to discuss equivalence classes of monos. We identify a mono and its equivalence class.

Example 2.3.11 Probably the most intuitive example of subobject fibrations is $p_{\mathbf{Sets}} : \mathbf{Sub}(\mathbf{Sets}) \to \mathbf{Sets}$. We give a description of $\mathbf{Sub}(\mathbf{Sets})$ as follows:

An object in $\mathbf{Sub}(\mathbf{Sets})$ is, according to the formal definition, an equivalence class of monos by $=_m$. However it is more convenient to represent it by a pair of sets (X, I) such that $X \subseteq I$. These two definitions are interchangeable; for an equivalence class X of monos and $m \in X$, we have a pair $(\{m(x) \in \mathrm{cod}(m) \mid x \in \mathrm{dom}(m)\}, \mathrm{cod}(m))$. This does not depend on the choice of m. On the other hand, a subset $X \subseteq I$ specifies a subobject $[\iota]_{=_m}$ of I, where $\iota : X \hookrightarrow I$ is the inclusion function.

A morphism from (X, I) to (Y, J) is a function $f : I \to J$ such that $f(x) \in Y$ holds for each $x \in X$.

When it is obvious that X and Y are subsets of I and J respectively, then we simply write $f : X \to Y$ to mean that f is a function from I to J satisfying $\forall x \in X \,.\, f(x) \in Y$.

The functor $p_{\mathbf{Sets}} : \mathbf{Sub}(\mathbf{Sets}) \to \mathbf{Sets}$ acts on objects and morphisms as follows:

$$p_{\mathbf{Sets}}(X, I) = I \quad p_{\mathbf{Sets}} f = f$$

$\mathbf{Sets}$ has pullbacks, thus $p_{\mathbf{Sets}}$ is a partially ordered fibration. For a set I, the fibre category $(p_{\mathbf{Sets}})_I$ is the preorder $(\mathcal{P}(I), \subseteq)$. □

2.4 Properties of Fibrations

We introduce structures over fibrations. We fix a cloven fibration $p : \mathbb{E} \to \mathbb{B}$.

2.4.1 Structures in Each Fibre Category

Definition 2.4.1 ([Jac99], definition 1.8.1) Let $\square$ be a name of a structure of categories (e.g. finite limits, coproducts, CCC, etc.). We say that p has *fibred* $\square$ if for each object I in $\mathbb{B}$, the fibre category $\mathbb{E}_I$ has structure $\square$ and for any morphism $f : I \to J$ in $\mathbb{B}$, the reindexing functor $f^* : \mathbb{E}_J \to \mathbb{E}_I$ preserves $\square$. □

We adopt the following notational conventions.

1. We use $\top, \wedge$ for fibred finite products.
2. We use $\bot, \vee$ for fibred finite coproducts.
3. We use $\Longrightarrow$ for fibred exponentials.

Example 2.4.2 (Continued from example 2.3.11) For a set I, the fibre category $(p_{\mathbf{Sets}})_I = (\mathcal{P}(I), \subseteq)$ has a CCC structure by taking $\top = I$, $X \wedge Y = X \cap Y$, $\bot = \emptyset$, $X \vee Y = X \cup Y$, $X \Longrightarrow Y = (I \backslash X) \vee Y$. Pullbacks in $\mathbf{Sets}$ preserve this CCC structure. Therefore $p_{\mathbf{Sets}}$ is a fibred CCC. □

Proposition 2.4.3 ([Jac99], lemma 1.8.4) *Consider the situation in theorem 2.3.5. If p has fibred $\square$, then the fibration i obtained by change-of-base also has fibred $\square$.* □

Choosing a terminal object $\top$ in the fibre category $\mathbb{E}_I$ for each object I in $\mathbb{B}$ is equivalent to giving a functor $\top : \mathbb{B} \to \mathbb{E}$ such that $p \circ \top = \mathrm{Id}_{\mathbb{B}}$.

Proposition 2.4.4 ([Jac99], lemma 1.8.8) *The fibred terminal objects functor $\top : \mathbb{B} \to \mathbb{E}$ is a right adjoint to $p : \mathbb{E} \to \mathbb{B}$.* □

Example 2.4.5 For a category $\mathbb{C}$ with pullbacks, the subobject fibration $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has fibred terminal objects given by $\top I = [\mathrm{id}_I]_{=_m}$. □

Proposition 2.4.6 *Assume that p has fibred finite (resp. small) products $\bigwedge$ and $\mathbb{B}$ has finite (resp.small) products $\prod$. Then $\mathbb{E}$ has finite (resp. small) products $\dot{\prod}$ which are strictly preserved by p.* □

PROOF The proof can be found in e.g. [Jac99], lemma 8.5.2. We only give the definition of $\dot{\prod}$. We define the finite product $\dot{\prod}_{i=1}^n X_i$ of objects $X_1, \cdots, X_n$ in $\mathbb{E}$ by $\dot{\prod}_{i=1}^n X_i = \bigwedge_{i=1}^n \pi_i^* X_i$ where $\pi_i : \prod_{i=1}^n pX_i \to pX_i$ is a projection in $\mathbb{B}$ for each $1 \leq i \leq n$. The terminal object $\dot{1}$ in $\mathbb{E}$ is the terminal object $\top$ in $\mathbb{E}_1$. ■

Example 2.4.7 (Continued from example 2.4.2) A binary product in $\mathbf{Sub}(\mathbf{Sets})$ is calculated by

$$(X, I) \mathbin{\dot{\times}} (Y, J) = \{(i, j) \in I \times J \mid i \in X \wedge j \in Y\}.$$

The following proposition is the dual of the above proposition.

Proposition 2.4.8 *Assume that p is a bifibration with fibred finite (resp. small) coproducts $\bigvee$ and $\mathbb{B}$ has finite (resp. small) coproducts $\coprod$. Then $\mathbb{E}$ has finite (resp. small) coproducts $\dot{\coprod}$ which are strictly preserved by p.* □

PROOF The proof can be found in e.g. [Jac99], lemma 9.2.2. We only give the definition of $\dot{\coprod}$. We define the finite coproduct $\dot{\coprod}_{i=1}^n X_i$ of objects in $X_1, \cdots, X_n$ in $\mathbb{E}$ by $\dot{\coprod}_{i=1}^n X_i = \bigvee_{i=1}^n (\iota_i)_* X_i$ where $\iota_i : pX_i \to \coprod_{i=1}^n pX_i$ is an injection in $\mathbb{B}$ for each $1 \leq i \leq n$. The initial object $\dot{0}$ in $\mathbb{E}$ is the initial object $\bot$ in $\mathbb{E}_0$. ■

Example 2.4.9 (Continued from example 2.4.2) A binary coproduct in $\mathbf{Sub}(\mathbf{Sets})$ is calculated by

$$(X, I) \mathbin{\dot{+}} (Y, J) = \{x \in I + J \mid (\exists i \in X \,.\, x = \iota_l(i)) \vee (\exists j \in Y \,.\, x = \iota_r(j)\}.$$

2.4.2 Structures Between Fibre Categories

In this section we assume that the base category $\mathbb{B}$ of the fibration $p : \mathbb{E} \to \mathbb{B}$ has finite products.

Definition 2.4.10 ([Jac99], definition 3.4.1) We say that p has *equality* if for each parameterised diagonal morphism $\delta_{I,J} = \langle \pi, \pi', \pi' \rangle : I \times J \to I \times J \times J$, we have a left adjoint to $\delta^*_{I,J} : \mathbb{E}_{I\times J\times J} \to \mathbb{E}_{I\times J}$, namely $Eq_{I,J} : \mathbb{E}_{I\times J} \to \mathbb{E}_{I\times J\times J}$, satisfying the Beck-Chevalley condition: for each morphism $f : K \to I$ in $\mathbb{B}$, the canonical natural transformation

$$\mathrm{Eq}_{I,J} \circ (f \times J)^* \to (f \times J \times J)^* \circ \mathrm{Eq}_{I,J}$$

is an isomorphism.

Furthermore, we say that the equality satisfies *Frobenius* if for any objects I and J in $\mathbb{B}$, X in $\mathbb{E}_{I\times J\times J}$ and Y in $\mathbb{E}_{I\times J}$, the canonical morphism in $\mathbb{E}_{I\times J\times J}$:

$$\mathrm{Eq}_{I,J}(\delta^*_{I,J}X \wedge Y) \to X \wedge \mathrm{Eq}_{I,J}Y$$

is an isomorhism. □

Proposition 2.4.11 ([Jac99], example 3.4.4) *For a category $\mathbb{C}$ with finite limits, $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has equality satisfying Frobenius.* □

PROOF The left adjoint to $\delta^*_{I,J}$ is given by $(\delta_{I,J})_*[m]_{=_m} = [\delta_{I,J} \circ m]_{=_m}$. This is well-defined, since $\delta : I \times J \to I \times J \times J$ is a monomorphism (actually, in a subobject fibration, for any monomorphism $f : I \to J$ in $\mathbb{C}$, f^* has a left adjoint). We leave readers to check that it satisfies Beck-Chevalley and Frobenius. ■

Example 2.4.12 (Continued from example 2.4.2) $p_{\mathbf{Sets}}$ has equality satisfying Frobenius. We define $Eq_{I,J}X$ for $X \subseteq I \times J \times J$ by:

$$Eq_{I,J}X = \{(i, j, j) \in I \times J \times J \mid (i, j) \in X\}.$$

This is a left adjoint of $\delta^*_{I,J}$, since for $Y \subseteq I \times J$, we have the following equivalence:

$$Y \subseteq \delta_* X \iff (\forall i \in I, j \in J \,.\, (i, j) \in Y \implies (i, j, j) \in X) \iff Eq_{I,J}Y \subseteq X.$$

The reader can check that the Beck-Chevalley and Frobenius are satisfied. □

Definition 2.4.13 ([Jac99], definition 1.9.1) We say that p has *simple products* if for each projection morphism $\pi : I \times J \to I$ in $\mathbb{B}$, we have a right adjoint to $\pi^* : \mathbb{E}_I \to \mathbb{E}_{I \times J}$, namely $\forall_{I,J} : \mathbb{E}_{I \times J} \to \mathbb{E}_I$, satisfying the *Beck-Chevalley* condition: for any morphism $f : K \to I$ in $\mathbb{B}$, the canonical natural transformation

$$f^* \circ \forall_{I,J} \to \forall_{K,J} \circ (f \times J)^*$$

is an isomorphism. □

Proposition 2.4.14 *For a category $\mathbb{C}$ with finite limits and exponentials, $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has simple products.* □

PROOF See [Jac99], corollary 1.9.9. ■

Example 2.4.15 (Continued from example 2.4.2) $p_{\mathbf{Sets}}$ has simple products. We define $\forall_{I,J} X$ for $X \subseteq I \times J$ by:

$$\forall_{I,J} X = \{i \in I \mid \forall j \in J \,.\, (i,j) \in X\}.$$

This is a right adjoint to π^*, since for any $Y \subseteq I$,

$$\begin{aligned} \pi^* Y \subseteq X &\iff (\forall i \in I, j \in J \,.\, i \in Y \implies (i,j) \in X) \\ &\iff (\forall i \in I \,.\, i \in Y \implies \forall j \in J \,.\, (i,j) \in X) \\ &\iff Y \subseteq \forall_{I,J} X. \end{aligned}$$

The reader can check that the Beck-Chevalley is satisfied. □

Definition 2.4.16 ([Jac99], definition 1.9.1) We say that p has *simple coproducts* if for each projection morphism $\pi : I \times J \to I$ in $\mathbb{B}$, we have a left adjoint to $\pi^* : \mathbb{E}_I \to \mathbb{E}_{I \times J}$, namely $\exists_{I,J} : \mathbb{E}_{I \times J} \to \mathbb{E}_I$, satisfying *Beck-Chevalley* condition: for any morphism $f : K \to I$ in $\mathbb{B}$, the canonical natural transformation

$$\exists_{I,J} \circ f^* \to (f \times J)^* \circ \exists_{K,J}$$

is an isomorphism.

Furthermore, we say that the simple coproducts *satisfy Frobenius* if for any objects I and J in $\mathbb{B}$, X in $\mathbb{E}_I$ and Y in $\mathbb{E}_{I\times J}$, the following canonical morphism in $\mathbb{E}_{I\times J}$:

$$\exists_{I,J}(\pi^* X \wedge Y) \to X \wedge \exists_{I,J}(Y) \quad (\pi : I \times J \to I \text{ is a projection})$$

is an isomorphism. □

Example 2.4.17 (Continued from example 2.4.2) $p_{\mathbf{Sets}}$ has simple coproducts. We define $\exists_{I,J} X$ for $X \subseteq I \times J$ by:

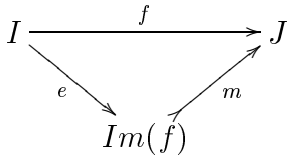
$$\exists_{I,J} X = \{i \in I \mid \exists j \in J \,.\, (i,j) \in X\}.$$

This is a left adjoint to π^*, since for any $Y \subseteq I$,

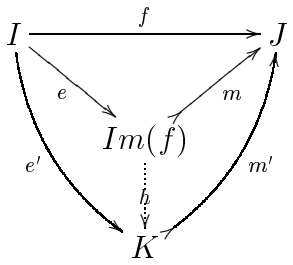
$$\begin{aligned} X \subseteq \pi^* Y &\iff (\forall i \in I, j \in J \,.\, (i,j) \in X \implies i \in Y) \\ &\iff (\forall i \in I \,.\, (\exists j \in J \,.\, (i,j) \in X) \implies i \in Y) \\ &\iff \exists_{I,J} X \subseteq Y. \end{aligned}$$

The reader can check that the Beck-Chevalley and Frobenius are satisfied. □

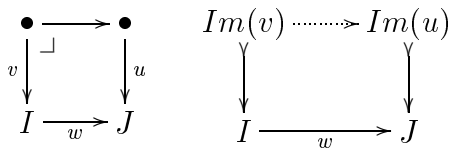
Definition 2.4.18 ([Jac99], definition 4.4.2) A category $\mathbb{C}$ with finite limits has *images* if for each morphism $f : I \to J$ in $\mathbb{C}$, there exists a factorisation (e, m) of f (that is, a pair (e, m) such that $f = m \circ e$) with m mono



satisfying the following universal property: for any factorisation (e', m') of f with m' mono, there exists a unique morphism h making the following triangles commute:



Furthermore, we say that images are *stable* if for each pullback square on the left,



the right square is also a pullback, where $Im(v) \to Im(u)$ is the morphism obtained by the universal property of image factorisation of u.

A category is *regular* if it has finite limits and stable images. □

Proposition 2.4.19 *Let $\mathbb{C}$ be a category with finite limits. Then $\mathbb{C}$ is a regular category if and only if $p_{\mathbb{C}}$ has simple coproducts satisfying Frobenius.* □

PROOF See [Jac99], theorem 4.4.4. ■

Proposition 2.4.20 ([Jac99], lemma 8.5.2) *Let $p : \mathbb{E} \to \mathbb{B}$ be a fibration such that $\mathbb{B}$ has finite products, p is a fibred CCC and p has simple products. Then $\mathbb{E}$ has exponentials.* □

PROOF For objects X and Y in $\mathbb{E}$ above I and J respectively, the following object:

$$X \dot{\Rightarrow} Y = \forall_{I \Rightarrow J, I}((\pi')^* X \implies @^* Y)$$

gives an exponential object in $\mathbb{E}$. For details, see [Jac99]. ■

Example 2.4.21 The exponential in $\mathbf{Sub}(\mathbf{Sets})$ is calculated by

$$(X, I) \dot{\Rightarrow} (Y, J) = \{f : I \Rightarrow J \mid \forall x \in X \,.\, f(x) \in Y\}.$$

This appears in the pattern defining *logical relations* (see section 1.1). □

2.4.3 Global Structure

Definition 2.4.22 ([Jac99], definition 4.6.1) We say that a fibration p with fibred terminal objects $\top : \mathbb{B} \to \mathbb{E}$ has *subset types* if $\top$ has a right adjoint $\{-\} : \mathbb{E} \to \mathbb{B}$. We write $m = p(\epsilon_{-}) : p\top\{-\} = \{-\} \to p$ for the *subset projection*, where $\epsilon : \top\{-\} \to \mathrm{Id}_{\mathbb{E}}$ is the counit of the adjunction $\top \dashv \{-\}$. □

Proposition 2.4.23 ([Jac99], example 4.6.3) *For a category $\mathbb{C}$ with pullbacks, the subobject fibration $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has subset types.* □

PROOF Under the axiom of choice, we can choose a representative monomorphism $m_P \in P$ for each subobject P in $\mathbf{Sub}(\mathbb{C})$. We then define $\{-\} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ to be $\mathrm{dom}(m_P)$. This determines a right adjoint to $\top$. ■

Proposition 2.4.24 ([Jac99], lemma 4.6.2) *Let $p : \mathbb{E} \to \mathbb{B}$ be a preordered fibration with fibred terminal objects $\top : \mathbb{B} \to \mathbb{E}$ and subset types $\{-\} : \mathbb{E} \to \mathbb{B}$. Then the subset projection $m_X : \{X\} \to pX$ is a monomorphism.* □

PROOF Let $f, g : I \to \{X\}$ be morphisms in $\mathbb{E}$ and assume $m_X \circ f = m_X \circ g$. We transpose f, g by the adjunction $\top \dashv \{-\}$ and obtain $\underline{f}, \underline{g} : \top I \to X$. Then we have

$$p\underline{f} = p(\epsilon_X \circ \top\underline{f}) = m_X \circ f = m_X \circ g = p\underline{g}.$$

Since p is preordered, we conclude $\underline{f} = \underline{g}$, that is, $f = g$. ■

Lifting of Initial Algebra

We introduce the concept of *lifting* of an endofunctor via a fibration.

Definition 2.4.25 Let $F : \mathbb{B} \to \mathbb{B}$ be an endofunctor. An endofunctor $\dot{F} : \mathbb{E} \to \mathbb{E}$ is a *lifting* of F (via p) if $p \circ \dot{F} = F \circ p$ holds. □

Example 2.4.26 ([Jac99], lemma 1.7.5) Let $\mathbb{C}$ be a category with pullbacks and $F : \mathbb{C} \to \mathbb{C}$ be an endofunctor preserving pullbacks. Then the following endofunctor $\dot{F} : \mathbf{Sub}(\mathbb{C}) \to \mathbf{Sub}(\mathbb{C})$ defined by:

$$\dot{F}[m]_{=_m} = [Fm]_{=_m}$$

is a lifting of F. □

For an endofunctor $F : \mathbb{B} \to \mathbb{B}$ and its lifting $\dot{F} : \mathbb{E} \to \mathbb{E}$ via a fibration $p : \mathbb{E} \to \mathbb{B}$, any $\dot{F}$-algebra $(X, u : \dot{F}X \to X)$ is above an underlying F-algebra $(pX, pu : p(\dot{F}X) = F(pX) \to pX)$. Like the way an object X in $\mathbb{E}$ specifies a part (or predicate) of the object pX in $\mathbb{B}$, we can regard (X, u) as a *sub-F-algebra* of (pX, pu).

Suppose we have an initial F-algebra $(A, a : FA \to A)$. One of the properties of the initial algebra is that it has no proper sub-algebra. However, this property holds only in $\mathbb{B}$, and does not take sub-algebras in the above sense into account. To make this property hold across the fibration, one might require that there is an initial $\dot{F}$-algebra $(\top A, \dot{a})$ above (A, a). This seems a natural requirement, since $\top A$ represents A itself in $\mathbb{E}$, and the initiality of $(\top A, \dot{a})$ guarantees that any $\dot{F}$-algebra above (A, a) is isomorphic to $(\top A, \dot{a})$, that is, there exists no proper sub-algebra in the above sense.

In the following theorem, we show that the above requirement is satisfied when p has subset types and there exists a natural vertical isomorphism $i : \dot{F} \circ \top \to \top \circ F$. This theorem is a mild generalisation of theorem 4.3 in [HJ95]. [1]

Theorem 2.4.27 *Let $p : \mathbb{E} \to \mathbb{B}$ be a fibration with fibred terminal objects $\top : \mathbb{B} \to \mathbb{E}$ and subset types $\{-\} : \mathbb{E} \to \mathbb{B}$. Let $F : \mathbb{B} \to \mathbb{B}$ be an endofunctor, $\dot{F} : \mathbb{E} \to \mathbb{E}$ be a lifting of F via p and $i : \dot{F} \circ \top \to \top \circ F$ be a natural vertical isomorphism. Then for any initial F-algebra $(A, a : FA \to A)$, $(\top A, \top a \circ i_A)$ is an initial $\dot{F}$-algebra above (A, a).* □

PROOF We write $\eta : \mathrm{Id}_{\mathbb{B}} \to \{\top -\}$ and $\epsilon : \top\{-\} \to \mathrm{Id}_{\mathbb{E}}$ for the unit and counit of the adjunction $\top \dashv \{-\}$.

Let $(B, b : \dot{F}B \to B)$ be an $\dot{F}$-algebra. We first find an $\dot{F}$-algebra morphism h from $(\top A, \top a \circ i_A)$ to (B, b). To do so, we construct an F-algebra $(\{B\}, \beta)$ in the following way:

$$\dfrac{\dfrac{\dfrac{b : \dot{F}B \to B \quad \dot{F}\epsilon_B : \dot{F}(\top\{B\}) \to \dot{F}B}{b \circ \dot{F}\epsilon_B : \dot{F}(\top\{B\}) \to B}}{b \circ \dot{F}\epsilon_B \circ i^{-1}_{\{B\}} : \top(F\{B\}) \to B}}{\beta = \{b \circ \dot{F}\epsilon_B \circ i^{-1}_{\{B\}}\} \circ \eta_{F\{B\}} : F\{B\} \to \{B\}.}$$

From initiality, we obtain a unique F-algebra morphism $h_0 : A \to \{B\}$. We give the

[1] The main difference from theorem 4.3 in [HJ95] is that theorem 2.4.27 is proved with respect to an arbitrary pair of an endofunctor F and its lifting $\dot{F}$ via p, rather than those constructed from a given polynomial using the property of bicartesian fibrations. On the other hand, theorem 2.4.27 is an instance of a 2-categorical fact on inserters (theorem A.7 in [HJ95]), which is also used to derive theorem 4.3 in [HJ95].

h in question by $h = \epsilon_B \circ \top h_0$. We check that this is indeed a $\dot{F}$-algebra morphism:

$$\begin{aligned}
(\epsilon_B \circ \top h_0) \circ \top a \circ i_A &= \epsilon_B \circ \top\beta \circ \top(Fh_0) \circ i_A \\
&= \epsilon_B \circ \top\beta \circ i_{\{B\}} \circ \dot{F}(\top h_0) \\
&= \epsilon_B \circ \top(\{b \circ \dot{F}\epsilon_B \circ i^{-1}_{\{B\}}\} \circ \eta_{F\{B\}}) \circ i_{\{B\}} \circ \dot{F}(\top h_0) \\
&= b \circ \dot{F}\epsilon_B \circ i^{-1}_{\{B\}} \circ \epsilon_{\top F\{B\}} \circ \top\eta_{F\{B\}} \circ i_{\{B\}} \circ \dot{F}(\top h_0) \\
&= b \circ \dot{F}(\epsilon_B \circ \top h_0).
\end{aligned}$$

To show that h is the unique $\dot{F}$-algebra morphism, let h' be another $\dot{F}$-algebra morphism from $(\top A, \top a \circ i_A)$ to (B, b). Then a calculation shows that $\{h'\} \circ \eta_A : A \to \{B\}$ is an F-algebra morphism from (A, a) to $(\{B\}, \beta)$. Thus $\{h'\} \circ \eta_A = h_0$, which implies $h' = h$. ∎

This theorem will be applied to the endofunctor corresponding to a typed binding signature in section 3.6.

2.4.3.1 Quotient Types

Definition 2.4.28 ([Jac99], definition 4.8.1) Assume that p has equality and fibred terminal objects $\top : \mathbb{B} \to \mathbb{E}$. We write $\Delta : \mathbb{B} \to \mathbb{B}$ for the functor sending an object I to $I \times I$. We obtain a new fibration $\Delta^* p : \Delta^*\mathbb{E} \to \mathbb{B}$ by change-of-base along Δ:

$$\begin{array}{ccc}
\Delta^*\mathbb{E} & \longrightarrow & \mathbb{E} \\
{\scriptstyle \Delta^* p}\downarrow & \lrcorner & \downarrow{\scriptstyle p} \\
\mathbb{B} & \xrightarrow[\Delta]{} & \mathbb{B}
\end{array}$$

Note that $\Delta^*\mathbb{E}$ has fibred finite products by proposition 2.4.3, thus $\Delta^*\mathbb{E}$ has finite products strictly preserved by $\Delta^*\mathbb{E}$ by proposition 2.4.6.

There is then an equality relation functor $EQ : \mathbb{B} \to \Delta^*\mathbb{E}$:

$$EQ : I \mapsto (h^*_I(Eq_{1,I}(\top(1 \times I))), I)$$

where $h_I = \langle !, \pi, \pi' \rangle : I \times I \to 1 \times I \times I$ (see [Jac99], section 4.8 for how EQ sends morphisms). We note that $r \circ EQ = \mathrm{Id}_{\mathbb{B}}$.

We say that p has *quotient types* if EQ has a left adjoint L. For an object R in $\Delta^*\mathbb{E}$ above an object I in $\mathbb{B}$, we often write I/R instead of LR to emphasise that L gives the quotient of I by (the equality relation generated from) R. We write $c = p(\eta_-) : r \to L$ for the *canonical quotient map*, where $\eta : \mathrm{Id}_{\mathbb{E}} \to EQ \circ L$ is the unit of the adjunction $L \dashv EQ$.

Furthermore, we say that the quotient types *satisfies Frobenius* if for any objects I and J in $\mathbb{B}$ and X in $\Delta^*\mathbb{E}_J$, the canonical map

$$(I \times J)/(EQ(I) \mathbin{\dot{\times}} X) \to I \times J/X$$

is an isomorphism. □

2.5 Internal Logic of a Fibration

In this thesis we use the internal logic of a fibration as a convenient tool to reason about objects and morphisms in the base and the total category of a fibration. You can think of internal logic as providing a logic-style user interface for structures on a fibration.

Logical connectives, quantifiers and their inference rules in the internal logic have a tight correspondence with structures on a fibration. For example, when a fibration has fibred finite coproducts, the internal logic admits falsum and disjunction. The presentation of internal logic in this section takes this point into account. First we introduce the base logic, which provides a basis for all variations of internal logic. Then it is followed by various extensions to the base logic by logical connectives and quantifications, such as equality predicate, universal quantification, etc. We associate each extension with a structure on a fibration, and give a categorical semantics of judgements and inference rules provided by the extension.

The internal logic has the following judgements.

A Type Judgement is of the form I where I is just a type.

A Term Judgement is of the form $\Gamma \vdash M : I$ where Γ is a context and M is a term of the internal logic. This judgement means that "term M has type I under the assignment Γ of types to free variables in M."

A Predicate Judgement is of the form $\Gamma \vdash P$, where Γ is a context and P is a formula of the internal logic. This judgement means that "P is a well-formed formula under the assignment Γ of types to free variables in P."

A Sequent is of the form $\Gamma \mid \Phi \vdash P$, where Γ is a context, Φ is a finite sequence $P_1, \cdots, P_n$ of predicates and P is a predicate. This judgement means that "$P_1 \wedge \cdots \wedge P_n$ implies P under the assignment Γ of types to free variables in $P_1, \cdots, P_n, P$."

These judgements will be interpreted in a preordered fibration $p : \mathbb{E} \to \mathbb{B}$:

- Type judgements are interpreted by objects in the base category $\mathbb{B}$.
- Term judgements are interpreted by morphisms in the base category $\mathbb{B}$.
- Predicate judgements are interpreted by objects in some fibre preorder.
- Sequents are interpreted by inequalities in some fibre preorder.

The logic has inference rules of the following form:

$$\frac{J_1 \quad \cdots \quad J_n}{J}$$

where $J, J_1, \cdots, J_n$ are judgements. We interpret an inference rule by a procedure for constructing an object / morphism $[\![J]\!]$ corresponding to the conclusion J of the rule from given objects / morphisms $[\![J_1]\!], \cdots, [\![J_n]\!]$ corresponding to the assumptions $J_1, \cdots, J_n$ of the rule. The fibration we consider is preordered, thus when all judgements $J, J_1, \cdots, J_n$ are sequents, the interpretation of an inference rule is just a statement that "inequalities $[\![J_1]\!], \cdots, [\![J_n]\!]$ imply $[\![J]\!]$."

For readability, we often write the formal proof in the internal logic in plain English. For example, "let $x : A''$ and ..." corresponds to the formal proof in the internal logic under the context $x : A''$.

The Base Logic

The base logic $\mathcal{L}_\wedge$ provides the inference rules which are common in various extensions, and those for truth and conjunction.

The base logic is designed to reason about the following fibration.

Definition 2.5.1 Let $p : \mathbb{E} \to \mathbb{B}$ be a fibration. We say that p *supports* $\mathcal{L}_\wedge$ if p is preordered, p has fibred finite products and $\mathbb{B}$ has finite products. □

Type Inference Rules of $\mathcal{L}_\wedge$ consists of:

$$\frac{}{I \in \mathbf{Obj}(\mathbb{B})} \qquad \frac{I_1 \quad \cdots \quad I_n}{\prod_{i=1}^n I_i}$$

Any object I in $\mathbb{B}$ is a type, and also the finite product $\prod_{i=1}^n I_i$ of types $I_1, \cdots, I_n$ is a type. The meaning of a type I in $\mathbb{B}$, denoted by $[\![I]\!]$, is straightforward.

We interpret a context Γ by the finite product object $\prod_{i=1}^{\#\mathrm{dom}(\Gamma)} \Gamma(\gamma^{-1}(i))$ in $\mathbb{B}$ denoted by $[\![\Gamma]\!]$. In the rest of this chapter, by $\Gamma, x_1 : I_1, \cdots, x_n : I_n$, we mean a context such that $\mathbf{x}_1 \leq_{\mathbf{v}} \cdots \leq_{\mathbf{v}} x_n$. Therefore we have an intuitive equation: $[\![\Gamma, x_1 : I_1, \cdots, x_n : I_n]\!] = [\![\Gamma]\!] \times [\![I_1]\!] \times \cdots \times [\![I_n]\!]$.

Term Inference Rules of $\mathcal{L}_\wedge$ consists of:

$$\frac{\Gamma(x) = I}{\Gamma \vdash x : I} \qquad \frac{\Gamma, x : I, y : I \vdash M : J}{\Gamma, x : I \vdash M[x/y] : J}$$

$$\frac{\Gamma \vdash M : I \quad f : [\![I]\!] \to [\![J]\!]}{\Gamma \vdash fM : J} \qquad \frac{\Gamma \vdash M_1 : I_1 \quad \cdots \quad \Gamma \vdash M_n : I_n}{\Gamma \vdash (M_1, \cdots, M_n) : \prod_{i=1}^n I_i}$$

In the first rule, we implicitly assume that all types in context Γ are correctly formed by type inference rules. The second rule is contraction of term variables. The third rule introduces a morphism in $\mathbb{B}$ as an operation in the internal logic. The fourth rule is for the tuple $(M_1, \cdots, M_n)$ of terms $M_1, \cdots, M_n$. We do not explicitly need terms for projection because we can express them by application:

$$\frac{\Gamma \vdash M : \prod_{i=1}^n I_i \quad \pi_j : [\![\prod_{i=1}^n I_i]\!] = \prod_{i=1}^n [\![I_i]\!] \to [\![I_j]\!]}{\Gamma \vdash \pi_j\ M : I_j}$$

We interpret a term judgement $\Gamma \vdash M : I$ as a morphism $[\![M]\!] : [\![\Gamma]\!] \to [\![I]\!]$ in $\mathbb{B}$. The above rules are interpreted as follows:

$$\begin{aligned} [\![x]\!] &= \pi_{\gamma(x)} \\ [\![M[x/y]]\!] &= [\![M]\!] \circ \langle \pi, \pi', \pi' \rangle \\ [\![(M_1, ..., M_n)]\!] &= \langle [\![M_1]\!], \cdots, [\![M_n]\!] \rangle \\ [\![fM]\!] &= f \circ [\![M]\!]. \end{aligned}$$

Predicate Inference Rules A predicate of $\mathcal{L}_\wedge$ is either an application PM of a term M to an object P in a fibre category, the truth $\top$ or a conjunction $P \wedge Q$.

$$\frac{\Gamma \vdash M : I \quad P \in \mathbf{Obj}(\mathbb{E}_I)}{\Gamma \vdash PM} \qquad \frac{}{\Gamma \vdash \top} \qquad \frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \wedge Q}$$

We interpret a predicate judgement $\Gamma \vdash P$ by an object in the fibre category $\mathbb{E}_{[\![\Gamma]\!]}$. An application PM is interpreted by reindexing of P along $[\![M]\!]^*$. Truth and conjunction are interpreted by the terminal object and binary product object in the fibre category.

$$\begin{aligned} [\![PM]\!] &= [\![M]\!]^* P \\ [\![\top]\!] &= \top \\ [\![P \wedge Q]\!] &= [\![P]\!] \wedge [\![Q]\!] \end{aligned}$$

We note that giving a sequent $\Gamma \mid \Phi \vdash PM$ is equivalent to asserting that $[\![M]\!] : [\![\Phi]\!] \to [\![P]\!]$ in $\mathbb{E}$ (c.f. proposition 2.3.4).

We interpret a sequence of predicates $\Phi = P_1, \cdots, P_n$ by the object $\bigwedge_{i=1}^n [\![P_i]\!]$ (denoted by $[\![\Phi]\!]$) in the fibre category $\mathbb{E}_{[\![\Gamma]\!]}$.

Sequent Inference Rules of $\mathcal{L}_\wedge$ consists of three groups of rules. The first group consists of axiom and substitution:

$$\frac{[\![\Phi]\!] \leq [\![P]\!] \text{ holds in } \mathbb{E}_{[\![\Gamma]\!]}}{\Gamma \mid \Phi \vdash P} \qquad \frac{\Gamma, x : I \mid \Phi \vdash P \quad \Gamma \vdash M : I}{\Gamma \mid \Phi[M/x] \vdash P[M/x]}$$

$$\frac{\Gamma \mid \Phi \vdash P \quad \Gamma \mid \Psi, P \vdash Q}{\Gamma \mid \Phi, \Psi \vdash Q}$$

The first rule asserts that any inequality $[\![\Phi]\!] \leq [\![P]\!]$ in $\mathbb{E}_{[\![\Gamma]\!]}$ is an axiom. This rule makes the internal logic reflect all inequalities between predicates in each fibre category. The second rule says that inequality is preserved by substitution. The third rule is the *cut* rule.

The second group consists of structural rules for assumptions.

$$\frac{\Gamma \mid P_1, \cdots, P_n \vdash Q}{\Gamma \mid P_{\theta(1)}, \cdots, P_{\theta(n)} \vdash Q} \qquad \frac{}{\Gamma \mid P_1, \cdots, P_n \vdash P_i} \qquad \frac{\Gamma \mid P_1, \cdots, P_n, P_n \vdash P_i}{\Gamma \mid P_1, \cdots, P_n \vdash P_i}$$

where $\theta : \{1, \cdots, n\} \to \{1, \cdots, n\}$ is a permutation.

The third group consists of the standard rules for reasoning about the truth and conjunctions in natural deduction style.

$$\frac{}{\Gamma \mid \Phi \vdash \top} \quad \frac{\Gamma \mid \Phi \vdash P \quad \Gamma \mid \Phi \vdash Q}{\Gamma \mid \Phi \vdash P \wedge Q} \quad \frac{\Gamma \mid \Phi \vdash P \wedge Q}{\Gamma \mid \Phi \vdash P} \quad \frac{\Gamma \mid \Phi \vdash P \wedge Q}{\Gamma \mid \Phi \vdash Q}$$

Fibred finite products are nothing but finite meets in the preorder $\mathbb{E}_{[\![\Gamma]\!]}$. We interpret the above inference rules by the following axioms on finite meets:

$$\begin{aligned} &[\![\Phi]\!] \leq \top \\ &[\![\Phi]\!] \leq [\![P]\!] \text{ and } [\![\Phi]\!] \leq [\![Q]\!] \implies [\![\Phi]\!] \leq [\![P \wedge Q]\!] \\ &[\![\Phi]\!] \leq [\![P \wedge Q]\!] \leq [\![P]\!], [\![Q]\!]. \end{aligned}$$

Later we consider a logic with set-indexed conjunctions. We refer to this logic by $\mathcal{L}_{\wedge\omega}$. This logic is supported by a preordered fibration $p : \mathbb{E} \to \mathbb{B}$ with fibred *small* products and finite products in $\mathbb{B}$. We write $\bigwedge_{i \in I} P_i$ for the conjunction of the I-indexed family of predicates P_i. Inference rules and their categorical interpretation are straightforward.

Extensions to the Base Logic

We introduce extensions to $\mathcal{L}_\wedge$. We fix a fibration p which supports $\mathcal{L}_\wedge$.

Equality Predicate

This fragment $\mathcal{L}_=$ provides an extension of the base logic $\mathcal{L}_\wedge$ with an equality predicate. We say that p *supports* $\mathcal{L}_=$ if p has equality satisfying Frobenius (see definition 2.4.10).

Predicate Inference Rule

$$\frac{\Gamma \vdash M : I \quad \Gamma \vdash N : I}{\Gamma \vdash M = N}$$

The equality predicate in the internal logic is called the *internal equality*, and is distinguished from the equality of morphisms $[\![M]\!] = [\![N]\!]$ which is called *external equality*.

We interpret the equality predicate $M = N$ by the following object in $\mathbb{E}_{[\![\Gamma]\!]}$:

$$\langle \mathrm{id}_{[\![\Gamma]\!]}, [\![M]\!], [\![N]\!] \rangle^* (Eq_{[\![\Gamma]\!],[\![I]\!]}(\top))$$

where $\top$ is the terminal object in $\mathbb{E}_{[\![\Gamma]\!] \times [\![I]\!]}$.

Sequent Inference Rules

$$\frac{\Gamma \vdash M : I \quad \Gamma \vdash N : I \quad [\![M]\!] = [\![N]\!]}{\Gamma \mid \Phi \vdash M = N} \qquad \frac{\Gamma, x : I \mid \Phi \vdash P[x/y]}{\Gamma, x : I, y : I \mid \Phi, x = y \vdash P}$$

The first rule guarantees that the external equality implies the internal equality. In general the internal and external equality do not coincide, but if they do, we say that the equality is *very strong*.

We interpret the first rule by $[\![\Phi]\!] \leq [\![M = N]\!] = [\![M = M]\!] = \top$.

The second rule is called *Lawvere equality* [Law70]. The double line means that you can use this rule in both directions. Lawvere equality captures the essence of the equality predicate which is usually presented by the following four rules (see [Jac99], lemma 3.2.2):

$$\frac{\Gamma \vdash M : I}{\Gamma \mid \Phi \vdash M = M} \qquad \frac{\Gamma \mid \Phi \vdash N = M}{\Gamma \mid \Phi \vdash M = N} \qquad \frac{\Gamma \mid \Phi \vdash M = N \quad \Gamma \mid \Phi \vdash N = L}{\Gamma \mid \Phi \vdash M = L}$$

$$\frac{\Gamma \mid \Phi \vdash N = L \quad \Gamma, x : I \vdash M : J}{\Gamma \mid \Phi \vdash M[N/x] = M[L/x]}$$

We interpret Lawvere equality by the following equivalence:

$$\begin{aligned} & [\![\Phi]\!] \leq \delta^*_{[\![\Gamma]\!],[\![I]\!]} [\![P]\!] = [\![P[x/y]]\!] \\ \iff \quad & [\![\Phi, x = y]\!] \cong \pi^* [\![\Phi]\!] \wedge \mathrm{Eq}_{[\![\Gamma]\!],[\![I]\!]}(\top) \cong \mathrm{Eq}_{[\![\Gamma]\!],[\![I]\!]}([\![\Phi]\!]) \leq [\![P]\!] \end{aligned}$$

where $\pi : [\![\Gamma]\!] \times [\![I]\!] \times [\![I]\!] \to [\![\Gamma]\!] \times [\![I]\!]$ is a projection.

Falsum and Disjunction

This fragment $\mathcal{L}_\vee$ provides an extension of the base logic with the falsum and disjunctions. We say that p *supports* $\mathcal{L}_\vee$ if p has fibred finite coproducts (see definition 2.4.1). We simply give inference rules; their interpretation is straightforward.

Predicate Inference Rule

$$\frac{}{\Gamma \vdash \bot} \qquad \frac{\Gamma \vdash P \quad \Gamma \vdash Q}{\Gamma \vdash P \vee Q}$$

Sequent Inference Rules

$$\frac{}{\Gamma \mid \Phi, \bot \vdash P} \qquad \frac{\Gamma \mid \Phi, P \vdash R \quad \Gamma \mid \Phi, Q \vdash R}{\Gamma \mid \Phi, P \vee Q \vdash R} \qquad \frac{\Gamma \mid \Phi \vdash P}{\Gamma \mid \Phi \vdash P \vee Q} \qquad \frac{\Gamma \mid \Phi \vdash Q}{\Gamma \mid \Phi \vdash P \vee Q}$$

Later we consider a logic with set-indexed disjunctions. We refer to this logic by $\mathcal{L}_{\vee\omega}$. This logic is supported by a preordered fibration $p : \mathbb{E} \to \mathbb{B}$ with fibred *small* coproducts. We write $\bigvee_{i \in I} P_i$ for the disjunction of the I-indexed family of predicates P_i. Inference rules and their categorical interpretation are straightforward.

Universal Quantifier

This fragment $\mathcal{L}_{\forall}$ provides an extension of the base logic with an universal quantifier. We say that p *supports* $\mathcal{L}_{\forall}$ if p has simple products (see definition 2.4.13).

Predicate Inference Rule

$$\frac{\Gamma, x : I \vdash P}{\Gamma \vdash \forall x : I \,.\, P}$$

We interpret a universally quantified predicate $\forall x : I \,.\, P$ as follows:

$$[\![\forall x : I \,.\, P]\!] = \forall_{[\![\Gamma]\!],[\![I]\!]}([\![P]\!]).$$

Seuqent Inference Rules

$$\frac{\Gamma, x : I \mid \Phi \vdash P \quad x \text{ is not free in } \Phi}{\Gamma \mid \Phi \vdash \forall x : I \,.\, P} \qquad \frac{\Gamma \mid \Phi \vdash \forall x : I \,.\, P \quad \Gamma \vdash M : I}{\Gamma \mid \Phi \vdash P[M/x]}$$

We notice that when $\Gamma, x : I \vdash \Phi$ contains no x, we can derive $\Gamma \vdash \Phi$ and show that $\pi^*[\![\Gamma \vdash \Phi]\!] = [\![\Gamma, x : I \vdash \Phi]\!]$. To interpret the first rule, assume that $\pi^*[\![\Phi]\!] \leq [\![P]\!]$ holds. By transposing this inequality by the adjunction $\pi^* \dashv \forall_{[\![\Gamma]\!],[\![I]\!]}$, we obtain $[\![\Phi]\!] \leq [\![\forall x : I \,.\, P]\!]$ in $\mathbb{E}_{[\![\Gamma]\!]}$.

To interpret the second rule, let $\pi^*(\forall_{[\![\Gamma]\!],[\![I]\!]}[\![P]\!]) \cong \pi^*([\![\forall x : I \,.\, P]\!]) \leq [\![P]\!]$ be the inequality corresponding to the counit of the adjunction $\pi^* \dashv \forall_{[\![\Gamma]\!],[\![I]\!]}$. For any $\Gamma \vdash M : I$, we have the following inequality:

$$\langle \mathrm{id}_{[\![\Gamma]\!]}, [\![M]\!] \rangle^* \pi^*([\![\forall x : I \,.\, P]\!]) \cong [\![\forall x : I \,.\, P]\!] \leq \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![M]\!] \rangle^*([\![P]\!]) = [\![P[M/x]]\!]$$

Therefore $[\![\Phi]\!] \leq [\![\forall x : I \,.\, P]\!]$ implies $[\![\Phi]\!] \leq [\![P[M/x]]\!]$.

Existential Quantifier

This fragment $\mathcal{L}_\exists$ provides an extension of the base logic with an existential quantifier. We say that p *supports* $\mathcal{L}_\exists$ if p has simple coproducts satisfying Frobenius (see definition 2.4.16).

Predicate Inference Rule

$$\frac{\Gamma, x : I \vdash P}{\Gamma \vdash \exists x : I \,.\, P}$$

We interpret an existentially quantified predicate $\exists x : I \,.\, P$ as follows:

$$[\![\exists x : I \,.\, P]\!] = \exists_{[\![\Gamma]\!],[\![I]\!]}([\![P]\!]).$$

Seuqent Inference Rules

$$\frac{\Gamma \mid \Phi \vdash \exists x : I \,.\, P \quad \Gamma, x : I \mid \Psi, P \vdash Q}{\Gamma \mid \Phi, \Psi \vdash Q} \qquad \frac{\Gamma \mid \Phi \vdash P[M/x] \quad \Gamma \vdash M : I}{\Gamma \mid \Phi \vdash \exists x : I \,.\, P}$$

In the first rule, we assume that Φ, Q does not contain x as a free variable.

To interpret the first rule, we assume that $[\![\Phi]\!] \leq [\![\exists x : I \,.\, P]\!]$ and $[\![\Psi, P]\!] = \pi^*[\![\Psi]\!] \wedge [\![P]\!] \leq \pi^*[\![Q]\!]$. From the mate rule of adjunction $\exists_{[\![\Gamma]\!],[\![I]\!]} \dashv \pi^*$ and Frobenius property, we have $[\![\Psi]\!] \wedge [\![\exists x : I \,.\, P]\!] \cong \exists_{[\![\Gamma]\!],[\![I]\!]}(\pi^*[\![\Psi]\!] \wedge [\![P]\!]) \leq [\![Q]\!]$. Therefore we obtain the inequality:

$$[\![\Psi]\!] \wedge [\![\Phi]\!] \leq [\![\Psi]\!] \wedge [\![\exists x : I \,.\, P]\!] \cong \exists_{[\![\Gamma]\!],[\![I]\!]}(\pi^*[\![\Psi]\!] \wedge [\![P]\!]) \leq [\![Q]\!].$$

To interpret the second rule, let $[\![P]\!] \leq \pi^*(\exists_{[\![\Gamma]\!],[\![I]\!]}[\![P]\!]) = \pi^*[\![\exists x : I \,.\, P]\!]$ be the inequality corresponding to the unit of adjunction $\exists_{[\![\Gamma]\!],[\![I]\!]} \dashv \pi^*$. For any $\Gamma \vdash M : I$, we have the following inequality:

$$[\![P[M/x]]\!] = \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![M]\!] \rangle^* [\![P]\!] \leq \langle \mathrm{id}_{[\![\Gamma]\!]}, [\![M]\!] \rangle^* \pi^* [\![\exists x : I \,.\, P]\!] \cong [\![\exists x : I \,.\, P]\!]$$

Thus $[\![\Phi]\!] \leq [\![P[M/x]]\!]$ implies $[\![\Phi]\!] \leq [\![\exists x : I \,.\, P]\!]$.

Subset Types

This fragment $\mathcal{L}_{\{-\}}$ provides an extension of the base logic with subset types. We say that p *supports* $\mathcal{L}_{\{-\}}$ if p has subset types (see definition 2.4.22).

Type Inference Rule Subset types embody *comprehension*: for a predicate P with a single variable x of type I, we can form a new type $\{x : I \mid P\}$ representing the collection of elements in I satisfying P. This is expressed by the following inference rule for types:

$$\frac{x : I \vdash P}{\{x : I \mid P\}}$$

We interpret subset types as follows:

$$[\![\{x : I \mid P\}]\!] = \{[\![P]\!]\}.$$

Term Inference Rule Subset types introduce two term constructs, $i_P\ M$ and $o_P\ M$. The term $i_P\ M$ asserts that M is included in a subset type $\{x : I \mid P\}$ provided that $P[M/x]$ holds, while $o_P\ M$ is just another notation for $m_{[\![P]\!]}M$.

$$\frac{x : I \vdash P \quad \Gamma \vdash M : I \quad \Gamma \mid \top \vdash P[M/x]}{\Gamma \vdash i_P\ M : \{x : I \mid P\}}$$

$$\frac{\Gamma \vdash M : \{x : I \mid P\} \quad m_{[\![P]\!]} : [\![\{x : I \mid P\}]\!] \to [\![I]\!]}{\Gamma \vdash o_P\ M = m_{[\![P]\!]}M : I}$$

We interpret $i_P\ M$ in the following way. Assume we have the inequality $\top \leq [\![P[M/x]]\!] = [\![M^*]\!]([\![P]\!])$ in $\mathbb{E}_{[\![\Gamma]\!]}$, where $\top$ is the terminal object in $\mathbb{E}_{[\![\Gamma]\!]}$. By applying proposition 2.3.4, we obtain a morphism $f : \top \to [\![P]\!]$ in $\mathbb{E}$. We transpose f by the adjunction $\top \dashv \{-\}$ and obtain a morphism $\underline{f} : \Gamma \to \{[\![P]\!]\} = [\![\{x : I \mid P\}]\!]$. With this, we interpret $i_P\ M$ by:

$$[\![i_P\ M]\!] = \underline{f}.$$

In the internal logic, we have the following axioms between terms:

$$i_P\ o_P\ M = [\![M]\!], \qquad [\![o_P\ i_P\ M]\!][\![M]\!].$$

which will soundly be interpreted in the fibration supporting $\mathcal{L}_{\{-\}}$.

Sequent Inference Rule Suppose that $P(x)$ implies $Q(x)$ for any $x : I$. Then $Q(x)$ is true for any x satisfying $P(x)$, i.e. for any x in $\{x : I \mid P(x)\}$. The sequent inference rule for subset types formulates this idea:

$$\frac{\Gamma, x : I \mid \Phi, P\ x \vdash Q \quad x : I \vdash P}{\Gamma, x : \{x : I \mid P\} \mid \Phi[o_P\ x/x] \vdash Q[o_P\ x/x]}$$

To interpret this rule, we assume the inequality $[\![\Phi]\!] \wedge \pi'^*[\![P]\!] \leq [\![Q]\!]$ in $\mathbb{E}_{[\![\Gamma]\!]\times[\![I]\!]}$, where $\pi' : [\![\Gamma]\!] \times [\![I]\!] \to [\![I]\!]$ is a projection. We send this by $([\![\Gamma]\!] \times m_{[\![P]\!]})^*$ and obtain $[\![\Phi[o_P\ x/x]]\!] \wedge \pi'^* m^*_{[\![P]\!]}[\![P]\!] \leq [\![Q[o_P\ x/x]]\!]$ in $\mathbb{E}_{[\![\Gamma]\!]\times\{[\![P]\!]\}}$. Since the counit $\epsilon_{[\![P]\!]} : \top\{[\![P]\!]\} \to [\![P]\!]$ of adjunction $\top \dashv \{-\}$ is above $m_{[\![P]\!]}$, the inequality $\top \cong \pi'^*\top \leq \pi'^* m^*_{[\![P]\!]}[\![P]\!]$ holds in $\mathbb{E}_{[\![\Gamma]\!]\times\{[\![P]\!]\}}$. Thus we obtain $[\![\Phi[o_P\ x/x]]\!] \leq [\![Q[o_P\ x/x]]\!]$. This shows that the above inference rule is sound with respect to the interpretation of sequents.

Quotient Types

This fragment $\mathcal{L}_{-/-}$ provides an extension of the base logic with quotient types. We say that p *supports* $\mathcal{L}_{-/-}$ if p supports $\mathcal{L}_{=}$ and has quotient types satisfying Frobenius (see definition 2.4.28).

Type Inference Rule The quotient type represents the quotient of a type I by an equivalence relation generated from a binary relation R over I. The type inference rule is the following:

$$\frac{x : I, y : I \vdash R}{I/R}$$

We interpret a quotient type as follows:

$$[\![I/R]\!] = L[\![R]\!]$$

where L is a left adjoint to $EQ : \mathbb{B} \to \Delta^*\mathbb{E}$ (see definition 2.4.28).

Term Inference Rule Quotient types introduce two term constructs, $[M]_R$ and "pick $x \in a$ in N". Let $x : I, y : I \vdash R(x, y)$ be a binary relation. The term

$[M]_R$ is just another notation for the term $c_{[\![R]\!]}\ M$:

$$\frac{\Gamma \vdash M : I \quad c_{[\![R]\!]}\ M : [\![I]\!] \to [\![I/R]\!]}{\Gamma \vdash [M]_R = c_{[\![R]\!]} M : I/R}$$

The term "pick $x \in a$ in N" extends the term N depending on a value in I to one depending on I/R, provided that N yields the same result for any values that are related by R.

$$\frac{\Gamma, x : I \vdash N : J \quad \Gamma, a : I, b : I \mid R(a,b) \vdash N[a/x] = N[b/x]}{\Gamma, a : I/R \vdash \text{pick } x \in a \text{ in } N : J}$$

We interpret "pick $x \in a$ in N" in the following way: we assume that $\Gamma = \overrightarrow{x : I}$. By applying Lawvere equality for all variables in Γ, we infer:

$$\frac{\overrightarrow{x : I}, a : I, b : I \mid R(a,b) \vdash N[a/x] = N[b/x]}{\overrightarrow{x : I}, a : I, \overrightarrow{x' : I}, b : I \mid \overrightarrow{x = x'}, R(a,b) \vdash N[a/x] = N[b/x]}$$

This means that in $\Delta^* \mathbb{E}_{[\![\Gamma]\!] \times [\![I]\!]}$ we have an inequality:

$$EQ([\![\Gamma]\!]) \mathbin{\dot\times} R \cong EQ([\![I_1]\!]) \mathbin{\dot\times} \cdots \mathbin{\dot\times} EQ([\![I_n]\!]) \mathbin{\dot\times} R \leq [\![N]\!]^* EQ([\![J]\!]),$$

which yields a morphism $f : EQ([\![\Gamma]\!]) \mathbin{\dot\times} R \to EQ([\![J]\!])$ in $\Delta^* \mathbb{E}$. By transposing this morphism by adjunction $L \dashv EQ$, we obtain $\underline{f} : ([\![\Gamma]\!] \times [\![I]\!])/(EQ([\![\Gamma]\!]) \times [\![R]\!]) \to J$. From the Frobenius property, we obtain $\underline{f} \circ h : [\![\Gamma]\!] \times [\![I]\!]/[\![R]\!] \to [\![J]\!]$, where $h : [\![\Gamma]\!] \times [\![I]\!]/[\![R]\!] \to ([\![\Gamma]\!] \times [\![I]\!])/(EQ([\![\Gamma]\!]) \dot\times [\![R]\!])$ is the isomorphism. With this morphism we interpret:

$$[\![\text{pick } x \in a \text{ in } N]\!] = \underline{f} \circ h.$$

In the internal logic, we have the following axioms between terms:

$$\text{pick } x \in [M]_R \text{ in } N]\!] = [\![N[M/x]]\!], \qquad \text{pick } x \in M \text{ in } N[[x]_R/y]]\!] \quad = \quad [\![N[M/y]$$

which will soundly be interpreted in the fibration supporting $\mathcal{L}_{-/-}$.

Sequent Inference Rule

$$\frac{\Gamma \vdash M : I \quad \Gamma \vdash N : I \quad x : I, y : I \vdash R}{\Gamma \mid R[M/x, N/y] \vdash [M]_R = [N]_R}$$

This rule asserts that any terms related by the relation belong to the same equivalence class.

To interpret this rule, we consider the unit $\eta_{[\![R]\!]} : [\![R]\!] \to EQ(L[\![R]\!])$ of the adjunction $L \dashv EQ$. This yields an inequality $[\![R]\!] \leq c^*_{[\![R]\!]}(EQ(L[\![R]\!]))$ in $\Delta^*\mathbb{E}_{[\![I]\!]}$. This inequality is equivalent to $[\![R]\!] \leq (c_{[\![R]\!]} \times c_{[\![R]\!]})^*(EQ(L[\![R]\!]))$ in $\mathbb{E}_{[\![I]\!]\times[\![I]\!]}$. Thus we obtain an inequality $[\![R[M/x, N/y]]\!] \leq \langle [\![[M]_R]\!], [\![[N]_R]\!] \rangle^*(EQ(L[\![R]\!])) \cong [\![[M]_R = [N]_R]\!]$, which gives the interpretation of the above rule.

2.5.1 Predicates Definable in Internal Logic

Let $p : \mathbb{E} \to \mathbb{B}$ be a fibration supporting $\mathcal{L}_\wedge, \mathcal{L}_=, \mathcal{L}_\exists$.

Definition 2.5.2 Let I, J be objects in $\mathbb{B}$ and R be an object in $\mathbb{E}_{I\times J}$.

1. R is *bijective* if the following holds in the internal logic.

$$x_1 : I, y_1 : J, x_2 : I, y_2 : J \mid R(x_1, y_1), R(x_2, y_2), x_1 = x_2 \vdash y_1 = y_2$$

$$x_1 : I, y_1 : J, x_2 : I, y_2 : J \mid R(x_1, y_1), R(x_2, y_2), y_1 = y_2 \vdash x_1 = x_2$$

2. R is *total* if the following holds in the internal logic.

$$x : I \vdash \top \mid \exists y : J \,.\, R(x, y)$$

$$x : J \vdash \top \mid \exists y : I \,.\, R(y, x).$$

2.5.2 Partial Equivalence Relations

Let $p : \mathbb{E} \to \mathbb{B}$ be a fibration supporting $\mathcal{L}_\wedge, \mathcal{L}_=, \mathcal{L}_{\{-\}}, \mathcal{L}_{-/-}$.

Definition 2.5.3 A *partial equivalence relation* (or simply *PER*) R over an object I in $\mathbb{B}$ is an object R in $\mathbb{E}_{I\times I}$ such that

- R is symmetric, that is, $x : I, y : I \mid R(x, y) \vdash R(y, x)$ holds, and
- R is transitive, that is, $x : I, y : I, z : I \mid R(x, y), R(y, z) \vdash R(x, z)$ holds.

We denote $\mathbf{PER}(p)$ for the full subcategory of $\Delta^*\mathbb{E}$ whose objects are partial equivalence relations. We write $q_p : \mathbf{PER}(p) \to \mathbb{B}$ for the restriction of $\Delta^* p$ to $\mathbf{PER}(p)$ (q_p is indeed a fibration). □

Proposition 2.5.4 *$\mathbf{PER}(p)$ has finite products which are strictly preserved by q_p.* □

PROOF Since p has fibred finite products and $\mathbb{B}$ has finite products, by proposition 2.4.6 and proposition 2.4.3, $\Delta^*\mathbb{E}$ also has finite products. We thus only need to check that for objects R and S in $\mathbf{PER}(p)$, $R \dot{\times} S$ is again an object in $\mathbf{PER}(p)$. Let R, S be objects in $\mathbf{PER}(p)$. We show that $R \dot{\times} S$ is a symmetric and transitive relation. We let $I = q_p R, J = q_p S, x : I \times J, y : I \times J$ and assume $R(\pi x, \pi y) \wedge S(\pi' x, \pi' y)$. Since R and S are symmetric relations, we have $R(\pi y, \pi x) \wedge S(\pi' y, \pi' x)$, which is $R \dot{\times} S(y, x)$. We can similarly show that $R \dot{\times} S$ is a transitive relation. ■

Proposition 2.5.5 *If p has fibred exponentials and simple products, then $\mathbf{PER}(p)$ has exponentials.* □

PROOF The assumption on p equips q_p with fibred exponentials and simple products. From proposition 2.4.20, $\Delta^* p$ has a CCC structure which is strictly preserved by q_p. We thus only need to check that $\mathbf{Obj}(\mathbf{PER})$ is closed under construction of exponential objects. Let R, S be objects in $\mathbf{PER}(p)$, $I = q_p R, J = q_p S$, $f : I \Rightarrow J, g : I \Rightarrow J$ and assume $\forall (x, y) \in I \times J \,.\, R(x, y) \implies S(f(x), g(y))$ (this formula expresses $R \dot{\Rightarrow} S$ through the internal logic of fibration). From the symmetry of R and S, it is easy to see that $R \dot{\Rightarrow} S$ is a symmetric relation. We can similarly show that $R \dot{\Rightarrow} S$ is a transitive relation. ■

Definition 2.5.6 We define a functor $|-| : \mathbf{PER}(p) \to \mathbb{B}$ and $[-] : \mathbf{PER}(p) \to \mathbb{B}$ as follows. Let R be an object in $\mathbf{PER}(p)$ above an object I in $\mathbb{B}$. By $x : I \vdash r(R)\ x$ we mean the predicate $R(x, x)$.

1. The functor $|-|$ sends R to the subset type $\{x : I \mid r(R)\ x\}$.

2. The functor $[-]$ sends R to the following quotient type:

$$\dfrac{\dfrac{\dfrac{x : |R| \vdash x : |R|}{x : |R| \vdash o_R\, x : I} \quad \dfrac{y : |R| \vdash y : |R|}{y : |R| \vdash o_R\, y : I}}{x : |R|, y : |R| \vdash R(o_R\, x, o_R\, y)}}{|R|/R(o_R\, x, o_R\, y)}$$

Lemma 2.5.7 *1. The functor $|-| : \mathbf{PER}(p) \to \mathbb{B}$ preserves finite products.*

2. The functor $[-] : \mathbf{PER}(p) \to \mathbb{B}$ preserves finite products. □

PROOF Let R and S be PERs over I and J respectively. We construct terms in the internal logic of fibration and show that they form an isomorphism by term calculation.

1. We define $p : |R| \times |S| \vdash F : |R \mathbin{\dot\times} S|$ and $p : |R \mathbin{\dot\times} S| \vdash G : |R| \times |S|$ by

$$\begin{aligned} F &= i_{r(R \mathbin{\dot\times} S)}(o_{r(R)}(\pi p), o_{r(S)}(\pi' p)) \\ G &= (i_{r(R)}(\pi(o_{r(R \mathbin{\dot\times} S)} p)), i_{r(R)}(\pi'(o_{r(R \mathbin{\dot\times} S)} p))) \end{aligned}$$

   Easy calculation shows that $G[F/p] = p$ and $F[G/p] = p$. Therefore, the morphisms corresponding to F and G give the desired isomorphism.

2. We define $p : [R \mathbin{\dot\times} S] \vdash F : [R] \times [S]$ and $q : [R] \times [S] \vdash G : [R \mathbin{\dot\times} S]$ by:

$$\begin{aligned} F &= \text{pick } x \in p \text{ in } ([i_R(\pi(o_{R \mathbin{\dot\times} S} x))]_R, [i_R(\pi'(o_{R \mathbin{\dot\times} S} x))]_S) \\ G &= \text{pick } r \in \pi\, q \text{ in pick } s \in \pi'\, q \text{ in } [i_{R \mathbin{\dot\times} S}(o_R r, o_R s)]_{R \mathbin{\dot\times} S}. \end{aligned}$$

   We show that $G[F/q] = p$ and $F[G/p] = q$. Below, for readability, we omit subscripts of i and o. We first show that $G[F/q] = p$.

$$\begin{aligned} G[F/q] &= \text{pick } r \in \pi F \text{ in pick } s \in \pi' F \text{ in } [i(or, os)]_{R \mathbin{\dot\times} S} \\ &= \text{pick } x \in p \text{ in } [i(o(i(\pi(ox))), o(i(\pi'(ox))))]_{R \mathbin{\dot\times} S} \\ &= \text{pick } x \in p \text{ in } [i(\pi(ox), \pi'(ox))]_{R \mathbin{\dot\times} S} \\ &= \text{pick } x \in p \text{ in } [i(ox)]_{R \mathbin{\dot\times} S} \\ &= \text{pick } x \in p \text{ in } [x]_{R \mathbin{\dot\times} S} \\ &= p. \end{aligned}$$

Next, we show $F[G/p] = q$.

$$
\begin{aligned}
& F[G/p] \\
= \;& \text{pick } x \in G \text{ in } ([i(\pi(ox))]_R, [i(\pi'(ox))]_S) \\
= \;& \text{pick } r \in \pi q \text{ in pick } s \in \pi' q \text{ in } ([i(\pi(o(i(or, os))))]_R, [i(\pi'(o(i(or, os))))]_S) \\
= \;& \text{pick } r \in \pi q \text{ in pick } s \in \pi' q \text{ in } ([i(or)]_R, [i(os)]_S \\
= \;& \text{pick } r \in \pi q \text{ in pick } s \in \pi' q \text{ in } ([r]_R, [s]_S) \\
= \;& (\pi q, \pi' q) \\
= \;& q.
\end{aligned}
$$

Therefore, the morphisms corresponding to F and G give the desired isomorphism. ∎

Chapter 3

Pre-Logical Predicates for Simply Typed Formal Systems

3.1 Introduction

Pre-logical predicates (relations) [HS02] are a generalisation of logical predicates. They are defined for the *simply typed lambda calculus* and its set-theoretic environmental models called *lambda applicative structures* [Mit96]. Two important properties are enjoyed by pre-logical predicates but not logical predicates. One is that pre-logical predicates are *equivalent* to predicates satisfying the basic lemma (interpretation of all terms respects predicates — this is the key to many applications of logical predicates), and the other is that binary pre-logical relations are closed under relational composition.

We aim to generalise pre-logical predicates from the simply typed lambda calculus to arbitrary *simply typed formal systems* (we just say *typed formal system* below) and their categorical models, then show that the above important properties hold in this generalised setting.

This generalisation enables us to extend pre-logical predicates systematically to other calculi, such as lambda calculus with various type constructors and variable binders, and calculi other than lambda calculus, such as logics and process calculi. This opens up the possibility of characterising observational equivalence [HS02] and

constructive data refinement [HLST00] in various non-lambda calculi.

There are three underlying elements on which pre-logical predicates are defined: syntax (the simply typed lambda calculus), semantics (set-theoretic environmental models) and predicates (as subsets of carrier sets). We generalise these three elements along the following dimensions:

- We generalise syntax to an arbitrary typed formal system described by a *typed binding signature* [MS03]. A typed formal system is a formal system whose inference rules fit within the following scheme:

$$\frac{\Gamma, \overrightarrow{x_1 : \tau_1} \vdash M_1 : \sigma_1 \quad \cdots \quad \Gamma, \overrightarrow{x_n : \tau_n} \vdash M_n : \sigma_n}{\Gamma \vdash o(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n) : \tau}$$

  This is general enough to subsume various simple type systems and calculi such as the simply typed lambda calculus, many-sorted first-order logic, pi-calculus, etc.

- We generalise from set-theoretic to category-theoretic semantics. Following the principle of categorical semantics, we give a semantics of a typed formal system in a Cartesian category $\mathbb{C}$ by mapping types to objects and terms to morphisms in $\mathbb{C}$.

- As we move to category theory, we need to change the notion of predicates from subsets to appropriate category-theoretic constructs. We use *subscones*, which is a mild generalisation of the injective scones of [MS93].

We represent all three elements as objects and morphisms in the category of *presentation models* $\mathbb{P}_T$, where T is the set of types [MS03]. In this category, the collection of well-formed terms modulo α-equivalence is represented as the initial algebra of an endofunctor corresponding to a typed binding signature.

After this generalisation, we formulate pre-logical predicates and predicates satisfying the basic lemma, and show their equivalence. Then we show that binary pre-logical relations are closed under composition of binary pre-logical relations.

In the next chapter, we examine the generalisation of pre-logical predicates in this chapter by instantiating these general definitions to several simply typed formal systems and their semantics. We consider the case of many-sorted algebra, the simply

typed lambda calculus with products, Moggi's computational metalanguage and first-order logic.

3.2 Category of Presentation Models

We introduce the category of *contexts* and the category of *presentation models* [MS03]. We represent all three elements involved in the notion of pre-logical predicates (syntax, semantics and predicates) in this category. We reserve the letter T to denote any set of types, which are ranged over by Greek letters τ and σ.

Definition 3.2.1 We define the *category $\mathbb{V}_T$ of T-contexts* as follows:

An object is a T-context (see section 2.1).

A morphism from Γ to Γ' is a function $f : \mathrm{dom}(\Gamma) \to \mathrm{dom}(\Gamma')$ such that $\Gamma = \Gamma' \circ f$.

For a morphism $f : \Gamma \to \Gamma'$ in $\mathbb{V}_T$, we define a *variable renaming substitution* θ_f to be $[f(x_1)/x_1, \cdots, f(x_n)/x_n]$, where $\{x_1, \cdots, x_n\} = \mathrm{dom}(\Gamma)$. □

Category $\mathbb{V}_T$ has finite coproducts (written $\uplus$). For any objects Γ_1 and Γ_2 in $\mathbb{V}_T$, we assume that $\Gamma_1 \uplus \Gamma_2$ is equal to $\Gamma_1 \cup \Gamma_2'$ where Γ_2' is chosen to be isomorphic to Γ_2 and satisfy $\mathrm{dom}(\Gamma_2') \cap \mathrm{dom}(\Gamma_1) = \emptyset$. By $\Gamma, x_1 : \tau_1, \cdots, x_n : \tau_n$, we mean a context $\Gamma \uplus \{\mathbf{v}_1 : \tau_1, \cdots, \mathbf{v}_n : \tau_n\}$ such that x_i is equal to $\iota_r(\mathbf{v}_i)$, where $\iota_r : \{\mathbf{v}_1 : \tau_1, \cdots, \mathbf{v}_n : \tau_n\} \to \Gamma \uplus \{\mathbf{v}_1 : \tau_1, \cdots, \mathbf{v}_n : \tau_n\}$ is the right injection.

We note that $\mathbb{V}_T$ is a free co-Cartesian category generated from T.

Definition 3.2.2 ([MS03]) We define the category of *presentation models* to be the functor category $\mathbb{P}_T = [\mathbb{V}_T \times T, \mathbf{Sets}]$. □

Intuitively, a presheaf (a covariant functor from $\mathbb{V}_T \times T$ to $\mathbf{Sets}$) F in $\mathbb{P}_T$ at an object (Γ, τ) in $\mathbb{V}_T \times T$ is a set of some entities which take inputs of shape Γ and output a value of type τ. Such an entity could be anything; for example, a well-formed term M of type τ under a context Γ of some type system can be such an entity by regarding free variables as inputs and the term itself as an output. Another example of an entity is a morphism in a category. That F is a presheaf means that such entities is equipped with

an *action* with respect to morphisms in $\mathbb{V}_T$. A typical action is to apply a permutation expressed by a morphism in $\mathbb{V}_T$ to the inputs of the entity. For example, for an entity $a \in F\Gamma$ and a morphism $f : \Gamma \to \Gamma'$ in $\mathbb{V}_T$, $Ff(a)$ represents the operation which first applies a permutation f to its input, then passes the result to Fa.

Proposition 3.2.3 *Category $\mathbb{P}_T$ has small limits, small colimits and exponentials.* □

PROOF See e.g. [MM92]. ■

In fact $\mathbb{P}_T$ is a topos, but we do not use this fact in this thesis.

In [MS03], the category of presentation models is defined as $[\mathbb{V}_T, [T, \mathbf{Sets}]]$, which is isomorphic to $\mathbb{P}_T$. For detail, see [MS03] and its precursor [FPT99].

We introduce a presheaf and endofunctors which are used in the next section to construct the endofunctor corresponding to a typed formal system.

Definition 3.2.4 1. We define the *presheaf Var of variables* in $\mathbb{P}_T$ by

$$Var(\Gamma, \tau) = \{x^\tau \mid \Gamma(x) = \tau\}.$$

2. For an object F in $\mathbb{P}_T$, by $F(- \uplus \Gamma, \tau)$ we mean the presheaf over $\mathbb{V}_T$ sending a context Γ' to the set $F(\Gamma' \uplus \Gamma, \tau)$. We then define an endofunctor $(-)^{\Gamma,\tau} : \mathbb{P}_T \to \mathbb{P}_T$ by $F^{\Gamma,\tau} = F(- \uplus \Gamma, \tau) \circ \pi$, where $\pi : \mathbb{V}_T \times T \to \mathbb{V}_T$ is the first projection.

3. We define an endofunctor $(-)|_\tau : \mathbb{P}_T \to \mathbb{P}_T$ as follows:

$$F|_\tau(\Gamma, \tau') = \begin{cases} \emptyset & (\tau \neq \tau') \\ F(\Gamma, \tau) & (\tau = \tau') \end{cases}$$

Proposition 3.2.5 *Endofunctors $(-)^{\Gamma,\tau}$ and $(-)|_\tau$ preserve pullbacks and colimits.* □

PROOF • We show a stronger statement that $(-)^{\Gamma,\tau}$ preserves limits and colimits. We notice that $F^{\Gamma,\tau} = F \circ g_{\Gamma,\tau}$ where $g_{\Gamma,\tau} : \mathbb{V}_T \times T \to \mathbb{V}_T \times T$ is a functor defined by $g_{\Gamma,\tau}(\Gamma', \tau') = (\Gamma' \uplus \Gamma, \tau)$. It is known that left and right Kan extension of $F : [\mathbb{V}_T \times T, \mathbf{Sets}]$ along $g_{\Gamma,\tau}$ exists (see e.g. [Bor94]). Therefore $(-)^{\Gamma,\tau}$ has both left and right adjoints, thus preserves limits and colimits.

- First we write $\pi' : \mathbb{V}_T \times T \to T$ for the second projection. Then we notice that $F|_\tau \cong (T(\tau, -) \circ \pi') \times F$, because

$$(T(\tau,-) \circ \pi' \times F)(\Gamma, \tau') = T(\tau, \tau') \times F(\Gamma, \tau') \cong \begin{cases} F(\Gamma, \tau') & (\tau = \tau') \\ \emptyset & (\tau \neq \tau') \end{cases} .$$

  The above isomorphism shows that $(-)|_\tau$ preserves pullbacks. Since $\mathbb{P}_T$ is a CCC, $T(\tau, -) \circ \pi' \Rightarrow -$ is a right adjoint of $(-)|_\tau$. Therefore it preserves colimits. ■

Proposition 3.2.6 *For any presheaves A, B and type $\tau \in T$, we have an isomorphism* $\mathbb{P}_T(A|_\tau, B) \cong [\mathbb{V}_T, \mathbf{Sets}](A(-, \tau), B(-, \tau))$. □

PROOF We give a function $h : \mathbb{P}_T(A|_\tau, B) \to [\mathbb{V}_T, \mathbf{Sets}](A(-, \tau), B(-, \tau))$ and its inverse h^{-1} by

$$\begin{aligned} h(\alpha)_\Gamma &= \alpha_{\Gamma,\tau} \\ h^{-1}(\beta)_{\Gamma,\sigma} &= \begin{cases} \beta_\Gamma & (\tau = \sigma) \\ ! & (\tau \neq \sigma) \end{cases} \end{aligned}$$

where $! : \emptyset \to B(\Gamma, \sigma)$ is the unique function in $\mathbf{Sets}$. It is easy to see that they define an isomorphism. ■

3.3 Syntax: Typed Formal Systems

We use *typed binding signatures* [MS03] for describing simply typed formal systems.

Definition 3.3.1 ([MS03]) A *typed binding signature* (ranged over by Π) is a tuple (T, O) where T is the set of *types* and O is the set of *operators* (ranged over by o), each of which is a pair of an *operator symbol* s and its *arity* $((\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n), \tau) \in (T^+)^* \times T$. We write $s^{(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau}$ for such a pair in O [1]. A *typed first-order signature* (ranged over by Σ) is just a typed binding signature (T, O) such that for each operator $s^{(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau} \in O$, $\overrightarrow{\tau_i} = \epsilon$. It coincides with the notion of many-sorted signature. □

[1] This definition of typed binding signature is a special case of the one in [MS03] where the set of types allowed for variables is equal to the set of all types. We also use Π to range over typed binding signatures instead of Σ, to distinguish them from typed first-order signatures.

A typed binding signature Π specifies a *typed formal system*. We first define the set of *raw*-Π *terms* (ranged over by M, N) by the BNF $M ::= x^\tau \mid o(\overrightarrow{x^\tau}.M, \cdots, \overrightarrow{x^\tau}.M)$. In this BNF, $\overrightarrow{x^\tau}.M$ means binding of variables $\overrightarrow{x^\tau}$ in M. As usual, we identify α-equivalent terms. The typed formal system is a system to derive judgements of the form $\Gamma \vdash_\Pi M : \tau$. The system consists of the following rules:

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash_\Pi x^\tau : \tau} \quad \frac{\Gamma' \vdash_\Pi M : \tau \quad f \in \mathbb{V}_T(\Gamma', \Gamma)}{\Gamma \vdash_\Pi M\theta_f : \tau}$$

$$\frac{\Gamma, \overrightarrow{x_1 : \tau_1} \vdash_\Pi M_1 : \sigma_1 \quad \cdots \quad \Gamma, \overrightarrow{x_n : \tau_n} \vdash_\Pi M_n : \sigma_n}{\Gamma \vdash_\Pi s^{(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau}(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n) : \tau}$$

The first rule is the introduction rule for variables. The second rule is a compact representation of the structural rules. It is equivalent to the following three rules (weakening, contraction and variable renaming):

$$\frac{\Gamma \vdash_\Pi M : \tau}{\Gamma, x : \tau \vdash_\Pi M : \tau} \quad \frac{\Gamma, y : \tau, z : \tau \vdash_\Pi M : \tau}{\Gamma, x : \tau \vdash_\Pi M[x/z, x/y] : \tau} \quad \frac{\Gamma, y : \tau \vdash_\Pi M : \tau}{\Gamma, x : \tau \vdash_\Pi M[x/y] : \tau} \tag{3.1}$$

The third rule is the rule for an operator $s^{(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau} \in O$.

Example 3.3.2 In this chapter we use the simply typed lambda calculus as a running example of a typed formal system. The lambda calculus we consider here is the minimal fragment built over a set of base types B. The set of types is defined by the BNF

$$\mathbf{Typ}^{\Rightarrow}(B) \ni \tau ::= b \mid \tau \Rightarrow \tau$$

and the calculus has the following standard rules (apart from the structural rules in (3.1)):

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x^\tau : \tau} \quad \frac{\Gamma, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x^\tau \,.\, M : \tau \Rightarrow \tau'} \quad \frac{\Gamma \vdash M : \tau \Rightarrow \tau' \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \tau'}$$

The above rules fit into the scheme of typed formal systems. The inference rule for lambda abstraction takes a term M of type τ' and binds the free variable x^τ in M, then yields a term $\lambda x^\tau \,.\, M$ of type $\tau \Rightarrow \tau'$. Thus this step can be regarded as an operation (namely lam) whose arity is $(\tau, \tau') \to \tau \Rightarrow \tau'$. We should not confuse $\to$ and $\Rightarrow$ here: $\to$ is used for writing the arity of the operator while $\Rightarrow$ is the function type in the

lambda calculus. The inference rule for application takes two terms M of type $\tau \Rightarrow \tau'$ and N of type τ, and constructs a term MN of type τ'. This step can be regarded as an operation (namely app) whose arity is $\tau \Rightarrow \tau', \tau \to \tau'$.

To summarise, the typed binding signature for the simply typed lambda calculus (over the set of base types B) is given by:

$$\Pi_\lambda = (\mathbf{Typ}^{\Rightarrow}(B), \{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}, \mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}\})$$

where τ, τ' ranges over $\mathbf{Typ}^{\Rightarrow}(B)$.

A more complex example involving bindings is case syntax for coproduct types. Suppose that the set of types is extended with coproduct types $\tau + \tau'$. Case syntax for coproducts is given as follows:

$$\frac{\Gamma \vdash M : \boldsymbol{\tau + \tau'} \quad \Gamma, x : \boldsymbol{\tau} \vdash N : \boldsymbol{\tau''} \quad \Gamma, y : \boldsymbol{\tau'} \vdash L : \boldsymbol{\tau''}}{\Gamma \vdash \text{case } M \text{ of } \mathsf{inl}(x^\tau) \to N \text{ or } \mathsf{inr}(y^{\tau'}) \to L : \boldsymbol{\tau''}}$$

To find the arity of this operator, we look at the types appearing in both contexts and entailment emphasised with bold face. This tells us that the arity of case syntax is $\tau + \tau', (\tau, \tau''), (\tau', \tau'') \to \tau''$. □

To a typed binding signature Π, we assign an endofunctor Π (we use the same letter as the signature) over $\mathbb{P}_T$ as follows.

$$\Pi A = Var + \coprod_{s(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau\in O} \left(\prod_{1\le i\le n} A^{\overrightarrow{\tau_i},\sigma_i}\right)\Bigg|_\tau .$$

We are interested in algebras of Π. We give an explicit description of a Π-algebra by the following lemma.

Lemma 3.3.3 *There is a bijective correspondence between a* Π*-algebra* $u : \Pi A \to A$ *and a choice of morphisms* $u_v : Var \to A$ *in* $\mathbb{P}_T$ *and* $u_o : \prod_{i=1}^m A(- \uplus \overrightarrow{\tau_i}, \sigma_i) \to A(-, \tau)$ *in* $[\mathbb{V}_T, \mathbf{Sets}]$ *for each operator* $o \in O$ *of arity* $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$. □

PROOF We expand the definition of Π and use the property of coproducts.

$$\begin{aligned}
& \mathbb{P}_T\left(\Pi A, A\right) \\
\cong\ & \mathbb{P}_T\left(Var + \coprod_{s(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau\in O}\left(\prod_{i=1}^{n} A^{\overrightarrow{\tau_i},\sigma_i}\right)\Big|_{\tau}, A\right) \\
\cong\ & \mathbb{P}_T\left(Var, A\right) \times \prod_{s(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau\in O} \mathbb{P}_T\left(\prod_{i=1}^{n} A^{\overrightarrow{\tau_i},\sigma_i}\Big|_{\tau}, A\right)
\end{aligned}$$

We note that $\prod_{i=1}^{n} A^{\overrightarrow{\tau_i},\sigma_i} = \prod_{i=1}^{n}(A(-\uplus\overrightarrow{\tau_i},\sigma_i)\circ\pi) \cong (\prod_{i=1}^{n} A(-\uplus\overrightarrow{\tau_i},\sigma_i))\circ\pi$. We expand the summand of the above big coproducts by proposition 3.2.6:

$$\begin{aligned}
& \mathbb{P}_T\left(\prod_{i=1}^{n} A^{\overrightarrow{\tau_i},\sigma_i}\Big|_{\tau}, A\right) \\
\cong\ & [\mathbb{V}_T, \mathbf{Sets}]\left(\left(\left(\prod_{i=1}^{n} A(-\uplus\overrightarrow{\tau_i},\sigma_i)\right)\circ\pi\right)(-,\tau), A(-,\tau)\right) \\
=\ & [\mathbb{V}_T, \mathbf{Sets}]\left(\prod_{i=1}^{n} A(-\uplus\overrightarrow{\tau_i},\sigma_i), A(-,\tau)\right)
\end{aligned}$$

The above isomorphism tells us that Π-algebra structure over a presheaf A consists of

1. a morphism $u_v : Var \to A$ in $\mathbb{P}_T$ and
2. a morphism $u_o : \prod_{i=1}^{n} A(-\uplus\overrightarrow{\tau_i},\sigma_i) \to A(-,\tau)$ in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O$ of arity $(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau$. ■

From proposition 3.2.5 and the fact that finite products preserve ω-colimits, the above endofunctor preserves ω-colimits (see e.g. [AC98]). Therefore we can construct an initial Π-algebra using ω-colimits. As one can imagine, this initial algebra corresponds to the set of well-formed terms of the simply typed formal system described by Π. The next theorem states that this intuition is indeed true.

Definition 3.3.4 Let Π be a typed binding signature. We define the presheaf S_Π in $\mathbb{P}_T$ by

$$\begin{aligned}
S_\Pi(\Gamma,\tau) &= \{M \mid \Gamma \vdash_\Pi M : \tau\} \\
S_\Pi(f,\tau)(M) &= M\theta_f
\end{aligned}$$

□

Theorem 3.3.5 ([MS03]) *S_Π has an initial Π-algebra structure $\alpha_\Pi : \Pi S_\Pi \to S_\Pi$.* □

PROOF We first give a Π-algebra structure α_Π over S_Π. From lemma 3.3.3, it suffices to give a morphism $\alpha_v : Var \to S_\Pi$ in $\mathbb{P}_T$ and a morphism $\alpha_o : \prod_{i=1}^n S_\Pi(- \uplus \overrightarrow{\tau_i}, \sigma_i) \to S_\Pi(-, \tau)$ in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$.

- We define α_v by $(\alpha_v)_{(\Gamma,\tau)}(x^\tau) = x^\tau$ for each $(\Gamma, \tau) \in \mathbb{V}_T \times T$.
- We define α_o for each operator $o \in O$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$ and well-formed terms $\Gamma, \overrightarrow{x_i : \tau_i} \vdash_\Pi M_i : \sigma_i \ (1 \leq i \leq n)$ by

$$(\alpha_o)_\Gamma(M_1, \cdots, M_n) = o(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n)$$

To show initiality, let $(A, \beta : \Pi A \to A)$ be a Π-algebra. We write β_v, β_o for the morphisms specified by lemma 3.3.3. We then define a Π-algebra morphism $h : S_\Pi \to A$ by induction on the structure of terms:

$$\begin{aligned} h_{(\Gamma,\tau)}(x^\tau) &= \beta_v(x^\tau) \\ h_{(\Gamma,\tau)}(o(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n)) &= \beta_o(h_{(\Gamma \uplus \overrightarrow{\tau_1}, \sigma_1)}(M_1), \cdots, h_{(\Gamma \uplus \overrightarrow{\tau_n}, \sigma_n)}(M_n)). \end{aligned}$$

It is easy to see that h is the unique Π-algebra morphism by induction. ■

3.4 Semantics: Weak Categorical Interpretation

We formulate the semantics of a typed formal system $\Pi = (T, O)$ by a morphism to an object in $\mathbb{P}_T$ which has the structure of a Cartesian category. The notion of semantics considered here is very weak in the sense that it does not exploit any categorical structure other than finite products. The semantics only keeps the basic principle of categorical model theory: that is, types are interpreted as objects and terms are interpreted as morphisms (but not substitution-as-composition).

Definition 3.4.1 1. An *interpretation of types* is just a functor $F : T \to \mathbb{C}$ where $\mathbb{C}$ is a Cartesian category. We extend it to a functor $F^* : \mathbb{V}_T \to \mathbb{C}^{op}$ by

$$\begin{aligned} F^*\Gamma &= F(\gamma^{-1}(1)) \times \cdots \times F(\gamma^{-1}(n)) \\ F^*(f : \Gamma \to \Gamma') &= \langle \pi_{\gamma'(f(\gamma^{-1}(1)))}, \cdots, \pi_{\gamma'(f(\gamma^{-1}(n)))} \rangle. \end{aligned}$$

Here, γ and γ' are the indexing functions for Γ and Γ' respectively (see section 2.1). In this thesis we write $s^F_{\Gamma_1,\cdots,\Gamma_n} : \prod_{i=1}^n F^*\Gamma_i \to F^*(\biguplus_{i=1}^n \Gamma_i)$ for the canonical isomorphism. The superscript F may be omitted if it is obvious from context.

2. For an interpretation of types $F : T \to \mathbb{C}$, we define an object H^F in $\mathbb{P}_T$ by $H^F(\Gamma, \tau) = \mathbb{C}(F^*\Gamma, F\tau)$. Let $\mathbb{D}$ be a Cartesian category. For a functor $G : \mathbb{C} \to \mathbb{D}$ preserving finite products strictly, we define a morphism $H^G : H^F \to H^{GF}$ in $\mathbb{P}_T$ by $H^G(\Gamma, \tau)(f : F^*\Gamma \to F\tau) = Gf : (GF)^*\Gamma = G(F^*\Gamma) \to GF\tau$.

3. A *categorical interpretation of* Π consists of a Cartesian category $\mathbb{C}$, an interpretation of types $F : T \to \mathbb{C}$ and a morphism $i : S_\Pi \to H^F$ in $\mathbb{P}_T$ called the *interpretation of terms*.

   Unless stated explicitly, we use the notation $\mathbb{C}[\![-]\!]_F$ to range over categorical interpretations. In this notation, $\mathbb{C}$ stands for the interpretation category, F for the interpretation of types, and $\mathbb{C}[\![-]\!]_F$ itself for the interpretation of terms. The interpretation of a well-formed term $\Gamma \vdash_\Pi M : \tau$ by $\mathbb{C}[\![-]\!]_F$ is written by $\mathbb{C}[\![M]\!]_F$ (component selection of $\mathbb{C}[\![-]\!]_F$ at (Γ, τ) is implicit).

4. We say that a categorical interpretation $\mathbb{C}[\![-]\!]_F$ satisfies the *semantic substitution lemma* if for all well-formed terms $\Delta_1 \vdash_\Pi M_1 : \tau_1, \cdots, \Delta_n \vdash_\Pi M_n : \tau_n$ and $x_1 : \tau_1, \cdots, x_n : \tau_n \vdash_\Pi M : \tau$, the following holds:

   $$\mathbb{C}[\![M]\!]_F \circ \langle \mathbb{C}[\![M_1]\!]_F, \cdots, \mathbb{C}[\![M_n]\!]_F \rangle = \mathbb{C}[\![M[M_1\theta_{\iota_1}/x_1, \cdots, M_n\theta_{\iota_n}/x_n]]\!]$$

   where $\iota_i : \Delta_i \to \uplus_{j=1}^n \Delta_j$ is the i-th injection. □

Definition 3.4.2 Let $\mathbb{C}[\![-]\!]_{F_1}, \mathbb{C}[\![-]\!]_{F_2}$ be categorical interpretations of Π in $\mathbb{C}$. We define the product interpretation $\mathbb{C}[\![-]\!]_{F_1} \times \mathbb{C}[\![-]\!]_{F_2}$ of Π in $\mathbb{C} \times \mathbb{C}$ by the following data:

- The interpretation of types is given by $\langle F_1, F_2 \rangle : T \to \mathbb{C} \times \mathbb{C}$. We note that $H^{\langle F_1, F_2 \rangle} = H^{F_1} \times F^{F_2}$.
- The interpretation of terms is given by $\langle \mathbb{C}[\![-]\!]_{F_1}, \mathbb{C}[\![-]\!]_{F_2} \rangle : S_\Pi \to H^{F_1} \times H^{F_2}$.

□

Example 3.4.3 (Continued from example 3.3.2) A *lambda applicative structure* $\mathcal{A}$ consists of a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of non-empty sets A called *carrier sets* (which can be identified with an interpretation of types $A : \mathbf{Typ}^{\Rightarrow}(B) \to \mathbf{Sets}$), a family of *application operators* $- \bullet^{\tau,\tau'} - : A(\tau \Rightarrow \tau') \times A\tau \to A\tau'$ and a *meaning function* $\mathcal{A}[\![-]\!]$ which maps a well-formed term $\Gamma \vdash_{\Pi_\lambda} M : \tau$ to a function $A^*\Gamma \to A\tau$, such that: [2]

$$\begin{aligned}
\mathcal{A}[\![x^\tau]\!]\rho &= \pi_{\gamma(x)}\rho \\
\mathcal{A}[\![MN]\!]\rho &= \mathcal{A}[\![M]\!]\rho \bullet \mathcal{A}[\![N]\!]\rho \\
\mathcal{A}[\![\lambda x^\tau \,.\, M]\!]\rho &= \mathcal{A}[\![\lambda y^\tau \,.\, M[y^\tau/x^\tau]]\!]\rho \\
\mathcal{A}[\![M\theta_f]\!]\rho &= \mathcal{A}[\![M]\!]((A^* f)\rho) \quad (\Gamma \vdash M : \tau, f \in \mathbb{V}_T(\Gamma, \Gamma')).
\end{aligned}$$

A lambda applicative structure specifies an interpretation $(A, \mathcal{A}[\![-]\!])$ of Π_λ in $\mathbf{Sets}$. We write $\mathcal{A}[\![-]\!]$ for this interpretation. □

Interpretation of Terms by Initiality

The presheaf of well-formed terms is equipped with an initial Π-algebra structure (theorem 3.3.5). Thus we can use initiality to obtain an interpretation of terms. Indeed H^F is often equipped with a Π-algebra structure. In this case initiality yields the unique Π-algebra morphism $! : S_\Pi \to H^F$. This is so-called *initial algebra semantics* [FPT99, MS03].

For the presheaf H^F of an interpretation of types $F : T \to \mathbb{C}$, there always exists an evident morphism $u_v : Var \to H^F$ defined by $(u_v)_{(\Gamma,\tau)}(x^\tau) = \pi_{\gamma(x)}$, which gives the natural interpretation of variables by projections. Thus to specify a Π-algebra structure over H^F, it suffices to specify a morphism

$$u_o : \prod_{i=1}^{m} H^F(- \uplus \overrightarrow{\tau_i}, \sigma_i) \to H^F(-, \tau)$$

in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$ (see lemma 3.3.3).

[2] The above conditions are an adaptation of *acceptable meaning function* in [Mit96], section 8.2.2. In this thesis we added the last condition, which is a natural requirement missing in [Mit96].

Example 3.4.4 ([FPT99, MS03]) (Continued from example 3.3.2) We give a categorical interpretation of Π_λ in a CCC $\mathbb{C}$. We take an interpretation of types $F : \mathbf{Typ}^{\Rightarrow}(B) \to \mathbb{C}$ satisfying $F(\tau \Rightarrow \tau') = F(\tau) \Rightarrow F(\tau')$.

To give a Π_λ-algebra structure over H^F, we specify two morphisms in $[\mathbb{V}_T, \mathbf{Sets}]$:

$$u_{\mathsf{lam}^{\tau,\tau'\to\tau\Rightarrow\tau'}} : H^F(- \uplus \tau, \tau') \to H^F(-, \tau \Rightarrow \tau')$$
$$u_{\mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}} : H^F(-, \tau \Rightarrow \tau') \times H^F(-, \tau) \to H^F(-, \tau').$$

- We define $(u_{\mathsf{lam}})_\Gamma(f) = \lambda(f \circ s_{\Gamma,\tau})$ for $f \in H^F(\Gamma \uplus \tau, \tau')$. The following steps show how u_{lam} forms the result.

$$\frac{\dfrac{f : F^*(\Gamma \uplus \tau) \to F\tau'}{f \circ s_{\Gamma,\tau} : F^*\Gamma \times F\tau \to F\tau'}}{\lambda(f \circ s_{\Gamma,\tau}) : F^*\Gamma \to F\tau \Rightarrow F\tau'}$$

- We define $(u_{\mathsf{app}})_\Gamma(f, g) = @ \circ \langle f, g\rangle$ for $f \in H^F(\Gamma, \tau \Rightarrow \tau')$ and $g \in H^F(\Gamma, \tau)$, where $@ : F(\tau) \Rightarrow F(\tau') \times F(\tau) \to F(\tau')$ is the evaluation map in $\mathbb{C}$. The following step shows how u_{app} forms the result.

$$\frac{f : F^*\Gamma \to F\tau \Rightarrow F\tau' \quad g : F^*\Gamma \to F\tau}{@ \circ \langle f, g\rangle : F^*\Gamma \to F\tau'}.$$

These morphisms plus $v : Var \to H^F$ determine a Π_λ-algebra structure over H^F. From initiality, there exists a unique Π_λ-algebra morphism $\mathbb{C}[\![-]\!]_F : S_{\Pi_\lambda} \to H^F$. Thus we obtain a categorical interpretation $(F, \mathbb{C}[\![-]\!]_F)$ ($\mathbb{C}[\![-]\!]_F$ for short). By expanding the definitions, $\mathbb{C}[\![-]\!]_F$ coincides with the standard interpretation of the lambda terms in $\mathbb{C}$ defined by induction on the derivation:

$$\begin{aligned}
\mathbb{C}[\![x^\tau]\!]_F &= \pi_{\gamma(x)} \\
\mathbb{C}[\![\lambda x^\tau\ .\ M]\!]_F &= \lambda(\mathbb{C}[\![M]\!]_F \circ s_{\Gamma,\tau}) \\
\mathbb{C}[\![MN]\!]_F &= @ \circ \langle \mathbb{C}[\![M]\!]_F, \mathbb{C}[\![N]\!]_F\rangle.
\end{aligned}$$

□

Other Interpretations of Terms

Not all interpretations of terms arise from initiality. Let $\Pi_1, \cdots, \Pi_n$ be typed binding signatures having the same set of types, $\mathbb{C}[\![-]\!]_F$ be an interpretation of Π_n and suppose

that each $S_{\Pi_{i+1}}$ $(1 \leq i \leq n-1)$ has a Π_i-algebra structure. From initiality of S_{Π_i}, we have a series of morphisms:

$$S_{\Pi_1} \xrightarrow{!_1} \cdots \xrightarrow{!_{n-1}} S_{\Pi_n} \xrightarrow{\mathbb{C}[\![-]\!]_F} H^F.$$

This situation happens when we give the semantics of a source language S_{Π_1} by multi-stage translation through intermediate languages $S_{\Pi_2}, \cdots, S_{\Pi_n}$. Now $(F, \mathbb{C}[\![-]\!]_F \circ !_{n-1} \circ \cdots \circ !_1)$ gives a categorical interpretation of Π_1, but generally the above composite is not a Π_1-algebra morphism. [3] In section 4.2, we see an example of an interpretation of the simply typed lambda calculus within combinatory algebra through a compilation to combinatory logic terms, and compare the pre-logical relations defined over the lambda calculus and the combinatory logic.

3.5 Predicates: Subscone

We introduce the notion of predicate over a categorical interpretation of types. When types are interpreted as carrier sets, the natural notion of predicate is simply a subset of each carrier set. In categorical settings, carrier sets are replaced by objects in a category, and the notion of predicates is more subtle.

First we recall *injective scones* in [MS93]. An injective scone is the category obtained by pulling back the forgetful functor $p_{\mathbf{Sets}} : \mathbf{Sub}(\mathbf{Sets}) \to \mathbf{Sets}$ along the global section functor $\mathbb{C}(1, -)$ [Jac99]. In this approach, the notion of predicates over an object C in $\mathbb{C}$ is represented as subsets of the global elements of C.

$$\begin{array}{ccc} iSc(\mathbb{C}) & \longrightarrow & \mathbf{Sub}(\mathbf{Sets}) \\ \downarrow & & \downarrow \\ \mathbb{C} & \xrightarrow[\mathbb{C}(1,-)]{} & \mathbf{Sets} \end{array}$$

We use the *subscone* approach [MR92, Laf88, MS93, PPST00], which is a mild generalisation of injective scones. We replace $\mathbf{Sets}$ with a category $\mathbb{D}$ with finite limits and global section functor with finite-product preserving functor.

[3] The possibility of such interpretations seems not to be discussed in [FPT99, MS03]. Definition 3.4.1 is intended to include both initial algebra semantics and non-algebraic semantics.

Definition 3.5.1 ([MR92, PPST00]) Let $\mathbb{C}$ be a category with finite products, $\mathbb{D}$ be a category with finite limits and $G : \mathbb{C} \to \mathbb{D}$ be a functor preserving finite products. From proposition 2.3.10, $p_{\mathbb{D}} : \mathbf{Sub}(\mathbb{D}) \to \mathbb{D}$ is a partially ordered fibration. The *category* $\mathbf{Pred}(G)$ *of* G*-predicates* is the total category obtained by the following change-of-base of $p_{\mathbb{D}}$ along G (see the left diagram):

$$\begin{array}{ccc} \mathbf{Pred}(G) & \longrightarrow & \mathbf{Sub}(\mathbb{D}) \\ {\scriptstyle \pi_G}\downarrow & & \downarrow{\scriptstyle p_{\mathbb{D}}} \\ \mathbb{C} & \xrightarrow[G]{} & \mathbb{D} \end{array} \qquad \begin{array}{ccc} \mathbf{Rel}_n(G) & \longrightarrow & \mathbf{Sub}(\mathbb{D}) \\ {\scriptstyle \pi^n_G}\downarrow & & \downarrow{\scriptstyle p_{\mathbb{D}}} \\ \mathbb{C}^n & \xrightarrow[G\circ\prod]{} & \mathbb{D} \end{array}$$

Similarly, the *category* $\mathbf{Rel}_n(G)$ *of* n*-ary* G*-relations* in the above right diagram is obtained by the change-of-base of $p_{\mathbb{D}}$ along the functor $G \circ \prod$, where $\prod$ is the n-ary product functor. An explicit description of $\mathbf{Pred}(G)$ is the following.

An object in $\mathbf{Pred}(G)$ is a pair (P, C) where P is a subobject of GC in $\mathbb{D}$. Using the internal logic of $p_{\mathbb{D}}$, P can be identified with a predicate judgement $x : GC \vdash P(x)$.

A morphism in $\mathbf{Pred}(G)$ from (P, C) to (P', C') is a morphism $f : C \to C'$ in $\mathbb{C}$ such that $Gf : P \to P'$. In other words, f is a morphism such that $x : GC \mid P(x) \vdash P'(Gf(x))$ holds in the internal logic of $p_{\mathbb{D}}$.

From theorem 2.3.5, π_G (π^n_G) is a partially ordered fibration and is faithful. □

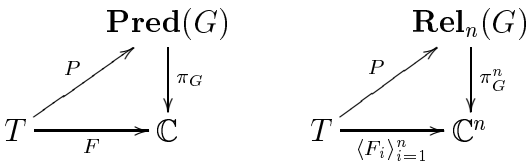
Proposition 3.5.2 *Category* $\mathbf{Pred}(G)$ *(resp.* $\mathbf{Rel}_n(G)$*) has finite products which are strictly preserved by* π_G *(resp.* π^n_G*).* □

PROOF Recall that $p_{\mathbb{D}}$ has fibred finite products (proposition 2.3.10) . Thus π_G (resp. π^n_G) also has fibred finite products (proposition 2.4.3). This implies the condition described in proposition 2.4.6. Therefore $\mathbf{Pred}(G)$ (resp. $\mathbf{Pred}(G)$) has finite products.

■

Definition 3.5.3 In the situation of definition 3.5.1, let $F : T \to \mathbb{C}$ be an interpretation of types. A G*-predicate* (or simply a *predicate*) P over F (written $P \subseteq_G F$)

is an interpretation of types $P : T \to \mathbf{Pred}(G)$ making the following left diagram commute:



Similarly an *n-ary G-relation* (or simply a *relation*) P between $F_i (1 \leq i \leq n)$ is just a predicate $P \subseteq_{(G-)^n} \langle F_i \rangle_{i=1}^n$, see the right diagram. □

Example 3.5.4 We consider the simplest case of $\mathbb{C} = \mathbb{D} = \mathbf{Sets}$ and $G = \mathrm{Id}_{\mathbf{Sets}}$ in definition 3.5.1. In this case the square in definition 3.5.1 collapses into $p_{\mathbf{Sets}} : \mathbf{Sub}(\mathbf{Sets}) \to \mathbf{Sets}$.

Let T be a set of types and F^τ be a T-indexed family of sets. We can identify this family of sets with an interpretation of types $F : T \to \mathbf{Sets}$. Now we consider a predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} F$. Recall that an object in $\mathbf{Sub}(\mathbf{Sets})$ is a pair of sets (X, I) such that $X \subseteq I$ (see example 2.3.11). Therefore for each type $\tau \in T$, $P\tau$ specifies such a pair, and the condition $p_{\mathbf{Sets}} \circ P = F$ asserts that the second component of $P\tau$ is $F\tau$. Therefore P can be identified with a T-indexed family of subsets $\{P\tau \subseteq F\tau\}_{\tau \in T}$. □

Proposition 3.5.5 *Let $G : \mathbb{C} \to \mathbb{D}$ be a functor preserving finite products and $F : T \to \mathbb{C}$ be an interpretation of types. For any predicate $P \subseteq_G F$, $H^{\pi_G} : H^P \to H^F$ is a monomorphism.* □

PROOF Immediate from the fact that π_G (π_G^n) is faithful and preserves finite products strictly. ■

3.6 Pre-logical Predicates

In this section, we fix a Cartesian category $\mathbb{C}$, a category $\mathbb{D}$ with finite limits, a finite product preserving functor $G : \mathbb{C} \to \mathbb{D}$, a typed binding signature $\Pi = (T, O)$, a categorical interpretation $\mathbb{C}[\![-]\!]_F$ of Π in $\mathbb{C}$ and a predicate $P \subseteq_G F$.

Predicates Satisfying the Basic Lemma

The predicate P is often chosen by specifying a "good part" of the interpretation F of types. We are then interested in seeing that the good part P is large enough to interpret a typed formal system. In other words, we would like to show the following statement:

$$\forall \Gamma \vdash_{\Pi} M : \tau \,.\, \mathbb{C}[\![M]\!]_{F_1} \,:\, P^*\Gamma \to P\tau. \tag{3.2}$$

(see definition 2.3.3 and example 2.3.11 for the meaning of $P^*\Gamma \to P\tau$.) This entails that all closed terms $\emptyset \vdash_{\Pi} M : \tau$ are included in $P\tau$, that is, all closed terms satisfy the good property specified by P.

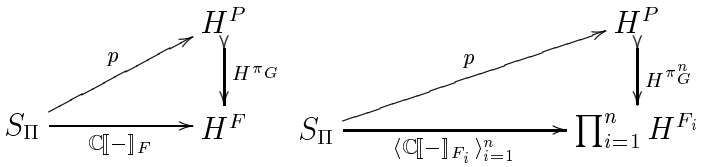
Example 3.6.1 (Continued from 3.4.3) In the study of the simply typed lambda calculus, we often take a good part Pb for each base type $b \in B$, then lift it up by induction to all types using the following scheme:

$$P(\tau \Rightarrow \tau') = \{f \in A(\tau \Rightarrow \tau') \mid \forall x : A\tau \,.\, x \in P\tau \implies f \bullet x \in P\tau'\}.$$

The predicate P constructed in this way is called a *logical predicate*. After establishing a logical predicate, we show (3.2) referred to as the *basic lemma* or *fundamental lemma*, to conclude that the good property holds for all closed terms. This proof technique is useful and can be applied to show various good properties of the simply typed lambda calculus, such as strong normalisation [Tai67], computational adequacy [Plo76], representation independence [Mit91], etc. See also section 1.1. □

We formulate (3.2) in $\mathbb{P}_T$.

Definition 3.6.2 Let $\mathbb{C}[\![-]\!]_F$ be a categorical interpretation of Π. We say that a predicate $P \subseteq_G F$ *satisfies the basic lemma for* Π *along* $\mathbb{C}[\![-]\!]_F$ if there exists a (necessarily unique) morphism $p : S_\Pi \to H^P$ making the following left diagram commute.



For the case of n-ary relations between $\mathbb{C}[\![-]\!]_{F_i} (1 \leq i \leq n)$, see the above right diagram. For a predicate $P \subseteq_G F$ satisfying the basic lemma, we denote the unique morphism which exists according to the above definition by the small letter corresponding to the predicate. □

Example 3.6.3 (Continued from example 3.4.3) A predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} A$ satisfying the basic lemma for Π_λ along $\mathcal{A}[\![-]\!]$ is a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of subsets $\{P\tau \subseteq A\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$ such that for every well-formed term $\Gamma \vdash_{\Pi_\lambda} M : \tau$ and $\rho \in P^*\Gamma$, $\mathcal{A}[\![M]\!]\rho \in P\tau$ holds. □

Example 3.6.4 (Continued from example 3.4.4) Let $G : \mathbb{C} \to \mathbb{D}$ be a functor preserving finite products. A predicate $P \subseteq_G F$ satisfying the basic lemma for Π_λ along $\mathbb{C}[\![-]\!]_F$ is a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of objects $\{P\tau\}_{\tau \in \mathbf{Typ}^{\Rightarrow}(B)}$ such that for every well-formed term $\Gamma \vdash_{\Pi_\lambda} M : \tau$, $\mathbb{C}[\![M]\!]_F : P^*\Gamma \to P\tau$ holds. □

Pre-Logical Predicates

It is often hard to directly prove (3.2) for a predicate P. Therefore it is desirable to have an equivalent characterisation of the basic lemma.

We say that a well-formed term $\Gamma \vdash_\Pi M : \tau$ is *invariant under* P if $\mathbb{C}[\![M]\!]_F : P^*\Gamma \to P\tau$ holds. The basic lemma is then saying that all well-formed terms are invariant under P.

Consider the pullback of H^{π_G} along $\mathbb{C}[\![-]\!]_F$ in $\mathbb{P}_T$:

$$\begin{array}{ccc} S^P_\Pi & \longrightarrow & H^P \\ {\scriptstyle i}\downarrow & & \downarrow{\scriptstyle H^{\pi_G}} \\ S_\Pi & \xrightarrow[\mathbb{C}[\![-]\!]_F]{} & H^F \end{array} \tag{3.3}$$

The vertex S^P_Π of the above pullback in $\mathbb{P}_T$ can be given by the following set for an object (Γ, τ) in $\mathbb{V}_T \times T$:

$$S^P_\Pi(\Gamma, \tau) = \{M \mid \Gamma \vdash_\Pi M : \tau \wedge \mathbb{C}[\![M]\!]_F : P^*\Gamma \to P\tau\}.$$

For this vertex, $i_{(\Gamma,\tau)} : S^P_\Pi(\Gamma, \tau) \to S_\Pi(\Gamma, \tau)$ is the inclusion function. We notice that $S^P_\Pi(\Gamma, \tau)$ is the set of all the invariant terms under P (having a type τ under a context Γ). We look at the syntactic structure of S^P_Π and obtain the following observation.

Suppose that the set of invariant terms S^P_Π is closed under all operators in Π. This is equivalent to the following two conditions:

1. For every context Γ and variable x such that $\Gamma(x) = \tau$, we have $x^\tau \in S^P_\Pi(\Gamma, \tau)$.

2. For each operator $o \in O$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$,

$$M_1 \in S^P_\Pi((\Gamma, \overrightarrow{x_1 : \tau_1}), \sigma_1) \cdots M_n \in S^P_\Pi((\Gamma, \overrightarrow{x_n : \tau_n}), \sigma_n)$$

   implies

$$o(\overrightarrow{x_1^{\tau_1}}.M_1, \cdots, \overrightarrow{x_n^{\tau_n}}.M_n) \in S^P_\Pi(\Gamma, \tau)$$

We can then prove by induction that all the well-formed terms $\Gamma \vdash_\Pi M : \tau$ are included in $S^P_\Pi(\Gamma, \tau)$, that is, the basic lemma holds.

Conversely if P satisfies the basic lemma, then S^P_Π trivially satisfies the above conditions.

This can be formulated by the existence of morphisms β_v in $\mathbb{P}_T$ and β_o in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$ making the following diagrams commute:

$$\begin{array}{ccc} Var & \overset{\beta_v}{\dashrightarrow} & S^P_\Pi \\ \| & & \downarrow i \\ Var & \xrightarrow[\alpha_v]{} & S_\Pi \end{array} \qquad \begin{array}{ccc} \prod S^P_\Pi(- \uplus \overrightarrow{\tau_i}, \sigma_i) & \overset{\beta_o}{\dashrightarrow} & S^P_\Pi(-, \tau) \\ {\scriptstyle \prod i(- \uplus \overrightarrow{\tau_i}, \sigma_i)} \downarrow & & \downarrow i \\ \prod S_\Pi(- \uplus \overrightarrow{\tau_i}, \sigma_i) & \xrightarrow[\alpha_o]{} & S_\Pi(-, \tau) \end{array}$$

where α_v, α_o are morphisms defined in the proof of theorem 3.3.5. From lemma 3.3.3, β_v and β_o specifies a Π-algebra $(S^P_\Pi, \beta : \Pi S^P_\Pi \to S^P_\Pi)$ and i is a Π-algebra morphism. Now we reach the following formulation of pre-logical predicates:

Definition 3.6.5 Let $\mathbb{C}[\![-]\!]_F$ be a categorical interpretation of Π. We call a predicate $P \subseteq_G F$ *pre-logical for* Π *along* $\mathbb{C}[\![-]\!]_F$ if in diagram (3.3), there exists a (necessarily unique) Π-algebra (S^P_Π, β) such that the projection i induced by the pullback is a Π-algebra morphism to the initial Π-algebra (S_Π, α_Π). □

Example 3.6.6 (Continued from example 3.4.3) We consider a pre-logical predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} A$ for $\mathcal{A}[\![-]\!]$ along Π_λ. It is the family of subsets $\{P\tau \subseteq A\tau\}_{\tau \in \mathbf{Typ}^\Rightarrow(B)}$ satisfying the following conditions.

1. For any context Γ, type τ, variable x such that $\Gamma(x) = \tau$ and $\rho \in P^*\Gamma$, $\mathcal{A}[\![x]\!]\rho \in P\tau$ holds.

2. For any well-formed term $\Gamma, x : \tau \vdash_{\Pi_\lambda} M : \tau'$, we have:

$$\begin{array}{ll} & (\forall \rho \in P^*(\Gamma, x : \tau) \,.\, \mathcal{A}[\![M]\!]\rho \in P\tau') \\ \Longrightarrow & \forall \rho \in P^*\Gamma \,.\, \mathcal{A}[\![\lambda x^\tau \,.\, M]\!]\rho \in P(\tau \Rightarrow \tau') \end{array}$$

3. For any well-formed terms $\Gamma \vdash_{\Pi_\lambda} M : \tau \Rightarrow \tau'$ and $\Gamma \vdash_{\Pi_\lambda} N : \tau$, we have

$$\begin{array}{ll} & (\forall \rho \in P^*\Gamma \,.\, \mathcal{A}[\![M]\!]\rho \in P(\tau \Rightarrow \tau')) \wedge (\forall \rho \in P^*\Gamma \,.\, \mathcal{A}[\![N]\!]\rho \in P\tau) \\ \Longrightarrow & (\forall \rho \in P^*\Gamma \,.\, \mathcal{A}[\![M]\!]\rho \bullet \mathcal{A}[\![N]\!]\rho \in P\tau'). \end{array}$$

We compare these conditions with the original pre-logical predicates [HS02] defined with respect to a lambda applicative structure $\mathcal{A}$ (see example 3.4.3). There, a family of subsets $P\tau \subseteq A\tau$ was said to be *pre-logical* if the following holds:

- For any well-formed term $\Gamma, x : \tau \vdash_{\Pi_\lambda} M : \tau'$ and $\rho \in P^*\Gamma$, we have

$$\begin{array}{ll} & (\forall v \in P\tau \,.\, \mathcal{A}[\![M]\!](s_{\Gamma,\tau}(\rho, v)) \in P\tau') \\ \Longrightarrow & \mathcal{A}[\![\lambda x^\tau \,.\, M]\!]\rho \in P(\tau \Rightarrow \tau') \end{array} \tag{3.4}$$

- For any types τ and τ' in $\mathbf{Typ}^{\Rightarrow}(B)$, we have

$$\forall x \in P(\tau \Rightarrow \tau'), y \in P\tau \,.\, x \bullet y \in P\tau' \tag{3.5}$$

Condition 1 always holds by the definition of meaning function. Thus we ignore it.

A calculation shows that condition (3.5) is equivalent to condition 3. To show that condition 3 implies condition (3.5), we instantiate condition 3 with a well-formed term $x : \tau \Rightarrow \tau', y : \tau \vdash xy : \tau'$. The converse is immediate.

It is easy to see that (3.4) implies condition 2 by distributing the universal quantifier at the head, while condition 2 itself does not imply (3.4). However, we show below that conditions 1–3 are equivalent to P satisfying the basic lemma (example 3.6.3), which immediately implies (3.4).

At this moment we do not know the right setting which precisely yields the original definition of pre-logical predicates. However, the point of pre-logical predicates is to give a syntactic characterisation of the basic lemma, which is the most important aspect, and our generalisation shares this with the original pre-logical predicates. Thus despite a minor difference of presentation, our pre-logical predicates and the original pre-logical predicates are equal. □

Example 3.6.7 (Continued from example 3.4.4) The predicate $P \subseteq_G F$ for Π_λ along $\mathbb{C}[\![-]\!]_F$ is pre-logical if the following three conditions hold.

1. For any (Γ, τ) and $x \in Var(\Gamma, \tau)$, $\mathbb{C}[\![x^\tau]\!]_F : P^*\Gamma \to P\tau$.
2. For any well-formed term $\Gamma, x : \tau \vdash_{\Pi_\lambda} M : \tau'$, if $\mathbb{C}[\![M]\!]_F : P^*(\Gamma, x : \tau) \to P\tau'$ then $\mathbb{C}[\![\lambda x^\tau \,.\, M]\!]_F : P^*\Gamma \to P(\tau \Rightarrow \tau')$.
3. For any well-formed terms $\Gamma \vdash_{\Pi_\lambda} M : \tau \Rightarrow \tau'$ and $\Gamma \vdash_{\Pi_\lambda} N : \tau$, if $\mathbb{C}[\![M]\!]_F : P^*\Gamma \to P(\tau \Rightarrow \tau')$ and $\mathbb{C}[\![N]\!]_F : P^*\Gamma \to P\tau$ then $\mathbb{C}[\![MN]\!]_F : P^*\Gamma \to P\tau'$.

The first condition trivially holds, as variables are interpreted by projections. Condition 3 is equivalent to:

$$@ : P(\tau \Rightarrow \tau') \times P\tau \to P\tau'.$$

as we discussed in the previous example. □

Theorem 3.6.8 *Let $\mathbb{C}[\![-]\!]_F$ be a categorical interpretation of Π. A predicate $P \subseteq_G F$ is pre-logical if and only if P satisfies the basic lemma.* □

PROOF (if) If P satisfies the basic lemma, we have an isomorphism $f : S^P_\Pi \cong S_\Pi$. Then $f : (S^P_\Pi, f^{-1} \circ \alpha_\Pi \circ (\Pi f)) \to (S_\Pi, \alpha_\Pi)$ is a Π-algebra morphism. Therefore P is pre-logical.

(only if) Suppose there exists a Π-algebra (S^P_Π, β). Now we show that the following diagram commutes in the category $\mathbb{P}^\Pi_T$ of Π-algebras:

$$(S^P_\Pi, \beta) \overset{i}{\rightarrowtail} (S_\Pi, \alpha_\Pi) \overset{!}{\twoheadrightarrow} (S^P_\Pi, \beta) \overset{i}{\rightarrowtail} (S_\Pi, \alpha_\Pi), \qquad id : (S^P_\Pi, \beta) \to (S^P_\Pi, \beta)$$

From the universal property of the initial Π-algebra, we have $i \circ ! = id$. Now we have $i \circ ! \circ i = i = i \circ id$, and since i is mono, $! \circ i = id$. Therefore (S_Π^P, β) and (S_Π, α_Π) are isomorphic, thus S_Π and S_Π^P are so. ■

This proof is a categorical re-formulation of the inductive proof of the basic lemma for pre-logical relations in [HS02]. From now on we identify pre-logical predicates and predicates satisfying the basic lemma.

Lifted Endofunctor and Pre-Logical Predicates

In this section we present a sufficient condition for a predicate $P \subseteq_G F$ being pre-logical using Hermida and Jacobs' formulation of induction principles in a fibred setting[HJ95, Jac99]. First, we recall that the subobject fibration $p_{\mathbb{P}_T} : \mathbf{Sub}(\mathbb{P}_T) \to \mathbb{P}_T$ has fibred terminal objects $\top : \mathbb{P}_T \to \mathbf{Sub}(\mathbb{P}_T)$ (example 2.4.5) and subset types $\{-\} : \mathbf{Sub}(\mathbb{P}_T) \to \mathbb{P}_T$ (proposition 2.4.23). They form the following adjunctions (proposition 2.4.4, definition 2.4.22):

$$p_{\mathbb{P}_T} \dashv \top \dashv \{-\}.$$

We recall the definition of the lifting of an endofunctor (definition 2.4.25). We say that an endofunctor $\dot{F}$ over $\mathbf{Sub}(\mathbb{P}_T)$ is a lifting of an endofunctor F over $\mathbb{P}_T$ if the following diagram commutes:

$$\begin{array}{ccc} \mathbf{Sub}(\mathbb{P}_T) & \xrightarrow{\dot{F}} & \mathbf{Sub}(\mathbb{P}_T) \\ {\scriptstyle p_{\mathbb{P}_T}}\downarrow & & \downarrow{\scriptstyle p_{\mathbb{P}_T}} \\ \mathbb{P}_T & \xrightarrow[F]{} & \mathbb{P}_T \end{array}$$

Proposition 3.6.9 *Let Π be a typed binding signature. Then the corresponding endofunctor Π over $\mathbb{P}_T$ has a lifting $\dot{\Pi}$ over $\mathbf{Sub}(\mathbb{P}_T)$.* □

PROOF First, endofunctors $(-)^{\Gamma,\sigma}$ and $(-)|_\tau$ preserve pullbacks from proposition 3.2.5. Therefore from example 2.4.26, they have liftings, which we write by $(-)^{\dot{\Gamma,\sigma}}$ and $(-)||_\tau$ respectively. Next from proposition 2.4.6 and proposition 2.4.8, we have small products and coproducts in $\mathbf{Sub}(\mathbb{P}_T)$ which are strictly preserved by $p_{\mathbb{P}_T}$. With these,

we define a lifting $\dot{\Pi}$ of Π as follows.

$$\dot{\Pi}F = \top Var \dot{+} \dot{\coprod_{s(\overrightarrow{\tau_1},\sigma_1),\cdots,(\overrightarrow{\tau_n},\sigma_n)\to\tau \in O}} \left(\dot{\prod_{1\leq i\leq n}} F^{\overrightarrow{\tau_i},\sigma_i} \right) \Big|\Big|_\tau$$

Proposition 3.6.10 *There is a natural isomorphism $i : \dot{\Pi}\top \to \top\Pi$.* □

PROOF From adjunctions $p_{\mathbb{P}_T} \dashv \top \dashv \{-\}$, the fibred terminal object functor $\top$ preserves both limits and colimits. Thus we have natural isomorphisms $\dot{\coprod}_{i\in I}(\top X_i) \cong \top(\coprod_{i\in I} X_i)$ and $\dot{\prod}_{i\in I}(\top X_i) \cong \top(\prod_{i\in I} X_i)$.

Next from the definition of the lifting of $(-)^{\Gamma,\tau}$, we have:

$$(\top F)^{\dot{\Gamma,\tau}} = ([\mathrm{id}_F]_{=m})^{\dot{\Gamma,\tau}} = [(\mathrm{id}_F)^{\Gamma,\tau}]_{=m} = [\mathrm{id}_{F^{\Gamma,\tau}}]_{=m} = \top(F^{\Gamma,\tau}).$$

Similarly we have $(\top F)||_\tau = \top(F|_\tau)$. Therefore all constructs in functor $\dot{\Pi}$ commutes with $\top$, thus $\dot{\Pi}\top \cong \top\Pi$. ■

Proposition 3.6.11 *Let $P \subseteq_G F$ be a predicate and assume that the subobject $H^{\pi_G} : H^P \rightarrowtail H^F$ has a $\dot{\Pi}$-algebra structure (thus H^F has a Π-algebra structure). Then P is pre-logical for Π along the Π-algebra morphism $! : S_\Pi \to H^F$ induced by initiality of the initial algebra (S_Π, α_Π) (see theorem 3.3.5).* □

PROOF By applying proposition 2.4.27, we can lift the initial Π-algebra (S_Π, α_Π) to an initial $\dot{\Pi}$-algebra $(\top S_\Pi, \top\alpha_\Pi \circ i_{S_\Pi})$. Thus we have the unique $\dot{\Pi}$-algebra morphism $\dot{!} : \top S_\Pi \to H^{\pi_G}$ which is above the unique Π-algebra morphism $! : S_\Pi \to H^F$. The following diagram describes this situation:

$$\begin{array}{ccc} S_\Pi & \xrightarrow{\dot{!}} & H^P \\ {\scriptstyle \top S_\Pi}\| & & \downarrow{\scriptstyle H^{\pi_G}} \\ S_\Pi & \xrightarrow[!]{} & H^F \end{array}$$

This implies that there exists a morphism $p : S_\Pi \to H^P$ such that $H^{\pi_G} \circ p = !$. Therefore P is a pre-logical predicate. ■

We expand the condition that H^{π_G} has a $\dot{\Pi}$-algebra structure. This implies the existence of morphisms u_v, β_v in $\mathbb{P}_T$ and u_o, β_o in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in$

O of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$, such that they make the following diagrams commute:

$$\begin{array}{ccc} Var & \xrightarrow{\beta_v} & H^P \\ \| & & \downarrow{\scriptstyle H^{\pi_G}} \\ Var & \xrightarrow[v]{} & H^F \end{array} \qquad \begin{array}{ccc} \prod_{i=1}^n H^P(- \uplus \overrightarrow{\tau_i}, \sigma_i) & \xrightarrow{\beta_o} & H^P(-, \tau) \\ {\scriptstyle \prod H^{\pi_G}(- \uplus \overrightarrow{\tau_i}, \sigma_i)} \downarrow & & \downarrow {\scriptstyle H^{\pi_G}} \\ \prod_{i=1}^n H^F(- \uplus \overrightarrow{\tau_i}, \sigma_i) & \xrightarrow[u_o]{} & H^F(-, \tau) \end{array}$$

The above left diagram commutes when u_v is the standard interpretation by projections. So we only need a morphism β_o for each operator $o \in O$ making the above right diagram commute. This requirement is expressed as follows:

$$\forall f_i : P^*(\Gamma \uplus \overrightarrow{\tau_i}) \to P\sigma_i \,.\, u_o(f_1, \cdots, f_n) : P^*\Gamma \to P\tau.$$

Example 3.6.12 (Continued from example 3.6.7) Let $P \subseteq_G F$ be a predicate satisfying the following conditions:

1. For any $f : P^*(\Gamma \uplus \tau) \to P\tau'$, $\lambda(f \circ s_{\Gamma,\tau}) : P^*\Gamma \to P(\tau \Rightarrow \tau')$.

2. For any $f : P^*\Gamma \to P(\tau \Rightarrow \tau')$ and $g : P^*\Gamma \to P\tau$, $@ \circ \langle f, g \rangle \in P^*\Gamma \to P\tau'$.

This implies that H^{π_G} has a $\dot{\Pi}_\lambda$ algebra structure, and from proposition 3.6.11, P is pre-logical. One can also directly conclude that P satisfies the basic lemma by induction on the derivation of well-formed terms. The above conditions are reminiscent of a characterisation of lax logical predicates for the simply typed lambda calculus in terms of categorical combinators [PPST00]. □

3.7 Relational Composition of Binary Pre-Logical Relations

We move to showing that binary pre-logical relations are closed under relational composition. The goal of this section is to adapt the composability result [HS02] to our framework. Composability is one distinguishing property of binary pre-logical relations, since this result does not hold for binary logical relations in general.

In set theory, the *composition* $c(R, S)$ of binary relations $R \subseteq A \times B$ and $S \subseteq B \times C$ is defined by:

$$c(R, S) = \{(a, c) \in A \times C \mid \exists b \in B \,.\, (a, b) \in R \wedge (b, c) \in S\}. \tag{3.6}$$

We first recall the composability result in [HS02].

Theorem 3.7.1 ([HS02], proposition 5.6) *Let $\mathcal{A}, \mathcal{B}, \mathcal{C}$ be lambda applicative structures (c.f. example 3.4.3) and $R \subseteq \mathcal{A} \times \mathcal{B}$ and $S \subseteq \mathcal{B} \times \mathcal{C}$ be binary pre-logical relations. Then the relational composition $c(R, S)$ (the same notation as the composition of binary relations) defined by:*

$$c(R, S)\tau = c(R\tau, S\tau) \tag{3.7}$$

is a binary pre-logical relation. [4] □

To adapt the above result to our framework, we need to formulate the composition of binary relations categorically. We do this without getting involved with morphisms and objects; we define the composition and show the above result within the internal logic of fibrations.

In subsconing, G-predicates and relations correspond to judgments in the internal logic of a subobject fibration $p_{\mathbb{D}} : \mathbf{Sub}(\mathbb{D}) \to \mathbb{D}$ (definition 3.5.1). In order to use (3.6) to define the composition of binary relations, we need an existential quantifier to hide the mediating element of a pair in each relation. We therefore take a *regular* category $\mathbb{D}$ (hence $p_{\mathbb{D}}$ supports $\mathcal{L}_{\exists}$), a Cartesian category $\mathbb{C}$ and a functor $G : \mathbb{C} \to \mathbb{D}$ preserving finite products. We write $\mathrm{fst}, \mathrm{snd} : \mathbb{C} \times \mathbb{C} \to \mathbb{C}$ for the first and second projections.

We take objects of $\mathbf{Rel}_2(G)$ as a categorical formulation of binary relations (see definition 3.5.1). An explicit description of $\mathbf{Rel}_2(G)$ is the following:

An object in $\mathbf{Rel}_2(G)$ is a triple (C, C', P) where C, C' are objects in $\mathbb{C}$ and P is an object in $\mathbf{Sub}(\mathbb{D})_{GC \times GC'}$. Using the internal logic of $p_{\mathbb{D}}$, P can be identified with a predicate judgement $x : GC, y : GC' \vdash P(x, y)$.

[4] The reason why the composition of binary logical relations is not logical is because we merely have the following inclusion:

$$c(R(\tau \Rightarrow \tau'), S(\tau \Rightarrow \tau')) \subseteq c(R, S)(\tau \Rightarrow \tau') = c(R\tau, S\tau) \Rightarrow c(R\tau', S\tau').$$

A concrete example is shown in [HS02], example 5.4.

A morphism in $\mathbf{Rel}_2(G)$ from (C, C', P) to (D, D', Q) is a pair of morphisms $f : C \to D, g : C' \to D'$ in $\mathbb{C}$ such that $x : GC, y : GC' \mid P(x, y) \vdash Q(Gf(x), Gg(y))$ holds in the internal logic of $p_{\mathbb{D}}$.

We then formulate the composition of binary relations in the following way. Given objects (C, C', P) and (C', C'', P') in $\mathbf{Rel}_2(G)$, we define their relational composition $c(C, C', C'', P, P')$ by:

$$c(C, C', C'', P, P') = (C, C'', x : GC, z : GC'' \vdash \exists y : GC' \,.\, P(x, y) \wedge P'(y, z))$$

where $x : GC, z : GC'' \vdash \exists y : GC' \,.\, P(x, y) \wedge P'(y, z)$ is the predicate judgement derived in the internal logic of $p_{\mathbb{D}}$. This is just a re-interpretation of (3.6) in the internal logic. By letting $\mathbb{C} = \mathbb{D} = \mathbf{Sets}$ and $G = \mathrm{Id}_{\mathbf{Sets}}$, the above composition coincides with the set-theoretic composition of binary relations given by (3.6).

We notice that relational composition is defined over binary relations (C, C', P') and (D, D', Q') such that $C' = D$. In other words, the relational composition is a mapping of objects $c : \mathbf{Obj}(\mathbf{Rel}_c(G)) \to \mathbf{Obj}(\mathbf{Rel}_2(G))$, where $\mathbf{Rel}_c(G)$ is the category of composable pairs of relations. Formally, $\mathbf{Rel}_c(G)$ is obtained by the following change-of-base:

$$\begin{array}{ccc} \mathbf{Rel}_c(G) & \longrightarrow & \mathbf{Rel}_2(G) \\ \downarrow & \lrcorner & \downarrow{\scriptstyle \partial_2} \\ \mathbf{Rel}_2(G) & \xrightarrow[\partial_1]{} & \mathbb{C} \end{array}$$

where $\partial_1 = \mathrm{fst} \circ \pi_{(G-)^2}$ and $\partial_2 = \mathrm{snd} \circ \pi_{(G-)^2}$. The category $\mathbf{Rel}_c(G)$ can be described as follows:

An object in $\mathbf{Rel}_c(G)$ is a tuple (C, C', C'', P, P') such that (C, C', P) and (C', C'', P') are objects in $\mathbf{Rel}_2(G)$.

A morphism in $\mathbf{Rel}_c(G)$ from (C, C', C'', P, P') to (D, D', D'', Q, Q') in $\mathbf{Rel}_c(G)$ is a triple $(f : C \to D, g : C' \to D', h : C'' \to D'')$ such that $(f, g) : (C, C', P) \to (D, D', Q)$ and $(g, h) : (C', C'', P') \to (D', D'', Q')$ are morphisms in $\mathbf{Rel}_2(G)$.

We then extend c to a functor $c : \mathbf{Rel}_c(G) \to \mathbf{Rel}_2(G)$ by specifying the way c acts on morphisms in $\mathbf{Rel}_c(G)$.

Definition 3.7.2 We define a functor $c : \mathbf{Rel}_c(G) \to \mathbf{Rel}_2(G)$ as follows:

$$\begin{aligned} c(C, C', C'', P, P') &= (C, C'', x : GC, z : GC'' \vdash \exists y : GC' \,.\, P(x, y) \wedge P'(y, z)) \\ c(f, g, h) &= (f, h). \end{aligned}$$

□

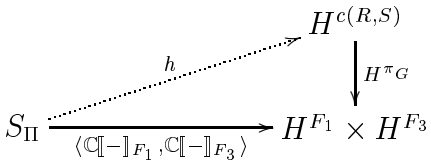
We state the main theorem of this section.

Theorem 3.7.3 *Let $R \subseteq_{(G-)^2} \langle F_1, F_2 \rangle$ and $S \subseteq_{(G-)^2} \langle F_2, F_3 \rangle$ be binary pre-logical relations for Π along $\mathbb{C}[\![-]\!]_{F_1} \times \mathbb{C}[\![-]\!]_{F_2}$ and $\mathbb{C}[\![-]\!]_{F_2} \times \mathbb{C}[\![-]\!]_{F_3}$ respectively. We define the relational composition $c(R, S)$ of R and S as follows:*

$$c(R, S)\tau = c(R\tau, S\tau)$$

which specifies a binary relation $c(R, S) \subseteq_{(G-)^2} \langle F_1, F_3 \rangle$. Then $c(R, S)$ is a binary pre-logical relation for Π along $\mathbb{C}[\![-]\!]_{F_1} \times \mathbb{C}[\![-]\!]_{F_3}$. □

PROOF We show that $c(R, S)$ satisfies the basic lemma. We find a morphism h making the following diagram commute:



provided that R, S are binary pre-logical relations. Since binary pre-logical relations and binary relations satisfying the basic lemma are equivalent, we have morphisms $r : S_\Pi \to H^R$ and $s : S_\Pi \to H^S$ in $\mathbb{P}_T$. Then we define $h : S_\Pi \to H^{c(R,S)}$ for a well-formed terms $\Gamma \vdash_\Pi M : \tau$ as follows:

$$h_{(\Gamma,\tau)}(M) = c(r_{(\Gamma,\tau)}(M), s_{(\Gamma,\tau)}(M)).$$

In chapter 6, we introduce pre-logical data refinement of typed formal systems. The closure property under the relational composition of binary pre-logical relations will be used to show that data refinement composes.

3.8 The Least Pre-Logical Extension

Given a predicate for a subset of the set of types, we can extend it to a pre-logical predicate. This extension is the least one among all possible pre-logical extensions of the given predicate. This is suggested in [HS02], and we examine this statement in the context of our generalised pre-logical predicates.

Let $\Pi = (T, O)$ be a typed binding signature, $\mathbb{C}$ be a Cartesian category, $\mathbb{C}[\![-]\!]_F$ be a categorical semantics of Π satisfying the semantic substitution lemma, $\mathbb{D}$ be a regular category such that $p_{\mathbb{D}} : \mathbf{Sub}(\mathbb{D}) \to \mathbb{D}$ has fibred small coproducts and $G : \mathbb{C} \to \mathbb{D}$ be a functor preserving finite products. The subobject fibration $p_{\mathbb{D}}$ supports $\mathcal{L}_\wedge, \mathcal{L}_=, \mathcal{L}_\exists, \mathcal{L}_{\vee\omega}$.

Definition 3.8.1 Let $U \subseteq T$ be a subset. We write $\iota : U \hookrightarrow T$ for the inclusion function. For a G-predicate $R \subseteq_G F \circ \iota$, we define the *least pre-logical extension* I_R *of* R by the following G-predicate $I_R \subseteq_G F$ in the internal logic of $p_{\mathbb{D}}$:

$$\begin{aligned} & x : GF\tau \vdash (I_R\tau)x \\ = & \bigvee_{\Delta \in \mathbf{Obj}(\mathbb{V}_U), \Delta \vdash M : \tau} \exists \rho : (GF)^*\Delta \,.\, (R^*\Delta)\rho \wedge x = G(\mathbb{C}[\![M]\!]_F)\rho \end{aligned}$$ □

Theorem 3.8.2 *1. $I_R \subseteq_G F$ is a pre-logical predicate for Π along $\mathbb{C}[\![-]\!]_F$ that includes R, that is, the following holds in the internal logic:*

$$x : GF\sigma \mid (R\sigma)x \vdash (I_R\sigma)x \quad (\sigma \in U).$$

2. Any pre-logical predicate $S \subseteq_G F$ for Π along $\mathbb{C}[\![-]\!]_F$ includes I_R, that is,

$$x : GF\tau \mid (I_R\tau)x \vdash_\Pi (S\tau)x$$

holds for each type $\tau \in T$ in the internal logic.

Therefore I_R is the least pre-logical predicate that includes R. □

PROOF We first show that I_R satisfies the basic lemma for Π along $\mathbb{C}[\![-]\!]_F$. Let $\Gamma = \{z_1 : \tau_1, \cdots, z_n : \tau_n\}$ be a context (for simplicity we assume that $z_1 \leq_{\mathbf{v}} \cdots \leq_{\mathbf{v}} z_n$), $\Gamma \vdash_\Pi N : \tau$ be a well-formed term, $\eta : (GF)^*\Gamma$ and assume $(I_R^*\Gamma)\eta$. We find an U-context Δ, a well-formed term $\Delta \vdash_\Pi M : \tau$ and $\rho : (GF)^*\Delta$ such that $G(\mathbb{C}[\![N]\!]_F)\eta = G(\mathbb{C}[\![M]\!]_F)\rho$.

We explicitly write down η by a tuple $(x_1, \cdots, x_n)$ with $x_i : GF\tau_i$. From the assumption $(I_R\tau_i)x_i$ for each $1 \leq i \leq n$, there exists a U-context Δ_i, a well-formed term $\Delta_i \vdash_\Pi M_i : \tau_i$ and $\eta_i : (GF)^*\Delta_i$ such that $(R^*\Delta_i)\eta_i$ and $x_i = G(\mathbb{C}[\![M_i]\!]_F)\eta_i$ holds. Now

$$\begin{aligned} G(\mathbb{C}[\![N]\!]_F)\eta &= G(\mathbb{C}[\![N]\!]_F)\langle G(\mathbb{C}[\![M_1]\!]_F)\eta_1, \cdots, G(\mathbb{C}[\![M_n]\!]_F)\eta_n\rangle \\ &= G(\mathbb{C}[\![N[M_1\theta_{\iota_1}/z_1, \cdots, M_n\theta_{\iota_n}/z_n]]\!]_F)\ s^{GF}_{\Delta_1,\cdots,\Delta_n}\langle \eta_1, \cdots, \eta_n\rangle \end{aligned}$$

where $s^{GF}_{\Delta_1,\cdots,\Delta_n}$ is the canonical isomorphism (see definition 3.4.1) and $\iota_i : \Delta_i \to \biguplus_{j=1}^n \Delta_j$ is the i-th injection. Thus we take $\biguplus_{j=1}^n \Delta_j, N[M_1/z_1, \cdots, M_n/z_n]$ and $s^{GF}_{\Delta_1,\cdots,\Delta_n}\langle \eta_1, \cdots, \eta_n\rangle$ for Δ, M and ρ.

We next show $x : GF\sigma \mid (R\sigma)x \vdash_\Pi (I_R\sigma)x$ for each type $\sigma \in U$. Let $x : GF\sigma$ and assume $(R\sigma)x$. By letting $\Delta = \{z : \sigma\}, M = z$ and $\rho = x$ in the definition of I_R (definition 3.8.1), we have $(I_R\sigma)x$.

To show that I_R is included in any pre-logical predicate including R, let $S \subseteq_G F$ be a pre-logical predicate satisfying $x : GF\sigma \mid (R\sigma)x \vdash_\Pi (S\sigma)x$ for each type $\sigma \in U$ (we name this assumption (*)). We show $x : GF\tau \mid (I_R\tau)x \vdash_\Pi (S\tau)x$ for each type $\tau \in T$. Let $x : GF\tau$ and assume $(I_R\tau)x$. From this assumption, there exists an U-context Δ, a well-formed term $\Delta \vdash_\Pi M : \tau$ and $\rho : (GF)^*\Gamma$ such that $(R^*\Delta)\rho$ and $x = G(\mathbb{C}[\![M]\!])\rho$ holds. From assumption (*), we have $(S^*\Delta)\rho$. Since S is pre-logical, we have $(S\tau)x$. ∎

Example 3.8.3 (Continued from example 3.4.3) In definition 3.8.1, we let $U = \emptyset$ and R be the empty predicate. We furthermore assume that $\mathcal{A}[\![-]\!]$ satisfies the semantic substitution lemma. Then I_R is the pre-logical predicate consisting of values definable by closed lambda terms, that is,

$$I_R\tau = \{\mathcal{A}[\![M]\!] \mid \emptyset \vdash_{\Pi_\lambda} M : \tau\}.$$

Related Work

For the history and applications of logical predicates, which is the precursor of pre-logical predicates, see section 1.1. This thesis adopts the subscone approach in [MR92].

Besides pre-logical predicates, we mention two other generalisations of logical predicates. *Lax logical predicates* [PPST00] are functors from the free CCC to the category $\mathbf{Pred}(G)$ which preserve finite products. Thus they are a weakening of Ma and Reynolds' formulation of logical predicates, which also require preservation of exponentials, but the basic lemma still holds for lax logical predicates, and furthermore the converse holds as with theorem 3.6.8 above. In this sense lax logical predicates and pre-logical predicates are the same. In [PPST00], lax logical predicates are extended to other Cartesian categories equipped with a discrete algebraic structure over $\mathbf{Cat}$, which includes finite coproducts, monoidal structures, etc. Lax logical predicates are considered in the computational lambda calculus [KP99] and Moggi's computational metalanguage [GLLNZ04]. A formal relationship between lax logical predicates and pre-logical predicates is shown in section 4.3.

In [KOPT97], Kinoshita et al. introduced *L-predicates*, which is another weakening and generalisation of Ma and Reynolds' formulation of logical predicates. L-predicates are closed under relational composition in more abstract sense. They abstracted the situation that a category is equipped with a category of binary relations and relational composition in terms of a category object in $\mathbf{Cat}$. They then showed that L-predicates are closed under the composition of category objects. It is possible to adopt their approach in our framework and show the closure property.

In [Lei01], Leiss extended pre-logical predicates to System $F\omega$, and characterised observational equivalence in terms of the existence of a binary pre-logical relation of a certain kind. System $F\omega$ is beyond the expressive power of simply typed formal systems due to type variables and two-layered structure of the lambda calculus. Therefore the generalisation of pre-logical predicates in this thesis is not applicable to system $F\omega$. One interesting phenomenon is that binary pre-logical relations for system $F\omega$ are not closed under relational composition. The same problem occurs in the much weaker calculus $\lambda\underline{\omega}$ (see appendix A). From the counterexample constructed in $\lambda\underline{\omega}$, the problem seems to arise from unrestricted relations between types rather than from polymorphism. Leiss pointed out that binary pre-logical relations for $F\omega$ are closed under relational composition if the relations between types are restricted to functional relations. We do not know if this restriction can be relaxed.

This work refers to the framework by Miculan and Scagnetto [MS03] on a categorical model of typed higher-order abstract syntax. This category is considered in a different form in [Fio02], and is a natural extension of the presheaf category considered in [FPT99, Hof99] to take types into account.

3.9 Conclusion

We have given a generalisation of pre-logical predicates to arbitrary simply typed formal systems, and shown that they are equivalent to predicates satisfying the basic lemma, and that binary pre-logical relations are closed under composition. We represent three underlying components of pre-logical predicates — syntax, semantics and predicates — in a presheaf category. Then we formulate pre-logical predicates and predicates satisfying the basic lemma, and show their equivalence.

It is interesting to extend our framework for defining formal systems. One direction is to allow type variables so that we can cover type systems such as System F or FPC [FP94]. The other direction is to modify the notion of contexts from the Cartesian one to the linear one to cover *linear logic* [Gir87]. In both cases we also have to switch the notion of models from Cartesian categories to more elaborate categorical structures such as symmetric monoidal categories and polymorphic fibrations [Jac99], etc.

Chapter 4

Examples of Pre-logical Predicates

In the previous chapter, we generalised pre-logical predicates for typed formal systems. To test the adequacy of this generalisation, we derive pre-logical predicates for various existing calculi which fit within the scheme of typed formal systems and their categorical models. The examples we consider in this chapter are the following. First we consider the simple case of many-sorted algebras. It turns out that pre-logical predicates for many-sorted algebras coincide with the notion of subalgebras. This fact is used in the next example to investigate the relationship between pre-logical predicates for the simply typed lambda calculus and those for combinatory algebras. Next we establish a formal relationship between pre-logical predicates and lax logical predicates [PPST00] for the simply typed lambda calculus. To demonstrate that the generalisation works for extensions of lambda calculi, we derive pre-logical predicates for Moggi's computational metalanguage [Mog91] and give a concrete example inspired by Stark and Lindley's leapfrog method [Lin04, LS05]. Furthermore, we consider pre-logical predicates for first-order logic and see that the notion of elementary submodel (see e.g. [Doe96]) can be characterised in terms of pre-logical relations.

4.1 Pre-Logical Predicates for First-order Typed Signatures

Let $\Sigma = (T_0, O_0)$ be a many-sorted signature, that is, a typed first-order signature (definition 3.3.1). This determines a typed formal system deriving well-formed terms $\Gamma \vdash_\Sigma M : \tau$.

A *many-sorted Σ-algebra* $\mathcal{A}$ consists of a T_0-indexed family of sets $\{A^\tau\}_{\tau \in T_0}$ (equivalently an interpretation of types $A : T_0 \to \mathbf{Sets}$) together with an assignment of a function $o_\mathcal{A} : A\tau_1 \times \cdots \times A\tau_n \to A\tau$ to each operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$. [1] We distinguish the term "many-sorted Σ-algebra" in this classical sense from the term "Σ-algebra", which means an algebra of the endofunctor Σ over $\mathbb{P}_T$ assigned to Σ. A *subalgebra* of a Σ-algebra $\mathcal{A}$ is a T_0-indexed family of subsets $\{P\tau \subseteq A\tau\}_{\tau \in T_0}$ such that for each operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$, we have $o_\mathcal{A} : P\tau_1 \times \cdots \times P\tau_n \to P\tau$.

For a well-formed term $\Gamma \vdash_\Sigma M : \tau$, we assign its *denotation* $\mathcal{A}[\![M]\!] : A^*\Gamma \to A\tau$ by induction on the structure of M:

$$\begin{aligned} \mathcal{A}[\![x^\tau]\!] &= \pi_{\gamma(x)} \\ \mathcal{A}[\![o(M_1, \cdots, M_n)]\!] &= o_\mathcal{A} \circ \langle \mathcal{A}[\![M_1]\!], \cdots, \mathcal{A}[\![M_n]\!] \rangle \end{aligned}$$

(see section 2.1 for the meaning of $\gamma(x)$).

In order to apply the general theory of pre-logical predicates, we explain this semantics within our ambient category $\mathbb{P}_T$. For each operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$, we define a morphism $u_o : \prod_{i=1}^n H^A(-, \tau_i) \to H^A(-, \tau)$ in $[\mathbb{V}_T, \mathbf{Sets}]$ by

$$(u_o)_\Gamma(f_1, \cdots, f_n) = o_\mathcal{A} \circ \langle f_1, \cdots, f_n \rangle$$

where $f_i : A^*\Gamma \to A\tau_i$. Together with the standard interpretation of variables $v : Var \to H^A$ by projections, we have a Σ-algebra structure over H^A by lemma 3.3.3. From initiality of S_Σ, there exists a unique Σ-algebra morphism in $\mathbb{P}_T$, namely $\mathcal{A}[\![-]\!]$:

[1] The development of this section is carried out in $\mathbf{Sets}$ for compatibility with the classical theory of many-sorted algebras. However, this does not mean that the development is specialised to $\mathbf{Sets}$; one can replace $\mathbf{Sets}$ with some Cartesian category $\mathbb{C}$.

$S_\Sigma \to H^A$. This overloaded notation is justified since this coincides with the above assignment of denotation. We call this the *standard interpretation* of Σ-terms in $\mathcal{A}$.

We discuss pre-logical predicates for Σ along the standard interpretation. First let $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} A$ be a predicate over A. This is just a T_0-indexed family of subsets $\{P\tau \subseteq A\tau\}_{\tau \in T_0}$ (see example 3.5.4).

Proposition 4.1.1 *A predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} A$ is pre-logical for Σ along the standard interpretation $\mathcal{A}[\![-]\!]$ if and only if for each operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$, we have $o_\mathcal{A} : P\tau_1 \times \cdots \times P\tau_n \to P\tau$, that is, P is a subalgebra of $\mathcal{A}$.* □

PROOF As we have seen in chapter 3, P is pre-logical if

1. For each object (Γ, τ) in $\mathbb{V}_T \times T$ and $x^\tau \in Var(\Gamma, \tau)$, $\mathcal{A}[\![x^\tau]\!] : P^*\Gamma \to P\tau$.

2. For each object (Γ, τ) in $\mathbb{V}_T \times T$ and $o^{\tau_1, \cdots, \tau_n \to \tau} \in O_0$ and well-formed terms $\Gamma \vdash_\Sigma M_i : \tau_i$, $\mathcal{A}[\![M_i]\!] : P^*\Gamma \to P\tau_i$ implies $\mathcal{A}[\![o(M_1, \cdots, M_n)]\!] : P^*\Gamma \to P\tau$.

The first condition automatically holds: for a variable $x^\tau \in Var(\Gamma, \tau)$, we have $\mathcal{A}[\![x^\tau]\!] = \pi_{\gamma^{-1}(x)} : P^*\Gamma \to P\Gamma(\gamma(\gamma^{-1}(x))) = P\tau$.

So we show that the second condition is equivalent to $o_\mathcal{A} : P\tau_1 \times \cdots \times P\tau_n \to P\tau$ for each operator o of arity $\tau_1, \cdots, \tau_n \to \tau \in O_o$.

Suppose that the second condition holds. We have well-formed terms $\overrightarrow{\tau} \vdash_\Sigma \mathbf{v}_i^{\tau_i} : \tau_i$, and the first condition implies $\mathcal{A}[\![x_i^{\tau_i}]\!] : P^*\overrightarrow{\tau} \to P\tau_i$. From the the second condition, we have $\mathcal{A}[\![o(\mathbf{v}_1, \cdots, \mathbf{v}_n)]\!] : P^*\overrightarrow{\tau} \to P\tau$. In fact:

$$\begin{aligned} \mathcal{A}[\![o(\mathbf{v}_1^{\tau_1}, \cdots, \mathbf{v}_n^{\tau_n})]\!] &= (u_o)_\Gamma(\pi_1, \cdots, \pi_n) \\ &= o_\mathcal{A} \circ \langle \pi_1, \cdots, \pi_n \rangle = o_\mathcal{A} \end{aligned}$$

thus we conclude $o_\mathcal{A} : P^*\overrightarrow{\tau} \to P\tau$.

Conversely suppose that $o_A : P^*\overrightarrow{\tau} \to P\tau$. Let $\Gamma \vdash M_i : \tau_i$ $(1 \leq i \leq n)$ be well-formed terms and assume that $\mathcal{A}[\![M_i]\!] : P^*\Gamma \to P\tau_i$. Then

$$\begin{aligned} \mathcal{A}[\![o(M_1, \cdots, M_n)]\!] &= (u_o)_\Gamma(\mathcal{A}[\![M_1]\!], \cdots, \mathcal{A}[\![M_n]\!]) \\ &= o_\mathcal{A} \circ \langle \mathcal{A}[\![M_1]\!], \cdots, \mathcal{A}[\![M_n]\!] \rangle : P^*\Gamma \to P\tau. \end{aligned}$$

■

To summarise, the notion of pre-logical predicate along the standard interpretation coincides with the notion of subalgebra.

We instantiate the above discussion with *combinatory logic*. Combinatory logic is just a many-sorted algebra of the following signature:

$$\Sigma_{CL} \quad = \quad (\mathbf{Typ}^{\Rightarrow}(B), \{\bullet^{\tau\Rightarrow\tau',\tau\to\tau'}, K^{\tau\Rightarrow\tau'\Rightarrow\tau}, S^{(\tau\Rightarrow\tau'\Rightarrow\tau'')\Rightarrow(\tau\Rightarrow\tau')\Rightarrow\tau\Rightarrow\tau''}\})$$

where $\tau, \tau'\tau''$ range over $\mathbf{Typ}^{\Rightarrow}(B)$. We may omit type annotations for readability, and denote $\bullet$ as an infix left-associative operator. The typed formal system of this signature can be regarded as a term assignment system for a Hilbert style formulation of minimal propositional logic [TS96]. A *typed combinatory algebra* (*combinatory algebra* for short) $\mathcal{U}$ is just a many-sorted Σ_{CL}-algebra satisfying the following equations:

$$\begin{aligned} K_{\mathcal{U}} \bullet_{\mathcal{U}} x \bullet_{\mathcal{U}} y &= x \\ S_{\mathcal{U}} \bullet_{\mathcal{U}} x \bullet_{\mathcal{U}} y \bullet_{\mathcal{U}} z &= x \bullet_{\mathcal{U}} z \bullet_{\mathcal{U}} (y \bullet_{\mathcal{U}} z) \end{aligned}$$

(type annotations for variables and application operators are omitted for readability).

An *algebraic predicate* [Mit90, HS02] over a combinatory algebra $\mathcal{U}$ is a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of subsets $\{P\tau \subseteq U\tau\}_{\tau\in\mathbf{Typ}^{\Rightarrow}(B)}$ satisfying the following conditions:

$$\begin{aligned} & K_{\mathcal{U}}^{\tau\Rightarrow\tau'\Rightarrow\tau} \in P(\tau \Rightarrow \tau' \Rightarrow \tau) \\ & S_{\mathcal{U}}^{(\tau\Rightarrow\tau'\Rightarrow\tau'')\Rightarrow(\tau\Rightarrow\tau')\Rightarrow(\tau\Rightarrow\tau'')} \in P((\tau \Rightarrow \tau' \Rightarrow \tau'') \Rightarrow (\tau \Rightarrow \tau') \Rightarrow (\tau \Rightarrow \tau'')) \\ & \forall x \in P(\tau \Rightarrow \tau'), y \in P\tau \,.\, x \bullet_{\mathcal{U}}^{\tau\Rightarrow\tau',\tau\to\tau'} y \in P\tau' \end{aligned}$$

where $\tau, \tau'\tau''$ range over $\mathbf{Typ}^{\Rightarrow}(B)$. This exactly says that P is a submodel of $\mathcal{U}$. From proposition 4.1.1, we conclude the following corollary.

Corollary 4.1.2 *Algebraic predicates over a combinatory algebra $\mathcal{U}$ are exactly pre-logical predicates for Σ_{CL} along the standard interpretation $\mathcal{U}[\![-]\!]$.* □

Another example is *correspondences* by Schoett [Sch90]. Let $\Sigma = (T_0, O_0)$ be a many-sorted signature, $OBS \subseteq T_0$ be a set of types called *observable types* and $\mathcal{A}, \mathcal{B}$ be Σ-algebras [2]. A *correspondence* is a T_0-indexed family of binary relations

[2] Schoett originally considered partial algebras, but for simplicity here we just discuss correspondences for total algebras.

$\{R\tau \subseteq A\tau \times B\tau\}_{\tau \in T_0}$ such that R_σ is total bijective for each $\sigma \in OBS$, and for each operator $o \in T_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$,

$$\forall (a_1, b_1) \in R\tau_1, \cdots, (a_n, b_n) \in R\tau_n \,.\, (o_{\mathcal{A}}(a_1, \cdots, a_n), o_{\mathcal{B}}(b_1, \cdots, b_n)) \in R\tau$$

holds. The latter condition says that R is just a subalgebra of the product Σ-algebra $\mathcal{A} \times \mathcal{B}$. From proposition 4.1.1, R is a binary pre-logical relation for Σ along $\mathcal{A}[\![-]\!] \times \mathcal{B}[\![-]\!]$. Correspondences are the basis of a characterisation of behavioural inclusion and behavioural equivalence, which will be seen in the next chapter.

4.2 Interpretation of Lambda Terms via Combinatory Logic

In this example, we examine the relationship between pre-logical predicates for combinatory algebras and pre-logical predicates for the simply typed lambda calculus in our framework. This revisits proposition 3.3 in [HS02].

The standard abstraction mechanism in the combinatory logic is defined as follows (see definition 7.1.5 in [Bar84], types are omitted for readability):

$$\begin{aligned} \lambda^* x.x &= SKK \\ \lambda^* x.P &= KP \quad (x \notin \mathrm{FV}(P)) \\ \lambda^* x.PQ &= S(\lambda^* x.P)(\lambda^* x.Q) \end{aligned}$$

This induces a Π_λ-algebra structure (see example 3.3.2) over $S_{\Sigma_{CL}}$. By initiality, there exists a unique Π_λ-algebra morphism $(-)_{CL} : S_{\Pi_\lambda} \to S_{\Sigma_{CL}}$, which coincides with the standard lambda-to-CL translation (see definition 7.3.1, [Bar84]). Then for a combinatory algebra $\mathcal{U}$, the composition $\mathcal{U}[\![(-)_{CL}]\!]$ gives an interpretation of the simply typed lambda calculus in $\mathcal{U}$.

Conversely, representing S and K combinators by lambda terms equips S_{Π_λ} with a Σ_{CL} algebra structure. By initiality, there exists a unique Σ_{CL}-algebra morphism, namely $(-)_\lambda : S_{\Pi_\lambda} \to S_{\Sigma_{CL}}$.

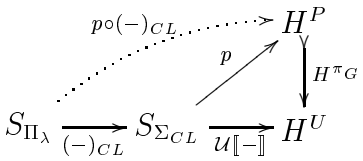
We note that the above is not the unique translation between lambda terms and

combinatory terms; both S_{Π_λ} and $S_{\Sigma_{CL}}$ may have other algebra structures depending on the choice of an abstraction mechanism and a representation of S and K.

We now relate algebraic predicates over a combinatory algebra (that is, pre-logical predicates for the combinatory logic along $\mathcal{U}[\![-]\!]$) and those for the lambda calculus along $\mathcal{U}[\![(-)_{CL}]\!]$.

Proposition 4.2.1 ("if" part of [HS02], proposition 3.3) *Let $\mathcal{U}$ be a combinatory algebra. If $P \subseteq_{Id_{\mathbf{Sets}}} U$ is a pre-logical predicate for Σ_{CL} along $\mathcal{U}[\![-]\!]$, then P is pre-logical for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$.* □

PROOF Let $P \subseteq_{Id_{\mathbf{Sets}}} U$ be a pre-logical predicate for Σ_{CL} along $\mathcal{U}[\![-]\!]$. From theorem 3.6.8, we have a morphism $p : S_{\Sigma_{CL}} \to H^P$ such that $H^{\pi_G} \circ p = \mathcal{U}[\![-]\!]$. By precomposing $(-)_{CL}$ with p, we obtain a morphism $p \circ (-)_{CL} : S_{\Pi_\lambda} \to H^P$ making the outer triangle in the following diagram commute:



This means that P satisfies the basic lemma for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$, that is, P is pre-logical by theorem 3.6.8. ■

What about the converse? The "only if" part of [HS02], proposition 3.3 claims that any pre-logical predicate for the lambda calculus along $\mathcal{U}[\![(-)_{CL}]\!]$ is an algebraic predicate. In our framework, this claim is reformulated as follows:

> ("only if" part of [HS02], proposition 3.3) Let $\mathcal{U}$ be a combinatory algebra. If $P \subseteq_{Id_{\mathbf{Sets}}} U$ is a pre-logical predicate for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$, then P is pre-logical for Σ_{CL} along $\mathcal{U}[\![-]\!]$.

However, the above claim is not true in general. Below we construct a counterexample to their claim.

Proposition 4.2.2 *There exists a combinatory algebra $\mathcal{U}_0$ and a pre-logical predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} U_0$ for Π_λ along $\mathcal{U}_0[\![(-)_{CL}]\!]$ which is not pre-logical for Σ_{CL} along $\mathcal{U}_0[\![-]\!]$.*

□

PROOF The proof uses the fact that the image of the standard lambda-to-CL translation does not cover the entire set of combinatory logic terms. In particular the terms S and K themselves are not in the image of this translation. To exploit this fact, we take $\mathcal{U}_0$ to be the closed term algebra, and show that the predicate consisting of the values definable by $\mathcal{U}_0[\![(-)_{CL}]\!]$, which is pre-logical for Π_λ, is not pre-logical for Σ_{CL} along $\mathcal{U}_0[\![-]\!]$.

We recall that *weak reduction* (written $\rightarrow_w$, see [Bar84], definition 7.2.1) is the least compatible relation generated from the following relation:

$$\begin{aligned} & \{(KMN, M) \mid \Gamma \vdash_{\Sigma_{CL}} KMN, M : \tau\} \\ \cup \quad & \{(SLMN, LN(MN) \mid \Gamma \vdash_{\Sigma_{CL}} SLMN, LN(MN) : \tau\}. \end{aligned}$$

We write $\twoheadrightarrow_w$ for the transitive reflexive closure of $\rightarrow_w$, and $=_w$ for the symmetric closure of $\twoheadrightarrow_w$. We assume the Church-Rosser theorem and strong normalisation theorem of weak reduction [HS86].

We define the *closed term combinatory algebra* $\mathcal{U}_0$ as follows.

- The carrier set U_0 is defined by:

$$U_0\tau = \{[M]_w \mid M \in \emptyset \vdash_{\Sigma_{CL}} M : \tau\}$$

  where $[M]_w$ is the equivalence class of combinatory terms by $=_w$.

- To each operator in Σ_{CL}, we assign the following constants and functions.

$$\begin{aligned} K_{\mathcal{U}_0}^{\tau \Rightarrow \tau' \Rightarrow \tau} &= [K^{\tau \Rightarrow \tau' \Rightarrow \tau}]_w \\ S_{\mathcal{U}_0}^{(\tau \Rightarrow \tau' \Rightarrow \tau'') \Rightarrow (\tau \Rightarrow \tau') \Rightarrow (\tau \Rightarrow \tau'')} &= [S^{(\tau \Rightarrow \tau' \Rightarrow \tau'') \Rightarrow (\tau \Rightarrow \tau') \Rightarrow (\tau \Rightarrow \tau'')}]_w \\ [M]_w \bullet_{\mathcal{U}_0}^{\tau \Rightarrow \tau', \tau \rightarrow \tau'} [N]_w &= [M \bullet N]_w \end{aligned}$$

It is easy to see that the above choice of combinators satisfies the axioms of combinatory algebra. As we have seen in section 4.1, we obtain an interpretation function of combinatory logic terms in $\mathcal{U}_0$, namely $\mathcal{U}_0[\![-]\!]$. Routine calculation shows that

$$\mathcal{U}_0[\![M]\!]\{x_1 \mapsto [M_1]_w, \cdots, x_n \mapsto [M_n]_w\} = [M[M_1/x_1, \cdots, M_n/x_n]]_w.$$

Lambda terms are interpreted by $\mathcal{U}_0[\![(-)_{CL}]\!]$.

Now we define the definability predicate $D\tau \subseteq U_0\tau$ by

$$D\tau = \{\mathcal{U}_0[\![M_{CL}]\!] \in U_0\tau \mid \emptyset \vdash_{\Pi_\lambda} M : \tau\}.$$

Proposition 4.2.3 *D is a pre-logical predicate for Π_λ along $\mathcal{U}_0[\![(-)_{CL}]\!]$.* □

PROOF We directly prove the basic lemma. Let $\Gamma \vdash_{\Pi_\lambda} M : \tau$ be a well-formed term and $\rho \in D^*\Gamma$. From the definition of D, for each $x \in \mathrm{dom}(\Gamma)$, there exists a term M_x such that $\rho(x) = [M_x]_w$. Therefore we have

$$\mathcal{U}_0[\![M]\!]\rho = \mathcal{U}_0[\![M]\!][[M_x]_w/x]_{x\in\mathrm{dom}(\Gamma)} = [M[M_x/x]_{x\in\mathrm{dom}(\Gamma)}]_w \in D\tau.$$

However, as we show below, $D(\tau \Rightarrow \tau' \Rightarrow \tau)$ does not include $[K^{\tau\Rightarrow\tau'\Rightarrow\tau}]_w$ for any type τ and τ'! Therefore D is not a pre-logical predicate for Σ_{CL} along $\mathcal{U}_0[\![-]\!]$. To see this, we first prove the following lemmas. We omit typings for readability.

Lemma 4.2.4 *There exists no closed term M such that $M_{CL} = K$.* □

PROOF Easy induction on the structure of M. ■

Lemma 4.2.5 *For any closed lambda term M and any combinatory term N, $M_{CL} \twoheadrightarrow_w N$ implies that there exists a closed lambda term N' such that $N = N'_{CL}$.* □

PROOF When M begins with a lambda abstraction, M_{CL} is always a normal form. Thus the claim clearly holds by taking $N' = M$. We do not consider the case where M is a variable, since we assume that M is closed. So we think of the case where $M = M_0M_1$ with two closed lambda terms M_0 and M_1. There are several possible causes of $M_{CL} \twoheadrightarrow_w N$.

- $N = M_{CL}$. We just take $N' = M$.
- There exists a combinatory term L such that $(M_0)_{CL} \twoheadrightarrow_w L$ and $L \twoheadrightarrow_w N$. From IH, there exists a combinatory term L' such that $L = L'_{CL}$. Again from IH, there exists a combinatory term N' such that $N = N'_{CL}$.
- $(M_0)_{CL} \twoheadrightarrow_w N_0$ and $N = N_0(M_1)_{CL}$. From IH, there exists a closed lambda term N'_0 such that $(N'_0)_{CL} = N_0$. Thus $N = (N'_0)_{CL}(M_1)_{CL} = (N'_0M_1)_{CL}$.

- $(M_1)_{CL} \twoheadrightarrow_w N_1$ and $N = (M_0)_{CL} N_1$. The proof is similar to the above case.
- $(M_0)_{CL} = K N_0$ with a combinatory term N_0 and $N = N_0$. From the definition of lambda-to-CL translation, M_0 should be equal to $\lambda x\,.\,N_0'$ where N_0' is a closed lambda term. Thus $N_0 = (N_0')_{CL}$.
- $(M_0)_{CL} = S N_0 N_1$ with combinatory terms N_0, N_1 and $N = N_0(M_1)_{CL}(N_1(M_1)_{CL})$. From the definition of lambda-to-CL translation, we have $M_0 = \lambda x\,.\,(N_0' N_1'), N_0 = (N_0')_{CL}$ and $N_1 = (M_1')_{CL}$. Thus we take $N' = N_0' M_1 (N_1' M_1)$. ■

Proposition 4.2.6 *For all types τ and τ', the following holds.*

1. *For any well-formed term $\Gamma \vdash_{\Pi_\lambda} M : \tau \Rightarrow \tau' \Rightarrow \tau$ and $\rho \in D^*\Gamma$, $\mathcal{U}_0[\![M_{CL}]\!]\rho \neq [K^{\tau \Rightarrow \tau' \Rightarrow \tau}]_w$.*
2. *$[K^{\tau \Rightarrow \tau' \Rightarrow \tau}]_w \notin D(\tau \Rightarrow \tau' \Rightarrow \tau)$.* □

PROOF 1. By definition, for each $x \in \mathrm{dom}(\Gamma)$, there exists a closed lambda term M_x such that $\rho(x) = \mathcal{U}_0[\![(M_x)_{CL}]\!]$. Thus

$$\begin{aligned} \mathcal{U}_0[\![M_{CL}]\!]\rho &= [M_{CL}[(M_{x_1})_{CL}/x_1, \cdots, (M_{x_n})_{CL}/x_n]]_w \\ &= [(M[M_{x_1}/x_1, \cdots, M_{x_n}/x_n])_{CL}]_w. \end{aligned}$$

From the Church-Rosser property, $\mathcal{U}_0[\![M_{CL}]\!]\rho = [K]_w$ implies

$$(M[M_{x_1}/x_1, \cdots, M_{x_n}/x_n])_{CL} \twoheadrightarrow_w K,$$

but this contradicts lemma 4.2.5.

2. By letting $\Gamma = \emptyset$ in the above case. ■

Two Corrections to [HS02], Proposition 3.3

Now the question is to seek a condition under which pre-logical predicates for combinatory algebras and those for the lambda calculus coincide. We try to repair [HS02], proposition 3.3 formulated as follows:

> ([HS02], proposition 3.3) Let $\mathcal{U}$ be a combinatory algebra and $P \subseteq_{Id_{\mathbf{Sets}}} U$ be a predicate. Then P is pre-logical for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$ if and only if P is pre-logical for Σ_{CL} along $\mathcal{U}[\![-]\!]$.

Here we provide two answers.

Answer One: Restricting the Class of Combinatory Algebras

From easy diagram chasing, we obtain the following lemma.

Lemma 4.2.7 *Let $\mathcal{U}$ be a combinatory algebra. If $P \subseteq_{Id_{\mathbf{Sets}}} U$ is a pre-logical predicate for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$, then P is pre-logical for Σ_{CL} along $\mathcal{U}[\![((-)_\lambda)_{CL}]\!]$.* □

Only the difference between the above proposition and "only if" part of [HS02], proposition 3.3 is the interpretation of combinatory logic terms: $\mathcal{U}[\![((-)_\lambda)_{CL}]\!]$ and $\mathcal{U}[\![-]\!]$. From this observation, we obtain the first answer:

Proposition 4.2.8 *Let $\mathcal{U}$ be a combinatory algebra satisfying $\mathcal{U}[\![((-)_\lambda)_{CL}]\!] = \mathcal{U}[\![-]\!]$. Then for any predicate $P \subseteq_{Id_{\mathbf{Sets}}} U$, P is pre-logical for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$ if and only if P is pre-logical for Σ_{CL} along $\mathcal{U}[\![-]\!]$.* □

An example of the class of the combinatory algebras satisfying $\mathcal{U}[\![((-)_\lambda)_{CL}]\!] = \mathcal{U}[\![-]\!]$ is the class of the typed version of *lambda algebras* (definition 5.2.2, [Bar84]). A lambda algebra $\mathcal{U}$ is a combinatory algebra satisfying the following set of axioms (see definition 7.3.6, [Bar84]):

$$\begin{aligned}
K_{\mathcal{U}} = \mathcal{U}[\![\lambda^* xy.Kxy]\!] &= \mathcal{U}[\![\lambda^* xy.x]\!] \\
S_{\mathcal{U}} = \mathcal{U}[\![\lambda^* xyz.Sxyz]\!] &= \mathcal{U}[\![\lambda^* xyz.xz(yz)]\!] \\
\mathcal{U}[\![\lambda^* xy.S(Kx)(Ky)]\!] &= \mathcal{U}[\![\lambda^* xy.K(xy)]\!] \\
\mathcal{U}[\![\lambda^* xy.S(S(KK)x)y]\!] &= \mathcal{U}[\![\lambda^* xyz.xz]\!] \\
\mathcal{U}[\![\lambda^* xyz.S(S(S(KS)x)y)z]\!] &= \mathcal{U}[\![\lambda^* xyz.S(Sxz)(Syz)]\!]
\end{aligned}$$

The first and second line guarantees that we have

$$K_{\mathcal{U}} = \mathcal{U}[\![(\lambda xy.x)_{CL}]\!], \quad S_{\mathcal{U}} = \mathcal{U}[\![(\lambda xyz.xz(yz))_{CL}]\!].$$

By easy induction we have $\mathcal{U}[\![((-)_\lambda)_{CL}]\!] = \mathcal{U}[\![-]\!]$. We summarise this example as a corollary of proposition 4.2.8:

Corollary 4.2.9 *Let $\mathcal{U}$ be a lambda algebra. Then for any predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} U$, P is pre-logical for Σ_{CL} along $\mathcal{U}[\![-]\!]$ if and only if P is pre-logical for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!]$.*

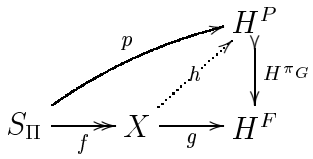
□

Lambda algebras include many model classes for the lambda calculus, such as the typed version of lambda models, Henkin models and full type hierarchies. Therefore [HS02], proposition 3.3 still holds for these classes of models, and the value of the characterisation of pre-logical predicates remains.

Answer Two: Changing Lambda-to-CL Translation

Another answer is to replace the lambda-to-CL translation $(-)_{CL}$. In the proof of proposition 4.2.2, we use the fact that $(-)_{CL}$ is not surjective. So what if we use another translation which is surjective? The following lemma supports this idea.

Lemma 4.2.10 *Let $\mathbb{C}[\![-]\!]_F$ be an interpretation of Π and $P \subseteq_G F$ be a pre-logical predicate for Π along $\mathbb{C}[\![-]\!]_F$. Suppose $\mathbb{C}[\![-]\!]_F$ factors to an epi $f : S_\Pi \to X$ followed by $g : X \to H^F$. Then there exists a unique morphism $h : X \to H^P$ making the following triangles commute:*



PROOF Recall that $\mathbb{P}_T$ is a topos. Thus all epimorphisms are orthogonal to monomorphisms, that is, in the given situation there exists a unique morphism $h : X \to H^P$ making the above triangles commute (see e.g. [Bor94]). ∎

We let $X = S_{\Sigma_{CL}}$, $g = \mathcal{U}[\![-]\!]$ and find an epimorphism $(-)_{CL'} : S_{\Pi_\lambda} \to S_{\Sigma_{CL}}$. Then from the above lemma, we obtain a morphism $h : S_{\Sigma_{CL}} \to H^P$ such that $h \circ H^{\pi_G} = \mathcal{U}[\![-]\!]$, which implies that P is pre-logical for $\Pi_{\Sigma_{CL}}$ along $\mathcal{U}[\![-]\!]$.

We construct $(-)_{CL'}$ by finding another Π_λ algebra structure over $S_{\Sigma_{CL}}$ which is different from the one obtained by the standard abstraction mechanism. We consider

the following modified abstraction mechanism which can yield bare S and K:

$$\begin{aligned}
&\lambda' x.x = SKK \\
&\lambda' x.M = KM \quad (x \notin \mathrm{FV}(M)) \\
&\lambda' x.Kx = K \\
&\lambda' x.S(K(S(S(Kx)(SKK))))(S(S(KS)(S(KK)(SKK)))(K(SKK))) = S \\
&\lambda' x.MN = S(\lambda' x.M)(\lambda' x.N) \quad \text{(otherwise)}.
\end{aligned}$$

The third and fourth lines are actually the expansion of the following definition:

$$\begin{aligned}
&\lambda' x.\lambda^* y.x = K \\
&\lambda' x.\lambda^* y.\lambda^* z.xz(yz) = S.
\end{aligned}$$

Intuitively, $\lambda' x.-$ behaves almost the same as $\lambda^* x.-$ except that when it recognises that the input is $\lambda xyz.xz(yz)$ or $\lambda xy.x$, it produces S and K respectively. The lambda-to-CL translation constructed from this abstraction mechanism satisfies the following property.

Proposition 4.2.11 *For all combinatory logic term M, $(M_\lambda)_{CL'} = M$.* □

PROOF We prove this by induction on the structure of M.

- Case $M = x$. Clearly $(x_\lambda)_{CL'} = x$.
- Case $M = K$. Note that $\lambda' y.x = Kx = \lambda^* y.x$. Thus

$$((K_\lambda)_{CL'}) = (\lambda xy\ .\ x)_{CL'} = \lambda' x.(\lambda' y.x) = \lambda' x.(\lambda^* y.x) = K.$$

- Case $M = S$. Note that $\lambda' yz.xz(yz) = \lambda^* yz.xz(yz)$. Thus

$$((S_\lambda)_{CL'}) = (\lambda xyz\ .\ xz(yz))_{CL'} = \lambda' x.(\lambda' yz.xz(yz)) = \lambda' x.(\lambda^* yz.xz(yz)) = S.$$

- Case $M = M_0 M_1$. We have

$$((M_0 M_1)_\lambda)_{CL'} = ((M_0)_\lambda)_{CL'}((M_1)_\lambda)_{CL'} = M_0 M_1.$$

We write $(-)_{CL'}$ for the lambda-to-CL translation using $\lambda' x.-$. The above proposition implies that $(-)_{CL'}$ considered as an interpretation of Π_λ is an epimorphism.

Proposition 4.2.12 *Let $\mathcal{U}$ be a combinatory algebra. Then for any predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} U$, P is pre-logical for Σ_{CL} along $\mathcal{U}[\![-]\!]$ if and only if P is pre-logical for Π_λ along $\mathcal{U}[\![(-)_{CL'}]\!]$.* □

4.3 Lax Logical Predicates and Pre-Logical Predicates

Lax logical predicates [PPST00] are a weakening of categorical formulations of logical predicates [MR92, MS93], and are closely related to pre-logical predicates.

We first recall a categorical formulation of logical predicates in the functorial semantics of the lambda calculus [MR92]. We write $\mathbf{L}$ for the CCC generated from the lambda calculus (see definition 4.3.1). The categorical semantics of the lambda calculus is given by a CCC $\mathbb{C}$ and a functor $[\![-]\!] : \mathbf{L} \to \mathbb{C}$ which strictly preserves the Cartesian closed structure. Now suppose that we have another CCC $\mathbb{D}$ with pullbacks and a functor $G : \mathbb{C} \to \mathbb{D}$ preserving finite products. It is well-known that $\mathbf{Pred}(G)$ has a CCC structure which is strictly preserved by the forgetful functor $\pi_G : \mathbf{Pred}(G) \to \mathbb{C}$ (see e.g. [MR92]). In this situation, to give a logical predicate over $[\![-]\!]$ is equivalent to giving a functor $P : \mathbf{L} \to \mathbf{Pred}(G)$ which strictly preserves the Cartesian closed structure and satisfies $\pi_G \circ P = [\![-]\!]$.

In [PPST00], Plotkin et al. discovered that the basic lemma still holds without the condition that P preserves exponentials, and its converse also holds. They called such functors *lax logical predicates*. That is, a lax logical predicate is a functor $P : \mathbf{L} \to \mathbf{Pred}(G)$ which strictly preserves finite products (but may not exponentials) and satisfies $\pi_G \circ P = [\![-]\!]$. Furthermore, binary lax logical predicates are closed under composition.

The major differences between lax logical predicates and the original pre-logical predicates [HS02] are threefold.

- Lax logical predicates are for the lambda calculus with finite products, while

pre-logical predicates are for the minimal lambda calculus.

- Lax logical predicates are defined with respect to the functorial semantics, while the original pre-logical predicates are defined with respect to interpretations in lambda applicative structures (example 3.4.3).

- Lax logical predicates are defined as G-predicates (definition 3.5.1), while the original pre-logical predicates are defined as subsets.

Despite these differences, both lax logical predicates and the original pre-logical predicates serve to give a characterisaion of the basic lemma. We can resolve these differences with our generalised framework of pre-logical predicates, and establish a formal relationship between them. In this section, we show that lax logical predicates are equivalent to pre-logical predicates for the lambda calculus with finite products along the standard interpretation in $\mathbb{C}$.

We define the syntax of the lambda calculus with finite products. We fix a set of base types B and define the set of types including finite products by the BNF:

$$\mathbf{Typ}^{\Rightarrow\times}(B) \ni \tau ::= b \mid \prod_{i=1}^{n} \tau_i \mid \tau \Rightarrow \tau$$

where b, n range over $B, \mathbf{N}$ respectively. The calculus extends the typing rules of the lambda calculus in example 3.3.2 with the following rules for finite products.

$$\frac{\Gamma \vdash M_i : \tau_i \quad 1 \leq i \leq n}{\Gamma \vdash \langle M_1, \cdots, M_n \rangle : \prod_{i=1}^{n} \tau_i} \qquad \frac{\Gamma \vdash M : \prod_{i=1}^{n} \tau_i}{\Gamma \vdash \pi_i(M) : \tau_i}$$

The typing rules fit within the scheme of simply typed formal systems. We define the typed binding signature $\Pi_{\lambda\times}$ for the lambda calculus with finite products by

$$\begin{aligned} \Pi_{\lambda\times} \quad = \quad & (\mathbf{Typ}^{\Rightarrow\times}(B), \\ & \{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}, \mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}, \mathsf{tuple}^{\tau_1,\cdots,\tau_n\to\prod_{i=1}^{n}\tau_i}, \mathsf{proj}^{\prod_{i=1}^{n}\tau_i\to\tau_i}\}) \end{aligned}$$

where $\tau, \tau', \tau_1, \cdots$ and n range over $\mathbf{Typ}^{\Rightarrow\times}(B)$ and $\mathbf{N}$ respectively.

The semantics of the lambda calculus with finite products in a CCC $\mathbb{C}$ is fairly standard. An interpretation of types $F : \mathbf{Typ}^{\Rightarrow\times}(B) \to \mathbb{C}$ is *standard* if it satisfies

$F(\tau \Rightarrow \tau') = F\tau \Rightarrow F\tau'$ and $F\left(\prod_{i=1}^n \tau_i\right) = \prod_{i=1}^n F\tau_i$. For a standard interpretation of types F, H^F has the following $\Pi_{\lambda\times}$-algebra structure (c.f. example 3.4.4):

$$\begin{aligned}
(u_{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}})_\Gamma(f) &= \lambda(f \circ s_{\Gamma,\tau}) \\
(u_{\mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}})_\Gamma(f,g) &= @ \circ \langle f, g\rangle \\
(u_{\mathsf{pair}^{\tau_1,\cdots,\tau_n\to\prod_{i=1}^n\tau_i}})_\Gamma(f_1,\cdots,f_n) &= \langle f_1,\cdots,f_n\rangle \\
(u_{\mathsf{proj}^{\prod_{i=1}^n\tau_i\to\tau_i}})_\Gamma(f) &= \pi_i \circ f
\end{aligned}$$

From initiality, we obtain a morphism $\mathbb{C}[\![-]\!]_F : S_{\Pi_\lambda} \to H^F$. We call this the *standard interpretation* of the lambda calculus with finite products in $\mathbb{C}$.

Definition 4.3.1 We define the *term category* $\mathbf{L}$ by the following data.

An object in $\mathbf{L}$ is a type $\tau \in \mathbf{Typ}^{\Rightarrow\times}(B)$.

A morphism from τ to τ' in $\mathbf{L}$ is the $\beta\eta$-equivalence class of a well-formed term $x : \tau \vdash_{\Pi_{\lambda\times}} M : \tau'$.

We write $I : \mathbf{Typ}^{\Rightarrow\times}(B) \to \mathbf{L}$ for the evident inclusion functor. □

Proposition 4.3.2 *1. Category* $\mathbf{L}$ *is a CCC.*

2. *For each well-formed term* $\Gamma \vdash_{\Pi_{\lambda\times}} M : \tau$*, we have*

$$\mathbf{L}[\![M]\!]_I = [M[\pi_{\gamma(y)}(x)/y]_{y\in\mathrm{dom}(\Gamma)}]_{\beta\eta}.$$

3. *Let* $\mathbb{C}$ *be a CCC. For any standard interpretation of types* $F : \mathbf{Typ}^{\Rightarrow\times}(B) \to \mathbb{C}$*, there exists a unique functor* $\overline{F} : \mathbf{L} \to \mathbb{C}$ *preserving the Cartesian closed structure strictly such that* $\overline{F} \circ I = F$ *and* $H^{\overline{F}} \circ \mathbf{L}[\![-]\!]_I = \mathbb{C}[\![-]\!]_F$. □

PROOF 1. See e.g. [Cro94].

2. Easy induction on the structure of M.

3. We write $i_{F\tau} : F\tau \to F^*\{x : \tau\}$ for the canonical isomorphism. We note that $i_{F\tau}^{-1} = \mathbb{C}[\![x]\!]_F$. We define $\overline{F} : \mathbf{L} \to \mathbb{C}$ by $\overline{F}\tau = F\tau$ for an object τ in $\mathbf{L}$, and $\overline{F}([M]_{\beta\eta}) = \mathbb{C}[\![M]\!]_F \circ i_{F\tau}$ for a morphism $[M]_{\beta\eta} : \tau \to \tau'$

in $\mathbf{L}$. It is easy to check that $\overline{F}$ is a functor and satisfies $\overline{F} \circ I = F$. We check that $\overline{F}$ strictly preserves Cartesian closed structure. For each projection $x : \prod_{i=1}^{n} \tau_i \vdash_{\Pi\lambda\times} \pi_i(x) : \tau_i$, we have:

$$\overline{F}([\pi_i(x)]_{\beta\eta}) = \mathbb{C}[\![\pi_i(x)]\!]_F \circ i_{F\tau} = \pi_i \circ \mathbb{C}[\![x]\!] \circ i_{F\tau} = \pi_i \circ i_{F\tau}^{-1} \circ i_{F\tau} = \pi_i.$$

Therefore he comparison map $\langle \overline{F}([\pi_1(x)]_{\beta\eta}), \cdots, \overline{F}([\pi_n(x)]_{\beta\eta}) \rangle$ is equal to id, that is, $\overline{F}$ strictly preserves finite products.

To see that $\overline{F}$ strictly preserves exponentials, we show $\lambda(\overline{F}([\pi_1(x)\pi_2(x)]_{\beta\eta})) = \mathrm{id}$ where $[\pi_1(x)\pi_2(x)]_{\beta\eta} : (\tau \Rightarrow \tau') \times \tau \to \tau'$ is an evaluation morphism in $\mathbf{L}$.

$$\begin{aligned}
& \lambda(\overline{F}([\pi_1(x)\pi_2(x)]_{\beta\eta})) \\
= \;& \lambda(\mathbb{C}[\![\pi_1(x)\pi_2(x)]\!]_F \circ i_{F\tau}) \\
= \;& \lambda(@ \circ \langle \mathbb{C}[\![\pi_1(x)]\!]_F, \mathbb{C}[\![\pi_2(x)]\!]_F \rangle \circ i_{F\tau}) \\
= \;& \lambda(@ \circ \langle \pi_1 \circ i_{F\tau}^{-1}, \pi_2 \circ i_{F\tau}^{-1} \rangle \circ i_{F\tau}) \\
= \;& \lambda(@) = \mathrm{id}
\end{aligned}$$

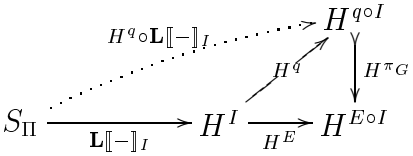
Now we check for each well-formed term $\Gamma \vdash M : \tau$, $\overline{F} \circ \mathbf{L}[\![M]\!]_I = \mathbb{C}[\![-]\!]_F$.

$$\begin{aligned}
\overline{F} \circ \mathbf{L}[\![M]\!]_I &= \mathbb{C}[\![M[\pi_{\gamma(y)}(x)/y]_{y \in \mathrm{dom}(\Gamma)}]\!]_F \circ i_{F\tau} \\
&= \mathbb{C}[\![M]\!]_F \circ \langle \mathbb{C}[\![\pi_1(x)]\!]_F, \cdots, \mathbb{C}[\![\pi_n(x)]\!]_F, \rangle \circ i_{F\tau} \\
&= \mathbb{C}[\![M]\!]_F \circ \langle \pi_1 \circ i_{F\tau}^{-1}, \cdots, \pi_n \circ i_{F\tau}^{-1} \rangle \circ i_{F\tau} = \mathbb{C}[\![M]\!]_F
\end{aligned}$$ ■

Definition 4.3.3 ([PPST00]) Let $\mathbb{C}$ be a CCC, $E : \mathbf{L} \to \mathbb{C}$ be a functor preserving Cartesian closed structure strictly, $\mathbb{D}$ be a Cartesian closed category with pullbacks and $G : \mathbb{C} \to \mathbb{D}$ be a functor preserving finite products. A *lax logical predicate* over E is a functor $q : \mathbf{L} \to \mathbf{Pred}(G)$ which strictly preserves finite products and satisfies $\pi_G \circ q = E$. □

Proposition 4.3.4 (Basic Lemma[PPST00]) *In the situation in definition 4.3.3, the lax logical predicate q determines a pre-logical predicate $q \circ I \subseteq_G E \circ I$ for $\Pi_{\lambda\times}$ along $\mathbb{C}[\![-]\!]_{E \circ I}$.* □

PROOF From $E = \pi_G \circ q$, we have a G-predicate $q \circ I$ over the standard interpretation of types $E \circ I$ that makes the lower right triangle commute.



Thus we have a morphism $H^q \circ \mathbf{L}[\![-]\!]_I$ making the outer triangle commute. ■

To show the converse, we first show that $\mathbf{L}[\![-]\!]_I$ is an epimorphism.

Lemma 4.3.5 *The morphism $\mathbf{L}[\![-]\!]_I : S_{\Pi_{\lambda\times}} \to H^I$ is an epimorphism.* □

PROOF We show that $\mathbf{L}[\![-]\!]_I$ is an epimorphism at each object (Γ, τ) in $\mathbb{V}_T \times T$. We take an equivalence class $[M]_{\beta\eta} \in H^I(\Gamma, \tau)$ with a representative well-formed term $x : I^*\Gamma \vdash M : \tau$. We calculate $\mathbf{L}[\![M[\langle \gamma^{-1}(1), \cdots, \gamma^{-1}(n)\rangle / x]]\!]_I = [M]_{\beta\eta}$ as follows:

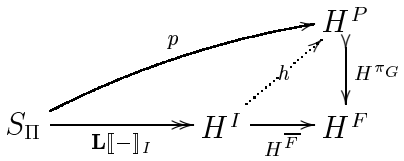
$$
\begin{aligned}
& \mathbf{L}[\![M[\langle \gamma^{-1}(1), \cdots, \gamma^{-1}(n)\rangle / x]]\!]_I \\
= \; & [M[\langle \gamma^{-1}(1), \cdots, \gamma^{-1}(n)\rangle / x][\pi_{\gamma(y)}(x)/y]_{y \in \mathrm{dom}(\Gamma)}]_{\beta\eta} \qquad \text{(proposition 4.3.2)} \\
= \; & [M[\langle \pi_1(x), \cdots, \pi_n(x)\rangle / x]]_{\beta\eta} \\
= \; & [M]_{\beta\eta}
\end{aligned}
$$

Thus $\mathbf{L}[\![-]\!]_I$ is epi. ■

Theorem 4.3.6 *Let $\mathbb{C}$ be a CCC, $F : \mathbf{Typ}^{\Rightarrow\times}(B) \to \mathbb{C}$ be a standard interpretation of types and $P \subseteq_G F$ be a pre-logical predicate for $\Pi_{\lambda\times}$ along $\mathbb{C}[\![-]\!]_F$. Then there exists a lax logical predicate $q : \mathbf{L} \to \mathbf{Pred}(G)$ such that for each well-formed term $\Gamma \vdash_{\Pi_{\lambda\times}} M : \tau$, $H^q \circ \mathbf{L}[\![M]\!]_I = p(\Gamma, \tau)(M)$.* □

PROOF The assumption says that there exists a morphism $p : S_{\Pi_{\lambda\times}} \to H^P$ such that $H^{\pi_G} \circ p = \mathbb{C}[\![-]\!]_F$. From proposition 4.3.2, there exists a functor $\overline{F} : \mathbf{L} \to \mathbb{C}$ preserving Cartesian closed structure strictly such that $F = \overline{F} \circ I$ and $H^{\overline{F}} \circ \mathbf{L}[\![-]\!]_I = \mathbb{C}[\![-]\!]_F$. Morphism $\mathbf{L}[\![-]\!]_I$ is an epimorphism by corollary 4.3.5 and morphism H^{π_G} is

mono. From lemma 4.2.10, there exists a unique morphism $h : H^I \to H^P$ such that $H^{\pi_G} \circ h = H^{\overline{F}}$ and $h \circ \mathbf{L}[\![-]\!]_I = p$.



Now we define the functor $q : \mathbf{L} \to \mathbf{Pred}(G)$ in question by $q\tau = P\tau$ for an object τ in $\mathbf{L}$ and $qf = h_{\{x:\tau\},\tau'}(f \circ i_{I\tau}^{-1}) \circ i_{P\tau}$ for a morphism $f : \tau \to \tau'$ in $\mathbf{L}$, where $i_{I\tau}^{-1} : I^*\{x : \tau\} \to I\tau$ and $i_{P\tau} : P\tau \to P^*\{x : \tau'\}$ are isomorphisms. We first show that $\pi_G \circ q(f) = \overline{F}f$.

$$
\begin{aligned}
& \pi_G(h_{\{x:\tau\},\tau'}(f \circ i_{I\tau}^{-1}) \circ i_P) \\
= \;& \pi_G(h_{\{x:\tau\},\tau'}(f \circ i_{I\tau}^{-1})) \circ \pi_G(i_{P\tau}) \\
= \;& \overline{F}(f \circ i_{I\tau}^{-1}) \circ \pi_G(i_{P\tau}) \quad (H^{\overline{F}} = H^{\pi_G} \circ h) \\
= \;& \overline{F}f \circ \overline{F}(i_{I\tau}^{-1}) \circ \pi_G(i_{P\tau}) \quad (\overline{F}(i_{I\tau}^{-1}) = i_{\overline{F}I\tau}^{-1} = i_{F\tau}^{-1}, \pi_G(i_{P\tau}) = i_{\pi_G P\tau} = i_{F\tau}) \\
= \;& \overline{F}f.
\end{aligned}
$$

We show that q is indeed a functor, that is, it preserves identity morphisms and composition of morphisms, using the faithfulness of π_G.

$$
\begin{aligned}
\pi_G \circ q(\mathrm{id}) &= \overline{F}(\mathrm{id}) = \mathrm{id} = \pi_G(\mathrm{id}) \\
\pi_G \circ q(f \circ g) &= \overline{F}f \circ \overline{F}g = (\pi_G \circ q(f)) \circ (\pi_G \circ q(g)) = \pi_G(q(f) \circ q(g))
\end{aligned}
$$

Since π_G is faithful, we conclude that q is a functor. Next, we show that q strictly preserves finite products. To see this, we show that the comparison map $\langle q\pi_1, \cdots, q\pi_n \rangle$ is the identity morphism using the faithfulness of π_G.

$$
\begin{aligned}
& \pi_G\langle q\pi_1, \cdots, q\pi_n \rangle \\
= \;& \langle \pi_G \circ q(\pi_1), \cdots, \pi_G \circ q(\pi_n) \rangle \quad (\pi_G \text{ strictly preserves finite products}) \\
= \;& \langle \overline{F}\pi_1, \cdots, \overline{F}\pi_n \rangle \quad (\pi_G \circ q = \overline{F}) \\
= \;& \overline{F}\langle \pi_1, \cdots, \pi_n \rangle \quad (\overline{F} \text{ strictly preserves finite products}) \\
= \;& \mathrm{id} = \pi_G(\mathrm{id})
\end{aligned}
$$

To show $H^q \circ \mathbf{L}[\![-]\!]_I = p$, we calculate the following:

$$H^{\pi_G} \circ H^q \circ \mathbf{L}[\![-]\!]_I = H^{\overline{F}} \circ \mathbf{L}[\![-]\!]_I = H^{\pi_G} \circ p$$

Now H^{π_G} is mono, thus $H^q \circ \mathbf{L}[\![-]\!]_I = p$. ∎

In the latter part of [PPST00], Plotkin et al. extend lax logical predicates to the languages described by a descrete algebraic structure extending finite product structure. We do not know if their extension can also be characterised by means of our generalised pre-logical predicates.

4.4 An Example from Moggi's Computational Metalanguage

Moggi's computational metalanguage [Mog91] is an extension of the simply typed lambda calculus so that various computational effects, such as non-termination, exceptions, I/O, states, nondeterminism and jumps, can be handled explicitly. This language is used as an intermediate platform of semantics, compilation and program transformation for call-by-value languages [Sta96, HD97, BKR99, BHM02].

The type system of the computational metalanguage fits within the scheme of a typed formal system, and its semantics can be described as the initial algebra semantics. As we have seen in chapter 3, we use this fact to derive pre-logical predicates for the computational metalanguage. Then we give a concrete example of a pre-logical predicate, inspired by Stark and Lindley's leapfrog method [Lin04, LS05], which is used to prove a strong normalisation theorem for the computational metalanguage.

Syntax of Computational Metalanguage

We begin with the syntax of the computational metalanguage. The language we consider here is the minimal fragment having only base types, arrow types and computational types. Let B be the set of base types. We define the set of types $\mathbf{Typ}^{\Rightarrow T}(B)$ for the computational metalanguage by the following BNF:

$$\mathbf{Typ}^{\Rightarrow T}(B) \ni \tau ::= b \mid \tau \Rightarrow \tau \mid T\tau$$

where b ranges over B. Compared to the simply typed lambda calculus, new types $T\tau$ called *computational types* are added to express the type of a program yielding a value of τ involving computational effects.

The computational metalanguage extends the simply typed lambda calculus (example 3.3.2) with two new term constructs related to computational types: $[-]^{\tau \to T\tau}$ and "let $x^\tau = M$ in N". Their typing rules are the following:

$$\frac{\Gamma \vdash M : \tau}{\Gamma \vdash [M] : T\tau} \qquad \frac{\Gamma \vdash M : T\tau \quad \Gamma, x : \tau \vdash N : T\tau'}{\Gamma \vdash \text{let } x^\tau = M \text{ in } N : T\tau'}$$

The term $[M]^{\tau \to T\tau}$ casts the value M of type τ into the computational type with a trivial computational effect. The term "let $x^\tau = M$ in N" composes the computational effect of M and that of N. The above rules fit within the scheme of simply typed formal systems, and the type system can be described by the following typed binding signature.

$$\Pi_M \quad = \quad (\mathbf{Typ}^{\Rightarrow T}(B), \{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}, \mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}, [-]^{\tau\to T\tau}, \mathsf{let}^{T\tau,(\tau,T\tau')\to T\tau'}\})$$

where τ, τ' range over $\mathbf{Typ}^{\Rightarrow T}(B)$.

Semantics of Computational Metalanguage

Before we move to the semantics of the computational metalanguage, we recall the notion of strong monad. A *monad* (T, η, μ) over a category $\mathbb{C}$ consists of an endofunctor $T : \mathbb{C} \to \mathbb{C}$ together with natural transformations $\eta : \mathrm{Id}_{\mathbb{C}} \to T$ and $\mu : T \circ T \to T$ satisfying the following equations for all objects in A:

$$\begin{aligned} \mu_A \circ \eta_{TA} &= \mathrm{id}_{TA} \\ \mu_A \circ T\eta_A &= \mathrm{id}_{TA} \\ \mu_A \circ \mu_{TA} &= T_A \circ T\mu_A \end{aligned}$$

A *strength* of a monad (T, η, μ) is given by a natural transformation $\theta_{A,B} : A \times TB \to T(A \times B)$ satisfying the following equations for all objects A, B, C in $\mathbb{C}$:

$$\begin{aligned} Tr_A \circ \theta_{1,A} &= r_{TA} \\ \theta_{A,B\times C} \circ \mathrm{id}_A \times \theta_{B,C} \circ \alpha_{A,B,TC} &= T\alpha_{A,B,C} \circ \theta_{A\times B,C} \\ \theta_{A,B} \circ \mathrm{id}_A \times \eta_B &= \eta_{A\times B} \\ \mu_{A\times B} \circ T\theta_{A,B} \circ \theta_{A,TB} &= \theta_{A,B} \circ \mathrm{id}_A \times \mu_B \end{aligned}$$

where $r_A : 1 \times A \to A$ and $\alpha_{A,B,C} : (A \times B) \times C \to A \times (B \times C)$ are natural isomorphisms. A *strong monad* over $\mathbb{C}$ is just a pair of a monad and a strength over it. For a morphism $f : A \to TB$ in $\mathbb{C}$, we write $f^{\#} = \mu \circ Tf : TA \to TB$. For details, we refer to [Mac71, Mog91].

Moggi gave a semantics of the computational metalanguage in CCCs with strong monads. This semantics can be understood in terms of our initial algebra semantics. Let $\mathbb{C}$ be a CCC with a strong monad $(T, \eta, \mu, theta)$. First we fix an interpretation of types $F_M : \mathbf{Typ}^{\Rightarrow T}(B) \to \mathbb{C}$ satisfying $F_M(\tau \Rightarrow \tau') = F_M\tau \Rightarrow F_M\tau'$ and $F_M(T\tau) = T(F_M\tau)$. As usual, we specify morphisms corresponding to each operator:

$$\begin{aligned} (u_{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}})_\Gamma(f) &= \lambda(f \circ s_{\Gamma,\tau}) \\ (u_{\mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}})_\Gamma(f,g) &= @ \circ \langle f, g\rangle \\ (u_{[-]^{\tau\to T\tau}})_\Gamma(f) &= \eta_{F_M\tau} \circ f \\ (u_{\mathsf{let}^{T\tau,(\tau,T\tau')\to T\tau'}})_\Gamma(f,g) &= (g \circ s_{\Gamma,\tau})^{\#} \circ \theta_{\Gamma,\tau} \circ \langle \mathrm{id}_{F^*\Gamma,}, f\rangle \end{aligned}$$

The above choice of morphisms induces a Π_M-algebra structure over H^{F_M}. From initiality, we obtain an interpretation of terms $\mathbb{C}[\![-]\!]_{F_M}$ which coincides with the standard interpretation of the computational metalanguage given by Moggi.

Pre-logical Predicates for Computational Metalanguage

We discuss pre-logical predicates over the computational metalanguage. To simplify the discussion, we consider the standard interpretation of the computational metalanguage in $\mathbf{Sets}$ with a strong monad (T, η, μ, θ). Concrete examples of such monads can be found in [Mog91] presented in great detail.

We derive conditions for a predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} F_M$ to be pre-logical for Π_M along $\mathbf{Sets}[\![-]\!]_{F_M}$. As we have discussed in chapter 3, P is pre-logical if the set of invariant terms under P:

$$S^P_{\Pi_M}(\Gamma, \tau) = \{M \mid \Gamma \vdash_{\Pi_M} M : \tau, \mathbf{Sets}[\![M]\!] : P^*\Gamma \to P\tau\}$$

is closed under the syntactic constructs in Π_M. This boils down to the following four conditions corresponding to lambda abstraction, application, $[-]$ and let.

Proposition 4.4.1 *A predicate $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} F_M$ is pre-logical for Π_M along $\mathbf{Sets}[\![-]\!]_{F_M}$ if the following conditions are satisfied.*

1. *For all $\Gamma, x : \tau \vdash_{\Pi_M} M : \tau'$,*

$$\begin{array}{rl} & \forall \rho \in P^*(\Gamma, x : \tau) \,.\, \mathbf{Sets}[\![M]\!]_{F_M}\rho \in P\tau' \\ \Longrightarrow & \forall \rho \in P\Gamma \,.\, \mathbf{Sets}[\![\lambda x : \tau \,.\, M]\!]_{F_M}\rho \in P(\tau \Rightarrow \tau'). \end{array}$$

2. *For all $f \in P(\tau \Rightarrow \tau')$ and $g \in P\tau$, $f(g) \in P\tau'$.*

3. *For all $x \in P\tau$, $\eta_{F_M\tau}(x) \in P(T\tau)$.*

4. *For all $\Gamma, x : \tau \vdash_{\Pi_M} N : T\tau'$,*

$$\begin{array}{rl} & (\forall \rho \in P^*(\Gamma, x : \tau) \,.\, \mathbf{Sets}[\![N]\!]_{F_M}\rho \in P(T\tau')) \\ \Longrightarrow & (\forall \rho \in P^*\Gamma \times PT\tau \,.\, (\mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#} \circ \theta_{\Gamma,\tau}(\rho) \in P(T\tau')). \end{array}$$

□

PROOF The first and second rules are obtained in the same way as in example 3.6.7. We show that condition 3 and 4 are equivalent to

$$M \in S^P_{\Pi_M}(\Gamma, \tau) \implies [M] \in S^P_{\Pi_M}(\Gamma, T\tau) \tag{4.1}$$

and

$$\begin{array}{l} M \in S^P_{\Pi_M}(\Gamma, T\tau) \wedge N \in S^P_{\Pi_M}((\Gamma, x : \tau), T\tau') \\ \implies \text{let } x = M \text{ in } N \in S^P_{\Pi_M}(\Gamma, T\tau'). \end{array} \tag{4.2}$$

respectively.

- ((4.1) $\Longrightarrow$ 3) From $x \in S^P_{\Pi_M}(\{x:\tau\},\tau)$, we have $[x] \in S^P_{\Pi_M}(\{x:\tau\},T\tau)$, that is, $\mathbf{Sets}[\![[x]]\!]_{F_M} = \eta_{F_M\tau} \circ \pi : P^*\{x:\tau\} \to PT\tau$, where $\pi : P^*\{x:\tau\} \to P\tau$ is the isomorphism. Thus we have $\eta_{F_M\tau} : P\tau \to PT\tau$.

  (3 $\Longrightarrow$ (4.1)) Let $M \in S^P_{\Pi_M}(\Gamma,\tau)$, that is, $\mathbf{Sets}[\![M]\!]_{F_M} : P^*\Gamma \to P\tau$. Then $\eta_{F_M\tau} \circ \mathbf{Sets}[\![M]\!]_{F_M} = \mathbf{Sets}[\![[M]]\!]_{F_M} : P^*\Gamma \to PT\tau$.

- ((4.2) $\Longrightarrow$ 4) First we assume the premise of 4, which is equivalent to $N \in S^P_{\Pi_M}((\Gamma,x:\tau),T\tau')$. We have $y \in S^P_{\Pi_M}((\Gamma,y:T\tau),T\tau)$ and $N \in S^P_{\Pi_M}((\Gamma,x:\tau,y:T\tau),T\tau')$ by weakening. From (4.2), we have let $x=y$ in $N \in S^P_{\Pi_M}((\Gamma,y:T\tau),T\tau')$, that is,

  $$\forall \rho \in P^*(\Gamma,y:T\tau)\ .\ \mathbf{Sets}[\![\text{let } x=y \text{ in } N]\!]_{F_M}\rho \in P(T\tau').$$

  An easy calculation shows that this is equivalent to the conclusion of 4.

  (4 $\Longrightarrow$ (4.2)) Let $M \in S^P_{\Pi_M}(\Gamma,T\tau)$ and $N \in S^P_{\Pi_M}((\Gamma,x:\tau),T\tau')$, that is, $\mathbf{Sets}[\![M]\!]_{F_M} : P^*\Gamma \to PT\tau$ and $\mathbf{Sets}[\![N]\!]_{F_M} : P^*(\Gamma,x:\tau) \to PT\tau'$. From 4, we have

  $$\begin{aligned} & \mathbf{Sets}[\![\text{let } x=M \text{ in } N]\!]_{F_M} \\ = \ & (\mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#} \circ \theta_{\Gamma,\tau} \circ \langle \mathrm{id}_{F^*\Gamma}, \mathbf{Sets}[\![M]\!]_{F_M}\rangle : P^*\Gamma \to P(T\tau'). \end{aligned}$$ ∎

Moggi's *computational lambda calculus* [Mog91] is closely related to the computational metalanguage. It provides an adequate framework for modeling *call-by-value* programming languages with side effects, such as partiality, nondeterminism, etc. *Typed partial combinatory algebras* [Mit96] also provide a suitable setting for modeling such languages. In [HS02], Honsell and Sannella discuss pre-logical predicates for typed partial combinatory algebras. In [KP99], Kinoshita and Power demonstrate that the concept of lax logical predicates is extensible to closed Freyd categories. At this moment, our generalisation is not applicable to derive the concept of pre-logical predicates for these settings. In the case of computational lambda calculus, this is because we interpret the calculus in the *Kleisli category* of a category with finite products and a strong monad having Kleisli exponents, and that Kleisli category may have no finite products.

An Example of a Pre-logical Predicate

We give an example of a pre-logical predicate for the computational metalanguage.

In [Lin04, LS05], Stark and Lindley developed *leapfrog method* to extend Tait's strong normalisation proof [Tai67] to Moggi's computational metalanguage. They introduced an operation called $\top\top$ to calculate the predicate at a computational type $T\tau$ from the one at type τ.

In this example, we propose a semantic formulation of the leapfrog method, and show that the predicate P such that $P(T\tau) = P\tau^{\top\top}$ satisfies the conditions of pre-logical predicates related to computational types.

We first recall Lindley and Stark's $\top\top$-operation.

Definition 4.4.2 ([Lin04, LS05]) 1. We define the set of *raw continuations* by the following BNF:

$$K ::= \mathrm{Id} \mid K \circ (x^{\tau}.N)$$

where N ranges over the set of computational metalanguage terms. The notation $x^{\tau}.N$ denotes binding of a variable x^{τ} in N. A judgement for a continuation is a triple (τ, K, τ') denoted by $T\tau \vdash_C K : T\tau'$. We have the following rules to derive well-formed continuations:

$$\frac{}{T\tau \vdash_C \mathrm{Id} : T\tau} \qquad \frac{x : \tau \vdash N : T\tau' \quad T\tau' \vdash_C K : T\tau''}{T\tau \vdash_C K \circ (x^{\tau}.N) : T\tau''}$$

We write $T\tau \vdash_C K$ to mean that there exists a (unique) type $T\tau'$ such that $T\tau \vdash_C K : T\tau'$ is a well-formed continuation.

2. We define *application* $K@M$ of an open term M of type $T\tau$ to a continuation $T\tau \vdash_C K$ as follows:

$$\mathrm{Id}@M = M, \qquad (K \circ (x^{\tau}.N))@M = K@(\text{let } x^{\tau} = M \text{ in } N).$$

3. Given a set ϕ of open terms of type τ, we define a set $\phi^{\top\top}$ of open terms of type $T\tau$ by

$$\begin{aligned} \phi^{\top} &= \{T\tau \vdash_C K \mid \forall M \in \phi \,.\, K@[M] \in SN\} \\ \phi^{\top\top} &= \{M : T\tau \mid \forall K \in \phi^{\top} \,.\, K@M \in SN\}. \end{aligned}$$

where SN is the set of strongly normalising terms. □

We propose a semantic formulation of the $\top\top$-operation. To do so, we need to find semantic counterparts of continuations and applications. We fix a set X and define a continuation to be a function $f : F_M\tau \to TX$. The inductive definition of application above shows that a continuation $\mathrm{Id} \circ (x_n^{\tau_n}.M_n) \circ \cdots \circ (x_1^{\tau_1}.M_1)$ can be regarded as a context as expressed by the following sequence of lets:

$$\text{let } x_1^{\tau_1} = [-] \text{ in let } x_2^{\tau_2} = M_1 \text{ in } \cdots \text{ in } M_n.$$

The essential information in the continuation is its body, which has the following type:

$$x_1 : \tau_1 \vdash \text{let } x_2^{\tau_2} = M_1 \text{ in } \cdots \text{ in } M_n : T\tau.$$

Our formalisation abstracts this information as a function of type $F_M\tau_1 \to TX$. The syntax $K \circ (x^{\tau}.N)$ is then semantically translated to the following construction of a morphism:

$$\frac{f : F_M\tau \to F_M(T\tau') = TF_M\tau' \quad g : F_M\tau' \to TX}{g^{\#} \circ f : F_M\tau \to TX}$$

The definition of continuation relies on a fixed X. This is a compromise of our formulation, while in definition 4.4.2 continuations can take any result type. On the other hand, the $\top\top$-operation in definition 4.4.2 is designed only for the proof of strong normalisation. We relax this by taking a subset $S \subseteq TX$, which plays the same role as SN in the definition of $(-)^{\top}$ and $(-)^{\top\top}$. We define the application of a continuation $f : F_M\tau \to TX$ to $x \in F_M(T\tau)$ by $f^{\#}x$.

Once continuation and application are defined, it is straightforward to define semantic version of the $\top\top$-operation. The following definition summarises the above discussion.

Definition 4.4.3 Let X and $S \subseteq TX$ be sets.

1. A *continuation* is a function $f : F_M\tau \to TX$.

2. We define *application* of a continuation $f : F_M\tau \to TX$ to $x \in F_M(T\tau)$ to be $f^{\#}x$.

3. Let $\phi \subseteq F_M\tau$ be a set. We define a set $\phi^{\top\top} \subseteq F_M(T\tau)$ by

$$\begin{aligned} \phi^{\top} &= \{f : F_M\tau \to TX \mid \forall x \in \phi \,.\, f(x) \in S\} \\ \phi^{\top\top} &= \{x \in F_M(T\tau) \mid \forall f \in \phi^{\top} \,.\, f^{\#}(x) \in S\}. \end{aligned}$$ □

To summarise, we have:

$$\phi^{\top\top} = \{x \in F_M(T\tau) \mid \forall f : F_M\tau \to TX \,.\, (\forall x \in \phi \,.\, f(x) \in S) \implies f^{\#}(x) \in S\}$$

Proposition 4.4.4 *Let $P \subseteq_{\mathrm{Id}_{\mathbf{Sets}}} F_M$ be a predicate satisfying $P(T\tau) = (P\tau)^{\top\top}$. Then P satisfies the conditions of pre-logical predicates related to computational types, i.e. conditions 3 and 4 in proposition 4.4.1.* □

PROOF
- (Condition 3) Let $x \in P\tau$. We show for any continuation $f : F_M\tau \to TX$, $\forall x \in P\tau \,.\, f(x) \in S$ implies $f^{\#} \circ \eta_{F_M\tau}(x) \in S$. This is trivial, as $f^{\#} \circ \eta_{F_M\tau}(x) = f(x)$.

- (Condition 4) Let $\Gamma, x : \tau \vdash_{\Pi_M} N : T\tau'$ be a well-formed term and assume $\mathbf{Sets}[\![N]\!]_{F_M} : P^*(\Gamma, x : \tau) \to P(T\tau')$ (we call this assumption (*)). Let $\rho \in P^*\Gamma$, $v \in P(T\tau)$ and $f : F_M\tau' \to TX$. We assume $f : P\tau' \to S$ and show $f^{\#} \circ (\mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#} \circ \theta_{\Gamma,\tau}(\rho, v) \in S$.

  First we have $f^{\#} \circ (\mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#} = (f^{\#} \circ \mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#}$. We define $f' : F_M\tau \to TX$ by

$$f'(v) = f^{\#} \circ \mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau}(\rho, v)$$

  then from assumption (*) and $P(T\tau') = (P\tau')^{\top\top}$, we have $f' : P\tau \to S$. Next from $v \in P(T\tau) = (P\tau)^{\top\top}$, we have $f'^{\#}v \in S$, which is equal to $f^{\#} \circ (\mathbf{Sets}[\![N]\!]_{F_M} \circ s_{\Gamma,\tau})^{\#} \circ \theta_{\Gamma,\tau}(\rho, v)$, which proves the required result. ■

The $\top\top$-operation can be extended to binary relations over $F_M\tau$ in the following way. Let $S \subseteq TX \times TX$ be a subset. A continuation is a pair (f, g) of functions $f, g : F_M\tau \to TX$. An application of $(x, y) : F_M(T\tau) \times F_M(T\tau)$ to a continuation (f, g) is $(f^{\#}x, g^{\#}y)$. For a binary relation $\phi \subseteq F_M\tau \times F_M\tau$, we define $\phi^{\top\top}$ as follows:

$$\begin{aligned} \phi^{\top} &= \{(f, g) \mid f, g \in F_M\tau \to TX, \forall (x, y) \in \phi \,.\, (fx, gy) \in S\} \\ \phi^{\top\top} &= \{(x, y) \mid x, y \in F_M(T\tau), \forall (f, g) \in \phi^{\top} \,.\, (f^{\#}x, g^{\#}y) \in S\} \end{aligned}$$

Example 4.4.5 We demonstrate a calculation of $\phi^{\top\top}$ for a binary relation $\phi \subseteq F_M\tau \times F_M\tau$ with the following T, X and S.

- We let T be the finite powerset monad:

$$\mathcal{P}_{fin}X = \{P \mid P \text{ is a finite subset of } X\}.$$

  together with the following unit η, multiplication μ and strength θ:

$$\eta(e) = \{e\}, \quad \mu(x) = \bigcup_{e \in x} e, \quad \theta(a, x) = \{(a, e) \mid e \in x\}.$$

- We let X be a one-point set $\{*\}$. We have $TX = \{\emptyset, X\}$.

- We let $S \subseteq TX \times TX$ be $\{(\emptyset, \emptyset), (\emptyset, X), (X, X)\}$. In other words,

$$(x, y) \in S \iff (x = X \implies y = X).$$

We identify a function $f : F_M\tau \to TX$ and a subset (written with the capital letter of the function) $F = \{x \in F_M\tau \mid fx = X\}$. Under this identification, for each $x \in F_M(T\tau) = TF_M\tau = \mathcal{P}_{fin}(F_M\tau)$, we have

$$f^{\#}x = X \iff \bigcup_{e \in x} fe = X \iff \exists e \in x \,.\, e \in F.$$

Therefore $\phi^{\top\top}$ is defined as:

$$\begin{aligned}
\phi^{\top} &= \{(F, G) \in (F_M\tau)^2 \mid \forall (x, y) \in \phi \,.\, x \in F \implies y \in G\} \\
\phi^{\top\top} &= \{(p, q) \in (F_M(T\tau))^2 \mid \\
&\qquad \forall (F, G) \in \phi^{\top} \,.\, (\exists e \in p \,.\, e \in F) \implies (\exists e' \in q \,.\, e' \in G)\}
\end{aligned}$$

that is,

$$\begin{aligned}
\phi^{\top\top} = \{(p, q) \mid \forall F, G \subseteq F_M\tau \,.\, (\forall (x, y) \in \phi \,.\, x \in F \implies y \in G) \implies \\
\forall e \in p \,.\, \exists e' \in q \,.\, e \in F \implies e' \in G\}.
\end{aligned}$$

This is still not intuitive, but interestingly we have the following simpler description of $\phi^{\top\top}$. This pattern appears in the definition of a *pre-bisimulation relation* in concurrency.

Proposition 4.4.6 $\phi^{\top\top} = \{(p,q) \mid \forall a \in p\,.\,\exists b \in q\,.\,(a,b) \in \phi\}$. □

PROOF ($\subseteq$) Let $(p,q) \in \phi^{\top\top}$ and $a \in p$. We show $\exists b \in q\,.\,(a,b) \in \phi$. We supply $\{a\}$ and $\{b \mid (a,b) \in \phi\}$ to F and G in the definition of $(p,q) \in \phi^{\top\top}$. We obtain:

$$\begin{array}{rl} & (\forall (x',y') \in \phi\,.\,x' = a \implies (a,y') \in \phi\}) \\ \implies & (\forall e \in p\,.\,\exists e' \in q\,.\,e = a \implies (a,e') \in \phi\}) \end{array}$$

whose premise part is trivially true. Thus

$$\forall e \in p\,.\,\exists e' \in q\,.\,(e,e') \in \phi$$

holds. By letting $e = a$ in the above, we have $\exists e' \in q\,.\,(a,e') \in \phi$.

($\supseteq$) We take $p, q \in F_M(T\tau)$ such that $\forall a \in p\,.\,\exists b \in q\,.\,(a,b) \in \phi$. Let $F, G \subseteq F_M\tau$, $e \in p$ and assume $\forall (x,y) \in \phi\,.\,x \in F \implies y \in G$ (we call this assumption (*)). We show $\exists e' \in q\,.\,e \in F \implies e' \in G$. Since $e \in p$, there exists $e' \in q$ such that $(e,e') \in \phi$. This e' is an answer, since from (*), we have $e \in F \implies e' \in G$. ■

In [Aba00], Abadi introduced $\top\top$-operation in a slightly different way from our formulation, inspired by [PS98] which is a precursor of the leapfrog method [Lin04, LS05]. At this point it is not known if our semantic formulation of $\top\top$-operation can explain the $\top\top$-operation in the sense of [Aba00].

4.5 An Example from First-Order Logic

In this example we see a characterisation of elementary submodels for a first-order logic in terms of the existence of a binary pre-logical relation. We also have a closer look at Tarski's criterion, which is equivalent to a submodel being an elementary submodel.

For this example, we fix a typed first-order signature $\Sigma = (T_0, O_0)$. The typed binding signature for first-order classical logic over Σ is given as follows:

$$\begin{array}{rcl} \Pi_{\Sigma\text{-fol}} & = & (T_0 \cup \{\boldsymbol{\Omega}\}, \\ & & O_0 \cup \{\mathsf{exists}^{(\tau,\boldsymbol{\Omega})\to\boldsymbol{\Omega}} \mid \tau \in T_0\} \cup \{\mathsf{not}^{\boldsymbol{\Omega}\to\boldsymbol{\Omega}}, \mathsf{or}^{\boldsymbol{\Omega},\boldsymbol{\Omega}\to\boldsymbol{\Omega}}\}) \end{array}$$

If $\Gamma \vdash_{\Pi_{\Sigma\text{-fol}}} M : \tau$ and $\tau \in T_0$, we can assert that M consists of only operators in O_0.

Let $\mathcal{A}$ be a many-sorted Σ-algebra. We give an interpretation of $\Pi_{\Sigma\text{-fol}}$ in $\mathbf{Sets}$, which coincides with the standard interpretation of first-order logic in the model constructed over $\mathcal{A}$. The interpretation of types $M_A : T_0 \cup \{\mathbf{\Omega}\} \to \mathbf{Sets}$ is given by

$$M_A\tau = \begin{cases} A\tau & \tau \in T_0 \\ \mathbf{2} = \{\mathrm{tt}, \mathrm{ff}\} & \tau = \mathbf{\Omega} \end{cases}$$

As usual, we specify morphisms corresponding to each operator.

$$\begin{aligned}
(u_{s^{\tau_1,\cdots,\tau_n\to\tau}})_\Gamma(x_1,\cdots,x_n)(\rho) &= s_{\mathcal{A}}(x_1\rho,\cdots,x_n\rho) \quad (s^{\tau_1,\cdots,\tau_n\to\tau} \in O_0) \\
(u_{\mathsf{exists}^{(n,\mathbf{\Omega})\to\mathbf{\Omega}}})_\Gamma(f)(\rho) &= \begin{cases} \mathrm{tt} & \exists n \in \mathbf{N} \,.\, f(\rho,n) = \mathrm{tt} \\ \mathrm{ff} & \text{otherwise} \end{cases} \\
(u_{\mathsf{not}^{\mathbf{\Omega}\to\mathbf{\Omega}}})_\Gamma(f)(\rho) &= \begin{cases} \mathrm{tt} & f\rho = \mathrm{ff} \\ \mathrm{ff} & \text{otherwise} \end{cases} \\
(u_{\mathsf{or}^{\mathbf{\Omega},\mathbf{\Omega}\to\mathbf{\Omega}}})_\Gamma(f,g)(\rho) &= \begin{cases} \mathrm{tt} & f\rho = \mathrm{tt} \vee g\rho = \mathrm{tt} \\ \mathrm{ff} & \text{otherwise} \end{cases}
\end{aligned}$$

We write $\mathcal{M}_{\mathcal{A}}[\![-]\!]$ for the induced interpretation of terms by initiality of $S_{\Pi_{\Sigma\text{-fol}}}$.

Let $\mathcal{A}$ be a sub-Σ algebra of $\mathcal{B}$. For any T_0-context Γ and any well-formed term $\Gamma \vdash_{\Pi_{\Sigma\text{-fol}}} M : \mathbf{\Omega}$ which does not include the existential quantifier, an easy induction shows that for any $\rho \in A^*\Gamma$, $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho$. When M has an existential quantifier, this does not hold in general because there might be a formula $\exists x^\tau \,.\, M$ which holds in $\mathcal{M}_{\mathcal{B}}$ by a witness that is only included in $\mathcal{B}$. However, if $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho$ still holds for any $\rho \in A^*\Gamma$, we say that $\mathcal{A}$ is an *elementary submodel* of $\mathcal{B}$ (see e.g. [Doe96], 2.8). In other words, first-order logic can not distinguish the two algebras.

Definition 4.5.1 We say that $\mathcal{A}$ is an *elementary submodel* of $\mathcal{B}$ if the following holds for each T_0-context Γ:

$$\forall \Gamma \vdash_{\Pi_{\Sigma\text{-fol}}} M : \mathbf{\Omega} \,.\, \forall \rho \in A^*\Gamma \,.\, \mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho.$$

When is a subalgebra an elementary submodel? One characterisation is given by the following condition which is called *Tarski's criterion* (see e.g. [Doe96], 2.10).

Definition 4.5.2 (Tarski's Criterion) We say *Tarski's criterion* holds for a submodel $\mathcal{A}$ of $\mathcal{B}$ if for each T_0-context Γ and $\tau \in T_0$,

$$\begin{aligned} & \forall \Gamma, x : \tau \vdash_{\Pi_{\Sigma\text{-fol}}} M : \mathbf{\Omega} \,.\, \forall \rho \in A^*\Gamma \,.\, \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{tt} \\ \Longrightarrow \quad & \exists v \in A\tau \,.\, \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho\{x^\tau \mapsto v\} = \mathrm{tt} \end{aligned}$$

holds. □

Tarski's criterion plays an important role in showing that the inclusion relation $I \subseteq_{-\times-} \langle M_A, M_B \rangle$ is a binary pre-logical relation between $\mathcal{M}_{\mathcal{A}}[\![-]\!]$ and $\mathcal{M}_{\mathcal{B}}[\![-]\!]$. In this proof, Tarski's criterion is used when we show that the condition related to existential quantification, which is derived from the goal that I is pre-logical, holds. Thus we could think of Tarski's criterion as an essential condition for I being pre-logical. When I is pre-logical, one can easily show that $\mathcal{A}$ is an elementary submodel of $\mathcal{B}$ (which implies Tarski's criterion). We summarise this discussion with the following proposition.

Proposition 4.5.3 *The following are equivalent:*

1. *$\mathcal{A}$ is an elementary submodel of $\mathcal{B}$.*
2. *Tarski's criterion holds for a submodel $\mathcal{A}$ of $\mathcal{B}$.*
3. *The inclusion relation $R \subseteq_{-\times-} M_A \times M_B$, that is, $R\mathbf{\Omega} = \mathrm{Id}_{\mathbf{\Omega}}$ and $R\tau = \{(x, x) \mid x \in A\tau\}$, is pre-logical for $\Pi_{\Sigma\text{-fol}}$ along $\mathcal{M}_{\mathcal{A}}[\![-]\!] \times \mathcal{M}_{\mathcal{B}}[\![-]\!]$.* □

PROOF

- (1 $\Longrightarrow$ 2) Let Γ be a T_0-context, $\tau \in T_0$, $\Gamma, x : \tau \vdash_{\Pi_{\Sigma\text{-fol}}} M : \mathbf{\Omega}$ be a well-formed term and $\rho \in A^*\Gamma$. Since $\mathcal{A}$ is an elementary submodel of $\mathcal{B}$, $\mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{tt} = \mathcal{M}_{\mathcal{A}}[\![\exists x^\tau \,.\, M]\!]\rho$. Thus there exists an element $v \in A\tau$ such that $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho\{x^\tau \mapsto v\} = \mathrm{tt}$, which implies $\mathcal{M}_{\mathcal{B}}[\![M]\!]\rho\{x^\tau \mapsto v\} = \mathrm{tt}$.

- (2 $\Longrightarrow$ 3) We show that the inclusion relation R is pre-logical. We obtain three subgoals to be proved by expanding the definition of R being pre-logical:

  1. We show that for each first-order operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$ and well-formed terms $\Gamma \vdash_{\Pi_{\Sigma\text{-fol}}} M_i : \tau_i \ (1 \leq i \leq n)$, we have

$$\begin{aligned} & \left(\forall \rho \in M_A^*\Gamma \,.\, \bigwedge_{i=1}^{n} \mathcal{M}_{\mathcal{A}}[\![M_i]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M_i]\!]\rho \right) \Longrightarrow \\ & (\forall \rho \in M_A^*\Gamma \,.\, \mathcal{M}_{\mathcal{A}}[\![o(M_1, \cdots, M_n)]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![o(M_1, \cdots, M_n)]\!]\rho). \end{aligned}$$

This is equivalent to

$$\forall v_1 \in M_A\tau_1, \cdots, v_n \in M_A\tau_n \,.\, o_{\mathcal{A}}(v_1, \cdots, v_n) = o_{\mathcal{B}}(v_1, \cdots, v_n)$$

and clearly this holds since $\mathcal{A}$ is a subalgebra of $\mathcal{B}$, and all the operators behave in the same way in $\mathcal{A}$ and $\mathcal{B}$.

2. Similarly, we can show that for any well-formed terms $\Gamma \vdash_{\Pi_{\Sigma\text{-fol}}} M, N : \mathbf{\Omega}$ and $\rho \in M_A^*\Gamma$, $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho$ and $\mathcal{M}_{\mathcal{A}}[\![N]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![N]\!]\rho$ implies $\mathcal{M}_{\mathcal{A}}[\![\mathsf{not}^{\mathbf{\Omega}\to\mathbf{\Omega}}(M)]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![\mathsf{not}^{\mathbf{\Omega}\to\mathbf{\Omega}}(M)]\!]\rho$ and $\mathcal{M}_{\mathcal{A}}[\![\mathsf{or}^{\mathbf{\Omega},\mathbf{\Omega}\to\mathbf{\Omega}}(M,N)]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![\mathsf{or}^{\mathbf{\Omega},\mathbf{\Omega}\to\mathbf{\Omega}}(M,N)]\!]\rho$.

3. We show that for each well-formed term $\Gamma, x : \tau \vdash_{\Pi_{\Sigma\text{-fol}}} M : \mathbf{\Omega}$ where $\tau \in T_0$, we have

$$\begin{aligned}&(\forall \rho \in A^*(\Gamma, x:\tau) \,.\, \mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho) \implies \\ &(\forall \rho \in A^*\Gamma \,.\, \mathcal{M}_{\mathcal{A}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho).\end{aligned}$$

It is clear that $\mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{ff}$ implies $\mathcal{M}_{\mathcal{A}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{ff}$. Thus we reduce the above to the following goal (we name it $C\exists$):

$$\begin{aligned}&(\forall \rho \in A^*(\Gamma, x:\tau) \,.\, \mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho) \implies \\ &(\forall \rho \in A^*\Gamma \,.\, \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{tt} \implies \mathcal{M}_{\mathcal{A}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{tt}).\end{aligned}$$

Now we assume $\forall \rho \in A^*(\Gamma, x:\tau) \,.\, \mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho$, take $\rho \in A^*\Gamma$ and assume $\mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathrm{tt}$. In order to use Tarski's criterion, we first split the context Γ into two disjoint contexts, a T_0-context Γ_1 and a $\{\mathbf{\Omega}\}$-context Γ_2 such that $\Gamma = \Gamma_1 \cup \Gamma_2$. We then split ρ into two as well; we take $\rho_1 \in A^*\Gamma_1$ and $\rho_2 \in A^*\Gamma_2$ such that $\rho = \rho_1 \cup \rho_2$. We can denote tt, ff by closed terms of type $\mathbf{\Omega}$, say M_{tt} and M_{ff}. Using these two terms, we define a term M' by $M[M_{\rho_2(x_1)}/x_1, \cdots, M_{\rho_2(x_n)}/x_n]$. It is easy to see that $\mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M']\!]\rho_1$ since

$$\begin{aligned}& \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M[M_{\rho_2(x_1)}/x_1, \cdots, M_{\rho_2(x_n)}/x_n]]\!]\rho_1 \\ = \;& \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho_1\{x_1 \mapsto \mathcal{M}_{\mathcal{B}}[\![M_{\rho_2(x_1)}]\!]\rho_1, \cdots, x_n \mapsto \mathcal{M}_{\mathcal{B}}[\![M_{\rho_2(x_n)}]\!]\rho_1\} \\ = \;& \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho_1\{x_1 \mapsto \rho_2(x_1), \cdots, x_n \mapsto \rho_2(x_n)\} \\ = \;& \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho_1 \cup \rho_2 = \mathcal{M}_{\mathcal{B}}[\![\exists x^\tau \,.\, M]\!]\rho\end{aligned}$$

We apply Tarski's criterion and obtain $v \in A\tau$ such that $\mathcal{M}_{\mathcal{B}}[\![M']\!]\rho_1\{x \mapsto v\} = \mathrm{tt}$. We repeat the same argument as above and obtain $\mathcal{M}_{\mathcal{B}}[\![M']\!]\rho_1\{x \mapsto v\} = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho\{x \mapsto v\}$, which is equal to $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho\{x \mapsto v\}$ from the assumption. Therefore $\mathcal{M}_{\mathcal{A}}[\![\exists x^{\tau} \,.\, M]\!]\rho = \mathrm{tt}$.

- (3 $\Longrightarrow$ 1) Let Γ be a T_0-context, $\rho \in A^*\Gamma$ and $\Gamma \vdash_{\Pi\Sigma\text{-fol}} M : \mathbf{\Omega}$ be a well-formed term. Since R is just an inclusion relation, we have $(\rho, \rho) \in R^*\Gamma$. Thus from the basic lemma of pre-logical relations, we have $(\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho, \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho) \in R\mathbf{\Omega}$. Since $R\mathbf{\Omega}$ is the identity relation in $\mathbf{\Omega}$, we have $\mathcal{M}_{\mathcal{A}}[\![M]\!]\rho = \mathcal{M}_{\mathcal{B}}[\![M]\!]\rho$. ∎

4.6 Conclusion

We have seen examples of generalised pre-logical predicates for typed formal systems. The examples we examined benefit the generalisation in several ways.

We applied our generalisation to many-sorted algebra, computational lambda calculus and first-order logic, which are new to the original work [HS02]. In the case of many-sorted algebra, we showed that pre-logical predicates and subalgebras coincide. In first-order logic, we used pre-logical predicates to characterise elementary submodels. In fact this is a special case of observational equivalence, which we will see in the next chapter. Our generalisation supports categorical semantics, which enables us to discuss the relationship with lax logical predicates.

By working in the category $\mathbb{P}_T$ where syntax, semantics and predicates appear as objects and morphisms, we could compare the notion of pre-logical predicates between different languages in a diagrammatical way. The comparison showed that pre-logical predicates for combinatory logic and those for the lambda calculus have a subtle difference.

Chapter 5

Behavioural Equivalence and Indistinguishability

This work is a contribution to the understanding of the relationship between *behavioural equivalence* and the *indistinguishability relation*. These notions arose from the study of data abstraction in the context of algebraic specifications. Behavioural equivalence identifies models which show the same behaviour for any program yielding an observable value. This formalises an intuitive equivalence between two programming environments that show the same behaviour to programmers, regardless of differences in the representation of non-observable data types. The indistinguishability relation is a partial equivalence relation which identifies values in a model that are interchangeable with each other in any program context. This provides an abstract view of the programming environment based on behaviour, rather than denotation.

Their relationship has been studied in a series of papers beginning with [BHW95] by Bidoit, Hennicker and Wirsing. They established the key idea of *factorisability* to relate behavioural equivalence and the indistinguishability relation. Their framework is infinitary first-order logic over many-sorted algebras. Their work has been extended by [HS96] to higher-order logic, where the indistinguishability relation can be finitely axiomatised when the underlying signature is finite.

We examine the situation for simply-typed formal systems and their categorical models. We characterise behavioural equivalence in terms of binary pre-logical rela-

tions, and show factorisability. As an example of this, we see that behavioural satisfaction and satisfaction over the quotient model are equivalent.

5.1 Behavioural Equivalence and Pre-Logical Relations

Behavioural equivalence identifies two models showing the same behaviour in response to all observations. Each observation examines the equality of two terms of observable types, whose values are directly accessible to programmers. This definition of observation formalises an experiment to test for a difference of behaviour of visible data types between two models. Thus, intuitively speaking, if two models are behaviourally equivalent, they provide the same programming environment to programmers, even though they may have different implementation of non-observable data types.

Schoett formulated behavioural equivalence for many-sorted partial algebras. He introduced the notion of *correspondence*, and showed that two models are behaviourally equivalent if and only if there exists a correspondence which is identity relation at observable types.

In the context of the lambda calculus, a restricted notion of behavioural equivalence called closed observational equivalence was studied in [Mit91]. He showed a *representation independence theorem*: if there exists a binary logical relation between two models such that the relation is partially bijective on any observable types, then these two models are closed observationally equivalent. He showed that the converse is also true when the underlying signature has only first-order constants. However this is not satisfactory for two reasons; one is the above restriction to first-order constants, and the other is that in general logical relations do not compose, despite the fact that behavioural equivalence is a transitive relation. Honsell and Sannella [HS02] removed the restriction on the constants by using pre-logical relations instead of logical relations.

In the following, we further generalise this result: two categorical semantics of a simply typed formal system are behaviourally equivalent if and only if there exists an

observational pre-logical relation between them.

5.1.1 Formulation of Behavioural Equivalence

In this section we go back to the traditional theory of many-sorted algebras, and introduce behavioural equivalence as in [HS96]. We first recall the *environmental interpretation* of algebras, which is a slight modification of the standard semantics in section 4.1. In this section, we allow the domains of contexts to be countably infinite. Let $\Sigma = (T_0, O_0)$ be a many-sorted signature. For a T_0-context Γ and T_0-indexed family of sets $\{A^\tau\}_{\tau \in T_0}$, we define the set A^Γ of Γ-*environments* by

$$A^\Gamma = \{\rho \in \mathrm{dom}(\Gamma) \to \bigcup_{\tau \in T_0} A^\tau \mid \forall x \in \mathrm{dom}(\Gamma)\,.\,\rho(x) \in A^{\Gamma(x)}\}.$$

We extend a T_0-indexed family of functions $\{f^\tau : A^\tau \to B^\tau\}_{\tau \in T_0}$ to $f^\Gamma : A^\Gamma \to B^\Gamma$ by $f^\Gamma(\rho)(x) = f^{\Gamma(x)}(\rho(x))$.

Let $\mathcal{A}$ be a many-sorted Σ-algebra. The environmental interpretation $\mathcal{A}[\![-]\!]$ [1] interprets a well-formed term $\Gamma \vdash_\Sigma M : \tau$ under an environment $\rho \in A^\Gamma$ as follows:

$$\begin{aligned} \mathcal{A}[\![x^\tau]\!]\rho &= \rho(x^\tau) \\ \mathcal{A}[\![o(M_1, \cdots, M_n)]\!]\rho &= o_{\mathcal{A}}(\mathcal{A}[\![M_1]\!]\rho, \cdots, \mathcal{A}[\![M_n]\!]\rho) \end{aligned}$$

We fix a many-sorted signature $\Sigma = (T_0, O_0)$, many-sorted Σ-algebras $\mathcal{A}$ and $\mathcal{B}$ and a set $\mathrm{OBS} \subseteq T_0$ of *observable types*.

Definition 5.1.1 ([HS96]) An environment ρ over $\mathcal{A}$ is OBS-surjective if for any type $\sigma \in \mathrm{OBS}$ and $e \in A^\sigma$, there exists a variable $x^\sigma \in \mathrm{dom}(\rho)$ such that $\rho(x^\sigma) = e$.

We say that a Σ-algebra $\mathcal{A}$ is OBS-*countable* if $\bigcup_{\sigma \in \mathrm{OBS}} A^\sigma$ is countable. Note that there exists an OBS-surjective environment over $\mathcal{A}$ if and only if $\mathcal{A}$ is OBS-countable.

□

Definition 5.1.2 ([HS96]) Two Σ-algebras $\mathcal{A}$ and $\mathcal{B}$ are *behaviourally equivalent* with respect to OBS (written by $\mathcal{A} \equiv^0_{\mathrm{OBS}} \mathcal{B}$) if there exists an OBS-context Δ, OBS-

[1] We deliberately overload this notation with the one for the standard interpretation of Σ in $\mathcal{A}$ that is introduced in section 4.1 because the only difference is the way that variables are handled.

surjective environments $\rho \in A^{\Delta}$ and $\rho' \in B^{\Delta}$ such that for any $\sigma \in \mathrm{OBS}$ and well-formed terms $\Delta \vdash M, N : \sigma$, we have $\mathcal{A}[\![M]\!]\rho = \mathcal{A}[\![N]\!]\rho \iff \mathcal{B}[\![M]\!]\rho' = \mathcal{B}[\![N]\!]\rho'$.

□

There are other possibilities for the treatment of free variables, and we leave this point to [HS96].

This formulation restricts the cardinality of the carrier sets of observable types to the cardinality of the set of variables. The surjective environment plays a role to adding constants for each element in the carrier set of an observable type. However, this is not an essential point when formulating behavioural equivalence. We introduce a slightly different, but equivalent formulation of behavioural equivalence from the above.

Definition 5.1.3 We write $\mathcal{A} \equiv_{\mathrm{OBS}} \mathcal{B}$ if there exists an OBS-indexed family of relations $\{R^{\sigma} \subseteq A^{\sigma} \times B^{\sigma}\}_{\sigma \in \mathrm{OBS}}$ such that for any OBS-context Δ whose domain is finite, type $\sigma \in \mathrm{OBS}$, well-formed terms $\Delta \vdash_{\Sigma} M, N : \sigma$, environments $\rho \in A^{\Delta}$ and $\rho' \in B^{\Delta}$,

$$\begin{aligned} & (\forall x \in \mathrm{dom}(\Delta) \,.\, (\rho(x), \rho'(x)) \in R(\Delta(x))) \\ \Longrightarrow \quad & (\mathcal{A}[\![M]\!]\rho = \mathcal{A}[\![N]\!]\rho \iff \mathcal{B}[\![M]\!]\rho' = \mathcal{B}[\![N]\!]\rho'). \end{aligned}$$

□

Proposition 5.1.4 *Suppose $\mathcal{A}$ is OBS-countable. Then $\mathcal{A} \equiv^{0}_{\mathrm{OBS}} \mathcal{B} \iff \mathcal{A} \equiv_{\mathrm{OBS}} \mathcal{B}$.*

□

PROOF ($\Longrightarrow$) Suppose $\mathcal{A} \equiv^{0}_{\mathrm{OBS}} \mathcal{B}$. Let Δ, ρ, ρ' be a context and environments which exist by definition 5.1.2. We define a relation $R^{\sigma} \subseteq A^{\sigma} \times B^{\sigma}$ for each $\sigma \in \mathrm{OBS}$ by

$$R^{\sigma} = \{(\rho(x), \rho'(x)) \mid x \in \mathrm{dom}(\Delta), \Delta(x) = \sigma\}.$$

We show that R^{σ} is total bijective for each $\sigma \in \mathrm{OBS}$. Totality is obvious as ρ and ρ' are OBS-surjective. Suppose $\rho(x) = \rho(y)$, which is equivalent to $\mathcal{A}[\![x]\!]\rho = \mathcal{A}[\![y]\!]\rho$. Since $\mathcal{A} \equiv^{0}_{\mathrm{OBS}} \mathcal{B}$, we have $\mathcal{B}[\![x]\!]\rho' = \mathcal{B}[\![y]\!]\rho'$, which is equivalent to $\rho'(x) = \rho'(y)$. Similarly, $\rho'(x) = \rho'(y)$ implies $\rho(x) = \rho(y)$. Therefore R^{σ} is total bijective.

Below for a type $\sigma \in \mathrm{OBS}$ and an element $a \in A^{\sigma}$, we write x_a for a variable $y \in \mathrm{dom}(\rho)$ such that $\rho(y) = a$. From this definition, $(a, \rho'(x_a)) \in R^{\sigma}$.

Let Δ be an OBS-context whose domain is finite, $\sigma \in \mathrm{OBS}$, $\Delta \vdash_\Sigma M, N : \sigma$, $\eta \in A^\Delta$ and $\eta' \in B^\Delta$. Suppose $(\eta(x), \eta'(x)) \in R^\sigma$ for each $x \in \mathrm{dom}(\Delta)$. Suppose $\mathcal{A}[\![M]\!]\eta = \mathcal{A}[\![N]\!]\eta$. Then we have:

$$\mathcal{A}[\![M]\!]\eta = \mathcal{A}[\![M]\!]\{x \mapsto \mathcal{A}[\![x_{\eta(x)}]\!]\rho\}_{x\in\mathrm{dom}(\eta)} = \mathcal{A}[\![M[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho$$

and $\mathcal{A}[\![N]\!]\eta = \mathcal{A}[\![N[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho$. From $\mathcal{A} \equiv^0_{\mathrm{OBS}} \mathcal{B}$, we have

$$\mathcal{B}[\![M[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho' = \mathcal{B}[\![N[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho'.$$

Then

$$\mathcal{B}[\![M[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho' = \mathcal{B}[\![M]\!]\{x \mapsto \mathcal{B}[\![x_{\eta(x)}]\!]\rho'\}_{x\in\mathrm{dom}(\eta)} = \mathcal{B}[\![M]\!]\eta'.$$

Similarly, we have $\mathcal{B}[\![M[x_{\eta(x)}/x]_{x\in\mathrm{dom}(\eta)}]\!]\rho' = \mathcal{B}[\![N]\!]\eta'$. Therefore we have $\mathcal{B}[\![M]\!]\eta' = \mathcal{B}[\![N]\!]\eta'$. The converse is by symmetry.

($\Longleftarrow$) Suppose $\mathcal{A} \equiv_{\mathrm{OBS}} \mathcal{B}$. Let $R^\sigma \subseteq A^\sigma \times B^\sigma$ be the relation which exists by definition 5.1.3 for each $\sigma \in \mathrm{OBS}$. Since $\mathcal{A}$ and $\mathcal{B}$ are OBS-countable, for each $\sigma \in \mathrm{OBS}$ and $(a, b) \in R\sigma$, we can assign a variable $x^\sigma_{a,b}$. With this assignment, we define an OBS-context Δ and OBS-surjective environments $\rho \in A^\Delta$ and $\rho' \in B^\Delta$ as follows:

$$\Delta(x^\sigma_{a,b}) = \sigma, \quad \rho(x^\sigma_{a,b}) = a, \quad \rho'(x^\sigma_{a,b}) = b.$$

Let $\sigma \in \mathrm{OBS}$ be a type and $\Delta \vdash_\Sigma M, N : \sigma$ be well-formed terms. Then there exists a finite context $\Delta_0 \subseteq \Delta$ such that we can still assert that $\Delta_0 \vdash_\Sigma M, N : \sigma$ is a well-formed term: such Δ_0 can be given by restricting the domain of Δ to $\mathrm{FV}(M) \cup \mathrm{FV}(N)$.

We write η and η' for the restriction of the domain of ρ and ρ' to $\mathrm{dom}(\Delta_0)$ respectively. They satisfy $(\eta(x), \eta'(x)) \in R(\Delta_0(x))$ for each $x \in \mathrm{dom}(\Delta_0)$. Then easy induction shows that $\mathcal{A}[\![M]\!]\eta = \mathcal{A}[\![M]\!]\rho$ and $\mathcal{B}[\![M]\!]\eta' = \mathcal{B}[\![M]\!]\rho'$ (the same for N). Now we assume $\mathcal{A}[\![M]\!]\rho = \mathcal{A}[\![N]\!]\rho$. Then $\mathcal{A}[\![M]\!]\eta = \mathcal{A}[\![N]\!]\eta$, and from $\mathcal{A} \equiv_{\mathrm{OBS}} \mathcal{B}$, we have $\mathcal{B}[\![M]\!]\rho' = \mathcal{B}[\![M]\!]\eta' = \mathcal{B}[\![N]\!]\eta' = \mathcal{B}[\![N]\!]\rho'$. The converse is by symmetry. Therefore $\mathcal{A} \equiv^0_{\mathrm{OBS}} \mathcal{B}$. ∎

5.1.2 A Characterisation of Behavioural Equivalence

We extend the notion of behavioural equivalence to typed formal systems and their categorical models, then characterise it in terms of the existence of an *observational pre-logical relation*.

We introduce an additional notation for the internal logic of fibrations. For $x_1 : F^*\Gamma_1, \cdots, x_n : F^*\Gamma_n$, by $(x_1, \cdots, x_n)_F^{\Gamma_1,\cdots,\Gamma_n}$ we mean the term $s^F_{\Gamma_1,\cdots,\Gamma_n}(x_1, \cdots, x_n)$, where $s^F_{\Gamma_1,\cdots,\Gamma_n}$ is the canonical isomorphism (see definition 3.4.1).

We fix a typed binding signature $\Pi = (T, O)$ and a set $\mathrm{OBS} \subseteq T$ of observable types. Let $\mathbb{C}$ be a regular category and $\mathbb{C}[\![-]\!]_{F_1}$ and $\mathbb{C}[\![-]\!]_{F_2}$ be categorical interpretations for Π in $\mathbb{C}$.

We formulate definition 5.1.3 in the internal logic of $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$.

Definition 5.1.5 We write $\mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$ if there exists an OBS-indexed family of total bijective relations $\{x : F_1\sigma, y : F_1\sigma \vdash R\sigma(x, y)\}_{\sigma\in\mathrm{OBS}}$ such that for any OBS-context Δ, type $\sigma \in \mathrm{OBS}$ and well-formed terms $\Delta \vdash_\Pi M, N : \sigma$, we have

$$\rho : F_1^*\Delta, \rho' : F_2^*\Delta \mid R^*\Delta(\rho, \rho'), \mathbb{C}[\![M]\!]_{F_1}\rho = \mathbb{C}[\![N]\!]_{F_1}\rho \vdash \mathbb{C}[\![M]\!]_{F_2}\rho' = \mathbb{C}[\![N]\!]_{F_2}\rho'$$
$$\rho : F_1^*\Delta, \rho' : F_2^*\Delta \mid R^*\Delta(\rho, \rho'), \mathbb{C}[\![M]\!]_{F_2}\rho' = \mathbb{C}[\![N]\!]_{F_2}\rho' \vdash \mathbb{C}[\![M]\!]_{F_1}\rho = \mathbb{C}[\![N]\!]_{F_1}\rho$$

□

Definition 5.1.6 An *observational pre-logical relation* for Π between $\mathbb{C}[\![-]\!]_{F_1}$ and $\mathbb{C}[\![-]\!]_{F_2}$ with respect to OBS is a pre-logical relation $R \subseteq_{-\times-} \langle F_1, F_2\rangle$ for Π along $\mathbb{C}[\![-]\!]_{F_1} \times \mathbb{C}[\![-]\!]_{F_2}$ such that $R\sigma$ is a total bijective relation for each type $\sigma \in \mathrm{OBS}$. The existence of such an observational pre-logical relation will be written $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. □

Theorem 5.1.7 $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$ *implies* $\mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. □

PROOF Let R be the observational pre-logical relation which exists by $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. By definition, $R\sigma$ is total bijective for each $\sigma \in \mathrm{OBS}$. Let Δ be an OBS-context, $\sigma \in \mathrm{OBS}$, $\Delta \vdash M, N : \sigma$ be well-formed terms, $\rho : F_1^*\Delta, \rho' : F_2^*\Delta$ and assume $(\rho, \rho') \in R^*\Delta$. From the basic lemma of pre-logical predicates, we have $R\sigma(\mathbb{C}[\![M]\!]_{F_1}\rho, \mathbb{C}[\![M]\!]_{F_2}\rho')$ and $R\sigma(\mathbb{C}[\![N]\!]_{F_1}\rho, \mathbb{C}[\![N]\!]_{F_2}\rho')$. Since $R\sigma$ is total bijective,

$\mathbb{C}[\![M]\!]_{F_1}\rho = \mathbb{C}[\![N]\!]_{F_1}\rho$ if and only if $\mathbb{C}[\![M]\!]_{F_2}\rho' = \mathbb{C}[\![N]\!]_{F_2}\rho'$. Hence $\mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$.

To show the converse, we further assume that the subobject fibration $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has fibred small coproducts, and $\mathbb{C}[\![-]\!]_{F_1}$ and $\mathbb{C}[\![-]\!]_{F_2}$ both satisfy the semantic substitution lemma. The assumption on $\mathbb{C}$ implies that $p_{\mathbb{C}}$ supports $\mathcal{L}_{\wedge}, \mathcal{L}_{=}, \mathcal{L}_{\exists}, \mathcal{L}_{\vee\omega}$. This comes from the requirement in theorem 3.8.2 which will be used to construct an actual pre-logical predicate from behavioural equivalence.

Theorem 5.1.8 $\mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$ *implies* $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. □

PROOF Let $x : F_1\sigma, y : F_2\sigma \vdash R\sigma(x,y)$ be the total bijective relation which exists by $\mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. By applying theorem 3.8.2, we obtain the least binary pre-logical relation $I_R \subseteq_{-\times-} \langle F_1, F_2\rangle$ for Π along $\mathbb{C}[\![-]\!]_{F_1} \times \mathbb{C}[\![-]\!]_{F_2}$ such that $x : F_1\sigma, y : F_2\sigma \mid R\sigma(x,y) \vdash I_R\sigma(x,y)$ holds for each $\sigma \in \mathrm{OBS}$. Totality of I_R comes from this property. So we show bijectivity of I_R. Let $x_1 : F_1\sigma, x_2 : F_1\sigma, y_1 : F_2\sigma, y_2 : F_2\sigma$ and assume $I_R\sigma(x_1,y_1)$ and $I_R\sigma(x_2,y_2)$. By definition of I_R, for each $i = 1,2$, there exists an OBS-context Δ_i, well-formed term $\Delta_i \vdash_{\Pi} M_i : \sigma$, $\rho_i : F_1^*\Delta$ and $\rho_i' : F_2^*\Delta$ such that $R^*\Delta_i(\rho_i,\rho_i'), x_i = \mathbb{C}[\![M_i]\!]_{F_1}\rho_i$ and $y_i = \mathbb{C}[\![M_i]\!]_{F_2}\rho_i'$ hold. From the definition of behavioural equivalence, $x_1 = x_2$ if and only if $y_1 = y_2$. Thus $I_R\sigma$ is bijective.

We summarise the above theorems.

Corollary 5.1.9 *Let $\mathbb{C}$ be a regular category such that $p_{\mathbb{C}}$ has fibred small coproducts, Π be a typed binding signature and $\mathbb{C}[\![-]\!]_{F_1}, \mathbb{C}[\![-]\!]_{F_2}$ be categorical interpretations of Π satisfying the semantic substitution lemma. Then we have the following equivalence:*

$$\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2} \iff \mathbb{C}[\![-]\!]_{F_1} \equiv_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}.$$

We postpone an application of this characterisation theorem to section 5.4.

5.2 Indistinguishability Relations

We introduce an equivalence of values called indistinguishability based on their behaviour rather than their denotation. We regard two values in a model as "behaviourally"

indistinguishable if they are interchangeable in any program. This is shown by performing a set of experiments; we fit one value into a program yielding a visible result, and see whether any difference is detected when we exchange that value with the other. If two values pass the above experiment over all possible programs, then we say that they are indistinguishable. This identification of values is more suitable to provide an abstract aspect of specifications. We express this idea in the internal logic of fibrations.

In this section we fix a typed binding signature $\Pi = (T, O)$, a category $\mathbb{C}$ with finite limits and a categorical interpretation $\mathbb{C}[\![-]\!]_F$ of Π. We assume that $p_{\mathbb{C}} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ has fibred small products, fibred small coproducts, simple products, simple coproducts and quotient types. Thus $p_{\mathbb{C}}$ supports $\mathcal{L}_{\wedge\omega}, \mathcal{L}_{=}, \mathcal{L}_{\forall}, \mathcal{L}_{\vee\omega}, \mathcal{L}_{\exists}, \mathcal{L}_{-/-}$.

Definition 5.2.1 We define the OBS-*reachability predicate* $R_0 \subseteq_{\mathrm{Id}_{\mathbb{C}}} F$ by the least pre-logical extension of $\top \circ F \circ \iota \subseteq_{\mathrm{Id}_{\mathbb{C}}} F \circ \iota$ by theorem 3.8.2, where $\iota : \mathrm{OBS} \hookrightarrow T$ is the inclusion and $\top$ is the truth functor (see proposition 2.4.4). Explicitly, for each type $\tau \in T$, the predicate $x : F\tau \vdash_{\Pi} (R_0\tau)x$ is defined as follows:

$$(R_0\tau)x = \bigvee_{\Delta \in |\mathbb{V}_{\mathrm{OBS}}|, \Delta \vdash_{\Pi} M : \tau} \exists \rho : F^*\Delta \,.\, x = \mathbb{C}[\![M]\!]_F \rho$$

Definition 5.2.2 An *observation* for a type $\tau \in T$ is a well-formed term $\Delta, x : \tau \vdash C : \sigma$ where $\sigma \in \mathrm{OBS}$ is a type and Δ is an OBS-context such that $x \notin \mathrm{dom}(\Delta)$. □

Definition 5.2.3 For a categorical interpretation $\mathbb{C}[\![-]\!]_F$ of Π satisfying the semantic substitution lemma, we define *indistinguishability relation* $x : F\tau, y : F\tau \vdash \approx_F^{\tau}(x, y)$ for each type $\tau \in T$ by the following formula:

$$(R_0\tau)x \wedge (R_0\tau)y \wedge \bigwedge_{\Delta, x:\tau \vdash C:\sigma} \forall \rho : F^*\Delta \,.\, \mathbb{C}[\![C]\!]_F(\rho, x)_F^{\Delta, \{x:\tau\}} = \mathbb{C}[\![C]\!]_F(\rho, y)_F^{\Delta, \{x:\tau\}}.$$

This specifies a predicate $\approx_F \subseteq_{-\times-} \langle F, F \rangle$. □

The indistinguishability relation is defined on each categorical interpretation $\mathbb{C}[\![-]\!]_F$ of Π. To show this dependency, we write the suffix of the interpretation of types used in the categorical interpretation of Π.

Theorem 5.2.4 *The indistinguishability relation* $\approx_F \subseteq_{-\times-} \langle F, F \rangle$ *is pre-logical for* Π *along* $\mathbb{C}[\![-]\!]_F \times \mathbb{C}[\![-]\!]_F$. *Furthermore,* $\approx_F^{\tau}$ *is a PER for each type* $\tau \in T$. □

PROOF Let $\Gamma \vdash M : \tau$ be a well-formed term where $\Gamma = \{w_1 : \tau_1, \cdots, w_n : \tau_n\}$ (without loss of generality we assume that $w_1 \leq_{\mathbf{v}} \cdots \leq_{\mathbf{v}} w_n$), $x_i : F\tau_i, y_i : F\tau_i$ ($1 \leq i \leq n$) and assume $x_i \approx_F^{\tau_i} y_i$ for $1 \leq i \leq n$. We show $\mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n) \approx_F^{\tau} \mathbb{C}[\![M]\!]_F(y_1, \cdots, y_n)$.

We first show $(R_0\sigma)\ \mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n)$. For each $1 \leq i \leq n$, $x_i \approx_F^{\tau_i} y_i$ implies $(R_0\tau_i)x_i$, that is, there exists a well-formed term $\Delta_i \vdash_{\Pi} M_i : \tau_i$ and $\rho_i : F^*\Delta_i$ such that $x_i = \mathbb{C}[\![M_i]\!]_F\rho_i$. Since $\mathbb{C}[\![-]\!]_F$ satisfies the semantic substitution lemma, we have

$$\begin{aligned} & \mathbb{C}[\![M]\!]_F(\mathbb{C}[\![M_1]\!]_F\rho_1, \cdots, \mathbb{C}[\![M_1]\!]_F\rho_n) \\ = \ & \mathbb{C}[\![M[M_1/w_1, \cdots, M_n/w_n]]\!]_F(\rho_1, \cdots, \rho_n)_F^{\Delta_1, \cdots, \Delta_n} \end{aligned}$$

Therefore we have $(R_0\sigma)\ \mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n)$. We can similarly show that $(R_0\sigma)\ \mathbb{C}[\![M]\!]_F(y_1, \cdots, y_n)$ holds.

Let $\Delta, x : \tau \vdash C : \sigma$ be an observation and $\rho : F^*\Delta$. We show

$$\mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n))_F^{\Delta, \{x:\tau\}} = \mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(y_1, \cdots, y_n))_F^{\Delta, \{x:\tau\}}.$$

First we have

$$\begin{aligned} & \mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n))_F^{\Delta, \{x:\tau\}} \\ = \ & \mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M[M_2/w_2, \cdots, M_n/w_n]]\!]_F(x_1, \rho_2, \cdots, \rho_n)_F^{\{x_1:\tau_1\}, \Delta_2, \cdots, \Delta_n})_F^{\Delta, \{x:\tau\}} \\ = \ & \mathbb{C}[\![C[M[M_2/w_2, \cdots, M_n/w_n]/x]]\!]_F(\rho, x_1, \rho_2, \cdots, \rho_n)_F^{\Delta, \{x_1:\tau_1\}, \Delta_2, \cdots, \Delta_n} \end{aligned}$$

Since $x_1 \approx_F^{\tau_1} y_1$, the above is equal to

$$\mathbb{C}[\![C[M[M_2/w_2, \cdots, M_n/w_n]/x]]\!]_F(\rho, y_1, \rho_2, \cdots, \rho_n)_F^{\Delta, \{x_1:\tau_1\}, \Delta_2, \cdots, \Delta_n}$$

and by the same discussion

$$\begin{aligned} & \mathbb{C}[\![C[M[M_2/w_2, \cdots, M_n/w_n]/x]]\!]_F(\rho, y_1)_F^{\Delta, \{x:\tau\}} \\ = \ & \mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(y_1, x_2, \cdots, x_n))_F^{\Delta, \{x:\tau\}}. \end{aligned}$$

By applying the same argument to $x_2, \cdots, x_n$, we can swap all x_i to y_i. Therefore we have

$$\mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(x_1, \cdots, x_n))_F^{\Delta, \{x:\tau\}} = \mathbb{C}[\![C]\!]_F(\rho, \mathbb{C}[\![M]\!]_F(y_1, \cdots, y_n))_F^{\Delta, \{x:\tau\}}.$$

From the definition, it is easy to see that $\approx_F^{\tau}$ is a PER for each $\tau \in T$. ∎

Proposition 5.2.5 *For each type $\sigma \in \mathrm{OBS}$, we have*

$$x : F\sigma, y : F\sigma \mid x \approx_F^\sigma y \vdash x = y \tag{5.1}$$

$$x : F\sigma, y : F\sigma \mid x = y \vdash x \approx_F^\sigma y. \tag{5.2}$$

□

PROOF Let $\sigma \in \mathrm{OBS}$ be a type and $x : F\sigma$ and $y : F\sigma$.

(5.1) Assume $x \approx_F^\sigma y$. From the definition of $x \approx_F^\sigma y$, for each observation $\Delta, z : \sigma \vdash C : \sigma$, we have $\forall \rho : F^*\Delta \,.\, \mathbb{C}[\![C]\!]_F(\rho, x)_F^{\Delta,\{x:\tau\}} = \mathbb{C}[\![C]\!]_F(\rho, y)_F^{\Delta,\{x:\tau\}}$. In particular, this holds when $\Delta = \emptyset$ and $C = z$. Thus we have $\mathbb{C}[\![z]\!]_F(x) = x = y = \mathbb{C}[\![z]\!]_F(y)$.

(5.2) It is sufficient to show that $x \approx_F^\sigma x$ for any $x : F\sigma$. This holds since any value $x : F\sigma$ is OBS-reachable, that is, $x = \mathbb{C}[\![z]\!]_F(x)$. ■

Corollary 5.2.6 *$r(\approx_F) \subseteq_{\mathrm{Id}_\mathbb{C}} F$ is a pre-logical predicate for Π along $\mathbb{C}[\![-]\!]_F$ (see definition 2.5.3 for the notation $r(\approx_F)$).* □

PROOF We show that for any well-formed term $\Gamma \vdash_\Pi M : \tau$ and $\rho : F^*\Gamma$, $(r(\approx_F)^*\Gamma)\rho$ implies $r(\approx_F^\tau)\mathbb{C}[\![M]\!]_F\rho$. First $(r(\approx_F)^*\Gamma)\rho$ implies that $\approx_F^*\Gamma(\rho, \rho)$. Since $\approx_F$ is pre-logical, we have $\mathbb{C}[\![M]\!]_F\rho \approx_F^\tau \mathbb{C}[\![M]\!]_F\rho$, that is, $r(\approx_F^\tau)\mathbb{C}[\![M]\!]_F\rho$. ■

Definition 5.2.7 The *quotient model* $\mathbb{C}[\![-]\!]_{[\approx_F]}$ of $\mathbb{C}[\![-]\!]_F$ by the indistinguishability relation $\approx_F \subseteq_{-\times-} \langle F, F\rangle$ consists of:

- the interpretation of types $[\approx_F]$ defined by

$$[\approx_F]\tau = [\approx_F^\tau]$$

  for each type $\tau \in T$, and

- the interpretation of terms $\mathbb{C}[\![-]\!]_{[\approx_F]}$ defined by

$$(\mathbb{C}[\![M]\!]_{[\approx_F]})_{\Gamma,\tau} = [\mathbb{C}[\![M]\!]_F] \circ k_\Gamma : [\approx_F]^*\Gamma \to [\approx_F]\tau \quad (\Gamma \vdash_\Pi M : \tau)$$

  where $k_\Gamma : [\approx_F]^*\Gamma \to [\approx_F^*\Gamma]$ is an isomorphism (see lemma 2.5.7). □

Proposition 5.2.8 $\mathbb{C}[\![-]\!]_F \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{[\approx_F]}$. □

PROOF Let $\tau \in T$ be a type. First we have a subset projection $m_{r(\approx_F^\tau)} : |\approx_F^\tau| \rightarrowtail F\tau$ and a canonical quotient map $c_{r(\approx_F^\tau)} : |\approx_F^\tau| \to [\approx_F^\tau]$ in $\mathbb{C}$. Thus $\langle m_{r(\approx_F^\tau)}, c_{r(\approx_F^\tau)} \rangle : |\approx_F^\tau| \rightarrowtail F\tau \times [\approx_F^\tau]$ is a monomorphism and specifies a subobject for each type $\tau \in T$. Therefore we have a predicate $|\approx_F| \subseteq_{-\times-} \langle F, [\approx_F] \rangle$. We next show that this is pre-logical for Π along $\mathbb{C}[\![-]\!]_F \times \mathbb{C}[\![-]\!]_{[\approx_F]}$.

From corollary 5.2.6 and theorem 3.6.8, we have a morphism $u : S_\Pi \to H^{r(\approx_F)}$ in $\mathbb{P}_T$ such that $H^{\pi_{\mathrm{Id}_\mathbb{C}}} \circ u = \mathbb{C}[\![-]\!]_F$. That is, for a well-formed term $\Gamma \vdash_\Pi M : \tau$, we have the following morphism in $\mathbf{Sub}(\mathbb{C})$:

$$u_{(\Gamma,\tau)}(M) : r(\approx_F)^*\Gamma \to r(\approx_F)\tau$$

such that $p_\mathbb{C}(u_{(\Gamma,\tau)}(M)) = \mathbb{C}[\![M]\!]_F$. Below we fix a well-formed term $\Gamma \vdash_\Pi M : \tau$.

The subset type functor $\{-\} : \mathbf{Sub}(\mathbb{C}) \to \mathbb{C}$ is a right adjoint to $\top$ (definition 2.4.22). Hence it preserves limits, and has the canonical isomorphism in $\mathbb{C}$:

$$h_\Gamma : |\approx_F|^*\Gamma = \{r(\approx_F)\}^*\Gamma \to \{r(\approx_F)^*\Gamma\}.$$

These morphisms make the following diagram commute in $\mathbb{C}$:

$$\begin{array}{ccccc}
F^*\Gamma & = & F^*\Gamma & \xrightarrow{\mathbb{C}[\![M]\!]_F} & F\tau \\
{\scriptstyle \prod m}\uparrow & & \uparrow{\scriptstyle m_{r(\approx_F)^*\Gamma}} & & \uparrow{\scriptstyle m_{r(\approx_F)\tau}} \\
|\approx_F|^*\Gamma & \xrightarrow{h_\Gamma} & \{r(\approx_F)^*\Gamma\} & \xrightarrow{\{u_{(\Gamma,\tau)}(M)\}} & |\approx_F|\tau \\
{\scriptstyle \prod c}\downarrow & & \downarrow{\scriptstyle c_{r(\approx_F)^*\Gamma}} & & \downarrow{\scriptstyle c_{r(\approx_F)\tau}} \\
[\approx_F]^*\Gamma & \xrightarrow[k_\Gamma]{} & [\approx_F^*\Gamma] & \xrightarrow[[\mathbb{C}[\![M]\!]_F]]{} & [\approx_F]\tau
\end{array}$$

where $\prod m$ and $\prod c$ are shorthand for the morphisms $\prod_{i=1}^{\#\mathrm{dom}(\Gamma)} m_{r(\approx_F)(\gamma^{-1}i)}$ and $\prod_{i=1}^{\#\mathrm{dom}(\Gamma)} c_{r(\approx_F)(\gamma^{-1}i)}$ respectively. This implies that there exists a morphism in $\mathbf{Sub}(\mathbb{C})$ from $|\approx_F|^*\Gamma$ to $|\approx_F|\tau$ above $\mathbb{C}[\![M]\!]_F \times \mathbb{C}[\![M]\!]_{[\approx_F]}$ for any well-formed term M. Thus $|\approx_F|$ is pre-logical. ∎

5.3 Factorisability

We have seen two approaches to obtain abstract models of specifications. The first involves behavioural equivalence, whereby the models of a specification are taken to be

all those models that are behaviourally equivalent to models which satisfy the axioms of the specification. The second involves indistinguishability, where equality in axioms is taken to refer to indistinguishability rather than identity. Both of them naturally arise from the motivation of reasoning about specification from a behavioural point of view.

We are interested in considering their relationship. The key idea is the notion of *factorisability* [BHW95]. We go back to the traditional theory of many-sorted algebras. Let Σ be a many-sorted signature. We write $\mathbf{Mod}(\Sigma)$ for the collection of Σ-algebras. We suppose that there exists an equivalence relation $\equiv$ over $\mathbf{Mod}(\Sigma)$ and a Σ-congruence $\approx_{\mathcal{A}}$ for each $\mathcal{A} \in \mathbf{Mod}(\Sigma)$. We say that $\equiv$ is *left-factorisable* by $\approx$ if the existence of a Σ-isomorphism $\mathcal{A}/\approx_{\mathcal{A}} \cong \mathcal{B}/\approx_{\mathcal{B}}$ implies $\mathcal{A} \equiv \mathcal{B}$. The converse of left-factorisability is called *right-factorisablity*. When $\equiv$ is both left and right factorisable by $\approx$, we simply say that $\equiv$ is factorisable by $\approx$.

In example 5.4 of [BHW95], Bidoit, Hennicker and Wirsing show that behavioural equivalence $\equiv_{\mathrm{OBS}}$ is factorisable by indistinguishability. In this section, we show this factorisability result in the context of simply typed formal systems and their categorical models. To do so, we replace the semantics of the traditional many-sorted algebras by categorical semantics and use internal logic to establish factorisability. We fix a typed binding signature $\Pi = (T, O)$, a set $\mathrm{OBS} \subseteq T$ of observable types, a category $\mathbb{C}$ with finite limits such that $p_{\mathbb{C}}$ supports $\mathcal{L}_{\wedge\omega}, \mathcal{L}_{=}, \mathcal{L}_{\forall}, \mathcal{L}_{\vee\omega}, \mathcal{L}_{\exists}, \mathcal{L}_{-/-}$ and categorical interpretations $\mathbb{C}[\![-]\!]_{F_1}$ and $\mathbb{C}[\![-]\!]_{F_2}$ for Π satisfying the semantic substitution lemma.

First we prove left-factorisabilty. This is rather easy to prove, thanks to the characterisation of behavioural equivalence and composability of pre-logical relations. In the following formulation of left-factorisability, we replace isomorphism with a pre-logical total bijective relation.

Theorem 5.3.1 *Suppose there exists a pre-logical total bijective relation* $R \subseteq_{-\times-} \langle [\approx_{F_1}], [\approx_{F_2}] \rangle$ *for* Π *along* $\mathbb{C}[\![-]\!]_{[\approx_{F_1}]} \times \mathbb{C}[\![-]\!]_{[\approx_{F_2}]}$. *Then* $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. □

PROOF The existence of the above R implies that $\mathbb{C}[\![-]\!]_{[\approx_{F_1}]} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{[\approx_{F_2}]}$. Thus from proposition 5.2.8 we have:

$$\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{[\approx_{F_1}]} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{[\approx_{F_2}]} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}.$$

Theorem 5.3.2 *Suppose* $\mathbb{C}[\![-]\!]_{F_1} \sim_{\mathrm{OBS}} \mathbb{C}[\![-]\!]_{F_2}$. *Then there is a pre-logical total bijective relation* $R \subseteq_{-\times-} \langle [\approx_{F_1}], [\approx_{F_2}] \rangle$ *for* Π *along* $\mathbb{C}[\![-]\!]_{[\approx_{F_1}]} \times \mathbb{C}[\![-]\!]_{[\approx_{F_2}]}$. □

PROOF The proof takes few steps. First we define the following program equivalence between terms having observable inputs:

Definition 5.3.3 Let Γ, Δ be OBS-contexts, $\sigma \in \mathrm{OBS}$, $\Gamma + \Delta \vdash_{\Pi} M, N : \sigma$ be well-formed terms and $\rho : F_1^* \Delta$. We write $\mathbb{C}[\![-]\!]_{F_1} \models \Gamma + \Delta \vdash_{\rho} M \sim N : \sigma$ if for any $\eta : F_1^* \Gamma$, we have $\mathbb{C}[\![M]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta} = \mathbb{C}[\![N]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta}$. □

From the assumption and theorem 5.1.8, there exists a pre-logical relation R between $\mathcal{A}$ and $\mathcal{B}$ such that $R\sigma$ is a total bijective relation for each $\sigma \in \mathrm{OBS}$. Then we show a couple of lemmas;

Lemma 5.3.4 *Let* Δ *be an* OBS*-context and assume* $R^*\Delta(\rho, \rho')$. *Then for any* OBS*-context* Γ, $\sigma \in \mathrm{OBS}$ *and well-formed terms* $\Gamma + \Delta \vdash_{\Pi} M, N : \sigma$, *we have*

$$\mathbb{C}[\![-]\!]_{F_1} \models \Gamma + \Delta \vdash_{\rho} M \sim N : \sigma \iff \mathbb{C}[\![-]\!]_{F_2} \models \Gamma + \Delta \vdash_{\rho'} M \sim N : \sigma.$$

PROOF We prove $\Longrightarrow$; the converse is by symmetry. Let $\eta' : F_2^* \Gamma$. We have a unique $\eta : F_1^* \Gamma$ such that $R^*\Gamma(\eta, \eta')$ since $R\sigma$ is total bijective for each type $\sigma \in \mathrm{OBS}$. From the basic lemma we have

$$R\sigma(\mathbb{C}[\![M]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta}, \mathbb{C}[\![M]\!]_{F_2}(\eta', \rho')_{F_2}^{\Gamma,\Delta})$$

$$R\sigma(\mathbb{C}[\![N]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta}, \mathbb{C}[\![N]\!]_{F_2}(\eta', \rho')_{F_2}^{\Gamma,\Delta}).$$

Now we assumed that $\mathbb{C}[\![M]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta} = \mathbb{C}[\![N]\!]_{F_1}(\eta, \rho)_{F_1}^{\Gamma,\Delta}$. Since $R\sigma$ is total bijective, we have $\mathbb{C}[\![M]\!]_{F_2}(\eta', \rho')_{F_2}^{\Gamma,\Delta} = \mathbb{C}[\![N]\!]_{F_2}(\eta', \rho')_{F_2}^{\Gamma,\Delta}$. ■

Lemma 5.3.5 *Let* Δ *be an* OBS*-context,* $\tau \in T$ *and* $\Delta \vdash_{\Pi} X, Y : \tau$ *be well-formed terms.*

1. *Let* $\rho : F_1^* \Delta$. *Then* $\mathbb{C}[\![X]\!]_{F_1} \rho \approx_{F_1}^{\tau} \mathbb{C}[\![Y]\!]_{F_1} \rho$ *if and only if for any* OBS*-context* Γ, $\sigma \in \mathrm{OBS}$ *and observation* $\Gamma, x : \tau \vdash_{\Pi} M : \sigma$ *for type* τ, *we have* $\mathbb{C}[\![-]\!]_{F_1} \models \Gamma + \Delta \vdash_{\rho} M[X/x] \sim M[Y/x] : \sigma$.

2. *Let* $\rho : F_1^*\Delta, \rho' : F_2^*\Delta$ *and assume* $R^*\Delta(\rho,\rho')$. *Then* $\mathbb{C}[\![X]\!]_{F_1}\rho \approx^{\tau}_{F_1} \mathbb{C}[\![Y]\!]_{F_1}\rho$ *if and only if* $\mathbb{C}[\![X]\!]_{F_2}\rho' \approx^{\tau}_{F_2} \mathbb{C}[\![Y]\!]_{F_2}\rho'$. □

PROOF 1. From the semantic substitution lemma, for any context Γ and $\eta : F_1^*\Gamma$, we have

$$\begin{aligned}\mathbb{C}[\![M[X/x]]\!]_{F_1}(\eta,\rho)^{\Gamma,\Delta}_{F_1} &= \mathbb{C}[\![M]\!]_{F_1}(\eta,\mathbb{C}[\![X]\!]_{F_1}\rho)^{\Gamma,\{x:\tau\}}_{F_1}\\ \mathbb{C}[\![M[Y/x]]\!]_{F_1}(\eta,\rho)^{\Gamma,\Delta}_{F_1} &= \mathbb{C}[\![M]\!]_{F_1}(\eta,\mathbb{C}[\![Y]\!]_{F_2}\rho)^{\Gamma,\{x:\tau\}}_{F_2}\end{aligned}$$

Then for any $\eta : F_1^*\Gamma$, $\mathbb{C}[\![M[X/x]]\!]_{F_1}(\eta,\rho)^{\Gamma,\Delta}_{F_1} = \mathbb{C}[\![M[Y/x]]\!](\eta,\rho)^{\Gamma,\Delta}_{F_1}$ if and only if $\mathbb{C}[\![M]\!](\eta,\mathbb{C}[\![X]\!]_{F_1}\rho)^{\Gamma,\{x:\tau\}}_{F_1} = \mathbb{C}[\![M]\!](\eta,\mathbb{C}[\![Y]\!]\rho)^{\Gamma,\{x:\tau\}}_{F_1}$, which implies the lemma.

2. We use lemma 5.3.4 and the above lemma.

$$\begin{aligned}& \mathbb{C}[\![X]\!]_{F_1}\rho \approx^{\tau}_{F_1} \mathbb{C}[\![Y]\!]_{F_1}\rho\\ \iff\ & \forall\Gamma : \text{OBS-context}, \Gamma, x:\tau \vdash_{\Pi} M : \sigma\ .\\ & \quad \mathbb{C}[\![-]\!]_{F_1} \models \Gamma + \Delta \vdash_{\rho} M[X/x] \sim M[Y/x] : \sigma\\ \iff\ & \forall\Gamma : \text{OBS-context}, \Gamma, x:\tau \vdash_{\Pi} M : \sigma\ .\\ & \quad \mathbb{C}[\![-]\!]_{F_2} \models \Gamma + \Delta \vdash_{\rho'} M[X/x] \sim M[Y/x] : \sigma\\ \iff\ & \mathbb{C}[\![X]\!]_{F_2}\rho' \approx^{\tau}_{F_2} \mathbb{C}[\![Y]\!]_{F_1}\rho'.\end{aligned}$$ ■

We return to the proof of right factorisability. We define a relation $x : [\approx^{\tau}_{F_1}], y : [\approx^{\tau}_{F_2}] \vdash h^{\tau}(x,y)$ as follows:

$$h^{\tau}(x,y) = \bigvee_{\Delta\in|\mathbb{V}_{\text{OBS}}|,\Delta\vdash_{\Pi}M:\tau} \exists\rho : F_1^*\Delta, \rho' : F_2^*\Delta\ .\ R^*\Delta(\rho,\rho') \wedge x = [\mathbb{C}[\![M]\!]_{F_1}\rho] \wedge y = [\mathbb{C}[\![M]\!]_{F_2}\rho']$$

From lemma 5.3.5(2), h is a bijective relation.

We next show that h^{τ} is a total relation for each $\tau \in T$. Let $x : [\approx^{\tau}_{F_1}]$. By definition 2.5.3, there exists $z : F_1$ such that $x = [z]$ and $r(\approx^{\tau}_{F_1})z$, which is equivalent to $(R_0\tau)z$ from indistinguishability. Thus there exists an OBS-context Δ, $\rho : F_1^*\Delta$ and a well-formed term $\Delta \vdash_{\Pi} M : \tau$ such that $z = \mathbb{C}[\![M]\!]_{F_1}\rho$. Since $R^*\Delta$ is total, there exists

$\rho' : F_2^*\Delta$ such that $R^*\Delta(\rho, \rho')$. By definition, $h^\tau(x, [\mathbb{C}[\![M]\!]_{F_2}\rho'])$. We have shown that $x : [\approx^\tau_{F_1}]\tau \mid \top \vdash \exists y : [\approx^\tau_{F_1}] \,.\, h^\tau(x, y)$; the other direction can be shown similarly.

It is easy to see that h is a pre-logical total bijective relation. ∎

5.4 Standard Satisfaction and Behavioural Satisfaction of Higher-Order Logic

We consider a higher-order logic to reason about a many-sorted algebra, then introduce two models of the logic. One is the *standard model*, which is a natural interpretation of the logic, and the other is the *behavioural model*, which reasons about a given many-sorted algebra up to some pre-logical PER over the algebra. The behavioural model arose in the study of the logic for specifications taking behaviour into account [BHW95, HS96].

Intuitively speaking, to reason about an algebra up to some equivalence relation is equivalent to reasoning about its quotient algebra. The goal of this section is to prove this formally; we show that the standard model over the quotient algebra of an algebra by a pre-logical PER over it is elementary equivalent to the behavioural model. The key to prove this result is to show that these two models are behaviourally equivalent with respect to the type of *propositions*; then elementary equivalence follows immediately. The same idea has already appeared in section 4.5.

We do not claim that this is a completely new result; the original work goes back to [BHW95] for the first-order case, and a similar result is shown in [HS96] for a different form of higher-order logic. Instead, we emphasise that elementary equivalence between models of a logic can be captured by behavioural equivalence between them with respect to the type of proposition, thus the proof is reduced to finding an observational pre-logical relation between them.

5.4.1 Syntax

The higher-order logic considered in this section is the fairly standard one over a first-order signature $\Sigma = (T_0, O_0)$. The syntax of the higher-order logic can be formalised

in the framework of simply typed formal systems—in fact it is a simply typed lambda calculus over Σ extended with constants for logical connectives.

Definition 5.4.1 A *higher-order logic to reason about* Σ has the following set of types and raw terms:

$$\begin{aligned} \tau &::= \tau_0 \mid \mathbf{\Omega} \mid \tau \Rightarrow \tau \\ M &::= x \mid o(M_1, \cdots, M_n) \mid \lambda x^\tau \,.\, M \mid MM \mid M = M \mid M \supset M. \end{aligned}$$

where τ_0 ranges over T_0 and o ranges over O_0. The type system of the higher-order logic has the following rules.

$$\frac{\Gamma(x) = \tau}{\Gamma \vdash x : \tau} \quad \frac{\Gamma \vdash M_1 : \tau_1, \cdots, \Gamma \vdash M_n : \tau_n \quad o \in O_0 \text{ and has an arity } \tau_1, \cdots, \tau_n \to \tau}{\Gamma \vdash o(M_1, \cdots, M_n) : \tau}$$

$$\frac{\Gamma, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x^\tau \,.\, M : \tau \Rightarrow \tau'} \quad \frac{\Gamma \vdash M : \tau \Rightarrow \tau' \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \tau'}$$

$$\frac{\Gamma \vdash M : \tau \quad \Gamma \vdash N : \tau}{\Gamma \vdash M = N : \mathbf{\Omega}} \quad \frac{\Gamma \vdash M : \mathbf{\Omega} \quad \Gamma \vdash N : \mathbf{\Omega}}{\Gamma \vdash M \supset N : \mathbf{\Omega}}$$

The above rules fit within the scheme of simply typed formal systems, thus can be described by a typed binding signature. The corresponding typed binding signature $\Pi_{\Sigma\text{-hol}}$ for this higher-order logic is defined by $\Pi_{\Sigma\text{-hol}} = (T_{\Sigma\text{-hol}}, O_{\Sigma\text{-hol}})$:

$$\begin{aligned} T_{\Sigma\text{-hol}} &= \mathbf{Typ}^{\Rightarrow}(T_0 \cup \{\mathbf{\Omega}\}) \\ O_{\Sigma\text{-hol}} &= O_0 \cup \{\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}, \mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}, =^{\tau,\tau\to\mathbf{\Omega}}, \supset^{\mathbf{\Omega},\mathbf{\Omega}\to\mathbf{\Omega}}\}. \end{aligned}$$

where τ and τ' range over $T_{\Sigma\text{-hol}}$. We may omit types of constants in superscripts if they are obvious from the context. The constants $\supset, =$ are used as infix operators. □

In the following discussion, we are mainly interested in the semantics of the logic — thus we omit proof systems for the higher-order logic and its soundness and completeness in this thesis. For details see [Hen50]. Lambda abstraction plus logical constants $\supset$ and $=$ are powerful enough to derive other familiar logical constants such as $\mathsf{true}, \mathsf{false}, \neg, \wedge, \vee, \neq$ and quantifiers $\forall x : \phi \,.\, F$ and $\exists x : \phi \,.\, F$ [And86]. We call a (possibly open) term of type $\mathbf{\Omega}$ *formula* and a closed formula *sentence*.

5.4.2 Standard Satisfaction and Behavioural Satisfaction

We introduce an interpretation of the higher-order logic in $\mathbf{Sets}$. Below we fix a many-sorted Σ-algebra $\mathcal{A}$ in $\mathbf{Sets}$. The *standard model* interprets the type of propositions $\mathbf{\Omega}$ as the two-point set $\mathbf{2} = \{\mathrm{tt}, \mathrm{ff}\}$, arrow types as function spaces, and the equality predicate at type τ as the characteristic function of the identity relation over the carrier set of type τ.

Definition 5.4.2 1. We extend the carrier sets $A : T_0 \to \mathbf{Sets}$ of the many-sorted Σ-algebra $\mathcal{A}$ to the interpretation of types $\overline{A} : T_{\Sigma\text{-hol}} \to \mathbf{Sets}$ by induction:

$$\begin{aligned} \overline{A}\mathbf{\Omega} &= \mathbf{2} \\ \overline{A}\tau_0 &= A\tau_0 \quad (\tau_0 \in T_0) \\ \overline{A}(\tau \Rightarrow \tau') &= \overline{A}\tau \Rightarrow \overline{A}\tau'. \end{aligned}$$

In the third line, $\Rightarrow$ on the left is an arrow type, while $\Rightarrow$ on the right is a function space.

2. We give a $\Pi_{\Sigma\text{-hol}}$-algebra structure β^s over the presheaf $H^{\overline{A}}$ in $\mathbb{P}_T$ by specifying a morphism β^s_o in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O_{\Sigma\text{-hol}}$ (variables are interpreted by projections; c.f. example 3.4.4):

$$\begin{aligned} &(\beta^s_o)_\Gamma(f_1, \cdots, f_n) = o_{\mathcal{A}} \circ \langle f_1, \cdots, f_n \rangle \quad (o \in O_0) \\ &(\beta^s_{\mathsf{lam}^{\tau,\tau'\to\tau\Rightarrow\tau'}})_\Gamma(f) = \lambda x \in \overline{A}^*\Gamma \,.\, \lambda y \in A\tau \,.\, f \circ s_{\Gamma,\tau}(x, y) \\ &(\beta^s_{\mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}})_\Gamma(f, g) = @ \circ \langle f, g \rangle \\ &(\beta^s_{=^{\tau,\tau\to\mathbf{\Omega}}})_\Gamma(f, g) = \mathrm{tt} \iff f = g \\ &(\beta^s_{\supset})_\Gamma(f, g) = imp \circ \langle f, g \rangle \end{aligned}$$

where $imp : \mathbf{2} \times \mathbf{2} \to \mathbf{2}$ is defined by

$$imp(x, y) = \mathrm{tt} \iff (x = \mathrm{tt} \implies y = \mathrm{tt}).$$

We write $\mathcal{L}_{\mathcal{A}}[\![-]\!] : S_{\Pi_{\Sigma\text{-hol}}} \to H^{\overline{A}}$ for the interpretation of terms obtained by initiality. We say that a sentence M is *satisfiable* (written $\mathcal{A} \models M$) if $\mathcal{L}_{\mathcal{A}}[\![M]\!] = \mathcal{L}_{\mathcal{A}}[\![\mathsf{true}]\!] = \mathrm{tt}$. □

The other model is the *behavioural model* with respect to the indistinguishability relation over $\mathcal{A}$. The standard model is not appropriate when we reason about programs up to their behaviour, since the equality predicate may distinguish two terms having different denotations even though they have the same behaviour. The behavioural model solves this problem by interpreting each type τ as $|\overline{\approx}^{\tau}|$ where $\overline{\approx}$ is the extension of indistinguishability relation to all types using exponentiation, with the equality predicate over τ as the equivalence relation $\overline{\approx}$ over $|\overline{\approx}|$.

For the sake of generality, in the following definition of the behavioural model, we replace the indistinguishability relation with arbitrary pre-logical PER E for Σ along $\mathcal{A}[\![-]\!] \times \mathcal{A}[\![-]\!]$ (that is, a *partial congruence* for $\mathcal{A}$).

We write $\mathbf{PER}$ for the category $\mathbf{PER}(p_{\mathbf{Sets}})$ and $q : \mathbf{PER} \to \mathbf{Sets}$ for the fibration $q_{p_{\mathbf{Sets}}}$ in definition 2.5.3.

Definition 5.4.3 1. Let $E \subseteq_{-\times-} \langle A, A\rangle$ be a pre-logical PER for Σ along $\mathcal{A}[\![-]\!] \times \mathcal{A}[\![-]\!]$. This specifies an interpretation of types $E : T_0 \to \mathbf{PER}$. We extend E to the interpretation of types $\overline{E} : T_{\Sigma\text{-hol}} \to \mathbf{PER}$ by induction:

$$\begin{aligned} \overline{E}\mathbf{\Omega} &= (\mathbf{2}, \mathrm{Id}_{\mathbf{2}}) \\ \overline{E}\tau_0 &= E\tau_0 \quad (\tau_0 \in T_0) \\ \overline{E}(\tau \Rightarrow \tau') &= \overline{E}\tau \mathrel{\dot{\Rightarrow}} \overline{E}\tau'. \end{aligned}$$

In the third line, $\Rightarrow$ on the left is an arrow type, while $\mathrel{\dot{\Rightarrow}}$ on the right is an exponentiation in $\mathbf{PER}$ (see proposition 2.5.5).

We note that $q(\overline{E}\tau) = \overline{A}\tau$.

2. We define a function $eq_{\overline{E}\tau} : \overline{A}\tau \times \overline{A}\tau \to \mathbf{2}$ by

$$eq_{\overline{E}\tau}(x, y) = \mathrm{tt} \iff (x, y) \in \overline{E}\tau.$$

We see the following facts hold in $\mathbf{PER}$:

$$o_{\mathcal{A}} : \overline{E}\tau_1 \mathrel{\dot{\times}} \cdots \mathrel{\dot{\times}} \overline{E}\tau_n \to \overline{E}\tau \tag{5.3}$$

$$eq_{\overline{E}\tau} : \overline{E}\tau \mathrel{\dot{\times}} \overline{E}\tau \to \overline{E}\mathbf{\Omega} \tag{5.4}$$

$$imp : \overline{E}\mathbf{\Omega} \mathrel{\dot{\times}} \overline{E}\mathbf{\Omega} \to \overline{E}\mathbf{\Omega} \tag{5.5}$$

where $o \in O_0$ is an operator of arity $\tau_1, \cdots, \tau_n \to \tau$. (5.3) is equivalent to that $o_{\mathcal{A}} \times o_{\mathcal{A}} : \overline{E}\tau_1 \dot{\times} \cdots \dot{\times} \overline{E}\tau_n \to \overline{E}\tau$ holds in $\mathbf{Sub}(\mathbf{Sets})$. The latter is true, since $\overline{E}\tau = E\tau$ for each $\tau \in T_0$ and $o_{\mathcal{A}} = \mathcal{A}[\![o(\mathbf{v}_1, \cdots, \mathbf{v}_n)]\!]$ and E is pre-logical for Σ along $\mathcal{A}[\![-]\!] \times \mathcal{A}[\![-]\!]$. (5.4) follows from the transitivity of $\overline{E}\tau$. (5.5) is clear.

We give a $\Pi_{\Sigma\text{-hol}}$-algebra structure over the presheaf $H^{\overline{E}}$ in $\mathbb{P}_T$ by specifying a morphism β^b_o in $[\mathbb{V}_T, \mathbf{Sets}]$ for each operator $o \in O_{\Sigma\text{-hol}}$ (variables are interpreted by projections; c.f. example 3.4.4):

$$
\begin{aligned}
&(\beta^b_o)_\Gamma(f_1, \cdots, f_n) = o_{\mathcal{A}} \circ \langle f_1, \cdots, f_n \rangle \quad (o \in O_0)\\
&(\beta^b_{\mathsf{lam}^{\tau,\tau' \to \tau \Rightarrow \tau'}})_\Gamma(f) = \lambda x \in \overline{A}^*\Gamma \,.\, \lambda y \in \overline{A}\tau \,.\, f \circ s_{\Gamma,\tau}(x, y)\\
&(\beta^b_{\mathsf{app}^{\tau \Rightarrow \tau', \tau \to \tau'}})_\Gamma(f, g) = @ \circ \langle f, g \rangle\\
&(\beta^b_{=^{\tau,\tau \to \mathbf{\Omega}}})_\Gamma(f, g) = eq_{\overline{E}\tau} \circ \langle f, g \rangle\\
&(\beta^b_{\supset})_\Gamma(f, g) = imp \circ \langle f, g \rangle.
\end{aligned}
$$

We write $\mathcal{L}^E_{\mathcal{A}}[\![-]\!] : S_{\Pi_{\Sigma\text{-hol}}} \to H^{\overline{E}}$ for the interpretation of terms obtained by initiality. We say that a sentence M is *behaviourally satisfiable* (written by $\mathcal{A} \models^E F$) if $\mathcal{L}^E_{\mathcal{A}}[\![M]\!] = \mathcal{L}^E_{\mathcal{A}}[\![\mathsf{true}]\!] = \mathrm{tt}$. □

5.4.3 Equivalence of Standard Satisfaction and Behavioural Satisfaction

For a pre-logical PER E for Σ along $\mathcal{A}[\![-]\!] \times \mathcal{A}[\![-]\!]$, we consider the quotient algebra $\mathcal{A}/E$ defined as follows:

$$(\{[E]\tau\}_{\tau \in T_0}, \{o_{\mathcal{A}/E}\}_{o \in O_0})$$

where for each operator o of arity $\tau_1, \cdots, \tau_n \to \tau$, $o_{\mathcal{A}/E}$ is defined by:

$$o_{\mathcal{A}/E} = [o_{\mathcal{A}}] \circ k_{\tau_1, \cdots, \tau_n}.$$

with the canonical isomorphism $k_{\tau_1, \cdots, \tau_n} : [E]^*(\tau_1, \cdots, \tau_n) \to [E^*(\tau_1, \cdots, \tau_n)]$ (see lemma 2.5.7). Intuitively, reasoning about $\mathcal{A}$ modulo the PER E is equivalent to reasoning about $\mathcal{A}/E$. The main result formulates this intuition. We show that the standard satisfaction $\mathcal{A}/E$ is equivalent to behavioural satisfaction over $\mathcal{A}$ with respect to E.

Theorem 5.4.4 *For any sentence M, we have*

$$\mathcal{A}/E \models M \iff \mathcal{A} \models^E M.$$

PROOF Our primary goal is to show the following behavioural equivalence with respect to $\{\mathbf{\Omega}\}$:

$$|\mathcal{L}^E_{\mathcal{A}}[\![-]\!]| \circ l \equiv_{\{\mathbf{\Omega}\}} \mathcal{L}_{\mathcal{A}/E}[\![-]\!] \tag{5.6}$$

where $l_\Gamma : |\overline{E}|^*\Gamma \to |\overline{E^*}\Gamma|$ is the canonical natural isomorphism (see lemma 2.5.7).

Suppose (5.6) is proved. It is easy to see that for any sentence M and N, we have

$$|\mathcal{L}^E_{\mathcal{A}}[\![M]\!]| \circ l_\emptyset = |\mathcal{L}^E_{\mathcal{A}}[\![N]\!]| \circ l_\emptyset \iff \mathcal{L}^E_{\mathcal{A}}[\![M]\!] = \mathcal{L}^E_{\mathcal{A}}[\![N]\!].$$

Then for a sentence M, we have

$$\begin{aligned}
\mathcal{A}/E \models M &\iff \mathcal{L}_{\mathcal{A}/E}[\![M]\!] = \mathcal{L}_{\mathcal{A}/E}[\![\mathsf{true}]\!] \\
&\iff |\mathcal{L}^E_{\mathcal{A}}[\![M]\!]| \circ l_\emptyset = |\mathcal{L}^E_{\mathcal{A}}[\![\mathsf{true}]\!]| \circ l_\emptyset \quad \text{(from behavioural equivalence)} \\
&\iff \mathcal{L}^E_{\mathcal{A}}[\![M]\!] = \mathcal{L}^E_{\mathcal{A}}[\![\mathsf{true}]\!] \\
&\iff \mathcal{A} \models^E M.
\end{aligned}$$

Thus we aim to prove (5.6). From the characterisation theorem of behavioural equivalence (theorem 5.1.7), it is sufficient to show

$$|\mathcal{L}^E_{\mathcal{A}}[\![-]\!]| \circ l \sim_{\{\mathbf{\Omega}\}} \mathcal{L}_{\mathcal{A}/E}[\![-]\!].$$

We give the observational pre-logical relation in question by the following graph relation $G_p\tau$:

$$G_p\tau = \{(x, y) \in |\overline{E}| \times \overline{[E]} \mid y = p^\tau(x)\}$$

of a function p^τ, which is defined together with $i^\tau : |\overline{E}|\tau \to \overline{[E]}\tau$ so that they satisfy $p^\tau \circ i^\tau = \mathrm{id}_{\overline{[E]}\tau}$ (in other words, i^τ and p^τ gives a witness of $\overline{[E]}\tau$ being a *retract* of $|\overline{E}|\tau$). Below we see the detail of the proof.

By $i : A \lhd B : p$ we mean $i : A \to B$ and $p : B \to A$ are functions such that $p \circ i = \mathrm{id}_A$. A simple calculation shows that

$$\begin{aligned}
&i : A \lhd B : p \wedge i' : A' \lhd B' : p' \\
&\quad \Longrightarrow (\lambda f\,.\, i' \circ f \circ p) : A \Rightarrow A' \lhd B \Rightarrow B' : (\lambda g\,.\, p' \circ g \circ i).
\end{aligned} \tag{5.7}$$

Definition 5.4.5 We define a $T_{\Sigma\text{-hol}}$-indexed family of functions $i^\tau : \overline{[E]}\tau \lhd |\overline{E}|\tau : p^\tau$ satisfying

$$\forall x \in \overline{[E]}\tau \,.\, i^\tau x \in |\overline{E}|\tau \tag{5.8}$$

$$\forall (x,y) \in \overline{E}\tau \,.\, p^\tau x = p^\tau y \tag{5.9}$$

$$\forall x \in |\overline{E}\tau| \,.\, (i^\tau(p^\tau x), x) \in \overline{E}\tau \tag{5.10}$$

by induction on types.

- Case $\tau \in T_0$. We have $\overline{[E]}\tau = [E]\tau$ and $|\overline{E}|\tau = |\overline{E}|\tau$. We define $p^\tau x = [x]_{E\tau}$ and $i^\tau x$ to be some element in x (we use the axiom of choice). The functions i^τ, p^τ satisfy (5.8),(5.9) and (5.10).

- Case $\mathbf{\Omega}$. We have $\overline{[E]}\mathbf{\Omega} = \mathbf{2} = [\overline{E}]\mathbf{\Omega}$. We define $i^{\mathbf{\Omega}} = p^{\mathbf{\Omega}} = \mathrm{id}_{\mathbf{2}}$. The functions $i^{\mathbf{\Omega}}, p^{\mathbf{\Omega}}$ satisfy (5.8),(5.9) and (5.10).

- Case $\tau \Rightarrow \tau'$. We define

$$\begin{aligned} i^{\tau\Rightarrow\tau'} f &= i^{\tau'} \circ f \circ p^\tau \\ p^{\tau\Rightarrow\tau'} f &= p^{\tau'} \circ g \circ i^\tau. \end{aligned}$$

  From (5.7), we have

$$i^{\tau\Rightarrow\tau'} : \overline{[E]}(\tau \Rightarrow \tau') \lhd |\overline{E}|\tau \Rightarrow |\overline{E}|\tau' : p^{\tau\Rightarrow\tau'}.$$

  We check that for any $f \in \overline{[E]}(\tau \Rightarrow \tau')$, we have $i^{\tau\Rightarrow\tau'} f \in \overline{E}\tau \dot{\Rightarrow} \overline{E}\tau'$. Let $(x,y) \in \overline{E}\tau$. From (5.9), we have $p^\tau x = p^\tau y$. Therefore $(i^{\tau\Rightarrow\tau'} f)x = (i^{\tau\Rightarrow\tau'} f)y \in |\overline{E}|\tau'$. Thus $i^{\tau\Rightarrow\tau'} f \in \overline{E}\tau \dot{\Rightarrow} \overline{E}\tau'$ holds. This also proves (5.8), that is, $i^{\tau\Rightarrow\tau'} f : \overline{[E]}(\tau \Rightarrow \tau') \to |\overline{E}|(\tau \Rightarrow \tau')$.

  We show (5.9). Let $(f,g) \in \overline{E}(\tau \Rightarrow \tau')$ and $x \in \overline{[E]}\tau$. We have $i^\tau x \in |\overline{E}|\tau$ from (5.8). Therefore $(f(i^\tau x), g(i^\tau x)) \in \overline{E}\tau'$. From (5.9), we obtain $p^{\tau\Rightarrow\tau'} f = p^{\tau'}(f(i^\tau x)) = p^{\tau'}(g(i^\tau x)) = p^{\tau\Rightarrow\tau'} f$.

  We check (5.8). Let $f : |\overline{E}|(\tau \Rightarrow \tau')$and $(x,y) \in \overline{E}\tau$. We show

$$(i^{\tau'}(p^{\tau'}(f(i^\tau(p^\tau x)))), y) \in \overline{E}\tau'.$$

From (5.10), we have $(i^{\tau}(p^{\tau}x), x) \in \overline{E}\tau$ by IH. Thus we have $(i^{\tau}(p^{\tau}x), y) \in \overline{E}\tau$ by transitivity. From the assumption on f, we have $(f(i^{\tau}(p^{\tau}x)), fy) \in \overline{E}\tau'$. From (5.10), we have $(i^{\tau'}(p^{\tau'}(f(i^{\tau}(p^{\tau}x)))), f(i^{\tau}(p^{\tau}x))) \in \overline{E}\tau'$. Therefore by transitivity, we obtain $(i^{\tau'}(p^{\tau'}(f(i^{\tau}(p^{\tau}x)))), fy) \in \overline{E}\tau'$. □

We write $G_p\tau = \{(a, b) \mid p^{\tau}(a) = b\}$ for the graph relation of p^{τ}. Clearly $G_p \subseteq_{-\times-} \langle|\overline{E}|, \overline{[E]}\rangle$ is a binary relation and $G_p\mathbf{\Omega}$ is total bijective [2].

Lemma 5.4.6 *The binary relation $G_p \subseteq_{-\times-} \langle|\overline{E}|, \overline{[E]}\rangle$ is pre-logical for $\Pi_{\Sigma\text{-hol}}$ along $|\mathcal{L}^E_A[\![-]\!]| \circ l \times \mathcal{L}_{\mathcal{A}/E}[\![-]\!]$.* □

PROOF We show for each binding operator $o \in O_{\Sigma\text{-hol}}$ of arity $(\overrightarrow{\tau_1}, \sigma_1), \cdots, (\overrightarrow{\tau_n}, \sigma_n) \to \tau$ and well-formed terms $\Gamma, \overrightarrow{x_i : \tau_i} \vdash_{\Pi_{\Sigma\text{-hol}}} M_i : \sigma_i \ (1 \leq i \leq n)$,

$$|\mathcal{L}^E_A[\![M_i]\!]| \circ l_{\Gamma} \times \mathcal{L}_{\mathcal{A}/E}[\![M_i]\!] : G^*_p(\Gamma \uplus \overrightarrow{\tau_i}) \to G_p\sigma_i \quad (1 \leq i \leq n)$$

implies

$$|\mathcal{L}^E_A[\![o(\overrightarrow{x^{\tau_1}_1} . M_1, \cdots, \overrightarrow{x^{\tau_n}_n} . M_n)]\!]| \circ l_{\Gamma} \times \mathcal{L}_{\mathcal{A}/E}[\![o(\overrightarrow{x^{\tau_1}_1} . M_1, \cdots, \overrightarrow{x^{\tau_n}_n} . M_n)]\!] : G^*_p\Gamma \to G_p\tau.$$

We note that for any well-formed term $\Gamma \vdash_{\Pi_{\Sigma\text{-hol}}} M : \tau$,

$$|\mathcal{L}^E_A[\![M]\!]| \circ l_{\Gamma} \times \mathcal{L}_{\mathcal{A}/E}[\![M]\!] : G^*_p\Gamma \to G_p\tau$$

is equivalent to

$$p^{\tau} \circ |\mathcal{L}^E_A[\![M]\!]| \circ l_{\Gamma} = \mathcal{L}_{\mathcal{A}/E}[\![M]\!] \circ p^{\Gamma}$$

where $p^{\Gamma} = \prod_{i=1}^{\#\mathrm{dom}(\Gamma)} p^{\Gamma(\gamma^{-1}(i))}$.

- Case $\mathsf{lam}^{(\tau,\tau')\to\tau\Rightarrow\tau'}$. Let $\Gamma, x : \tau \vdash_{\Pi_{\Sigma\text{-hol}}} M : \tau'$ be a well-formed term. We assume

  $$|\mathcal{L}^E_A[\![M]\!]| \circ l_{\Gamma\uplus\tau} \times \mathcal{L}_{\mathcal{A}/E}[\![M]\!] : G^*_p(\Gamma \uplus \tau) \to G_p\tau'.$$

  For any $\rho \in |\overline{E}|^*\Gamma$ and $v \in |\overline{E}|\tau$, we have

  $$\begin{aligned} p^{\tau\Rightarrow\tau'}(|\mathcal{L}^E_A[\![\lambda x^{\tau} . M]\!]| \circ l_{\Gamma}\rho)v &= (p^{\tau\Rightarrow\tau'}(\mathcal{L}^E_A[\![\lambda x^{\tau} . M]\!]\rho))v \\ &= p^{\tau'}((\mathcal{L}^E_A[\![\lambda x^{\tau} . M]\!]\rho)(i^{\tau}v)) \\ &= p^{\tau'}(\mathcal{L}^E_A[\![M]\!] \circ s^{|\overline{E}|}_{\Gamma,\tau}(\rho, i^{\tau}v)) \\ &= \mathcal{L}_{\mathcal{A}/E}[\![M]\!] \circ s^{\overline{|E|}}_{\Gamma,\tau}(p^{\Gamma}\rho, v) \\ &= (\mathcal{L}_{\mathcal{A}/E}[\![\lambda x^{\tau} . M]\!](p^{\Gamma}\rho))v. \end{aligned}$$

[2] $G_p\tau$ is also total bijective for each type $\tau \in T_0$, but we do not use this fact in this proof.

This implies that $|\mathcal{L}^E_A[\![\lambda x^\tau \,.\, M]\!]| \circ l_\Gamma \times \mathcal{L}_{\mathcal{A}/E}[\![\lambda x^\tau \,.\, M]\!] : G^*_p\Gamma \to G_p(\tau \Rightarrow \tau')$.

- Case $\mathsf{app}^{\tau\Rightarrow\tau',\tau\to\tau'}$. It suffices to show

$$(|@| \circ l') \times @ : G_p(\tau \Rightarrow \tau') \times G_p\tau \to G_p\tau',$$

which is equivalent to

$$p^{\tau'}(|@|(l'(f,v))) = (p^{\tau\Rightarrow\tau'}f)(p^\tau v)$$

where $l' : |\overline{E}|(\tau \Rightarrow \tau') \times |\overline{E}|\tau \to |\overline{E}(\tau \Rightarrow \tau') \dot{\times} \overline{E}|$ is the canonical isomorphism. Let $f \in |\overline{E}(\tau \Rightarrow \tau')|$ and $v \in |\overline{E}\tau|$. By definition, we have $(p^{\tau\Rightarrow\tau'}f)(p^\tau v) = p^{\tau'} \circ f \circ i^{\tau'}(p^\tau v)$. From (5.10), we have $(f(i^{\tau'}(p^\tau v)), fv) \in \overline{E}\tau'$, and from (5.9), we obtain $p^{\tau'}(f(i^{\tau'}(p^\tau v))) = p^{\tau'}(fv) = p^{\tau'}(|@| \circ l'(f,v))$.

- Case $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$. It suffices to show

$$|o_\mathcal{A}| \circ l_{\tau_1,\cdots,\tau_n} \times [o_\mathcal{A}] \circ k_{\tau_1,\cdots,\tau_n} : G_p\tau_1 \times \cdots \times G_p\tau_n \to G_p\tau.$$

Let $a_i \in |E\tau_i|$ for $1 \le i \le n$. We have

$$\begin{aligned} p^\tau \circ |o_\mathcal{A}| \circ l_{\tau_1,\cdots,\tau_n}(a_1,\cdots,a_n) &= [o_\mathcal{A}(a_1,\cdots,a_n)]_{E\tau} \\ &= [o_\mathcal{A}] \circ k_{\tau_1,\cdots,\tau_n}([a_1]_{E\tau_1},\cdots,[a_n]_{E\tau_n}) \\ &= [o_\mathcal{A}] \circ k_{\tau_1,\cdots,\tau_n}(p^{\tau_1}a_1,\cdots,p^{\tau_n}a_n). \end{aligned}$$

- Case $=^{\tau,\tau\to\mathbf{\Omega}}$ and $\supset^{\mathbf{\Omega},\mathbf{\Omega}\to\mathbf{\Omega}}$. This case can be proved in the same way as the previous case. ■

Corollary 5.4.7 $|\mathcal{L}^E_A[\![-]\!]| \circ l \sim_{\{\mathbf{\Omega}\}} \mathcal{L}_{\mathcal{A}/E}[\![-]\!]$. □

PROOF By definition, $G_p\mathbf{\Omega}$ is total bijective. Thus G_p is an observational pre-logical relation. ■

From theorem 5.1.7, (5.6) holds. The rest of proof is already discussed in the beginning. ■

One would like to generalise this result further by replacing $\mathbf{Sets}$ with a "$\mathbf{Sets}$-like" category, i.e. a *topos* [MM92]. This is more natural setting, because topoi provide a natural class of models for higher-order logic. This direction is also suggested in [HS96]. There seems no technical difficulty in redoing the above proof in a topos admitting the axiom of choice (which was used to construct i^τ for $\tau \in T_0$). However we do not know if the axiom of choice is essential in showing (5.6).

5.5 Conclusion

We have extended the study of the relationship between behavioural equivalence and indistinguishability [BHW95, HS96] to simply typed formal systems. We characterised behavioural equivalence between two typed formal systems by the existence of a observational pre-logical relations, and showed that behavioural equivalence is factorised by indistinguishability.

We applied this characterisation theorem to show that two models of a higher-order logic for reasoning about a many-sorted algebra are elementary equivalent. The key observation is that elementary equivalence is a consequence of behavioural equivalence. Thus we actually constructed an observational pre-logical relation between models of a higher-order logic.

Related Work

The work by Bidoit, Hennicker and Wirsing [BHW95] established the key idea of factorisability for relating behavioural equivalence and the indistinguishability relation, and they used this to reason about the semantics of behavioural and abstractor specifications. In [BH96], Bidoit and Hennicker discussed a proof method for showing behavioural equivalence in first order logic, and considered finitary axiomatisation of behavioural equality. Hofmann and Sannella represented the indistinguishability relation and the "experiments" for behavioural equivalence in a higher-order logic, then showed that the satisfiability of the experiments coincides in each model when quotients of the two models are isomorphic [HS96].

In [BT96], Bidoit and Tarlecki gave a relationship between behavioural satisfaction, behavioural equivalence, indistinguishability and correspondences in concrete categories (a category which has a faithful functor to the category of (type-indexed) sets) satisfying certain properties. The main difference is that the semantics category in our framework need not be a concrete category. On the other hand, we examined the structure necessary for formulating behavioural equivalence and establishing characterisation results. The formal relationship between these two approaches is not clear.

Chapter 6

An Application of Pre-Logical Predicates to Data Refinement

Data refinement is an activity of constructing high-level, user-oriented data structures and accompanying operators by combining existing data structures provided by some basic libraries and programming constructs. We say such a data refinement is correct if the high-level data structures constructed over any implementation of the low-level data structures conform to the specification of the high-level data structures.

We explain the idea of data refinement with the following analogy. Suppose there is a library which provides a data type for files with flexible, useful operators on files. The library is written in the C language and uses the UNIX environment, and realises the data type for files and operators by means of the data types and system calls provided by the C language and the UNIX environment. This library corresponds to a data refinement, and the correctness of this data refinement corresponds to the fact that the library compiled on any C language compiler and operating system providing the environment shows the same expected behaviour to users.

Typically the target of a data refinement is abstract data types. The characteristic property of abstract data types is that programmers can not know anything about their internal representation, but can only create and inspect these values by means of the accompanying operators. The opposite of abstract data types is observable types; programmers can inspect their internal representation and check their equality.

This classification of types restricts programmers' knowledge about a programming environment to the values and programs over observable types. The behavioural equivalence introduced in the previous chapter can be regarded as the equivalence between programming environments up to programmers' knowledge.

Under this restricted knowledge, programmers may accept some implementations which do not strictly conform to a specification, because they still show the same behaviour as an implementation conforming to a specification. Such an implementation should be accepted as a possible realisation of the original specification.

The style of data refinement we consider in this chapter reflects this idea, but we use observational pre-logical relations instead of behavioural equivalence. As we have seen in chapter 5, the existence of the former relation implies the latter. The closure of binary pre-logical relations under relational composition is then used to show that data refinements compose.

6.1 Specification for Typed Formal Systems

We first clarify the notion of specification over typed formal systems. In the world of algebraic specification, a specification SP over a many-sorted signature Σ is a set of formulae of some logic which describes desirable properties of the data types and relevant operators defined by Σ. Such a specification stands for the collection of all of its possible realising Σ-algebras. This is the so-called *loose semantics of the specification*. Concretely speaking, the semantics of a Σ-specification SP is given by the following collection of all Σ-algebras in which each formula in the specification is satisfied:

$$\mathbf{Mod}(SP) = \{\mathcal{A} \mid \mathcal{A} \text{ is a } \Sigma\text{-algebra}, \forall F \in SP \,.\, \mathcal{A} \models F\}.$$

In this thesis, to avoid a discussion of the logic for specifications and its semantics, we just take what specifications stand for as the definition of specification.

Definition 6.1.1 A *specification* over a typed binding signature Π in a Cartesian category $\mathbb{C}$ is a collection of categorical interpretations of Π in $\mathbb{C}$. □

The followings are typical examples of specifications.

Definition 6.1.2 Let Π be a typed binding signature and $\mathbb{C}$ be a Cartesian category.

1. We define the specification $C_{\mathbb{C},\Pi}$ over Π in $\mathbb{C}$ by the following collection of categorical interpretations of Π in $\mathbb{C}$:

$$\begin{aligned} \mathbb{C}[\![-]\!]_F \in C_{\mathbb{C},\Pi} \quad &\Longleftrightarrow \quad \forall \Gamma \in \mathbf{Obj}(\mathbb{V}_T), x \in \mathrm{dom}(\Gamma) \,.\, \mathbb{C}[\![x^\tau]\!]_F = \pi_{\gamma(x)} \\ &\wedge \quad \mathbb{C}[\![-]\!]_F \text{ satisfies the semantic substitution lemma.} \end{aligned}$$

2. An *equational axiom* over Π is a tuple (Γ, M, N, τ) such that $\Gamma \vdash_\Pi M : \tau$ and $\Gamma \vdash_\Pi N : \tau$ are well-formed terms. We write $\Gamma \vdash_\Pi M = N : \tau$ for such a tuple.

   An *equational specification* $ES_{\mathbb{C},\Pi}(E)$ in $\mathbb{C}$ given by a set of equational axioms E over Π is the collection of categorical interpretations of Π in $\mathbb{C}$ such that for each axiom $\Gamma \vdash_\Pi M = N : \tau \in E$,

$$\mathbb{C}[\![M]\!]_F = \mathbb{C}[\![N]\!]_F$$

   holds.

When the category $\mathbb{C}$ is clear from the context, we simply write C_Π and $ES_\Pi(E)$ instead of $C_{\mathbb{C},\Pi}$ and $ES_{\mathbb{C},\Pi}(E)$. □

Example 6.1.3 Let $\Sigma = (T_0, O_0)$ be a typed first-order signature. We write Σ-$\mathbf{Alg}$ to denote the collection of many-sorted Σ-algebras in the traditional sense (see section 4.1). We show that C_Σ characterises the class of many-sorted Σ-algebras: that is, there exists an isomorphism

$$h_\Sigma : \Sigma\text{-}\mathbf{Alg} \cong C_\Sigma.$$

The function h_Σ sends a many-sorted Σ-algebra $\mathcal{A}$ to its standard interpretation (see section 4.1). It is easy to see that the standard interpretation satisfies the conditions defining C_Π. The function h_Σ^{-1} sends an interpretation $\mathbf{Sets}[\![-]\!]_F \in C_\Sigma$ of Σ to the following many-sorted Σ-algebra $\mathcal{F}$:

$$\mathcal{F} = (\{F\tau\}_{\tau \in T_0}, \{o_\mathcal{F}\}_{o \in O_0})$$

where for each operator $o \in O_0$ of arity $\tau_1, \cdots, \tau_n \to \tau$, $o_\mathcal{F}$ is defined by

$$o_\mathcal{F} = \mathbf{Sets}[\![o(\mathbf{v}_1, \cdots, \mathbf{v}_n)]\!]_F : F\tau_1 \times \cdots \times F\tau_n \to F\tau.$$

It is easy to see that $h_\Sigma^{-1} \circ h_\Sigma = \mathrm{id}_{\Sigma\text{-}\mathbf{Alg}}$, so we show that $h_\Sigma \circ h_\Sigma^{-1} = \mathrm{id}_{C_\Sigma}$. Let $\mathbf{Sets}[\![-]\!]_F \in C_\Sigma$. We write $\mathcal{F}$ for the many-sorted Σ-algebra $h_\Sigma^{-1}(\mathbf{Sets}[\![-]\!]_F)$ as defined above. We show by induction that the standard interpretation $\mathcal{F}[\![-]\!]$ is equal to $\mathbf{Sets}[\![-]\!]_F$.

- Case $\Gamma \vdash_\Sigma x^\tau : \tau$. We have $\mathcal{F}[\![x^\tau]\!] = \pi_{\gamma(x)} = \mathbf{Sets}[\![x^\tau]\!]_F$.
- Case $\Gamma \vdash_\Sigma o(M_1, \cdots, M_n) : \tau$ where $o \in O_0$ is an operator of arity $\tau_1, \cdots, \tau_n \to \tau$. We have

$$\begin{aligned} & \mathcal{F}[\![o(M_1, \cdots, M_n)]\!] \\ = \;& o_{\mathcal{F}} \circ \langle \mathcal{F}[\![M_1]\!], \cdots, \mathcal{F}[\![M_n]\!] \rangle \\ = \;& \mathbf{Sets}[\![o(\mathbf{v}_1, \cdots, \mathbf{v}_n)]\!]_F \circ \langle \mathbf{Sets}[\![M_1]\!]_F, \cdots, \mathbf{Sets}[\![M_n]\!]_F \rangle \\ = \;& \mathbf{Sets}[\![o(M_1, \cdots, M_n)]\!]_F. \end{aligned}$$

  We used the semantic substitution lemma to derive the last line. Thus $h \circ h^{-1} = \mathrm{id}_{C_\Sigma}$ is proved. □

Example 6.1.4 Recall that a combinatory algebra $\mathcal{U}$ is a many-sorted Σ_{CL}-algebra satisfying the following equations (type annotations are omitted):

$$\begin{aligned} K_{\mathcal{U}} \bullet_{\mathcal{U}} x \bullet_{\mathcal{U}} y &= x \\ S_{\mathcal{U}} \bullet_{\mathcal{U}} x \bullet_{\mathcal{U}} y \bullet_{\mathcal{U}} z &= x \bullet_{\mathcal{U}} z \bullet_{\mathcal{U}} (y \bullet_{\mathcal{U}} z). \end{aligned}$$

where x, y, z ranges over the carrier sets of appropriate types (see section 4.1).

The above two equations can equivalently be rewritten as the following two conditions:

1. for all well-formed terms

$$\Gamma_s^{\tau,\tau',\tau''} \vdash_{\Sigma_{CL}} S \bullet \mathbf{v}_1 \bullet \mathbf{v}_2 \bullet \mathbf{v}_3, (\mathbf{v}_1 \bullet \mathbf{v}_3) \bullet (\mathbf{v}_2 \bullet \mathbf{v}_3) : \tau''$$

   where $\Gamma_s^{\tau,\tau',\tau''} = \{\mathbf{v}_1 : \tau \Rightarrow \tau' \Rightarrow \tau'', \mathbf{v}_2 : \tau \Rightarrow \tau', \mathbf{v}_3 : \tau\}$, we have

$$\mathcal{U}[\![S \bullet \mathbf{v}_1 \bullet \mathbf{v}_2 \bullet \mathbf{v}_3]\!] = \mathcal{U}[\![(\mathbf{v}_1 \bullet \mathbf{v}_3) \bullet (\mathbf{v}_2 \bullet \mathbf{v}_3)]\!]$$

   and

2. for all well-formed terms

$$\Gamma_k^{\tau,\tau'} \vdash_{\Sigma_{CL}} K \bullet \mathbf{v}_1 \bullet \mathbf{v}_2, \mathbf{v}_1 : \tau$$

where $\Gamma_k^{\tau,\tau'} = \{\mathbf{v}_1 : \tau, \mathbf{v}_2 : \tau'\}$, we have

$$\mathcal{U}[\![K \bullet \mathbf{v}_1 \bullet \mathbf{v}_2]\!] = \mathcal{U}[\![\mathbf{v}_1]\!].$$

Furthermore, by using the function $h_{\Sigma_{CL}} : \Sigma_{CL} \to C_{\Sigma_{CL}}$ defined in the previous example, the above two conditions can be rephrased as

$$h_{\Sigma_{CL}}(\mathcal{U}) \in ES_{\Sigma_{CL}}(E_{CL})$$

where E_{CL} is the following equational axiom over Σ_{CL}:

$$\begin{array}{rcl} E_{CL} & = & \{\Gamma_s^{\tau,\tau',\tau''} \vdash_{\Sigma_{CL}} S \bullet \mathbf{v}_1 \bullet \mathbf{v}_2 \bullet \mathbf{v}_3 = (\mathbf{v}_1 \bullet \mathbf{v}_3) \bullet (\mathbf{v}_2 \bullet \mathbf{v}_3) : \tau'') \mid \\ & & \quad \tau, \tau', \tau'' \in \mathbf{Typ}^{\Rightarrow}(B)\} \\ & \cup & \{\Gamma_k^{\tau,\tau'} \vdash_{\Sigma_{CL}} K \bullet \mathbf{v}_1 \bullet \mathbf{v}_2 = \mathbf{v}_1 : \tau \mid \tau, \tau' \in \mathbf{Typ}^{\Rightarrow}(B)\} \end{array}$$ □

Conversely, let $\mathbf{Sets}[\![-]\!]_F \in C_{\Sigma_{CL}} \cap ES_{\Sigma_{CL}}(E_{CL})$ be an interpretation of Σ_{CL}. This means that $h_{\Sigma_{CL}}(h^{-1}_{\Sigma_{CL}}(\mathbf{Sets}[\![-]\!]_F)) \in ES_{\Sigma_{CL}}(E_{CL})$. Therefore $h^{-1}_{\Sigma_{CL}}(\mathbf{Sets}[\![-]\!]_F)$ is a combinatory algebra. To summarise, the collection $\mathbf{CA}$ of combinatory algebras can be characterised by the following isomorphism:

$$h_{\Sigma_{CL}} : \mathbf{CA} \cong C_{\Sigma_{CL}} \cap ES_{\Sigma_{CL}}(E_{CL}).$$

Example 6.1.5 We consider a specification over Π_λ in $\mathbf{Sets}$. A typical equational axioms over λ-terms is β-*equality*, which is an equivalence relation over the set of well-formed lambda terms generated by the following schemes:

$$\frac{M =_\beta M' \quad N =_\beta N'}{MN =_\beta M'N'} \qquad \frac{M =_\beta N}{\lambda x^\tau \,.\, M =_\beta \lambda x^\tau \,.\, N} \qquad \frac{}{(\lambda x^\tau \,.\, M)N =_\beta M[N/x^\tau]}$$

A natural specification over the lambda terms in $\mathbf{Sets}$ is then given as follows:

$$SP_\lambda = ES_{\Pi_\lambda}(\{\Gamma \vdash M = N : \tau \mid M =_\beta N\}).$$

6.2 Translation Between Simply Typed Formal Systems

Next we formulate the activity of implementing data types in a typed formal system in terms of another typed formal system. This is given by a pair of a type translation and a term translation which respects the type translation.

Let T and T' be two sets of types and $\delta : T \to T'$ be a function. We extend δ to a functor $\mathbb{V}_\delta : \mathbb{V}_T \to \mathbb{V}_{T'}$ in an obvious way. We then define a functor $- * \delta : \mathbb{P}_{T'} \to \mathbb{P}_T$ by $F * \delta = F \circ (\mathbb{V}_\delta \times \delta)$. This preserves limits and colimits in $\mathbb{P}_{T'}$. We note that $H^F * \delta = H^{F \circ \delta}$.

Definition 6.2.1 A *translation* (δ, t) from $\Pi = (T, O)$ to $\Pi' = (T', O')$ (we write $(\delta, t) : \Pi \to \Pi'$) is a pair such that $\delta : T \to T'$ is a function which translates types in T to those in T', and $t : S_\Pi \to S_{\Pi'} * \delta$ is a morphism in $\mathbb{P}_T$ which translates well-formed terms of Π to those of Π'. In other words, for each T-context $\Gamma = \{x_1 : \tau_1, \cdots, x_n : \tau_n\}$ and type τ, $t_{(\Gamma,\tau)}$ sends a well-formed term

$$x_1 : \tau_1, \cdots, x_n : \tau_n \vdash_\Pi M : \tau$$

to

$$x_1 : \delta(\tau_1), \cdots, x_n : \delta(\tau_n) \vdash_{\Pi'} t_{(\Gamma,\tau)}(M) : \delta(\tau).$$

We define the composition of two translations $(\delta, t) : \Pi \to \Pi'$ and $(\delta', t') : \Pi' \to \Pi''$ by

$$(\delta', t') \circ (\delta, t) = (\delta' \circ \delta, (t' * \delta) \circ t).$$

A translation $(\delta, t) : \Pi \to \Pi'$ induces a mapping of categorical interpretations of Π' to those of Π. Let $\mathbb{C}[\![-]\!]_F$ be an interpretation of Π' in $\mathbb{C}$. We have a new interpretation of Π given by the composition of the following morphisms:

$$S_\Pi \xrightarrow{\quad t \quad} S_{\Pi'} * \delta \xrightarrow{\mathbb{C}[\![-]\!]_F * \delta} H^F * \delta = H^{F \circ \delta}$$

We write $\mathbb{C}[\![t(-)]\!]_{F \circ \delta}$ for this composite.

Example 6.2.2 We consider the traditional example of implementing the data type for finite sets of elements by lists of elements. This implementation is an example

of translation from the signature of finite sets of elements to the signature of lists of elements. The idea behind this implementation is to express a finite set by the list which contains each element of the set regardless of duplication and order. In the following we fix the finite set of elements C.

We first define the following basic signature $\Sigma_{\mathrm{bool}} = (T_{\mathrm{bool}}, O_{\mathrm{bool}})$ for truth values and elements:

$$\begin{aligned} T_{\mathrm{bool}} &= \{\mathrm{bool}, \mathrm{elem}\} \\ O_{\mathrm{bool}} &= \{\mathsf{tt}^{\mathrm{bool}}, \mathsf{ff}^{\mathrm{bool}}, \mathsf{or}^{\mathrm{bool},\mathrm{bool}\to\mathrm{bool}}, \mathsf{eq}^{\mathrm{elem},\mathrm{elem}\to\mathrm{bool}}\} \cup \{c^{\mathrm{elem}} \mid c \in C\}. \end{aligned}$$

This signature provides primitive data types (bool for truth values and a for elements) and operators on them. We regard these data types as *observable*, that is, programmers can directly compare their representations.

The typed first-order signature $\Sigma_{\mathrm{set}} = (T_{\mathrm{set}}, O_{\mathrm{set}})$ for finite sets of elements extends Σ_{bool} as follows:

$$\begin{aligned} T_{\mathrm{set}} &= T_{\mathrm{bool}} \cup \{\mathrm{set}\} \\ O_{\mathrm{set}} &= O_{\mathrm{bool}} \cup \{\emptyset^{\mathrm{set}}, \{-\}^{\mathrm{elem}\to\mathrm{set}}, (- \cup -)^{\mathrm{set},\mathrm{set}\to\mathrm{set}}, (- \in -)^{\mathrm{elem},\mathrm{set}\to\mathrm{bool}}\}. \end{aligned}$$

These operators are for the empty set, creating a singleton set, taking the union of two sets and the membership predicate.

The typed binding signature for lists $\Pi_{\mathrm{list}} = (T_{\mathrm{list}}, O_{\mathrm{list}})$ extends Σ_{bool} as follows:

$$\begin{aligned} T_{\mathrm{list}} &= T_{\mathrm{bool}} \cup \{\mathrm{elem}^*\} \\ O_{\mathrm{list}} &= O_{\mathrm{bool}} \cup \{\mathsf{nil}^{\mathrm{elem}^*}, \mathsf{cons}^{\mathrm{elem},\mathrm{elem}^*\to\mathrm{elem}^*}, \mathsf{iter}^{\tau,(\mathrm{elem},\tau,\tau),\mathrm{elem}^*\to\tau}\}. \end{aligned}$$

where τ ranges over T_1. The first two operators are familiar constructors for lists. The third operator is the iterator of lists, which involves variable bindings in the second position. In the framework of many-sorted algebras, the iterator on lists is usually not available unless we do some kind of trick, such as extending the language with higher-order types and combinators.

We give a translation $(\delta_{\mathrm{set}}, t_{\mathrm{set}}) : \Sigma_{\mathrm{set}} \to \Pi_{\mathrm{list}}$. First the types in Σ_{set} are translated to those in Π_{list} by the following δ_{set}:

$$\delta_{\mathrm{set}} = \mathrm{id}_{T_{\mathrm{bool}}} \cup \{\mathrm{set} \mapsto \mathrm{elem}^*\}.$$

This reflects the idea that the abstract data type for finite sets of elements is realised by lists of elements. This implementation does not touch the data types for truth values and elements.

Next, we specify a Σ_{set}-algebra structure α over $S_{\Pi_{\mathrm{list}}} * \delta_{\mathrm{set}}$ to obtain a morphism $t_{\mathrm{set}} : S_{\Sigma_{\mathrm{set}}} \to S_{\Pi_{\mathrm{list}}} * \delta_{\mathrm{set}}$ in $\mathbb{P}_T$ by initiality. Since Π_{list} is an extension of Σ_{bool}, $S_{\Pi_{\mathrm{list}}}$ already has a Σ_{bool}-algebra structure. Therefore for each operator o of arity $\tau_1, \cdots, \tau_n \to \tau$ which is only included in Σ_{set}, we specify a morphism $\alpha_o : \prod_{i=1}^n S_{\Pi_{\mathrm{list}}}(\mathbb{V}_{\delta_{\mathrm{set}}}(-), \delta_{\mathrm{set}}(\tau_i)) \to S_{\Pi_{\mathrm{list}}}(\mathbb{V}_{\delta_{\mathrm{set}}}(-), \delta_{\mathrm{set}}(\tau))$ in $[\mathbb{V}_T, \mathbf{Sets}]$ as follows (type annotations are omitted):

$$\begin{aligned}
\alpha_{\emptyset} &= \mathsf{nil} \\
\alpha_{\{-\}} &= \lambda x \,.\, \mathsf{cons}(x, \mathsf{nil}) \\
\alpha_{-\cup-} &= \lambda(x, y) \,.\, \mathsf{iter}(x, (h, t \,.\, \mathsf{cons}(h, t)), y) \\
\alpha_{-\in-} &= \lambda(x, y) \,.\, \mathsf{iter}(\mathsf{ff}, (h, t \,.\, \mathsf{or}(\mathsf{eq}(x, h), t)), y).
\end{aligned}$$

From this, $S_{\Pi_{\mathrm{list}}} * \delta_{\mathrm{set}}$ is equipped with a Σ_{set}-algebra structure. Concretely speaking, $t_{\mathrm{set}} : S_{\Sigma_{\mathrm{set}}} \to S_{\Pi_{\mathrm{list}}} * \delta_{\mathrm{set}}$ is the following recursively defined function (in the following type annotations for t_{set} are omitted for readability):

$$\begin{aligned}
t_{\mathrm{set}}(x^{\tau}) &= x^{\delta(\tau)} \\
t_{\mathrm{set}}(o(M_1, \cdots, M_n)) &= o(t_{\mathrm{set}}(M_1), \cdots, t_{\mathrm{set}}(M_n)) \\
&\qquad (o \in O_{\mathrm{bool}} \text{ has arity } \tau_1, \cdots, \tau_n \to \tau) \\
t_{\mathrm{set}}(\emptyset) &= \mathsf{nil} \\
t_{\mathrm{set}}(\{M\}) &= \mathsf{cons}(t_{\mathrm{set}}(M), \mathsf{nil}) \\
t_{\mathrm{set}}(M \cup N) &= \mathsf{iter}(t_{\mathrm{set}}(M), (h, t \,.\, \mathsf{cons}(h, t)), t_{\mathrm{set}}(N)) \\
t_{\mathrm{set}}(M \in N) &= \mathsf{iter}(\mathsf{ff}, (h, t \,.\, \mathsf{or}(\mathsf{eq}(t_{\mathrm{set}}(M), h), t)), t_{\mathrm{set}}(N))
\end{aligned}$$

From straightforward induction, for well-formed terms $\Gamma \vdash_{\Sigma_{\mathrm{set}}} M_i : \tau_i \quad (1 \leq i \leq n)$ and $x_1 : \tau_1, \cdots, x_n : \tau_n \vdash_{\Sigma_{\mathrm{set}}} M : \tau$, this term translation satisfies the following property:

$$t_{\mathrm{set}}(M)[t_{\mathrm{set}}(M_1)/x_1^{\delta(\tau_1)}, \cdots, t_{\mathrm{set}}(M_n)/x_n^{\delta(\tau_n)}] = t_{\mathrm{set}}(M[M_1/x_1^{\tau_1}, \cdots, M_n/x_n^{\tau_n}]). \square$$

Example 6.2.3 We have seen in section 4.2 that choosing a compilation method of lambda abstraction using combinators equips $S_{\Sigma_{CL}}$ with a Π_λ-algebra structure, and initiality yields a morphism $(-)_{CL} : S_{\Pi_\lambda} \to S_{\Sigma_{CL}}$ in $\mathbb{P}_T$. Therefore $(\mathrm{id}_{\mathbf{Typ}^{\Rightarrow}(B)}, (-)_{CL}) : \Pi_\lambda \to \Sigma_{CL}$ is a translation. □

6.3 Pre-Logical Data Refinement

We introduce the concept of *pre-logical data refinement* for typed formal systems. This is a direct generalisation of [HLST00].

Definition 6.3.1 Let SP and SP' be specifications over Π and Π' in a Cartesian category $\mathbb{C}$ respectively. A translation $(\delta, t) : \Pi \to \Pi'$ is a *pre-logical data refinement* from SP to SP' with respect to OBS if for any $\mathbb{C}[\![-]\!]_F \in SP'$, there exists $\mathbb{C}[\![-]\!]_G \in SP$ such that $\mathbb{C}[\![-]\!]_G \sim_{\mathrm{OBS}} \mathbb{C}[\![t(-)]\!]_{F\circ\delta}$. We sometimes write such a pre-logical refinement in the following compact notation:

$$SP \xrightarrow[\mathrm{OBS}]{(\delta,t)} SP'$$

In this definition, observational pre-logical relations play a role of giving a witness for behavioural equivalence. From theorem 5.1.7, the existence of an observational pre-logical relation implies

$$\mathbb{C}[\![-]\!]_G \equiv_{\mathrm{OBS}} \mathbb{C}[\![t(-)]\!]_{F\circ\delta}\,.$$

This means that the realisation of Π over $\mathbb{C}[\![-]\!]_F \in SP'$ via translation (δ, t) conforms to the specification SP up to behavioural equivalence, as we discussed in the introduction of this chapter.

Example 6.3.2 We introduce specifications SP_{set} and SP_{list} over Σ_{set} and Π_{list} respectively, then show that the translation $(\delta_{\mathrm{set}}, t_{\mathrm{set}})$ in example 6.2.2 is a pre-logical data refinement from SP_{set} to SP_{list} with respect to $\{\mathrm{bool}, \mathrm{elem}\}$:

$$SP_{\mathrm{set}} \xrightarrow[\{\mathrm{bool},\mathrm{elem}\}]{(\delta_{\mathrm{set}},t_{\mathrm{set}})} SP_{\mathrm{list}}$$

First, we give a specification SP_{bool} over Σ_{bool}. This specification allows only one concrete implementation of Σ_{bool}, which is the standard semantics of $S_{\Sigma_{\mathrm{bool}}}$ in the following many-sorted Σ_{bool}-algebra $\mathcal{B}$ (type annotations are omitted):

$$
\begin{aligned}
&B(\mathrm{bool}) = \{\mathtt{tt}, \mathtt{ff}\}, \quad B(\mathrm{elem}) = C \\
&\mathsf{tt}_{\mathcal{B}} = \mathtt{tt}, \quad \mathsf{ff}_{\mathcal{B}} = \mathtt{ff}, \quad c_{\mathcal{B}} = c \quad (c \in C), \\
&\mathsf{or}_{\mathcal{B}}(x, y) = \mathtt{tt} \iff x = \mathtt{tt} \vee y = \mathtt{ff}, \\
&\mathsf{eq}_{\mathcal{B}}(x, y) = \mathtt{tt} \iff x = y.
\end{aligned}
$$

We define SP_{bool} by the following singleton collection:

$$SP_{\mathrm{bool}} = \{\mathcal{B}[\![-]\!]\}.$$

Next we give a specification SP_{list} over Π_{list}. This specification also allows only one concrete implementation of lists of elements by finite sequence of elements. In the following, we denote the inclusion of terms by $i_{\mathrm{list}} : S_{\Sigma_{\mathrm{bool}}} \to S_{\Pi_{\mathrm{list}}}$ (which is induced by the signature inclusion $\Sigma_{\mathrm{bool}} \hookrightarrow \Pi_{\mathrm{list}}$).

$$
\begin{aligned}
\mathbf{Sets}[\![-]\!]_F \in SP_{\mathrm{set}} \quad \iff \quad & \mathbf{Sets}[\![-]\!]_F \in C_{\Pi_{\mathrm{list}}} \\
\wedge \quad & \mathbf{Sets}[\![-]\!]_F \circ i_{\mathrm{list}} \in SP_{\mathrm{bool}} \\
\wedge \quad & F(\mathrm{elem}^*) = C^* \\
\wedge \quad & \mathbf{Sets}[\![\mathsf{nil}]\!]_F = \epsilon \\
\wedge \quad & \mathbf{Sets}[\![\mathsf{cons}(\mathbf{v}_1, \mathbf{v}_2)]\!]_F(x, y) = xy \quad \text{(concatenation)} \\
\wedge \quad & \mathbf{Sets}[\![\mathsf{iter}(\mathbf{v}_1, (h, t\,.\,M), \mathsf{nil})]\!]_F(x, \overrightarrow{z}) = x \\
\wedge \quad & \mathbf{Sets}[\![\mathsf{iter}(\mathbf{v}_1, (h, t\,.\,M), \mathsf{cons}(\mathbf{v}_2, \mathbf{v}_3))]\!]_F(x, c, w, \overrightarrow{z}) = \\
& \mathbf{Sets}[\![M[\mathbf{v}_1/h, \mathbf{v}_2/t]]\!]_F(c, v, \overrightarrow{z}) \\
& \text{where } \; v = \mathbf{Sets}[\![\mathsf{iter}(\mathbf{v}_1, (h, t\,.\,M), \mathbf{v}_2)]\!]_F(x, w, \overrightarrow{z})
\end{aligned}
$$

We give a specification SP_{set} over Σ_{set}. We first define a set of the equational

axioms over Σ_{set}:

$$\begin{aligned}
E_{\text{set}} \quad = \quad & \{x : \text{elem} \vdash_{\Sigma_{\text{set}}} x \in \emptyset = \mathsf{ff} : \text{bool}, \\
& x : \text{elem}, y : \text{elem} \vdash_{\Sigma_{\text{set}}} \mathsf{eq}(x, y) = x \in \{y\} : \text{bool}, \\
& z : \text{elem}, x : \text{set}, y : \text{set} \vdash_{\Sigma_{\text{set}}} \mathsf{or}(z \in x, z \in y) = z \in (x \cup y) : \text{bool} \\
& x : \text{set} \vdash_{\Sigma_{\text{set}}} x \cup \emptyset = x : \text{set} \\
& x : \text{set} \vdash_{\Sigma_{\text{set}}} x \cup x = x : \text{set} \\
& x : \text{set}, y : \text{set} \vdash_{\Sigma_{\text{set}}} x \cup y = y \cup x : \text{set} \\
& x : \text{set}, y : \text{set}, z : \text{set} \vdash_{\Sigma_{\text{set}}} (x \cup y) \cup z = x \cup (y \cup z) : \text{set}\}.
\end{aligned}$$

We then define SP_{set} by:

$$SP_{\text{set}} = C_{\Sigma_{\text{set}}} \cap ES_{\Sigma_{\text{set}}}(E_{\text{set}}).$$

Like example 6.1.4, this specification is isomorphic to the class of many-sorted Σ_{set} algebras satisfying equational axioms in E_{set}.

We show that the transformation $(\delta_{\text{set}}, t_{\text{set}})$ in example 6.2.2 is a pre-logical data refinement from SP_{set} to SP_{list} with respect to $\{\text{bool}, \text{elem}\}$.

We first define an equivalence relation $\approx$ over C^* by

$$x \approx y \iff \forall z \in C \,.\, z \text{ occurs in } x \iff z \text{ occurs in } y.$$

This equivalence relation identifies two list representations of sets when they contain the same elements regardless of order and number of occurrences.

We then define a many-sorted Σ_{set} algebra $\mathcal{G} = (\{G\tau\}_{\tau \in T_{\text{set}}}, \{o_{\mathcal{G}}\}_{o \in O_{\text{set}}})$ by

$$\begin{aligned}
& G(\text{set}) = C^*/\!\approx, \quad G(\text{elem}) = C, \quad G(\text{bool}) = \{\mathsf{tt}, \mathsf{ff}\} \\
& c_{\mathcal{G}} = c_{\mathcal{B}} \quad (c \in \Sigma_{\text{bool}}) \\
& \emptyset_{\mathcal{G}} = [\epsilon]_{\approx}, \quad \{a\}_{\mathcal{G}} = [a]_{\approx} \\
& [x]_{\approx} \cup_{\mathcal{G}} [y]_{\approx} = [xy]_{\approx} \\
& x \in_{\mathcal{G}} [y]_{\approx} = \mathsf{tt} \iff x \text{ occurs in } y.
\end{aligned}$$

The well-definedness of $\in_{\mathcal{G}}$ is clear by the definition of $\approx$. We check the well-definedness

of $\cup_{\mathcal{G}}$. Suppose $x \approx x'$ and $y \approx y'$. Then

$$\begin{array}{rcl} z \in C \text{ occurs in } xy & \Longleftrightarrow & (z \text{ occurs in } x) \vee (z \text{ occurs in } y) \\ & \Longleftrightarrow & (z \text{ occurs in } x') \vee (z \text{ occurs in } y') \\ & \Longleftrightarrow & z \text{ occurs in } x'y'. \end{array}$$

Therefore $xy \approx x'y'$. The standard semantics $\mathcal{G}[\![-]\!]$ is included in the specification SP_{set}. We leave this to the exercise to the readers.

Let $\mathbf{Sets}[\![-]\!]_F \in SP_{\text{list}}$. We establish an observational pre-logical relation between $\mathbf{Sets}[\![t_{\text{set}}(-)]\!]_{F \circ \delta_{\text{set}}}$ and $\mathcal{G}[\![-]\!]$. We define a relation $R \subseteq_{-\times-} \langle F \circ \delta_{\text{set}}, G \rangle$ by

$$R(\text{set}) = \{(x, y) \in C^* \times C^* / \approx \mid y \in x\}, \quad R(\text{elem}) = \text{Id}_C, \quad R(\text{bool}) = \text{Id}_{\{\text{tt},\text{ff}\}}.$$

It is observational by definition of Ra and $R\text{bool}$. We show that R is pre-logical for Σ_{set} along $\mathbf{Sets}[\![t_{\text{set}}(-)]\!]_{F \circ \delta_{\text{set}}} \times \mathcal{G}[\![-]\!]$.

In example 6.2.2, we checked that

$$t_{\text{set}}(M)[t_{\text{set}}(M_1)/x_1^{\delta_{\text{set}}(\tau_1)}, \cdots, t_{\text{set}}(M_n)/x_n^{\delta_{\text{set}}(\tau_n)}] \quad = \quad t_{\text{set}}(M[M_1/x_1^{\tau_1}, \cdots, M_n/x_n^{\tau_n}]).$$

This implies that for $\mathbf{Sets}[\![-]\!]_F \in SP_{\text{list}}$, $\mathbf{Sets}[\![t_{\text{set}}(-)]\!]_{F \circ \delta_{\text{set}}}$ satisfies:

1. $\mathbf{Sets}[\![t_{\text{set}}(x^\tau)]\!]_{F \circ \delta_{\text{set}}} = \mathbf{Sets}[\![x^{\delta_{\text{set}}(\tau)}]\!]_F = \pi_{\gamma(x)}$ and

2. the semantic substitution lemma, because

$$\begin{array}{rcl} & & \mathbf{Sets}[\![t_{\text{set}}(M)]\!]_{F \circ \delta_{\text{set}}} \circ \langle \overrightarrow{\mathbf{Sets}[\![t_{\text{set}}(N)]\!]_{F \circ \delta_{\text{set}}}} \rangle \\ & = & (\mathbf{Sets}[\![t_{\text{set}}(M)]\!]_F \circ \langle \overrightarrow{\mathbf{Sets}[\![t_{\text{set}}(N)]\!]_F} \rangle) * \delta_{\text{set}} \\ & = & \mathbf{Sets}[\![t_{\text{set}}(M)[\overrightarrow{t_{\text{set}}(N)/x^{\delta_{\text{set}}(\tau)}}]]\!]_F \\ & = & \mathbf{Sets}[\![t_{\text{set}}(M[\overrightarrow{N/x^\tau}])]\!]_F. \end{array}$$

Thus $\mathbf{Sets}[\![t_{\text{set}}(-)]\!]_{F \circ \delta_{\text{set}}} \in C_{\Sigma_{\text{set}}}$, and from example 6.1.3 we have a many-sorted Σ_{set}-algebra $\mathcal{S}$ such that $\mathbf{Sets}[\![t_{\text{set}}(-)]\!]_{F \circ \delta_{\text{set}}} = \mathcal{S}[\![-]\!]$. As we have seen in section 4.1, a pre-logical predicate for typed first-order signatures along the standard semantics (in this context the standard semantics of the product algebra $\mathcal{S} \times \mathcal{G}$) are just subalgebras. Thus it is sufficient to show that the following holds:

1. $(\mathbf{Sets}[\![t_{\mathrm{set}}(\emptyset)]\!]_{F\circ\delta_{\mathrm{set}}}, \mathcal{G}[\![\emptyset]\!]) \in R\mathrm{set}$,

2. for any $(x, [x]_\approx) \in R\mathrm{set}$ and $(y, [y]_\approx) \in R\mathrm{set}$,

$$(\mathbf{Sets}[\![t_{\mathrm{set}}(\mathbf{v}_1 \cup \mathbf{v}_2)]\!]_{F\circ\delta_{\mathrm{set}}}(x, y), \mathcal{G}[\![\mathbf{v}_1 \cup \mathbf{v}_2]\!]([x]_\approx, [y]_\approx)) \in R\mathrm{set},$$

3. for any $x \in C$ and $(y, [y]_\approx) \in R\mathrm{set}$,

$$\mathbf{Sets}[\![t_{\mathrm{set}}(\mathbf{v}_1 \in \mathbf{v}_2)]\!]_{F\circ\delta_{\mathrm{set}}}(x, y) = \mathcal{G}[\![\mathbf{v}_1 \in \mathbf{v}_2]\!](x, [y]_\approx),$$

4. for any $x \in C$,

$$(\mathbf{Sets}[\![t_{\mathrm{set}}(\{\mathbf{v}_1\})]\!]_{F\circ\delta_{\mathrm{set}}}(x), \mathcal{G}[\![\{\mathbf{v}_1\}]\!](x)) \in R\mathrm{set}.$$

1 and 4 are clear. 2 and 3 follow from the fact that

$$\mathbf{Sets}[\![t_{\mathrm{set}}(\mathbf{v}_1 \cup \mathbf{v}_2)]\!]_{F\circ\delta_{\mathrm{set}}}(x, x') = xx'$$
$$\mathbf{Sets}[\![t_{\mathrm{set}}(\mathbf{v}_1 \in \mathbf{v}_2)]\!]_{F\circ\delta_{\mathrm{set}}}(x, w) = \mathrm{tt} \iff x \text{ occurs in } w.$$ □

Example 6.3.3 We show that the translation $(\mathrm{id}_{\mathbf{Typ}^{\Rightarrow}(B)}, (-)_{CL}) : \Pi_\lambda \to \Sigma_{CL}$ in example 6.2.3 is a pre-logical data refinement from SP_λ to $C_{\Sigma_{CL}} \cap ES_{\Sigma_{CL}}(E_{CL})$ with respect to B.

Recall that the specification $C_{\Sigma_{CL}} \cap ES_{\Sigma_{CL}}(E_{CL})$ is isomorphic to the class of combinatory algebras $\mathbf{CA}$ (see example 6.1.4). Thus we take a combinatory algebra $\mathcal{U}$ and consider its standard interpretation $\mathcal{U}[\![-]\!]$ of Σ_{CL}. We show that there exists an interpretation $\mathbf{Sets}[\![-]\!]_G$ of Π_λ in SP_λ such that

$$\mathbf{Sets}[\![-]\!]_G \equiv_B \mathcal{U}[\![(-)_{CL}]\!].$$

We note that $\mathcal{U}[\![(-)_{CL}]\!] \notin SP_\lambda$ in general; when $\mathcal{U}$ is the term combinatory algebra $\mathcal{U}_0$ (see section 4.2), $\lambda x\ .\ (\lambda y\ .\ y)x =_\beta \lambda y\ .\ y$ but $\mathcal{U}_0[\![\lambda x\ .\ (\lambda y\ .\ y)x]\!] = [S(K(SKK))(SKK)]_w \neq [SKK]_w = \mathcal{U}_0[\![\lambda y\ .\ y]\!]$.

We construct an interpretation by the *extensional collapse* of $\mathcal{U}$, taking the quotient of the carrier set $U\tau$ for each type $\tau \in \mathbf{Typ}^{\Rightarrow}(B)$ by the logical PER R constructed

over the identity relations for base types. Formally, R is a $\mathbf{Typ}^{\Rightarrow}(B)$-indexed family of relations $R \subseteq_{-\times-} \langle U, U \rangle$ defined by induction on types:

$$\begin{aligned} Rb &= \mathrm{Id}_{Ub} \\ R(\tau \Rightarrow \tau') &= \{(f,g) \in (U(\tau \Rightarrow \tau'))^2 \mid \forall (x,y) \in R\tau \,.\, (f \bullet x, g \bullet y) \in R\tau'\}. \end{aligned}$$

It is routine to show that R is a pre-logical PER for Π_λ along $\mathcal{U}[\![(-)_{CL}]\!] \times \mathcal{U}[\![(-)_{CL}]\!]$ (see e.g. exercise 4.5.6, [AC98]). For an element $x \in |R\tau|$, we write $[x]_R$ for the equivalence class of x by R. That R is pre-logical implies that the following is a well-defined function for each well-formed term $\Gamma \vdash_{\Pi_\lambda} M : \tau$:

$$[\mathcal{U}[\![M_{CL}]\!]]_R([x_1]_R, \cdots, [x_n]_R) = [\mathcal{U}[\![M_{CL}]\!](x_1, \cdots, x_n)]_R.$$

Thus we take $([R], [\mathcal{U}[\![(-)_{CL}]\!]]_R)$ for the interpretation of Π_λ in question.

We show that $[\mathcal{U}[\![(-)_{CL}]\!]]_R \in SP_\lambda$, that is, for each well-formed term $x_1 : \tau_1, \cdots, x_n : \tau_n \vdash_{\Pi_\lambda} M_0, N_0 : \tau$, $M_0 =_\beta N_0$ implies

$$\forall (x_1, y_1) \in R\tau_1, \cdots, (x_n, y_n) \in R\tau_n \,.\, (\mathcal{U}[\![(M_0)_{CL}]\!]\overrightarrow{x}, \mathcal{U}[\![(N_0)_{CL}]\!]\overrightarrow{y}) \in R\tau.$$

We show this by induction on the derivation of $M_0 =_\beta N_0$. Let $(x_1, y_1) \in R\tau_1, \cdots, (x_n, y_n) \in R\tau_n$. The interesting cases are the following.

- $M_0 = \lambda x^\tau \,.\, M, N_0 = \lambda x^\tau \,.\, N$ and $M_0 =_\beta N_0$ is derived by the following rule:

$$\frac{M =_\beta N}{\lambda x^\tau \,.\, M =_\beta \lambda x^\tau \,.\, N}$$

  In combinatory logic, the following holds for any $x' \in U\tau$:

$$\mathcal{U}[\![M_{CL}]\!](\overrightarrow{x}, x') = (\mathcal{U}[\![(\lambda x^\tau \,.\, M)_{CL}]\!]\overrightarrow{x}) \bullet x'$$

  (see [Bar84]). Therefore from IH, we have

$$\begin{aligned} & \forall (x', y') \in R\tau \,.\, (\mathcal{U}[\![M_{CL}]\!](\overrightarrow{x}, x'), \mathcal{U}[\![N_{CL}]\!](\overrightarrow{y}, y') \in R\tau' \\ \iff & \forall (x', y') \in R\tau \,.\, (\mathcal{U}[\![(\lambda x^\tau \,.\, M)_{CL}]\!]\overrightarrow{x} \bullet x', \mathcal{U}[\![(\lambda x^\tau \,.\, N)_{CL}]\!]\overrightarrow{y} \bullet y') \in R\tau' \\ \iff & (\mathcal{U}[\![(\lambda x^\tau \,.\, M)_{CL}]\!]\overrightarrow{x}, \mathcal{U}[\![(\lambda x^\tau \,.\, N)_{CL}]\!]\overrightarrow{y}) \in R(\tau \Rightarrow \tau'). \end{aligned}$$

- $M_0 = (\lambda x^\tau \,.\, M)N, N_0 = M[N/x]$ and $M_0 =_\beta N_0$ is derived by the following axiom:

$$\frac{}{(\lambda x^\tau \,.\, M)N =_\beta M[N/x]}$$

  This is clear since the weak equivalence always validates $((\lambda x^\tau \,.\, M)N)_{CL} =_w (M[N/x^\tau])_{CL}$. Therefore $(\mathcal{U}[\![((\lambda x^\tau \,.\, M)N)_{CL}]\!]\overrightarrow{x}, \mathcal{U}[\![(M[N/x^\tau])_{CL}]\!]\overrightarrow{y}) \in R^\tau$.

We show $[\mathcal{U}[\![(-)_{CL}]\!]]_R \equiv_B \mathcal{U}[\![(-)_{CL}]\!]$. We give a witnessing observational pre-logical relation $S \subseteq \langle [R], U \rangle$ as a witness as follows:

$$S\tau = \{(a, b) \in [R\tau]_R \times U\tau \mid b \in a\}.$$

First Sb is a total bijective relation for each $b \in B$, as Rb is the identity relation over Ub. Next we show that S satisfies the basic lemma. Let $\Gamma \vdash_{\Pi_\lambda} M : \tau$ be a well-formed term and $((a_1, \cdots, a_n), (b_1, \cdots, b_n)) \in S^*\Gamma$. By definition of S, we have $[b_i]_R = a_i$ for $1 \leq i \leq n$. Thus

$$\begin{aligned} & ([\mathcal{U}[\![(-)_{CL}]\!]]_R([b_1]_R, \cdots, [b_n]_R), \mathcal{U}[\![(-)_{CL}]\!](b_1, \cdots, b_n)) \\ = \;& ([\mathcal{U}[\![(-)_{CL}]\!](b_1, \cdots, b_n)]_R, \mathcal{U}[\![(-)_{CL}]\!](b_1, \cdots, b_n)) \in S\tau. \end{aligned}$$

□

6.4 Stability and Composition of Data Refinement

It is natural to expect that data refinements compose, that is, for two data refinements:

$$SP \xrightarrow[\mathrm{OBS}]{(\delta, t)} SP' \qquad SP' \xrightarrow[\mathrm{OBS}']{(\delta', t')} SP''$$

we would expect that the composition of the above translation yields a pre-logical data refinement:

$$SP \xrightarrow[\mathrm{OBS}]{(\delta', t') \circ (\delta, t)} SP''.$$

To show this, it is sufficient to know that binary pre-logical relations are preserved by translations. Schoett called this sufficient condition *stability* [Sch85, Sch90].

We impose a constraint on the sets of observable types: the translation never maps observable types in the source specification to abstract types in the target specification.

Proposition 6.4.1 *Let (δ, t) be a translation from $\Pi = (T, O)$ to $\Pi' = (T', O')$. Let $\mathrm{OBS} \subseteq T, \mathrm{OBS}' \subseteq T'$ be sets of observable types and assume that $\delta(\sigma) \in \mathrm{OBS}'$ holds for each $\sigma \in \mathrm{OBS}$.*

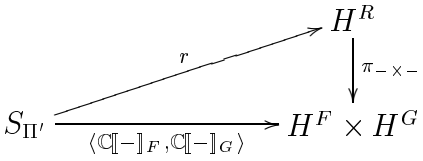
Then for any categorical interpretations $\mathbb{C}[\![-]\!]_F$ and $\mathbb{C}[\![-]\!]_G$ of Π' in a Cartesian category $\mathbb{C}$,

$$\mathbb{C}[\![-]\!]_F \sim_{\mathrm{OBS}'} \mathbb{C}[\![-]\!]_G$$

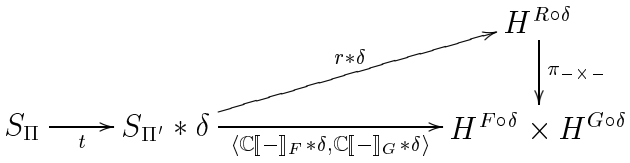
implies

$$\mathbb{C}[\![t(-)]\!]_{F\circ\delta} \sim_{\mathrm{OBS}} \mathbb{C}[\![t(-)]\!]_{G\circ\delta}.$$

PROOF From theorem 3.6.8, the existence of an observational pre-logical relation, say R, is equivalent to the existence of a morphism r making the following diagram in $\mathbb{P}_{T'}$ commute:



The functor $- * \delta$ sends this triangle to the following one in $\mathbb{P}_T$. The term translation morphism $t : S_\Pi \to S_{\Pi'} * \delta$ is also added into the diagram.



This implies that $R \circ \delta \subseteq_{-\times-} \langle F \circ \delta, G \circ \delta \rangle$ is a binary pre-logical relation for Π along $(\mathbb{C}[\![t(-)]\!]_{F\circ\delta}) \times (\mathbb{C}[\![t(-)]\!]_{G\circ\delta})$. The assumption on δ immediately implies that this binary pre-logical relation is total bijective for each $\sigma \in \mathrm{OBS}$. Hence $\mathbb{C}[\![t(-)]\!]_{F\circ\delta} \sim_{\mathrm{OBS}}$ $\mathbb{C}[\![t(-)]\!]_{G\circ\delta}$. ∎

Theorem 6.4.2 *Let SP, SP', SP'' be specifications of typed binding signatures $\Pi = (T, O), \Pi' = (T', O'), \Pi'' = (T'', O'')$ respectively. Suppose we have pre-logical data refinements*

$$SP \xrightarrow[\mathrm{OBS}]{(\delta,t)} SP' \qquad SP' \xrightarrow[\mathrm{OBS}']{(\delta',t')} SP''$$

such that for any $\sigma \in \mathrm{OBS}$, $\delta(\sigma) \in \mathrm{OBS}'$ *holds. Then we have the following pre-logical data refinement:*

$$SP \xrightarrow[\mathrm{OBS}]{(\delta',t')\circ(\delta,t)} SP''.$$

PROOF Let $\mathbb{C}[\![-]\!]_F \in SP''$. Since (δ', t') and (δ, t) are pre-logical data refinements, there exists interpretations $\mathbb{C}[\![-]\!]_{G_1} \in SP'$ and $\mathbb{C}[\![-]\!]_{G_2} \in SP$ such that

$$\begin{aligned} \mathbb{C}[\![-]\!]_{G_1} &\sim_{\mathrm{OBS}'} \mathbb{C}[\![t'(-)]\!]_{F\circ\delta'} \\ \mathbb{C}[\![-]\!]_{G_2} &\sim_{\mathrm{OBS}} \mathbb{C}[\![t(-)]\!]_{G_1\circ\delta}. \end{aligned} \tag{6.1}$$

We apply proposition 6.4.1 to equation 6.1 and obtain:

$$\mathbb{C}[\![t(-)]\!]_{G_1\circ\delta} \sim_{\mathrm{OBS}} \mathbb{C}[\![(t * \delta) \circ t'(-)]\!]_{F\circ\delta'\circ\delta}.$$

Since $\sim_{\mathrm{OBS}}$ is a transitive relation, we have

$$\mathbb{C}[\![-]\!]_{G_2} \sim_{\mathrm{OBS}} \mathbb{C}[\![(t * \delta) \circ t'(-)]\!]_{F\circ\delta'\circ\delta}.$$

Hence $(\delta', t') \circ (\delta, t)$ is a pre-logical data refinement from SP to SP'' with respect to OBS.

6.5 Related Work

The study of data refinement emerged in the 1970s. The pioneering work of this area is [Hoa72], where Hoare used "abstraction function" to relate abstract representations and concrete representations of data types. Later, Schoett and others noticed that functions are not enough, and a certain kind of relations are more appropriate ([Sho83, Sch85, Nip86]; c.f. [Mil71]). Schoett considered such relations, called *correspondences*, in partial algebras. When we restrict our attention to ordinary many-sorted algebras, a correspondences is just an observational pre-logical relation between the standard semantics of many-sorted algebras (see section 4.1). In this sense our generalisation is a natural extension of correspondences. A similar approach is taken in the simply typed lambda calculus to show the representation independence result [Mit86, Ten94].

In the field of algebraic specifications, data refinements are decomposed into two concepts: *constructors* on models (translations in this chapter is an example of constructors) and behavioural equivalence. In [ST88], Sannella and Tarlecki formulated data refinements in this way, and called them *abstractor implementations* or *behavioural implementations*. In this formulation, Schoett's stability is a crucial property for data refinements to compose. Another approach to achieve data abstraction, which is not covered in this thesis, is to take the quotient of an interpretation by the indistinguishability relation. This approach is compared to the former one in [BHW95].

The concept of abstractor implementation is applied to data refinement in the simply typed lambda calculus. Honsell et al. considered data refinement for the simply typed lambda calculus and its Henkin models in [HLST00], where they used binary pre-logical relations instead of binary logical relations as in [Ten94]. The material of this chapter is a direct adaptation of their work to simply typed formal systems and their categorical models. This thesis combines their idea and our generalisation to discuss pre-logical refinements in a wider context.

6.6 Conclusion

We have seen an application of binary pre-logical relations to data refinement. Pre-logical relations are used to give witnesses for behavioural equivalence, which is the key to achieving data abstraction. This idea is explained with two examples, one being the classical example of a refinement of the abstract data type of finite sets of elements by means of lists of elements. The second is to represent the lambda calculus by combinatory algebras through lambda-to-CL translations. In both cases type systems involving data refinements have variable bindings; iterators on lists in the former case and lambda abstraction in the latter case involve variable bindings. This is not covered in the traditional algebraic framework. We then showed that pre-logical data refinements are closed under composition.

Chapter 7

Conclusions

A generalisation of pre-logical predicates and their applications have been presented. Pre-logical predicates are reformulated in a categorical setting as a syntactic characterisation of the basic lemma. Our generalisation is strictly wider than the original formulation of pre-logical predicates [HS02] in the following sense:

1. The system is extended to simply typed formal systems, which includes the simply typed lambda calculus with various type constructors, many-sorted algebras and first-order logics.

2. The semantics is extended to categorical interpretations, which subsume set-theoretic interpretations of the lambda calculus.

To give a systematic explanation of our generalisation, we used the semantics of typed formal systems in a presheaf category and the characterisation of the well-formed terms as an initial algebra. Two desirable properties are preserved: one is the equivalence with the basic lemma and the other is the composability of binary pre-logical relations.

To test this generalisation, we have instantiated it to various type systems. In the case of many-sorted algebras, pre-logical predicates for the standard interpretations coincide with subalgebras. We then compared pre-logical predicates for combinatory algebras and the simply typed lambda calculus. In first-order logic, a well-known condition called Tarski's criterion turns out to be an essential condition of the inclusion relation to be pre-logical.

Pre-logical relations are then applied to characterise behavioural equivalence. We show that the indistinguishability relation is a pre-logical partial equivalence relation, and factorisability[BHW95] holds in this generalised setting. We then applied our generalisation of pre-logical relations and the characterisation result of behavioural equivalence to formulate data refinement in simply typed formal systems. By composability plus stability of observational pre-logical relations, pre-logical refinements are closed under composition. We showed two examples of pre-logical refinements: one is a traditional refinement of sets of alphabets by means of lists of alphabets, and the other is a refinement of lambda calculus by means of combinatory algebras.

7.1 Future Directions

7.1.1 Beyond Simply Typed Formal Systems

So far we have seen pre-logical predicates for simply typed formal systems, whose contexts are modeled by finite products.

One direction of extending our work is to consider type systems with more elaborate structures, such as type dependency [ML75, Hof97], type variables and polymorphism [Gir72, Rey83], linear contexts [Gir87], etc.

Pre-logical predicates are the syntactic characterisation of the notion of submodels. This is the right principle to derive pre-logical predicates for extended type systems with preserving the equivalence to the basic lemma. To establish a formal statement of this equivalence, it is desirable to have an initial algebra semantics for extended type systems; here we used a presheaf category to obtain such a semantics. For the system with linear contexts and bindings, see [Tan00].

The closure under composition of binary pre-logical relations for extended type systems is not obvious. The author obtained a counterexample of the closure property for Leiss's notion of pre-logical predicates for System $\lambda\underline{\omega}$, which is a weaker calculus than $F\omega$ (this example is in appendix A). This suggests that the problem arises due to relations between types, rather than impredicativity. The same problem exists in $F\omega$ too, as indicated by Leiss in [Lei01]. Leiss pointed out that binary pre-logical relations for $F\omega$ are closed under relational composition if relations between types are restricted

to functional relations. Our counterexample in appendix A violates this restriction. We do not know if this restriction can be relaxed.

One interesting question is whether it is possible to formulate the concept of parametricity [MR92] using binary pre-logical relations instead of binary logical relations. This question was addressed in [HKS03]. This is a potential application to resolving the problem of expressing data refinement involving higher-order constants in the logic of parametricity [Han01].

7.1.2 Applications of Pre-Logical Refinements

Applications to Program Transformations

A potential application of pre-logical refinements is the verification of *program transformations*.

The aim of program transformations is to remove redundant computations and intermediate data structures by analysing and rewriting input programs [BD77, Wad90, GLJ93]. An example of program transformation is *deforestation* [Wad90]. Deforestation is an algorithm which takes a program of the form $f(g(x))$ and generates an equivalent program $h(x)$ which does not construct the intermediate data structure passed from g to f. We say such a transformation is correct if running a program without transformation and with transformation show the same behaviour (we do not want to use any optimiser which changes behaviour of programs).

One may notice that program transformation and data refinement are very similar. Indeed, both are activities to implement a language via translations of types and terms to another language [1]. Their correctness is that the implemented language environment shows the same behaviour as the ideal language environment.

However, there are two reasons why the traditional algebraic framework for data refinement is not directly applicable to the field of program transformations. First, program transformations concentrate on programs which have a particular shape (like $f(g(x))$ for deforestation) and perform detailed analysis, while language translations considered in the algebraic framework are modeled by *signature morphisms* or *derived*

[1] Of course there are cases in which the source and target of a translation are the same language.

signature morphisms, which merely replace each operator in the source signature with an operator or a term of the target signature. They induce a transformation between languages, but do not perform any detailed analysis. Second, the target language of program transformations usually involves variable bindings, which are missing in the algebraic framework.

These two gaps naturally disappear in the framework of pre-logical refinements in chapter 6. First, we discussed the most general form of translations between languages, which are just mappings of well-formed terms respecting types. Program transformations are thus subsumed by translations. Second, our framework of simply typed formal systems covers any languages with variable bindings.

Using the concept of behavioural equivalence, we could prove the correctness of program transformations with respect to behaviour. This viewpoint is emerging for the verification of program transformations [Nie00, Nis03], and we believe that pre-logical refinements provide a bridge between the world of algebraic specifications and the world of program transformations.

Applications to Process Calculi

Process calculi such as CCS and pi-calculus fit within the framework of simply typed formal systems. This suggests another potential application area of pre-logical refinements. When we regard process calculi as a foundation of programming languages, it is natural to extend them with abstract data types and operators that satisfy specifications. Spi-calculus [AG99] is an example of such an extension; it is a pi-calculus extended with encryption and decryption operators satisfying certain equations. We can then think of implementing these operators with other basic operators and discuss the correctness of the implementation. We believe that pre-logical refinements provides a right framework to discuss such a correctness.

Appendix A

A Counterexample of Composability of Pre-logical Relations in $\lambda\underline{\omega}$

A.1 Introduction

Pre-logical relations were first proposed by Honsell and Sannella [HS02], and are a generalised notion of logical relations. In [HS02] various characterisation of pre-logical relations are studied. They showed that pre-logical binary relations between models of the lambda calculus are composable.

In [Lei01], Leiss extended the notion of pre-logical relations to system $F\omega$, the omega-order polymorphic lambda calculus. He pointed out that pre-logical binary relations between models of system $F\omega$ are not composable, although the concrete situation was not explained very well in [Lei01].

The aim of this appendix is to give a concrete counterexample of composability in a $\lambda\underline{\omega}$ calculus, which is a subcalculus of $F\omega$ [Bar91]. The counterexample relies on a common definition of pre-logical relations over models of system $F\omega$ and those of system $\lambda\underline{\omega}$, thus we can construct a similar counterexample in system $F\omega$.

A.2 System $\lambda\underline{\omega}$

Roughly speaking, system $\lambda\underline{\omega}$ consists of two layers of the simply typed lambda calculus, one for terms and the other for types. However, unlike $F\omega$, there is no type-dependent terms. In this paper, we consider a system $\lambda\underline{\omega}$ with one base type b, no term constants and a fixed set-theoretic semantics of the system. The counterexample is built on this model in the next section.

A.2.1 The Syntax and Type System of $\lambda\underline{\omega}$

- *Raw kinds, types and terms* are defined by the following BNF:

$$\begin{array}{rcl} \mathbf{K} \ni \kappa & ::= & T \mid \kappa \Rrightarrow \kappa \\ \mathbf{T} \ni \tau & ::= & \alpha \mid b \mid \Rightarrow \mid \lambda\alpha^{\kappa}.\tau \mid \tau\tau \\ \mathbf{\Lambda} \ni M & ::= & x \mid \lambda x^{\tau}.M \mid MM \end{array}$$

  where α, x ranges over the set of type variables and variables respectively. We identify α-equivalent raw types and terms, and adopt Barendregt's variable convention [Bar84]. We write $M[N/x]$ and $\tau[\tau'/\alpha]$ for the results of substituting N (τ') for the free variable of x (α) in M (τ) respectively. We treat $\Rightarrow$ as an infix operator.

- A *kind context* (ranged by meta variable Δ) is a mapping from a subset of type variables to $\mathbf{K}$. Similarly, a *type context* (ranged by meta variable Γ) is a mapping from a subset of term variables to $\mathbf{T}$.

- We say that $\Delta \vdash \tau : \kappa$ is a *well-formed type* if it is derived only from the following rules:

$$\frac{\alpha \in \mathrm{dom}(\Delta)}{\Delta \vdash \alpha : \Delta(\alpha)} \quad \frac{}{\Delta \vdash b : T} \quad \frac{}{\Delta \vdash \Rightarrow : T \Rrightarrow T \Rrightarrow T}$$

$$\frac{\Delta, \alpha : \kappa \vdash \tau : \kappa'}{\Delta \vdash \lambda\alpha^{\kappa}.\tau : \kappa \Rrightarrow \kappa'} \quad \frac{\Delta \vdash \tau : \kappa \Rrightarrow \kappa' \quad \Delta \vdash \tau' : \kappa}{\Delta \vdash \tau\tau' : \kappa'}$$

  We say that $\Delta; \Gamma$ is a *well-formed context* if $\Delta \vdash \Gamma(x) : T$ for any $x \in \mathrm{dom}(\Gamma)$.

- We introduce an equality between types. Two types τ, τ' of kind κ are equal under a kind context Δ if $\Delta \vdash \tau = \tau' : \kappa$ is derived only from the following rules:

$$\frac{\Delta \vdash \tau : \kappa}{\Delta \vdash \tau = \tau : \kappa} \quad \frac{\Delta \vdash \tau = \tau' : \kappa \quad \Delta \vdash \tau' = \tau'' : \kappa}{\Delta \vdash \tau = \tau'' : \kappa} \quad \frac{\Delta \vdash \tau' = \tau : \kappa}{\Delta \vdash \tau = \tau' : \kappa}$$

$$\frac{\Delta \vdash (\lambda\alpha^\kappa.\tau)\tau' : \kappa'}{\Delta \vdash (\lambda\alpha^\kappa.\tau)\tau' = \tau[\tau'/\alpha] : \kappa'} \quad \frac{\Delta \vdash \lambda\alpha^\kappa.\tau\alpha : \kappa \twoheadrightarrow \kappa'}{\Delta \vdash \lambda\alpha^\kappa.\tau\alpha = \tau : \kappa \twoheadrightarrow \kappa'}$$

- We say that $\Delta; \Gamma \vdash M : \tau$ is a *well-formed term* if it is derived only from the following rules:

$$\frac{\Delta; \Gamma \vdash M : \tau' \quad \Delta \vdash \tau = \tau' : T}{\Delta; \Gamma \vdash M : \tau} \quad \frac{x \in \mathrm{dom}(\Gamma) \quad \Delta; \Gamma \text{ well-formed}}{\Delta; \Gamma \vdash x : \Gamma(x)}$$

$$\frac{\Delta; \Gamma, x : \tau \vdash M : \tau'}{\Delta; \Gamma \vdash \lambda x^\tau.M : \tau \Rightarrow \tau'} \quad \frac{\Delta; \Gamma \vdash M : \tau \Rightarrow \tau' \quad \Delta; \Gamma \vdash M' : \tau}{\Delta; \Gamma \vdash MM' : \tau'}$$

  We note that if $\Delta; \Gamma \vdash M : \tau$ is a well-formed term, then $\Delta; \Gamma$ is a well-formed context.

- We introduce an equality between terms. Two terms M, N of type τ are equal under a kind context Δ and a type context Γ if $\Delta; \Gamma \vdash M = N : \tau$ is derived only from the following rules:

$$\frac{\Delta; \Gamma \vdash M : \tau}{\Delta; \Gamma \vdash M = M : \tau} \quad \frac{\Delta; \Gamma \vdash M' = M : \tau}{\Delta; \Gamma \vdash M = M' : \tau}$$

$$\frac{\Delta; \Gamma \vdash M = M' : \tau \quad \Delta; \Gamma \vdash M' = M'' : \tau}{\Delta; \Gamma \vdash M = M'' : \tau}$$

$$\frac{\Delta; \Gamma \vdash (\lambda x^\tau.M)M' : \tau'}{\Delta; \Gamma \vdash (\lambda x^\tau.M)M' = M[M'/\alpha] : \tau'} \quad \frac{\Delta; \Gamma \vdash \lambda x^\tau.Mx : \tau \Rightarrow \tau'}{\Delta; \Gamma \vdash \lambda x^\tau.Mx = M : \tau \Rightarrow \tau'}$$

A.2.2 A Set-Theoretic Semantics of $\lambda\underline{\omega}$

We introduce a set-theoretic model $\mathcal{F}$ of system $\lambda\underline{\omega}$. We define the following set $\mathbf{T}_0$, which is a subset of $\mathbf{T}$:

$$\mathbf{T}_0 \ni \tau ::= b \mid \tau \Rightarrow \tau$$

We define the following $\mathbf{T}_0$-indexed family of sets A:

$$Ab = \{\mathrm{tt}, \mathrm{ff}, \bot\}, \quad A(\tau \Rightarrow \tau') = (A\tau')^{A\tau}$$

where B^A is the set of all functions from A to B. We define a kind-indexed family of sets K to give the semantic domain of types:

$$KT = \{A\tau \mid \tau \in \mathbf{T}_0\}, \quad K(\kappa \Rrightarrow \kappa') = (K\kappa')^{K\kappa}$$

A Δ-*environment* is a mapping ρ satisfying $\mathrm{dom}(\rho) = \mathrm{dom}(\Delta)$ and $\rho(\alpha) \in K(\Delta(\alpha))$ for all $\alpha \in \mathrm{dom}(\Delta)$. Each well-formed type $\Delta \vdash \tau : \kappa$ is then interpreted in $K\kappa$ under a Δ-environment ρ. We write the value of this interpretation by $[\![\Delta \vdash \tau : \kappa]\!]\rho$. This interpretation is defined by induction on the derivation of the well-formed type as follows:

$$\begin{aligned}
[\![\Delta \vdash \alpha : \kappa]\!]\rho &= \rho(\alpha) \\
[\![\Delta \vdash b : T]\!]\rho &= Ab \\
[\![\Delta \vdash \Rightarrow : T \Rrightarrow T \Rrightarrow T]\!]\rho &= \lambda t \in KT\,.\,\lambda t' \in KT\,.\,t'^{t} \\
[\![\Delta \vdash \lambda\alpha^{\kappa}.\tau : \kappa \Rrightarrow \kappa']\!]\rho &= \lambda t \in K\kappa\,.\,[\![\Delta, \alpha : \kappa \vdash \tau : \kappa']\!]\rho\{\alpha : t\} \\
[\![\Delta \vdash \tau\tau' : \kappa']\!]\rho &= ([\![\Delta \vdash \tau : \kappa \Rrightarrow \kappa']\!]\rho)([\![\Delta \vdash \tau' : \kappa]\!]\rho)
\end{aligned}$$

Theorem A.2.1 *For any well-formed type $\Delta \vdash \tau : \kappa$ and Δ-environment ρ, we have $[\![\Delta \vdash \tau : \kappa]\!]\rho \in K\kappa$.* □

PROOF By induction on the derivation of $\Delta \vdash \tau : \kappa$. ■

Theorem A.2.2 (Soundness) *For any equation $\Delta \vdash \tau = \tau' : \kappa$ and Δ-environment ρ, we have $[\![\Delta \vdash \tau : \kappa]\!]\rho = [\![\Delta \vdash \tau' : \kappa]\!]\rho$.* □

PROOF By induction on the derivation of $\Delta \vdash \tau = \tau' : \kappa$. ■

We move to give the semantics of terms. For any well-formed context $\Delta; \Gamma$, a $\Delta; \Gamma$-*environment* is a pair of mappings $\rho; \eta$ where ρ is a Δ-environment and η is a mapping satisfying $\mathrm{dom}(\eta) = \mathrm{dom}(\Gamma)$ and $\eta(x) \in [\![\Delta \vdash \Gamma(x) : T]\!]\rho$ for any $x \in \mathrm{dom}(\Gamma)$. Each well-formed term $\Delta; \Gamma \vdash M : \tau$ is interpreted in the semantic domain

$[\![\Delta \vdash \tau : T]\!]\rho$ under a $\Delta;\Gamma$-environment $\rho;\eta$. We write the value of this interpretation by $[\![\Delta;\Gamma \vdash M : \tau]\!]\rho;\eta$. This interpretation is defined by induction on the derivation of the well-formed term.

$$\begin{aligned}
[\![\Delta;\Gamma \vdash x : \tau]\!]\rho;\eta &= \eta(x) \\
[\![\Delta;\Gamma \vdash M : \tau]\!]\rho;\eta &= [\![\Delta;\Gamma \vdash M : \tau']\!]\rho;\eta \quad (\Delta \vdash \tau = \tau' : T) \\
[\![\Delta;\Gamma \vdash \lambda x^{\tau}.M : \tau \Rightarrow \tau']\!]\rho;\eta &= \lambda v \in [\![\Delta \vdash \tau : T]\!]\rho.[\![\Delta;\Gamma, x : \tau \vdash M : \tau']\!]\rho;\eta\{x : v\} \\
[\![\Delta;\Gamma \vdash MM' : \tau']\!]\rho;\eta &= ([\![\Delta;\Gamma \vdash M : \tau \Rightarrow \tau']\!]\rho;\eta)([\![\Delta;\Gamma \vdash M' : \tau]\!]\rho;\eta)
\end{aligned}$$

Theorem A.2.3 *For any well-formed term $\Delta;\Gamma \vdash M : \tau$ and $\Delta;\Gamma$-environment $\rho;\eta$, we have $[\![\Delta;\Gamma \vdash M : \tau]\!]\rho;\eta \in [\![\Delta \vdash \tau : T]\!]\rho$.* □

PROOF By induction on the derivation of $\Delta;\Gamma \vdash M : \kappa$. ■

Theorem A.2.4 (Soundness) *For any equation $\Delta;\Gamma \vdash M = M' : \tau$ and $\Delta;\Gamma$-environment $\rho;\eta$, we have $[\![\Delta;\Gamma \vdash M : \tau]\!]\rho;\eta = [\![\Delta;\Gamma \vdash M' : \tau]\!]\rho;\eta$.* □

PROOF By induction on the derivation of $\Delta;\Gamma \vdash M = M' : \kappa$. ■

Notational convention Once we declare a well-formed type $\Delta \vdash \tau : \kappa$ and a well-formed term $\Delta;\Gamma \vdash M : \tau$, we refer them by τ and M.

A.3 Pre-logical Relations for $\lambda\underline{\omega}$

In this section we introduce the notion of pre-logical binary relations over $\mathcal{F}$. We only think of binary relations over $\mathcal{F}$ itself for the counterexample, although we can define relations between arbitrary models of $\lambda\underline{\omega}$ calculus.

Definition A.3.1 A *binary relation* over $\mathcal{F}$ is a pair (R, S) such that R is a kind-indexed family of sets satisfying $R\kappa \subseteq K\kappa \times K\kappa$ and S is an RT-indexed family of sets satisfying $S(t, t') \subseteq t \times t'$. □

Definition A.3.2 Let (R, S) and (R', S') be binary relations.

- The *inverse* of (R, S), written by $(R, S)^{-1}$, is the following binary relation (R'', S''):

$$\begin{aligned} R''\kappa &= \{(t', t) \mid (t, t') \in R\kappa\} \\ S''(t, t') &= \{(e', e) \mid (e, e') \in S(t', t)\} \end{aligned}$$

- The *composition* of (R, S) and (R', S'), written by $(R, S) \circ (R', S')$, is the following binary relation (R'', S''):

$$\begin{aligned} R''\kappa &= R\kappa \circ R'\kappa \\ S''(t, t'') &= \bigcup_{(t,t')\in RT\wedge(t',t'')\in R'T} S(t, t') \circ S'(t', t'') \end{aligned}$$ □

  where $- \circ -$ is ordinary composition of binary relations.

We define pre-logical relations in terms of the basic lemma. This way of defining pre-logical relations is an adoption of Leiss's definition of pre-logical binary relations between models of $F\omega$ [Lei01].

First, some notations. Let (R, S) be a binary relation. We write $(\rho, \rho') \in R^*\Delta$ if ρ and ρ' are mappings satisfying $(\rho(x), \rho'(x)) \in R(\Delta(x))$ for any $x \in \mathrm{dom}(\Delta)$. We also write $(\rho; \eta, \rho'; \eta') \in R; S^*(\Delta; \Gamma)$ for a well-formed context $\Delta; \Gamma$ if $(\rho, \rho') \in R^*\Delta$ and η and η' are mappings satisfying $(\eta(x), \eta'(x)) \in S([\![\Gamma(x)]\!]\rho, [\![\Gamma(x)]\!]\rho')$ for any $x \in \mathrm{dom}(\Gamma)$.

Definition A.3.3 ([Lei01]) A binary relation (R, S) is *pre-logical* if it satisfies the following statements:

1. For any well-formed type $\Delta \vdash \tau : \kappa$ and $(\rho, \rho') \in R^*\Delta$, we have

$$([\![\tau]\!]\rho, [\![\tau]\!]\rho') \in R\kappa.$$

2. For any well-formed term $\Delta; \Gamma \vdash M : \tau$ and $(\rho; \eta, \rho'; \eta') \in R; S^*(\Delta; \Gamma)$, we have

$$([\![M]\!]\rho; \eta, [\![M]\!]\rho'; \eta') \in S([\![\tau]\!]\rho, [\![\tau]\!]\rho').$$

It is clear that $(R, S)^{-1}$ is a pre-logical binary relation if and only if (R, S) is so.

A.4 A Counterexample

The goal of this section is to show that there are pre-logical binary relations whose composition is not a pre-logical binary relation. The goal is achieved by showing the following statement:

> There exists pre-logical binary relations (R, S) and (R', S') such that the composition $(R'', S'') = (R, S) \circ (R', S')$ satisfies the following: there exists a well-formed term $\Delta; \Gamma \vdash M : \tau$ and environments $(\rho; \eta, \rho''; \eta'') \in R''; S''^{*}(\Delta; \Gamma)$ such that
>
> $$([\![M]\!]\rho; \eta, [\![M]\!]\rho''; \eta'') \notin S''([\![\tau]\!]\rho, [\![\tau]\!]\rho'').$$

To simplify the situation, we presuppose that the above well-formed term M is the following:

$$\alpha : T, \beta : T; x : \alpha \Rightarrow \beta, y : \alpha \vdash xy : \beta.$$

and rewrite the goal as follows.

Theorem A.4.1 *There exists pre-logical binary relations (R, S) and (R', S') such that the composition $(R'', S'') = (R, S) \circ (R', S')$ satisfies the following: there exists $(t, t'') \in R''T, (u, u'') \in R''T, (e, e'') \in S''(t^u, t''^{u''}), (f, f'') \in S''(t, t'')$ such that*

$$(e(f), e''(f'')) \notin S''(u, u'').$$

PROOF We first define a pre-logical binary relation (R, S) and show that (R, S) and $(R, S)^{-1}$ satisfy the theorem A.4.1.

We define type substitutions θ and θ' to be $[b \Rightarrow b/\alpha]$ and $[b/\alpha]$, respectively. We define the relation R for types as follows:

$$R\kappa = \{([\![\emptyset \vdash \tau\theta : \kappa]\!], [\![\emptyset \vdash \tau\theta' : \kappa]\!]) \mid \alpha : T \vdash \tau : \kappa\}$$

Note that R is not functional, since we have both $([\![b \Rightarrow b]\!], [\![b \Rightarrow b]\!]) \in RT$ by letting $\tau = b \Rightarrow b$ and $([\![b \Rightarrow b]\!], [\![b]\!]) \in RT$ by letting $\tau = \alpha$.

Lemma A.4.2 *The relation R satisfies the condition 1 in definition A.3.3.* □

PROOF Let $\Delta \vdash \tau : \kappa$ be a well-formed type and $(\rho, \rho') \in R^*\Delta$. This means that for each $\beta_j \in \mathrm{dom}(\Delta)$ there exists a well-formed type $\alpha : T \vdash \tau_j : \Delta(\beta_j)$ such that $\rho(\beta_j) = [\![\tau_j\theta]\!]$ and $\rho'(\beta_j) = [\![\tau_j\theta']\!]$. Then we have

$$\begin{aligned} [\![\tau]\!]\rho &= [\![\tau[\tau_j\theta/\beta_j]_{\beta_j\in\mathrm{dom}(\Delta)}]\!] = [\![\tau[\tau_j/\beta_j]_{\beta_j\in\mathrm{dom}(\Delta)}\theta]\!] \\ [\![\tau]\!]\rho' &= [\![\tau[\tau_j\theta'/\beta_j]_{\beta_j\in\mathrm{dom}(\Delta)}]\!] = [\![\tau[\tau_j/\beta_j]_{\beta_j\in\mathrm{dom}(\Delta)}\theta']\!]. \end{aligned}$$

Therefore by the definition of R, the pair of the above two are included in $R\kappa$. ∎

Lemma A.4.3 *We write* $\mathbf{T}_0[\alpha]$ *for the set defined by the following BNF:*

$$\mathbf{T}_0[\alpha] \ni \tau ::= b \mid \alpha \mid \tau \Rightarrow \tau$$

where α *here is treated as a constant (not a metavariable ranging over the set of type variables). Then we have*

$$RT = \{([\![\emptyset \vdash \tau\theta : T]\!], [\![\emptyset \vdash \tau\theta' : T]\!] \mid \alpha : T \vdash \tau : T \wedge \tau \in \mathbf{T}_0[\alpha]\}.$$

PROOF ($\supseteq$) Trivial. ($\subseteq$) We define the following logical relation P:

$$\begin{aligned} PT &= \{([\![\tau\theta]\!], [\![\tau\theta']\!]) \mid \alpha : T \vdash \tau : T \wedge \tau \in \mathbf{T}_0[\alpha]\} \\ P(\kappa \twoheadrightarrow \kappa') &= \{(f, g) \mid \forall (x, y) \in P\kappa.(f(x), g(y)) \in P\kappa'\} \end{aligned}$$

It is easy to see $([\![b]\!], [\![b]\!]) \in PT$ and $([\![\Rightarrow]\!], [\![\Rightarrow]\!]) \in P(T \Rightarrow T \Rightarrow T)$. Therefore we obtain the basic lemma: for any well-formed type $\Delta \vdash \tau : \kappa$, we have

$$\forall (\rho, \rho') \in P\Delta.([\![\tau]\!]\rho, [\![\tau]\!]\rho') \in P\kappa.$$

Particularly for any well-formed term $\alpha : T \vdash \tau : T$ and $([\![b \Rightarrow b]\!], [\![b]\!]) \in PT$, we have

$$\begin{aligned} PT &\ni ([\![\tau]\!]\{\alpha : [\![b \Rightarrow b]\!]\}, [\![\tau]\!]\{\alpha : [\![b]\!]\}) \\ &= ([\![\tau\theta]\!], [\![\tau\theta']\!]). \end{aligned}$$

∎

Now we construct an RT-indexed family of relations S. This is constructed like logical relations by induction on $\tau \in \mathbf{T}_0[\alpha]$. This induction covers all elements in RT by the

previous lemma.

$$\begin{aligned} S([\![b\theta]\!],[\![b\theta']\!]) &= \{(\mathrm{tt},\mathrm{tt}),(\mathrm{ff},\mathrm{ff})\} \\ S([\![\alpha\theta]\!],[\![\alpha\theta']\!]) &= \{(\lambda x \in Ab.\bot,\bot)\} \\ S([\![(\tau \Rightarrow \tau')\theta]\!],[\![(\tau \Rightarrow \tau')\theta']\!]) &= \{(f,g) \mid \forall (x,y) \in S([\![\tau\theta]\!],[\![\tau\theta']\!]). \\ & \qquad (f(x),g(y)) \in R([\![\tau'\theta]\!],[\![\tau'\theta']\!])\}. \end{aligned}$$

We show a graphical explanation of a part of (R,S) in figure A.1. The set $2 \to 2$ in the figure represents the set of functions $f \in (Ab)^{Ab}$ such that for any $x \in \{\mathrm{tt},\mathrm{ff}\}$, $f(x) \in \{\mathrm{tt},\mathrm{ff}\}$. Dashed bold lines connect related semantic domains by R, and bold normal lines connect related values by S.

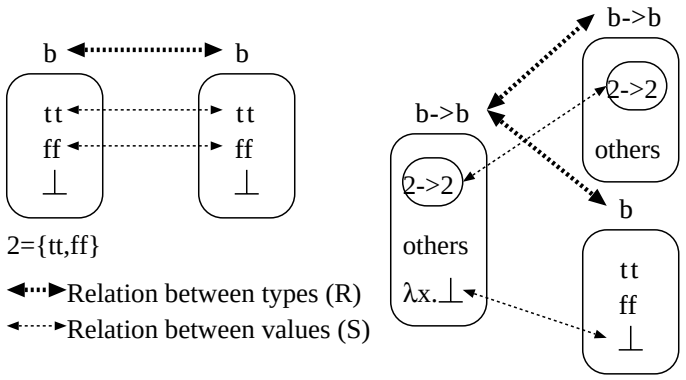


Figure A.1: Relation (R,S)

Lemma A.4.4 *The relation (R,S) is pre-logical.* □

PROOF We show (R,S) satisfies the condition 2 in definition A.3.3 by induction on the derivation of $\Delta;\Gamma \vdash M : \tau$. Let $(\rho;\eta,\rho';\eta') \in R;S^*(\Delta;\Gamma)$. We see interesting cases; applications and lambda abstractions.

- Case $M = M'M''$. There exists τ' and we have well-formed terms $\Delta;\Gamma \vdash M' : \tau' \Rightarrow \tau$ and $\Delta;\Gamma \vdash M'' : \tau'$. From IH, we have $([\![M']\!]\rho;\eta,[\![M']\!]\rho';\eta') \in S([\![\tau' \Rightarrow \tau]\!]\rho,[\![\tau' \Rightarrow \tau]\!]\rho')$ and $([\![M'']\!]\rho;\eta,[\![M'']\!]\rho';\eta') \in S([\![\tau']\!]\rho,[\![\tau']\!]\rho')$. Then from the lemma A.4.3, there exists $\tau_0,\tau_0' \in \mathbf{T}_0[\alpha]$ such that $([\![\tau]\!]\rho,[\![\tau]\!]\rho') =$

$(\llbracket\tau_0\theta\rrbracket, \llbracket\tau_0\theta'\rrbracket)$ and $(\llbracket\tau'\rrbracket\rho, \llbracket\tau'\rrbracket\rho') = (\llbracket\tau'_0\theta\rrbracket, \llbracket\tau'_0\theta'\rrbracket)$. From definition of S, we have

$$(\llbracket M'M''\rrbracket\rho; \eta, \llbracket M'M''\rrbracket\rho'; \eta') \in S(\llbracket\tau_0\theta\rrbracket, \llbracket\tau_0\theta'\rrbracket) = S(\llbracket\tau\rrbracket\rho, \llbracket\tau\rrbracket\rho').$$

- Case $M = \lambda x : \tau'.M'$ and $\tau = \tau' \Rightarrow \tau''$ for some $\Delta \vdash \tau', \tau'' : T$. We have a well-formed term $\Delta; \Gamma, x : \tau' \vdash M' : \tau''$. From lemma A.4.3 there exists $\tau'_0, \tau''_0 \in \mathbf{T}_0[\alpha]$ such that $(\llbracket\tau'\rrbracket\rho, \llbracket\tau'\rrbracket\rho') = (\llbracket\tau'_0\theta\rrbracket, \llbracket\tau'_0\theta'\rrbracket)$ and $(\llbracket\tau''\rrbracket\rho, \llbracket\tau''\rrbracket\rho') = (\llbracket\tau''_0\theta\rrbracket, \llbracket\tau''_0\theta'\rrbracket)$. From IH, for any $(e, e') \in S(\llbracket\tau'\rrbracket\rho, \llbracket\tau'\rrbracket\rho') = S(\llbracket\tau'_0\theta\rrbracket, \llbracket\tau'_0\theta'\rrbracket)$, we have

$$\begin{array}{rl} & ((\llbracket\lambda x^{\tau'}.M'\rrbracket\rho; \eta)(e), (\llbracket\lambda x^{\tau'}.M'\rrbracket\rho'; \eta')(e')) \\ = & (\llbracket M'\rrbracket\rho; \eta\{x : e\}, \llbracket M'\rrbracket\rho'; \eta'\{x : e'\}) \\ \in & S(\llbracket\tau''\rrbracket\rho, \llbracket\tau''\rrbracket\rho') = S(\llbracket\tau''_0\theta\rrbracket, \llbracket\tau''_0\theta'\rrbracket) \end{array}$$

Then from the definition of S, we have

$$\begin{array}{rl} & (\llbracket\lambda x^{\tau'}.M'\rrbracket\rho; \eta, \llbracket\lambda x^{\tau'}.M'\rrbracket\rho'; \eta') \\ \in & S(\llbracket(\tau'_0 \Rightarrow \tau''_0)\theta\rrbracket, \llbracket(\tau'_0 \Rightarrow \tau''_0)\theta'\rrbracket) = S(\llbracket\tau' \Rightarrow \tau''\rrbracket\rho, \llbracket\tau' \Rightarrow \tau''\rrbracket\rho'). \end{array}$$ ∎

We now consider the composition $(R', S') = (R, S) \circ (R, S)^{-1}$. Diagramatically, we add the mirror image of figure A.1 to itself, and regard connected values in both sides as a new binary relation (figure A.2).

We notice the following:

1. $(\lambda x \in Ab.\bot, \lambda x \in Ab.\bot) \in S'^{(\llbracket b \Rightarrow b\rrbracket, \llbracket b \Rightarrow b\rrbracket)}$,
2. $(\mathrm{tt}, \mathrm{tt}) \in S'^{(\llbracket b\rrbracket, \llbracket b\rrbracket)}$ and
3. $(\bot, \bot) \notin S'^{(\llbracket b\rrbracket, \llbracket b\rrbracket)}$.

Now we prove the theorem A.4.1; take $t = t'' = u = u'' = Ab, e = e'' = \lambda x \in Ab.\bot$ and $f = f'' = \mathrm{tt}$. ∎

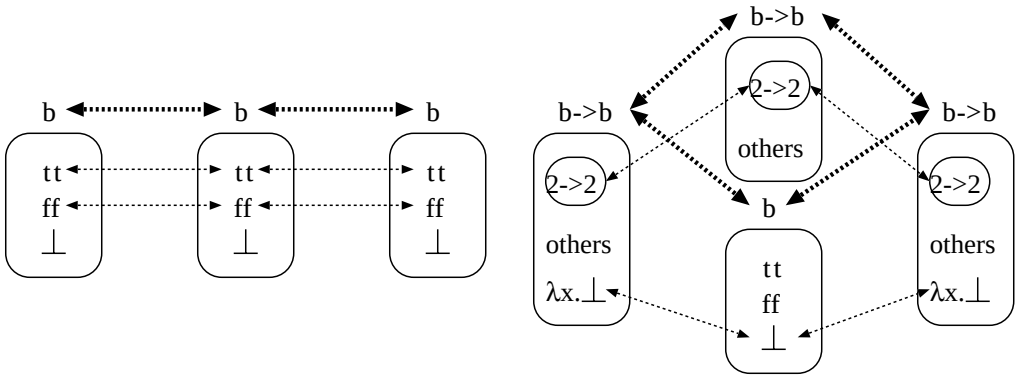


Figure A.2: Relation $(R, S) \circ (R, S)^{-1}$

A.5 Discussion

The above counterexample gives a detail of the failure of the closure property under the composition of pre-logical binary relations between models of $\lambda\underline{\omega}$. By constructing a similar binary relation, it is highly probable that $F\omega$ has the same problem, although we have not checked it in detail.

In [Lei01], Leiss mentioned that the pre-logical binary relations between models of $F\omega$ whose relations at types are functional are closed under composition. This seems to be a sensible solution, because this restriction excludes the above counterexample.

There is also a room to reconsider the definition of the composition of binary relations (definition A.3.2). For this, we speculate that it would be helpful to consider examples of data refinements in $F\omega$ (or $\lambda\underline{\omega}$), and to examine desirable properties of the composition of such refinements expressed by pre-logical relations.

Bibliography

[Aba00] M. Abadi. $\top\top$-closed relations and admissibility. *MSCS*, 10(3):313–320, 2000.

[AC98] R. Amadio and P.-L. Curien. *Domains and Lambda-Calculi*, volume 46 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1998.

[AG99] M. Abadi and A. Gordon. A calculus for cryptographic protocols: The spi calculus. *Information and Computation*, 148(1):1–70, 1999.

[And86] P. Andrews. *An introduction to mathematical logic and type theory: to truth through proof*. Academic Press, 1986.

[Bar84] H. Barendregt. *The Lambda Calculus-Its Syntax and Semantics*. North Holland, 1984.

[Bar91] H. Barendregt. Introduction to generalized type systems. *J. Funct. Program.*, 1(2):125–154, 1991.

[BD77] R. Burstall and J. Darlington. A transformation system for developing recursive programs. *ACM*, 24(1):44–67, 1977.

[BH96] M. Bidoit and R. Hennicker. Behavioural theories and the proof of behavioural properties. *Theoretical Computer Science*, 165(1):3–55, 1996.

[BHM02] N. Benton, J. Hughes, and E. Moggi. Monads and effects. In *Proc. APPSEM 2000*, volume 2395 of *LNCS*, pages 42–122. Springer, 2002.

[BHW95] M. Bidoit, R. Hennicker, and M. Wirsing. Behavioural and abstractor specifications. *Science of Computer Programming*, 25(2–3):149–186, 1995.

[BKR99] N. Benton, A. Kennedy, and G. Russell. Compiling standard ML to Java bytecodes. In *Proc. ICFP 1998*, volume 34(1) of *SIGPLAN Notices*, pages 129–140. ACM, 1999.

[Bor94] F. Borceux. *Handbook of Categorical Algebra 1*, volume 50 of *Encyclopedia of Mathematics and Its Applications*. Cambridge University Press, 1994.

[BT96] M. Bidoit and A. Tarlecki. Behavioural satisfaction and equivalence in concrete model categories. In *Proc. 21st Int. Coll. on Trees in Algebra and Programming (CAAP '96)*, volume 1059 of *LNCS*, pages 241–256. Springer, 1996.

[Cro94] R. Crole. *Categories for Types*. Cambridge Mathematical Textbooks. Cambridge, 1994.

[Doe96] K. Doets. *Basic Model Theory*. CSLI publications, 1996.

[Fio02] M. Fiore. Semantic analysis of normalisation by evaluation for typed lambda calculus. In *Proc. PPDP 2002*, pages 26–37. ACM, 2002.

[FP94] M. Fiore and G. Plotkin. An axiomatization of computationally adequate domain theoretic models of FPC. In *Proc. LICS 1994*, pages 92–102. IEEE, 1994.

[FPT99] M. Fiore, G. Plotkin, and D. Turi. Abstract syntax and variable binding. In *Proc. LICS 1999*, pages 193–202. IEEE Computer Society Press, 1999.

[Fri73] H. Friedman. Equality between functionals. In *Proc. Logic Colloquium*, volume 453 of *LNM*. Springer, 1973.

[FS99] M. Fiore and A. Simpson. Lambda definability with sums via grothendieck logical relations. In *Proc. TLCA 1999*, volume 1581 of *LNCS*, pages 147–161. Springer, 1999.

[Gir72] J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures dans l'arithmétique d'ordre supérieur*. PhD thesis, Université Paris VII, 1972.

[Gir87] J.-Y. Girard. Linear logic. *Theor. Comp. Sci.*, 50:1–102, 1987.

[GLJ93] A. Gill, J. Launchbury, and S. Jones. A short cut to deforestation. In *Proc. Functional Programming and Computer Architecture 1993*, pages 223–232. ACM Press, 1993, 1993.

[GLLN02] J. G.-Larrecq, S. Lasota, and D. Nowak. Logical relations for monadic types. In *Proc. CSL*, volume 2471 of *LNCS*, pages 553–568. Springer, 2002.

[GLLNZ04] J. G-Larrecq, S. Lasota, D. Nowak, and Y. Zhang. Complete lax logical relations for cryptographic lambda-calculus. In *Proc. CSL 2004*, volume 3210 of *LNCS*, pages 400–414. Springer, 2004.

[GLT88] J.-Y. Girard, Y. Lafont, and P. Taylor. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1988.

[Han01] J. Hannay. *Abstraction Barriers and Refinement in the Polymorphic Lambda Calculus*. PhD thesis, University of Edinburgh, 2001.

[HD97] J. Hatcliff and O. Danvy. A computational formalization for partial evaluation. *MSCS*, 7(5):507–541, 1997.

[Hen50] L. Henkin. Completeness in the theory of types. *Journal of Symbolic Logic*, 15:81–91, 1950.

[Her93] C. Hermida. *Fibrations, Logical Predicates and Indeterminants*. PhD thesis, University of Edinburgh, 1993.

[HJ95] C. Hermida and B. Jacobs. An algebraic view of structural induction. In *Proc. Computer Science Logic 1994*, volume 933 of *LNCS*, pages 412–426. Springer-Verlag, 1995.

[HKS03] J. Hannay, S. Katsumata, and D. Sannella. Semantic and syntactic approaches to simulation relations. In *Proc. MFCS*, volume 2747 of *LNCS*, pages 68–91. Springer, 2003.

[HLST00] F. Honsell, J. Longley, D. Sannella, and A. Tarlecki. Constructive data refinement in typed lambda calculus. In *Proc. FoSSACS 2000*, volume 1784 of *LNCS*, pages 161–176. Springer, 2000.

[Hoa72] C. Hoare. Proof of correctness of data representations. *Acta Informatica*, 1:271–281, 1972.

[Hof97] M. Hofmann. *Semantics of Logics of Computation*, chapter Syntax and semantics of dependent types, pages 79–130. Cambridge Univ. Press, 1997.

[Hof99] M. Hofmann. Semantical analysis of higher-order abstract syntax. In *Proc. LICS*, pages 204–213. IEEE, 1999.

[HS86] J. Hindley and J. Seldin. *Introduction to Combinators and λ-calculus*. Cambridge University Press, 1986.

[HS96] M. Hofmann and D. Sannella. On behavioural satisfaction and behavioural abstraction in higher-order logic. *Theoretical Computer Science*, 167(1–2):3–45, 1996.

[HS02] F. Honsell and D. Sannella. Prelogical relations. *Information and Computation*, 178(1):23–43, 2002.

[Jac99] B. Jacobs. *Categorical Logic and Type Theory*. Elsevier, 1999.

[JT93] A. Jung and J. Tiuryn. A new characterization of lambda definability. In *Proc. TLCA*, volume 664 of *LNCS*, pages 245–257. Springer, 1993.

[Kat04] S. Katsumata. A generalisation of pre-logical predicates to simply typed formal systems. In *Proc. ICALP '04*, volume 3142 of *LNCS*, pages 831–845. Springer, 2004.

[KOPT97] Y. Kinoshita, P. O'Hearn, A. Power, and M. Takeyama. An axiomatic approach to binary logical relations with applications to data refinement. In *Proc. TACS 1997*, volume 1281 of *LNCS*, pages 191–212. Springer, 1997.

[KP99] Y. Kinoshita and J. Power. Data-refinement for call-by-value programming languages. In *Proc. CSL 1999*, volume 1683 of *LNCS*, pages 562–576. Springr, 1999.

[Laf88] Y. Lafont. *Logiques, Categories et Machines*. PhD thesis, Université de Paris VII, 1988.

[Law70] F. Lawvere. Equality in hyperdoctrines and comprehension schema as an adjoint functor. In *Proc. AMS Symposium on Pure Mathematics XVII*, pages 1–14, 1970.

[Lei01] H. Leiss. Second-order pre-logical relations and representation independence. In *Proc. TLCA 2001*, volume 2044 of *LNCS*, pages 298–314. Springer, 2001.

[Lin04] S. Lindley. *Normalisation by Evaluation in the Compilation of Typed Functional Programming Languages*. PhD thesis, University of Edinburgh, 2004.

[LS86] J. Lambek and P. J. Scott. *Introduction to Higher Order Categorical Logic*. Cambridge studies in advanced mathematics. CUP, 1986.

[LS05] S. Lindley and I. Stark. Reducibility and $\top\top$-lifting for computation types. In *TLCA*, pages 262–277, 2005.

[Mac71] S. MacLane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics*. Springer, 1971.

[Mil71] R. Milner. An algebraic definition of simulation between programs. In *Second International Joint Conference on Artificial Intelligence*, pages 481–489. The British Computer Society, 1971.

[Mit86] J. Mitchell. Representation independence and data abstraction. In *Proc. POPL*, pages 263–276, 1986.

[Mit90] J. Mitchell. Type systems for programming languages. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 365–458. Elsevier and MIT Press, 1990.

[Mit91] J. Mitchell. On the equivalence of data representations. In V. Lifschitz, editor, *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 305–330. Academic Press, San Diego, 1991.

[Mit96] J. Mitchell. *Foundations for Programming Languages*. MIT Press, 1996.

[ML75] P. Martin-Löf. An intuitionistic theory of types: predicative part. In *Logic Colloquium '73*, pages 73–118. North-Holland, 1975.

[MM91] J. Mitchell and E. Moggi. Kripke-style models for typed lambda calculus. *Annals of Pure and Applied Logic*, 51:99–124, 1991.

[MM92] S. MacLane and I. Moerdijk. *Sheaves in geometry and logic*. LNMS. Springer Verlag, 1992. biblio entry inserted by Steve Vickers; modified by Simon Gay.

[Mog91] E. Moggi. Notions of computation and monads. *Information and Computation*, 93(1):55–92, 1991.

[MR92] Q. Ma and J. Reynolds. Types, abstractions, and parametric polymorphism, part 2. In *Proc. MFPS 1991*, volume 598 of *LNCS*, pages 1–40. Springer, 1992.

[MS93] J. Mitchell and A. Scedrov. Notes on sconing and relators. In *Proc. CSL 1992*, volume 702 of *LNCS*, pages 352–378. Springer, 1993.

[MS03] M. Miculan and I. Scagnetto. A framework for typed HOAS and semantics. In *Proc. PPDP 2003*, pages 184–194. ACM, 2003.

[Nie00] L. Nielsen. A denotational investigation of defunctionalization. Technical Report RS-00-47, BRICS, 2000.

[Nip86] T. Nipkow. Non-deterministic data types: Models and implementations. *Acta Informatica 22*, pages 629–661, 1986.

[Nis03] S. Nishimura. Correctness of a higher-order removal transformation through a relational reasoning. In *Proc. APLAS 2003*, volume 2895 of *LNCS*, pages 358–375, 2003.

[Plo76] G. Plotkin. A powerdomain construction. *SIAM Journal of Computaing*, 5:452–487, 1976.

[Plo77] G. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255, 1977.

[Plo80] G. Plotkin. Lambda-definability in the full type hierarchy. In *"To H.B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism"*, pages 367–373. Academic Press, San Diego, 1980.

[PPST00] G. Plotkin, J. Power, D. Sannella, and R. Tennent. Lax logical relations. In *Proc. ICALP 2000*, volume 1853 of *LNCS*, pages 85–102. Springer, 2000.

[PR00] J. Power and E. Robinson. Logical relations and data abstraction. In *CSL*, volume 1862 of *LNCS*, pages 497–511. Springer, 2000.

[PS98] A. Pitts and I. Stark. Operational reasoning for functions with local state. In A. D. Gordon and A. M. Pitts, editors, *Higher Order Operational Techniques in Semantics*, Publications of the Newton Institute, pages 227–273. Cambridge University Press, 1998.

[Rey83] J. Reynolds. Types, abstraction and parametric polymorphism. In *Proc. 9th IFIP World Computer Congress*, pages 513–523. North-Holland, 1983.

[Sch85] O. Schoett. Behavioural correctness of data representations. Technical Report CSR-185-85, Department of Computer Science, University of Edinburgh, 1985.

[Sch90] O. Schoett. Behavioural correctness of data representations. *Science of Computer Programming*, 14:43–57, 1990.

[Sho83] O. Shoett. A theory of program modules, their specification and implementation (extended abstract). Technical Report CSR-155-83, Department of Computer Science, University of Edinburgh, 1983.

[Sie92] K. Sieber. Reasoning about sequential functions via logical relations. In *Proc. LMS Symposium on Applications of Categories in Computer Science*, LMS Lecture Note Series 177, pages 258–269. Cambridge University Press, 1992.

[ST87] D. Sannella and A. Tarlecki. On observational equivalence and algebraic specification. *Journal of Computer and System Sciences*, 34:150–178, 1987.

[ST88] D. Sannella and A. Tarlecki. Essential concepts of algebraic specification and program development: implementations revisited. *Acta Informatica*, pages 233–281, 1988.

[Sta85] R. Statman. Logical relations and the typed lambda calculus. *Information and Control*, 65:85–97, 1985.

[Sta96] I. Stark. Categorical models for local names. *Lisp and Symbolic Computation*, 9(1):77–107, February 1996.

[Tai67] W. Tait. Intensional interpretation of functionals of finite type i. *Journal of Symbolic Logic*, 32, 1967.

[Tan00] M. Tanaka. Abstract syntax and variable binding for linear binders. In *Proc. MFCS*, volume 1893 of *LNCS*, pages 670–679. Springer, 2000.

[Ten94] R. Tennent. Correctness of data representation in algol-like languages. In *A Classical Mind: Essays in Honour of C.A.R. Hoare*. Prentice Hall, 1994.

[TS96] A. Troelstra and H. Schwichtenberg. *Basic Proof Theory*, volume 43 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1996.

[Wad90] P. Wadler. Deforestation: transforming programs to eliminate trees. *Theoretical Computer Science*, 73:231–248, 1990.