Interactive theorem proving and program extraction.

Hou, Tie

Interactive Theorem Proving and Program Extraction

Tie Hou

Submitted to Swansea University in fulfilment
of the requirements for the Degree of Doctor of Philosophy

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University

2014

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed .... (candidate)

Date 28/08/2014

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed .... (candidate)

Date 28/08/2014

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed .... (candidate)

Date 28/08/2014

Abstract

Nowadays more and more people are relying heavily on software and software controlled system. The failure of some software may result in serious consequences such as significant financial losses or substantial environmental damages. The methods for improving the reliability of software can be viewed as either based on traditional techniques that from programs to give proofs, or on automatic techniques that from proofs to generate programs.

In this dissertation, we are concerned with developing theories of program extractions from proofs via realisability in order to obtain better extracted programs from interactive theorem provers.

Firstly, we study the domain-theoretic semantics of a Church-style typed λ-calculus with constructors, pattern matching and recursion, and show that it is closely related to the semantics of its untyped counterpart. When extracting programs from proofs via realisability, one has the choice of extracting typed or untyped terms from proofs. Our result shows that under a certain regularity condition, the choice is irrelevant.

Secondly, we propose a realisability interpretation of an intuitionistic version of Church's Simple Theory of Types (CST) which can be viewed as a formalisation of intuitionistic higher-order logic. In this way, important syntactic properties of realisability (e.g. being well-behaved w.r.t. substitution) can be proven elegantly on an abstract lambda-calculus level. Our interpretation introduces a direct realisability of monotone induction and coinduction.

Thirdly, we develop a prototype interactive theorem prover in Haskell to demonstrate the usefulness of the theory described previously. In this prototype, motivated by the desire to facilitate the implementation of interactive proof systems with rich sets of proof rules, we implement a uniform system of rule schemata to generate proof rules for different styles of logical calculi.

Acknowledgements

First, I am most indebted to my supervisor, Ulrich Berger, for his valuable guidance, unselfish commitment of time, continuous support and encouragement throughout my study. He has been remarkably patient and a constant source of inspiration. Just knowing that I could go to him for help, about any matter whatsoever, was a great source of reassurance for me. He also provided a very nice research environment with good connections around the world. I hope that one day I will become as prolific and inspiring a supervisor as he is. Second, I would like to thank Arnold Beckmann for being my second supervisor. His concern, interest and support have been invaluable.

Furthermore, I am grateful to my examiners, Marcelo Fiore and Anton Setzer, for their detailed high-quality comments and the interesting and stimulating discussions. I would also like to thank Hajime Ishihara from Japan Advanced Institute of Science and Technology and Hideki Tsuiki from Kyoto University for inviting me to visit their research groups in Japan where most of this work has been built up.

I also take this opportunity to thank all the academic and non-academic staff of the Computer Science Department at Swansea University for providing an excellent working environment. In particular, I would like to thank all the staff members and research students of the Theoretical Computer Science group at Swansea University for the friendly atmosphere and good discussions. The MRes/PhD seminars have been a valuable experience and I am grateful for their wonderful talks and challenging questions. I would like to say a special thanks to Monika Seisenberger for her kindness and support.

I owe many thanks to my friends that always give full attention to me to solve my problems and put up with my moods.

Last but not least, I am deeply grateful to my father for his love, understanding and support throughout all my life. I owe him what I am today.

Table of Contents

Chapter 1

Introduction

Contents

1.1 Motivation

Ultimately, software problems are solved by building well-structured and comprehensible programs that are guaranteed correct with respect to the specification. However, in general, when developing complex computer systems, no matter how well designed, it is natural that the resulting system still is full of errors. With rapid growth of the size and complexity of software programs, the importance of the reliability of these programs arises.

A "correct" program is one that specifies the desired behaviour of the program. More precisely, it produces the correct output for every possible input if the program terminates successfully, and the program will always terminate successfully. The methodologies for designing reliable software can be viewed as either based on correctness proofs (i.e., giving proofs for existing programs) or on program extractions (i.e., from proofs to generate programs). To verify the correctness of a program empirically is a long-standing and widely-used practice in the software industry. Usually, in order to obtain full correctness, this would involve feeding all possible values of its input to the certified program and proving the correctness of the respective output. This can be really time consuming, or even impossible. Therefore, the correctness of the tested program cannot be guaranteed in general.

A "formally correct" program is one whose correctness is verified in a mathematical approach. In order to achieve this, the intended behaviour of programs has to be

specified precisely in a mathematical way. It is well known that (abstract) algorithms are often hidden inside mathematical proofs. Indeed, from a fully formalised constructive mathematical proof, one can easily extract a computer program of such an algorithm together with a proof that guarantees the correctness of the program.

The research presented in this dissertation began as an attempt to provide a new systematic approach to synthesise correct, error-free programs from proofs, and strong theoretical guarantees about their correctness. We focus on the applicability of constructive logics and type theories to the problem of program extraction from proofs via realisability. There are always gaps between our thinking and our communication. Consider the following sentence

$$\exists x(x \cdot x = 2) \tag{1.1}$$

Can we really assert this by saying x exists without giving the value of x but just thinking there is an x? Even if there exists an x, is it computable? The solution is when we assert the existence of something, we should be able to provide an algorithm to compute it, and a proof that the algorithm is correct. For example, to prove the formula (1.1), we need to find an instance of x, e.g. t, and a proof P of $t \cdot t = 2$. This computational view of existence can naturally be extended to all logical connectives. We are led to the *Brouwer-Heyting-Kolmogorov interpretation* (BHK interpretation) that explains the meaning of a proof of a given formula:

- A proof of $P \wedge Q$ is given as a pair (p,q) of proofs, where p is a proof of P and q is a proof of Q.
- A proof of $P \vee Q$ is given as a pair (p,q) of proofs, where either p is 0 and q is a proof of Q, or p is 1 and q is a proof of Q.
- A proof of $P \to Q$ is a construction which converts any proof of P into a proof of Q.
- There is no proof for $\bot$.
- A proof of $\forall x.P(x)$ is a construction which converts a proof of $d \in D$ (D is the intended range of the variable x) into a proof of $P(d)$.
- A proof of $\exists x.P(x)$ is given as a pair (d,q) with $d \in D$, where q is a proof of $P(d)$.

The BHK interpretation can be developed into a methodology by which one can use mathematical reasoning to derive programs whose correctness is developed during the construction. One important feature of constructive proofs is that the executable codes

extracted from formalisations of proofs are functional programs. This provides a particularly elegant way of connecting program extraction with lambda calculus and its corresponding type theories.

The *Curry-Howard isomorphism* ([Cur34, How80]), also known as *proofs - as - programs correspondence*, establishes a deep connection between programs and proofs. It is a generalisation of the following syntactic analogy: a proof is identified with a λ-term (in other words, a program), the formula it provides is encoded as a type for the program, logical rules can be represented as type inferences or programming constructs, and proof normalisations correspond to term reductions. This contributes a set of methods to extract programs from proofs so that the task of programming a function is cut down to reasoning from contexts.

In traditional verification of programs, the most natural way is to start with a particular specification that expresses the input-output behaviour of a desired program in terms of logical formulas, and then build a program by hand, finally prove that the program satisfies the specification. An alternative approach is instead of writing the program, to generate the proof of the formulas (semi-)automatically with the help of proof assistants. Through the power of the Curry-Howard isomorphism, a program in a suitably chosen (usually functional) programming language is created. The transformation process from proofs to correct-by-construction programs is called *program extraction*. Very often the generating programs contain parts that are irrelevant to compute the final result. How to synthesise efficient programs from proofs obeying their formal specifications has been a long sought after goal.

One method of program extraction is to employ a realisability interpretation. In 1945 Kleene [Kle45] first introduced the concept of realisability with the idea of defining a relation "*n realises A*" between natural numbers *n* and logical sentences *A*. Intuitively, a realiser is a solution of the computational problem expressed by *A*. Later many other notions on realisability were introduced. In particular Gödel's functional or Dialectica Interpretation [Goe58, Goe90] and Kreisel's modified realisability [Kre59] have a profound impact. The possibility of effectively obtaining a program and its verification proof is based on a sound realisability interpretation.

It is essential for people in mathematical, scientific and engineering fields to logically analyse problems or present mathematical models precisely. Among all the formal logics that they choose to express and prove mathematical facts, *first-order logic* has a prominent place. However, it is not an expressive tool in a practice sense, since it is impossible to prove about higher-order objects such as sets and functions in first-order logic directly. Some basic and reasonable notions fall outside the scope of first-order logic, such as the transitive closure of a relation and the completeness principle for the real numbers. Starting with a formal set theory to formalise e.g. abstract algebra in first-order logic is of course an alternative solution, especially when abbreviations are involved. Nevertheless, such formalisations are often quite expensive in terms of how much work needs to be carried out. Moreover, an additional limitation of

first-order logic lies in the expression of mathematical statements due to the missing of abstraction mechanisms for building predicates and functions and definition description mechanisms for specifying values.

The *Simple Theory of Types*, originating with Alonzo Church [Chu40], is a precise formulation of type theory which includes first-order logic. Farmer gave a detailed analysis of the virtues of Church's type theory in [Far08] and characterised Church's type theory as a simple, elegant, highly expressive and practical language and logic. In Church's type theory, functions are treated as primitives since properties and relations are expressed via functions from objects to truth values, and lambda-notation and lambda-conversion [Chu32, Chu41] are formulated for use in the logic. Church's type theory is also called *higher-order logic*, since it admits unrestricted quantification over higher order predicates. There are two kinds of mathematical objects in Church's type theory: terms and types. *Terms* are the terms of the λ-calculus, denoting values including truth values. Based on atomic types, e.g. the type of individuals and the type of propositions (truth values), *types* can be built as functional types $\rho \to \sigma$ where ρ and σ are types. Every term has been assigned to a type which denotes the kind of values it ranges over and should be consistent with the types of its subterms. In Church's original formalisation it is a classical type theory since it allows nonconstructive reasoning principles. However, we will work on a version that is based on intuitionistic logic (but is still impredicative).

Eventually we are interested in specifying the operational behaviour of a machine when it is running a program. However, an extracted program is a collection of abstract expressions with no observation of its behaviour, i.e. no operational semantics. Therefore, it is necessary to have a connection between the proof-theoretic view with the execution of a program. An elegant way to ensure the correctness of the behaviours of programs is to put denotational semantics in the middle. With denotational semantics, we can formalise mathematically rigorous descriptions of programming languages by assigning mathematical meanings to programs in terms of mathematical objects, such as strings, integers, booleans and functions. Aiming at this purpose, a wealth of different approaches are proposed. The theory of *Scott domains*, originally conceived by Scott [Sco70], is a powerful mathematical tool to give a meaning to the following two features of programming languages: recursion (least fixed point) and data types (e.g. lists, function spaces and recursive types). Instead of representing objects as elements of a data type in the sense of programming languages, elements of a domain are abstract representations of their partial properties that contain some notions of information. Thus, it is the input-output behaviour of programs that is finally being formalised. Domain-theoretic semantics provide very simple and elegant proofs of computational adequacy, and hence for the correctness of program extraction. Since domain theory combines the computational features of functions with the mathematical definition of function as a mapping from one domain to another, from a mathematical point of view, a functional language can be viewed as a description language for domain-theoretic

concepts.

1.2 Main Contributions

This section serves as a general description of the main achievements of this dissertation. In the next section, background information is provided, and more detailed explanations are given.

(1) We study the domain-theoretic semantics of a Church-style typed λ-calculus with constructors, pattern matching and recursion, and show that it is closely related to the semantics of its untyped counterpart. When extracting programs from proofs via realisability, one has the choice of extracting typed or untyped terms from proofs. Our result shows that under a certain regularity condition, the choice is irrelevant. Furthermore, we give a soundness proof for a language of realisers of proofs involving inductive and coinductive definitions. The proof uses logical relations, which are related to Tait's computability method and Girard's method of reducibility candidates.

("Typed vs. Untyped Realizability" [BH12])

(2) We give a realisability interpretation of an intuitionistic version of Church's Simple Theory of Types (CST) which can be viewed as a formalisation of intuitionistic higher-order logic. Although definable in CST we include operators for monotone induction and coinduction (not limited to the strictly positive case only) and provide simple realisers for them. Realisers are formally represented in an untyped lambda-calculus with pairing and case-construct. We introduce a general notion of interpretation of one instance of the simply typed lambda calculus in another, and define realisability as an instance of such an interpretation. In this way, important syntactic properties of realisability (e.g. being well-behaved w.r.t. substitution) can be proven elegantly on an abstract lambda-calculus level. Our interpretation introduces a direct realisability of monotone induction and coinduction.

("A realisability interpretation of Church's simple theory of types" [BH14a])

(3) In order to demonstrate the usefulness of the theory described in (2), we develop a prototype of the proof system in Haskell. In this prototype, motivated by the desire to facilitate the implementation of interactive proof systems with rich sets of proof rules, we implement a uniform system of rule schemata to generate proof rules for different styles of logical calculi. The system requires only one schema for each logical operator to generate introduction and elimination rules

in natural deduction and sequent calculus style. In addition, the system supports program extraction from proofs by generating realisers for the proof rules automatically.

("Uniform Schemata for Proof Rules" [BH14b])

1.3 Overview of the Dissertation

Chapter 2: Typed vs. Untyped Realisability

In this chapter, we introduce a natural language of realisers for inductive and coinductive definitions, which is a typed lambda calculus with types modelling initial algebras and final coalgebras, and terms modelling structural recursion and corecursion. In fact, we study a more general calculus that allows fixed points of arbitrary type operators and definitions of functions by general recursion. The advantage of this generality is that our results will apply to all conceivable extensions of our theory of realisers of inductive and coinductive definitions.

We study the domain-theoretic semantics of a Church-style typed λ-calculus with constructors, pattern matching and recursion, and compare it with its untyped counterpart. We work with polymorphic types that allow fixed points of arbitrary type operators. A type ρ is interpreted as (the image of) a finitary projection $\langle\rho\rangle$, following the idea of Amadio, Bruce and Longo [ABL86]. The main result (Theorem 2.4.16) relates the semantics of a typed term M with its untyped variant M^-: if M has type ρ, where ρ is a regular type, that is, fixed points are only taken of positive operators, then

$$\langle\rho\rangle[\![M^-]\!] = [\![M]\!],$$

where $[\![M]\!]$ is the value of M in a denotational model. The proof uses logical relations. We do not know whether the result also holds if ρ is not regular.

A similar problem was studied by Reynolds [Rey00, Rey03] who established a coherence between the typed and untyped meanings of expressions based on cpo models of a version of PCF. The main differences to our work are as follows: Reynolds considers simple types over the base types of natural numbers and booleans while we allow arbitrary recursive types. On the other hand, he includes subtyping which we do not. Regarding the typed semantics, Reynolds interprets typing derivations in a typed model while we interpret terms with a typed abstraction in an untyped model.

The motivation for this study comes from program extraction from proofs via realisability (see e.g. [BH08, Ber09, BS10, Ber11] for applications in constructive analysis) where one has the choice of extracting typed or untyped terms from proofs. Our result shows that if the extracted type is regular, the choice is irrelevant. In fact, regularity is a harmless restriction because in the intended realisability interpretation the

types of realising terms will always be regular. In [Ber10] the soundness of a realisability interpretation based on a fragment of the untyped version of our calculus was proven, and the calculus was shown to be computationally adequate with respect to a domain-theoretic semantics (the same semantics we are considering here). In [BS11] it was shown that the extracted programs admit a Curry-style typing. In this dissertation we provide the missing semantical link to Curry-style typing.

The application to realisability is also our motivation for working with Scott domains (instead of arbitrary cpos, as Reynolds does): the adequacy proof in [Ber10] uses the fact that all semantic objects can be approximated by compact ones, hence we have to ensure that types are interpreted in a cartesian closed category of algebraic domains. This is achieved by interpreting types as finitary projections. Apart from that, our results could also be obtained using arbitrary cpos and embedding-retraction pairs.

Chapter 3: Church's Simple Type Theory

In this chapter, we describe the basic logic of a constructive version of a type theory, namely Church's Simple Type Theory. We start with the Simply Typed Lambda Calculus (STLC), which is constructed freely from type atoms. And then we introduce the notion of an interpretation from one instances of STLC to another. One important aspect of this interpretation is that it preserves full β-equality.

Slightly differing from Church's original work, our version of CST is based on a particular instance of STLC where the set of base types contains a set $\mathscr{S}$ of base types for individuals and a type of proposition o, and the constant set consists of $\to$, $\wedge$, $\vee$, $\forall_\rho$, $\exists_\rho$, $=_\rho$, μ_ρ and ν_ρ. Here for $\forall_\rho$ and $\exists_\rho$, ρ is for arbitrary types, while for μ_ρ and ν_ρ, ρ is restricted to *predicate types*, i.e. types that are canonically (in any ccc) isomorphic to a finite product of types of the form $\rho \to o$. The constants μ_ρ and ν_ρ will be interpreted as least and greatest fixed point operators for monotone arguments. In order to precisely express their properties we give a definition of higher-order versions of inclusion between predicates, which can be declared meaningfully only by predicate types.

From a logical point of view, the constants $\to$ and $\forall_\rho$ would suffice to define all other constant. For the logical constants including equality this was already observed by Church [Chu40]. $\mu_\rho \Phi$ can be defined as the infimum of all $x : \rho$ such that $\Phi x \subseteq_\rho x$, and $\nu_\rho \Phi$ can be defined similarly. The reason why we prefer this richer set of constants is that they can be given simpler realisers.

In order to enable a realisability interpretation we have to deviate from Church's calculus in several aspects: first our system is intuitionistic while Church's is classical. Secondly, we dropped the choice operator since it does not appear to admit a realisability interpretation.

The reason why we choose Church's type theory is that, due to its simplicity, it

provides the necessary, but simple techniques for a realisability interpretation. Church presented his theory as a foundation of mathematics, but we make use of its property that it is free to be extended. Church's type theory is general in the sense that it has implicit constraints on terms and types. It can be directly extended with other datatypes such as Cartesian products, disjoint unions, records, and so on. Therefore many programming languages support a simply typed system. Also, in Church's type theory axioms for set existence are no longer necessary, since lambda-notations (or functions) provide an explicit representation of sets. In addition, type checking is decidable in Church's type theory.

Chapter 4: Program Extraction via Realisability

In this chapter, we give a definition of realisability RCST by extending CST to a more practical form. And then we give the soundness results for our realisability interpretation.

We extend CST to RCST by adding an extra base type δ for realisers and extra constants nil, in_L, in_R, pr_L, pr_R, pair, app, fun, case, rec which we call *program constants*. We also extend the ranges of the parameters of the constants $\forall_\rho$, $\exists_\rho$, $=_\rho$ and μ_ρ, ν_ρ to all types respectively predicate types ρ of RCST.

The new base type δ will be interpreted by a Scott domain D which is essentially the same as the domain used in Chapter 2.

In order to prove the soundness of induction and coinduction, we replace the rules for monotone induction in CST by rules which we call *general (co)induction*. That is, rules expressing that $\mu_\rho\Phi$ is the least fixed point of the operator $\breve{\Phi}(X) := \bigcup^\rho_{Y\subseteq_\rho X}\Phi(Y)$. The operator $\breve{\Phi}$ (which could be easily formally defined in RCST) is monotone for arbitrary $\Phi : \rho \to \rho$, hence the least fixed point exists. If Φ is monotone, then Φ is the same as $\breve{\Phi}$. Dually, the rules for coinduction are replaced by rules expressing that $\nu_\rho\Phi$ is the greatest fixed point of the operator $\widehat{\Phi}(X) := \bigcap^\rho_{Y\supseteq_\rho X}\Phi(Y)$. This general (co)induction is a generalisation to higher-order logic of Mendler-style (co)induction [Men91].

Our interpretation appears to be more general than related interpretations in Isabelle/HOL [Ber03b] and Coq [PM89b, PM89a] in that it covers unrestricted intuitionistic higher-order logic and induction is not confined to the strictly positive case.

Realisability interpretations for monotone coinduction have been given earlier by Tatsuta [Tat98] and Miranda-Perea [MP05]. The minor difference of Tatsuta's interpretation to ours is that he uses realisability with truth (q-realisability) whereas we omit the 'truth' component, he works in second-order logic while we use higher order logic, and in his system the programming language is part of the 'input system', that is, the formal system that is to be interpreted, while we keep the input and output systems apart. The major difference is that we can avoid Tatsuta's extra condition on monotone (co)induction namely that not only the operator has to be monotone, but also its

realisability interpretation. Tatsuta shows that this extra condition is necessary in his system. The reason why it is not necessary in ours is that our realising system (i.e. output system) has rules for a certain form of non-monotone inductive and coinductive definitions. Miranda-Perea extracts typed terms and uses a clausular form of monotone (co)induction in Krivine [Kri93] system AF2 of second-order logic.

Our main motivation for using higher-order logic and monotone instead of strictly positive induction/coinduction is not that applications would require this greater expressive power and generality. It is more that it allows to express problems much more naturally and less technically, and the analysis of monotone (co)induction is much simpler than that of the strictly positive case. In addition, in our system, induction and coinduction are completely dual to each other, which is not the case in current implementations (e.g. Coq [Coq], Minlog [Min]) which are restricted to the strictly positive case.

Our soundness result refers to the provability "on both sides", i.e. if A is provable then we get a term M that provably realises A where the realisability is defined formally. In order to conclude that realisers also *compute* results we refer to the Adequacy results in [Ber10] that relate formal realisability to the computation of witness.

To summarise, the main improvement in our work is that we prove soundness of realisability for monotone (co)induction

- without the extra condition that the realisability interpretation of the monotone operator in question is again monotone (these conditions are needed in [Tat98] and [MP05]);

- for higher-order logic (previous results are restricted to second-order logic [Tat98, MP05] or to a restricted form of strictly positive induction and coinduction [Ber03b, PM89b, PM89a]);

- w.r.t. type-free realisers for which a computational adequacy (i.e. normalisation) theorem has been shown earlier and once and for all (other approaches use typed realisers, which means that normalisation has to be proved again if the (co)induction scheme changes, since then type system changes as well).

Chapter 5: Uniform Schemata for Proof Rules

In this chapter, we introduce a uniform system of *rule schemata*, which directly express the meaning of logical operators and which, in a uniform way, allow to derive the rules of different styles of proof calculi, such as sequent calculus and natural deduction, but also further rules that are used in interactive proof assistants. Surprisingly, the approach requires only one schema for each logical operator. The introduction and elimination rules of natural deduction as well as left and right rules in sequent calculus are derived automatically. Moreover, our system is able to automatically derive realisers of intuitionistic proof rules, thus facilitating the implementation of proof systems that support program extraction from proofs, such as Coq [Coq] and Minlog [Min].

We are currently developing a prototype as shown in the next chapter of such a proof system using rule schemata as a basis of the implementation.

Briefly, the global strategy is as follows. First we introduce rule schemata, from which we derive generating rules. These generating rules are different rules that correspond to different styles of proving e.g. sequent calculus, or natural deduction or the mixture of these two. Then from generating rules we obtain the real rules in the proof system by instantiation and adding side formulas.

An additional advantage of rule schemata is the fact that they are built on a data structure of finitary sets, a generalisation of finite sets. Finitary sets have the structure of a monad and can therefore be very conveniently implemented and manipulated in a programming language that supports monads and provides a special syntax for them.

Chapter 6: Implementation

In this chapter, we describe the implementation of a prototype interactive theorem prover, which is an experiment with the usability of the concepts introduced in previous chapters. It is composed of two main parts: first, an automated proof checker for verifying mathematical arguments with the ability to store and replay proofs; second, an interactive interface for users to input logic formulas and give commands based on the given information, i.e. what goals remain to be proven and which assumptions are available to prove those goals.

We consider plain simple one-sorted predicate logic without complicated data structures in this prover. Modules with monad structures are used to provide a convenient framework for applying all necessary functions for different types.

Since it is very difficult to remove bugs when programs become bigger, we also give a precise proof for the correct behaviour of our prover before coding.

Chapter 2

Domain-theoretic Semantics of a Language of Realisers

Contents

A notation for arbitrary *functions* was not available until 250 years after the mathematical notation for *expressions* and *equations* was devised, when Church introduced the smallest universal programming language of the world, the *lambda calculus* [Chu32, Chu33]. Anything that can be computed can be expressed and evaluated via this system. The lambda calculus is an alternative to Turing machines [Tur36] in the sense that both models define the same class of computable functions [Tur37]. However, the lambda calculus is more related to the software aspect, not caring about the implementation of the machine evaluating it, whereas the Turing machine is more related to the hardware aspect. Beyond its great influence in the area of computability theory, the lambda calculus has many practical applications in the formal semantics of programming languages.

The central concepts in the lambda calculus are function abstraction and application by variable substitution. There are two versions of the lambda calculus, typed (also called Church-style) and untyped (also called Curry-style). The difference between the two styles is that in Church-style type assignment each bound variable is assigned a unique type, as in $\lambda x : \rho.M$, while in Curry-style the binding is untyped, as

in $\lambda x.M$. The untyped lambda calculus has no restrictions on the function application, so the domain of a function is not declared inside the system. Therefore, the untyped lambda calculus can formalise all effectively computable functions. On the other hand, in the typed lambda calculus, a function can only accept the inputs of a certain given type. Types play an important role in the development of software systems. Type checking allows us to mechanically ensure the compatibility of the constructed programs and the correctness of their functions. However, typed programming languages are sometimes too constraining, so that untyped programming languages are taken into account. In addition, the typed lambda calculus provides a mathematical connection to proof theory via the Curry-Howard isomorphism.

This chapter contributes to a soundness proof for a language of realisers of proofs involving inductive and coinductive definitions. The notion of realisability will be discussed in more detail in Chapter 4.

2.1 Preliminaries

In this section, we will review the basic theory of Scott Domains, and give explicit descriptions of some of its properties. More details can be found in [SHLG94, AJ94, AC98].

The primary motivation for the study of domains was raised by two problems, least fixed points as meanings of recursive definitions, and recursive domain equations. Let D be some mathematical structure. Given a recursive definition on D: $X = f(X)$, we want to find an element $d \in D$ such that if we substitute d for x in the equation, we will generate a valid equation s.t. $d = f(d)$. That is indeed looking for a fixed point of f. Furthermore, we want a uniform canonical method for constructing such fixed points for arbitrary D and f. However, usual mathematical structures like sets, topological spaces, groups, vector spaces and etc. cannot satisfy these. Apart from that, giving a mathematical semantics for the lambda calculus can also lead to problems. Take the self-application term $\lambda x.xx$ as an example. If the type of the second occurrence of x in xx is D and that of the whole term xx is D, then the type of the first occurrence must be $[D \to D]$. If $[\_ \to \_]$ is a functor $F : \mathbf{C}^{op} \times \mathbf{C} \to \mathbf{C}$ over some category $\mathbf{C}$, again recursive datatypes lead us to the requirement for a fixed point that is uniform and canonical.

A *partially ordered set* or *poset* is a set P with a binary relation $\sqsubseteq$ which is reflexive, transitive and antisymmetric. If an element $x \in P$ is above every element of $A \subseteq P$, then x is called an *upper bound*. If all elements of P are above a single element $x \in P$, then x is called the *least element*. *Lower bound* and *largest element* are defined dually. If the least element x of P is no less than all elements of $A \subset P$, x is called *supremum*. We write $x = \sqcup A$. A nonempty subset $A \sqsubseteq P$ is *directed* if for any $x, y \in A$ there exists an upper bound $z \in A$ with $x \sqsubseteq z$ and $y \sqsubseteq z$.

Definition 2.1.1 (Scott-domain) A triple $(D, \sqsubseteq, \bot)$ is a *Scott-domain* if it is

- directed complete: if $A \subseteq D$ is directed, then $\sqcup A \in D$
- algebraic: $\forall x \in D.\widehat{x} := \{x_0 \in D \mid x_0 \text{ is compact}\}$ is directed and $x = \sqcup\widehat{x}$
- bounded complete: $\forall B \subseteq D((\exists x \in D.B \sqsubseteq x) \Rightarrow \sqcup B \in D)$

where

$x_0 \in D$ is *compact* if for every directed $A \subseteq D$, A has a supremum $\sqcup A$ and $x_0 \sqsubseteq \sqcup A$ then $x_0 \sqsubseteq x$ for some $x \in A$. By D_c we denote the set of compact elements of D.

B is *bounded* if $\exists x \in D.\forall y \in B.y \sqsubseteq x$.

The reason for working with Scott domains is that all the semantic constructions we need are readily available, e.g. cartesian closure, solutions to recursive domain equations, recursive definition of functions, interpretation of types, including recursive types, as finitary projections. All these constructions are very elementary and do not require a heavy category-theoretical machinery.

By a *Scott-domain*, or *domain* for short, we mean a bounded complete ω-algebraic dcpo with least element. We will denote the least element of a domain by $\bot$. By $\mathbf{1}$ we denote the sole-element domain $\{\text{Nil}\}$, and by $(D_1 + \ldots + D_n)_\bot$, $D \times E$, $[D \to E]$ the separated sum [AC98], cartesian product, and continuous function space of domains[1]. Note that in a coalesced sum, the bottom elements are identified if they have them, while in the separated sum, a new bottom element is adjoined.

Due to ω-algebraicity, every element of a domain D is the directed countable supremum of compact elements.

Remark From the point of view of information, an element $x \in D$ can be interpreted as a datum that might not be fully defined. Thus $x \sqsubseteq y$ indicates that all the information represented by x is in y; the supremum of $A \subseteq D$ is the element that contains all the information only from every element of A; the least element contains no information at all. The algebraicity clause guarantees that each element obtains all the information from the ones lower down in the ordering.

Lemma 2.1.2 If for $F \subseteq [D \to E]$, there exists an $f \in [D \to E]$ s.t. $f = \sqcup F$, then $f(x) = \sqcup\{g(x) \mid g \in F\}$ for all $x \in D$.

Proof. Since $f = \sqcup F$, we have (1) $\forall g \in F(g \sqsubseteq f)$, and (2) for any $f' \in [D \to E]$, if $\forall g \in F(g \sqsubseteq f')$, then $f \sqsubseteq f'$.

Therefore we can get for all $x \in D$ and for all $g \in F$, $g(x) \sqsubseteq f(x)$, and if $g(x) \sqsubseteq f'(x)$, then $f(x) \sqsubseteq f'(x)$. That is $f(x) = \sqcup\{g(x) \mid g \in F\}$. □

[1]These domain operations should not be confused with the syntactic constructors for types which for simplicity we denoted by the same symbols.

Definition 2.1.3 (Subdomain) $E \subseteq D$ is a *subdomain* of D if

(i) $\perp_D \in E$.

(ii) If $A \subseteq E$ and $\sqcup_D A$ exists in D, then $\sqcup_D A \in E$ and $\sqcup_D A = \sqcup_E A$.

(iii) If x is compact in E, then x is compact in D.

(iv) $\forall y \in D_c \forall x \in E(y \sqsubseteq x \rightarrow \exists y' \in E_c(y \sqsubseteq y' \sqsubseteq x))$.

Lemma 2.1.4 Let $E \subseteq D$ be a subdomain of D. Then E is a domain.

Proof. By verifying the four clauses of Definition 2.1.1.

1. Assume $A \subseteq E$ is directed. Show $\sqcup A \in E$:

   Since $A \subseteq D$ is directed, we get $\sqcup A \in D$. By Definition 2.1.3 (ii), we get $\sqcup A \in E$.

2. We need to show $\forall x \in E(\hat{x}^E := \{y \in E \mid y \in E_c, y \sqsubseteq x\}$ is directed $\wedge\, x = \sqcup_E \hat{x}^E)$.

   Let $x \in E$. We have $\hat{x}^D := \{y \in D \mid y \in D_c, y \sqsubseteq x\}$ is directed $\wedge\, x = \sqcup_D \hat{x}^D$, since D is a domain.

   We have $\hat{x}^E \subseteq \hat{x}^D$ by Definition 2.1.3 (iii), and by Definition 2.1.3 (iv) we have $\hat{x}^E$ is bounded.

   So we get $\sqcup_D \hat{x}^E$ exists, and by Definition 2.1.3 (ii) we get $\sqcup_E \hat{x}^E = \sqcup_D \hat{x}^E \sqsubseteq \sqcup_D \hat{x}^D = x \in E$.

3. Assume $B \subseteq E$ and $\exists x \in E.B \sqsubseteq x$. Then by Definition 2.1.1 we have $\sqcup B \in D$. And then by Definition 2.1.3 (ii) we get $\sqcup B \in E$.

□

Following [ABL86] we interpret types as finitary projections in D. Since the range of a finitary projection is a subdomain of D the semantics of types can be viewed as a domain. This approach provides an easy solution to the problem of defining the semantics of a fixed point type: one can simply take the least fixed point of a suitable continuous function on the domain $[D \rightarrow D]$.

Definition 2.1.5 (Finitary projection) $f : D \rightarrow D$ is a *projection* if

- f is continuous,
- $f \sqsubseteq \mathrm{id}$, i.e. $\forall x \in D.f(x) \sqsubseteq x$,
- $f \circ f = f$, i.e. $\forall x \in D.f(f(x)) = f(x)$.

A projection f is *finitary* if the range of f, denoted by $f(D)$, is a subdomain of D.

For $f: X \to X$ we set $\mathrm{Fix}(f) := \{x \in X \mid f(x) = x\}$.

Lemma 2.1.6 If a function $f: X \to X$ is continuous and $f \circ f = f$, then $f(X) = \mathrm{Fix}(f)$.

Proof. If $f \circ f = f$, then $f(X) \subseteq \mathrm{Fix}(f)$. Trivially $\mathrm{Fix}(f) \subseteq f(X)$. Hence, $f(X) = \mathrm{Fix}(f)$. □

Lemma 2.1.7 E is a subdomain of D if and only if there exists a finitary projection $p: D \to D$ such that $E = p(D)$.

Proof. "⇒": Assume E is a subdomain of D. We have to find a projection p s.t. $E = p(D)$.

We define $p: D \to D$ by $p(x) = \sqcup_D\{y \in E \mid y \sqsubseteq x\} = \sqcup_E\{y \in E \mid y \sqsubseteq x\}$. We show first that p is a projection.

1. Obviously, $p(x) \sqsubseteq x$ for all $x \in D$, so we get $p \sqsubseteq \mathrm{id}$.

2. We need to show $p \circ p = p$. We have

$$\begin{aligned} p(p(x)) &= p(\sqcup\{y \in E \mid y \sqsubseteq x\}) \\ &= \sqcup\{z \in E \mid z \sqsubseteq \sqcup\{y \in E \mid y \sqsubseteq x\}\} \end{aligned}$$

   Let $A := \sqcup\{y \in E \mid y \sqsubseteq x\}$, and $B := \sqcup\{z \in E \mid z \sqsubseteq A\}$. We show $A = B$.

   - $B \sqsubseteq A$: It suffices to show that A is an upper bound of $\{z \in E \mid z \sqsubseteq A\}$. This is obvious.
   - $A \sqsubseteq B$: It suffices to show that B is an upper bound of $\{y \in E \mid y \sqsubseteq x\}$.
     Let $y \in E$ s.t. $y \sqsubseteq x$. We need to show $y \sqsubseteq B$.
     By definition of B, it suffices to show that $y \sqsubseteq A$. This follows by the fact that $y \sqsubseteq x$ and the definition of A.

3. We show p is continuous. Clearly, p is monotone by its definition.

   Let $A \subseteq D$ directed. We need to show $p(\sqcup A) \sqsubseteq \sqcup p(A)$. Since

$$\begin{aligned} p(\sqcup A) &= \sqcup\{y \in E \mid y \sqsubseteq \sqcup A\} \\ \sqcup p(A) &= \sqcup\{p(a) \mid a \in A\} \\ &= \sqcup\{\sqcup\{y \in E \mid y \sqsubseteq a\} \mid a \in A\}, \end{aligned}$$

   let $b := \sqcup\{\sqcup\{y \in E \mid y \sqsubseteq a\} \mid a \in A\}$, it is to show $z \sqsubseteq b$ for any $z \in E$ s.t. $z \sqsubseteq \sqcup A$. Since every element of D can be obtained as the supremum of a directed set of compact elements of D, we get $z = \sqcup\{x \in D_c \mid x \sqsubseteq z\}$. Then by Definition 2.1.3 (iv), there exists some $x' \in E_c$ s.t. $x \sqsubseteq x' \sqsubseteq z$. By Definition 2.1.3 (iv), we get $x' \in D_c$. Thus, by (1) we get $x' \sqsubseteq b$. So $x \sqsubseteq b$. Hence, b is an upper bound of $\{x \in D_c \mid x \sqsubseteq z\}$. Therefore, $z \sqsubseteq b$.

Second, we want to show that $E = p(D)$.

1. "$\subseteq$": Let $x \in E$. Then $p(x) = \sqcup\{y \in E \mid y \sqsubseteq x\} = x$. So $x \in p(D)$.

2. "$\supseteq$": Assume $p(x) = x$. We need to show $x \in E$.

   We get $\sqcup\{y \in E \mid y \sqsubseteq x\} = x$. By Definition 2.1.3 (ii), $\sqcup_E\{y \in E \mid y \sqsubseteq x\} \in E$, i.e. $x \in E$.

"$\Leftarrow$": Trivial since p is a finitary projection. □

Lemma 2.1.8 Let $p : D \to D$ be a projection. If $a, b \in p(D)$ and $a \sqcup b$ exists, then $a \sqcup b \in p(D)$.

Proof. We need to show $p(a \sqcup b) = a \sqcup b$.
Since $p(a \sqcup b) \sqsupseteq p(a) = a$ and $p(a \sqcup b) \sqsupseteq p(b) = b$, we get $p(a \sqcup b) \sqsupseteq a \sqcup b$. We get $p(a \sqcup b) \sqsubseteq a \sqcup b$ by $p \sqsubseteq \mathrm{id}$. Therefore, $p(a \sqcup b) = a \sqcup b$. □

In the following two lemmas we assume that $p : D \to D$ is a projection, and $p(D)_c := D_c \cap p(D)$.

Lemma 2.1.9 The following are equivalent:

(a) p is finitary

(b) $\forall x \in D(A_x := \{a \in p(D)_c \mid a \sqsubseteq x\})$ is directed and $p(x) = \sqcup A_x$.

(c) $\exists A \subseteq D_c . \forall x \in D(p(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$.

Proof. • We show $(b) \Rightarrow (a)$.

Since p is a projection on D, we get $p(D) = \mathrm{Fix}(p)$ by Lemma 2.1.6.

We need to show $p(D)$ is finitary, i.e. $p(D)$ is a subdomain of D.

(i) Since $p \sqsubseteq \mathrm{id}$ and $\bot \sqsubseteq p(\bot)$, we get $\bot = p(\bot) \in p(D)$.

(ii) Assume $A \subseteq p(D)$ and $\sqcup_D A$ exists in D. We need to show $\sqcup_D A \in p(D)$, i.e. to show $p(\sqcup_D A) = \sqcup_D A$ since $p(D) = \mathrm{Fix}(p)$.
Let $x := \sqcup_D A$. Then $p(x) = \sqcup\{a \in p(D)_c \mid a \sqsubseteq x\} \sqsubseteq x$.
We need to show $p(x) \sqsupseteq x$, i.e. $p(x) \sqsupseteq y$ for all $y \in A$.
Since $\sqcup_D A = x \sqsupseteq y$ for all $y \in A$, we get $p(x) \sqsupseteq p(y) = y$.

(iii) Assume x is compact in $p(D)$. Let $x \sqsubseteq \sqcup_D A$, where $A \subseteq D$ is directed. Then we get $x = p(x) \sqsubseteq p(\sqcup_D A) = \sqcup_D p(A)$. Therefore $\sqcup_D p(A) = p(\sqcup_D A) \in p(D)$. Trivially $p(A) \subseteq p(D)$. Hence, we get $x \sqsubseteq p(y)$ for some $y \in A$. Since $p \sqsubseteq \mathrm{id}$, we get $p(y) \sqsubseteq y$. So $x \sqsubseteq y$ for some $y \in A$. So $x \in D_c$.

(iv) Assume $y \sqsubseteq x$ for some $y \in D_c$ and $x \in p(D)$. We need to show $\exists y' \in (p(D))_c (y \sqsubseteq y' \sqsubseteq x)$.
Since $p(x) = \sqcup A_x$, we get $\forall y' \in A_x (y' \sqsubseteq p(x) = x)$.
Since $A_x \subseteq D_c$ is directed by (iii), we get $y \sqsubseteq y'$ for some $y' \in A_x$.

- We show $(a) \Rightarrow (b)$.

Assume $p(D)$ is finitary. We need to show $\forall x \in D (A_x := \{a \in p(D)_c \mid a \sqsubseteq x\}$ is directed and $p(x) = \sqcup A_x)$.

Since $\bot \in p(D)$, $\bot \in D_c$ and $\bot \sqsubseteq x$ for all $x \in D$, we get $\bot \in A_x \neq \emptyset$.

For all $a, b \in A_x$, we get $a \sqsubseteq x$ and $b \sqsubseteq x$. By Lemma 2.1.8, we get $a \sqcup b \in A_x$. So A_x is directed.

Since $p(x) \in p(D) \subseteq D$, we get $p(x) = \sqcup \{a \in p(D)_c \mid a \sqsubseteq p(x)\} = \sqcup \{a \in p(D)_c \mid p(a) \sqsubseteq p(x)\} \sqsupseteq \sqcup \{a \in p(D)_c \mid a \sqsubseteq x\} = \sqcup A_x$.

Therefore, $p(x) = \sqcup A_x$.

- We show $(c) \Rightarrow (a)$.

Since p is a projection on D, we get $p(D) = \mathrm{Fix}(p)$ by Lemma 2.1.6.

We need to show $p(D)$ is finitary, i.e. $p(D)$ is a subdomain of D.

(i) Since $p \sqsubseteq \mathrm{id}$ and $\bot \sqsubseteq p(\bot)$, we get $\bot = p(\bot) \in p(D)$.

(ii) Assume $B \subseteq p(D)$ and $\sqcup_D B$ exists in D. We need to show $\sqcup_D B \in p(D)$, i.e. to show $p(\sqcup_D B) = \sqcup_D B$ since $p(D) = \mathrm{Fix}(p)$.
Let $x := \sqcup_D B$. Then $p(x) = \sqcup \{a \in A \mid a \sqsubseteq x\}$ for some $A \subseteq D_c$.
Clearly $p(x) \sqsubseteq x$ by p is a projection.
We need to show $p(x) \sqsupseteq x$, i.e. $p(x) \sqsupseteq y$ for all $y \in B$.
Since $\sqcup_D B = x \sqsupseteq y$ for all $y \in B$, we get $p(x) \sqsupseteq p(y) = y$.

(iii) Assume x is compact in $p(D)$. Let $x \sqsubseteq \sqcup_D B$, where $B \subseteq D$ is directed. Then we get $x = p(x) \sqsubseteq p(\sqcup_D B) = \sqcup_D p(B)$. Now $p(B) \subseteq p(D)$ since p is monotone and $\sqcup_D p(B) \in p(D)$ by (b). Hence, we get $x \sqsubseteq p(y)$ for some $y \in B$. Since $p \sqsubseteq \mathrm{id}$, we get $p(y) \sqsubseteq y$. So $x \sqsubseteq y$ for some $y \in B$. So $x \in D_c$.

(iv) Assume $y \sqsubseteq x$ for all $y \in D_c$ and for all $x \in p(D)$. We need to show $\exists y' \in (p(D))_c (y \sqsubseteq y' \sqsubseteq x)$.
Since $p(x) = \sqcup \{a \in A \mid a \sqsubseteq x\}$ for some $A \subseteq D_c$, we get $\forall y' \in \{a \in A \mid a \sqsubseteq x\} (y' \sqsubseteq p(x) = x)$.
Since $\{a \in A \mid a \sqsubseteq x\} \subseteq D_c$ is directed, for all $y \in D_c$, we get $y \sqsubseteq y'$ for some $y' \in \{a \in A \mid a \sqsubseteq x\}$.

- We show $(c) \Rightarrow (a)$.

  Assume $p(D)$ is finitary. We need to show $\exists A \subseteq D_c.\forall x \in D(p(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$. By Lemma 2.1.9 we get $\forall x \in D(A_x := \{a \in p(D)_c \mid a \sqsubseteq x\}$ is directed and $p(x) = \sqcup A_x)$. Let $A := p(D)_c$. Since $p(D)_c \subseteq D_c$, we get $\exists A \subseteq D_c.\forall x \in D(p(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$.

□

Lemma 2.1.10 (Admissibility of finitary projections) If for all n, p_n is a finitary projection and $p_n \sqsubseteq p_{n+1}$, then $\sqcup_n p_n$ is a finitary projection.

Proof. Let $p := \sqcup_n p_n$. By Lemma 2.1.2, it is easy to see that p is a projection.

By Lemma 2.1.9 (c), it suffices to show $p(x) = \sqcup\{a \in p(D)_c \mid a \sqsubseteq x\}$.

We first show that p is idempotent:

$p(p(x)) = (\sqcup_n p_n)((\sqcup_m p_m)(x)) = \sqcup_n \sqcup_m (p_n(p_m(x))) \overset{(*)}{=} \sqcup_n(p_n(p_n(x))) = \sqcup_n(p_n(x))$
$= (\sqcup_n p_n)(x) = p(x)$.

Equation (*) easily follows from the fact that the double sequence $p_n(p_m(x))$ is increasing in m and in n. Hence, we get $p(D) = \{x \mid p(x) = x\}$, and therefore, $\sqcup\{a \in p(D)_c \mid a \sqsubseteq x\} = \sqcup\{a \in D_c \mid p(a) = a \wedge a \sqsubseteq x\} = \sqcup\{a \in D_c \mid \sqcup_n(p_n(a)) = a \wedge a \sqsubseteq x\} = \sqcup\{a \in D_c \mid \exists n.p_n(a) = a \wedge a \sqsubseteq x\}$ (by compactness) $= \sqcup_n(\sqcup\{a \in D_c \mid p_n(a) = a \wedge a \sqsubseteq x\}) = \sqcup_n(p_n(x)) = p(x)$. □

Lemma 2.1.11 Let $(L, \sqsubseteq)$ be a complete lattice, and $f_1, f_2 : L \to L$ are monotone.

Assume $\forall x \in L(f_1(x) \sqsubseteq f_2(x))$. Then $\mathrm{LFP}(f_1) \sqsubseteq \mathrm{LFP}(f_2)$.

Proof. First to show $\mathrm{LFP}(f_1)$ and $\mathrm{LFP}(f_2)$ exist.

If $f : L \to L$ is monotone, let $X = \{x \in L \mid f(x) \sqsubseteq x\}$. Since a complete latice cannot be empty, L has a maximal element. Thus, X is nonempty. Then because f is monotone, if $x \in X$, then we have $f(f(x)) \sqsubseteq f(x)$, i.e. $f(x) \in X$.

Let $x \in X$ and $x_0 = \sqcap X$. Then $x_0 \sqsubseteq x$, so $f(x_0) \sqsubseteq f(x) \sqsubseteq x$. Hence, $f(x_0)$ is a lower bound of X. But x_0 is the greatest lower bound, so $f(x_0) \sqsubseteq x_0$, i.e. $x_0 \in X$. Then $f(x_0) \in X$. Again since x_0 is the greatest lower bound of X, we get $x_0 \sqsubseteq f(x_0)$. Therefore $x_0 = f(x_0)$. Clearly x_0 is the least fixed point of f since every fixed point is in X.

Second to show $\mathrm{LFP}(f_1) \sqsubseteq \mathrm{LFP}(f_2)$.

Let $\mathrm{LFP}(f_i) = \sqcap X_i (i = 1, 2)$ where $X_i = \{x \in L \mid f_i(x) \sqsubseteq x\}$.

Since $\mathrm{LFP}(f_2)$ is the greatest lower bound of X_2, then it is to show $\mathrm{LFP}(f_1) \sqsubseteq X_2$.

Since $\mathrm{LFP}(f_1)$ is the greatest lower bound of X_1, i.e. $\mathrm{LFP}(f_1) \sqsubseteq X_1$, then it is to show $X_2 \subseteq X_1$.

Assume $x \in X_2$. Then $f_2(x) \sqsubseteq x$. Since $f_1(x) \sqsubseteq f_2(x)$, we get $f_1(x) \sqsubseteq x$. Then $x \in X_1$. □

2.2 Types and Terms

In this and the next section we study the syntax and semantics of types and typed terms. Untyped terms will be introduced in Section 2.4.

Definition 2.2.1 (Types) The set of types is defined by the following grammar:

$$\mathbf{Type} \ni \rho, \sigma, \tau ::= \alpha \mid \rho \to \sigma \mid \mathbf{1} \mid \rho \times \sigma \mid \rho + \sigma \mid \mathrm{fix}\,\alpha.\rho.$$

where α ranges over a set **TVar** of type variables. The fixed-point construction $\mathrm{fix}\,\alpha.\rho$ binds all free occurrences of α in ρ.

We work with a Church-style typed lambda-calculus with constructors, pattern matching and recursion which we call *Language of Realisers* (**LoR**) because its terms are intended to be used as extracted programs from proofs obtained by a realisability interpretation.

We consider only the constructors Nil (nullary), Pair (binary), and Left, Right, In (unary). The intention behind the first four constructors should be obvious. The constructor In is used to model type fixed points up to isomorphism. Many definitions and results could be extended to an arbitrary set of constructors.

Definition 2.2.2 (Terms) The set of (Church-style typed) terms is defined by

$$\begin{aligned}\mathbf{LoR} \ni M, N, R_i ::= x \mid \lambda x:\rho.M \mid MN \mid \mathrm{rec}\,x:\rho.M \mid C(M_1,\ldots,M_n) \mid \\ \mathrm{case}\,M\,\mathrm{of}\,\{C_i(\vec{x}_i) \to R_i\}_{i\in\{1,\ldots,n\}}.\end{aligned}$$

where x ranges over a set of variables **Var**, C is a constructor of arity n, and in $\mathrm{case}\,M\,\mathrm{of}\,\{C_i(\vec{x}_i) \to R_i\}_{i\in\{1,\ldots,n\}}$ all constructors C_i are distinct and each $\vec{x}_i$ is a vector of distinct variables whose length coincide with the arity of C_i. Lambda abstraction, $\lambda x:\rho.M$, and recursion, $\mathrm{rec}\,x:\rho.M$, bind all free occurrences of x in M, and a pattern matching clause, $C_i(\vec{x}_i) \to R_i$, binds all free occurrences of $\vec{x}_i$ in R_i.

We introduce typing rules for **LoR**-terms. A *type context* is a set of pairs $\Gamma := x_1 : \rho_1, \ldots, x_n : \rho_n$ (for notational convenience we omit the curly braces) where ρ_i are types and x_i are distinct variables. The set of variables $\{x_1, \ldots, x_n\}$ (which may be empty) is denoted by $\mathrm{dom}(\Gamma)$.

The relation $\Gamma \vdash M : \rho$ (M is a **LoR** term of type ρ in context Γ) is inductively defined as follows. Note that in the definition of terms (Definition 2.2.2), a case expression can have in general many clauses, but our typing rules only allow two or one

clauses.

$$\Gamma \vdash \mathrm{Nil} : \mathbf{1} \qquad \Gamma, x:\rho \vdash x:\rho$$

$$\frac{\Gamma, x:\rho \vdash M:\sigma}{\Gamma \vdash \lambda x:\rho.M:\rho\to\sigma} \qquad \frac{\Gamma, x:\tau \vdash M:\tau}{\Gamma \vdash \mathrm{rec}\, x:\tau.M:\tau}$$

$$\frac{\Gamma \vdash M:\rho\to\sigma \qquad \Gamma \vdash N:\rho}{\Gamma \vdash MN:\sigma} \qquad \frac{\Gamma \vdash M:\rho \qquad \Gamma \vdash N:\sigma}{\Gamma \vdash \mathrm{Pair}(M,N):\rho\times\sigma}$$

$$\frac{\Gamma \vdash M:\rho}{\Gamma \vdash \mathrm{Left}(M):\rho+\sigma} \qquad \frac{\Gamma \vdash M:\sigma}{\Gamma \vdash \mathrm{Right}(M):\rho+\sigma}$$

$$\frac{\Gamma \vdash M:\rho+\sigma \qquad \Gamma, x_1:\rho \vdash L:\tau \qquad \Gamma, x_2:\sigma \vdash R:\tau}{\Gamma \vdash \mathrm{case}M\ \mathrm{of}\{\mathrm{Left}(x_1)\to L; \mathrm{Right}(x_2)\to R\}:\tau}$$

$$\frac{\Gamma \vdash M:\rho\times\sigma \qquad \Gamma, x:\rho, y:\sigma \vdash N:\tau}{\Gamma \vdash \mathrm{case}M\ \mathrm{of}\{\mathrm{Pair}(x,y)\to N\}:\tau}$$

$$\frac{\Gamma \vdash M:\rho[\mathrm{fix}\,\alpha.\rho/\alpha]}{\Gamma \vdash \mathrm{In}(M):\mathrm{fix}\,\alpha.\rho}$$

$$\frac{\Gamma \vdash M:\mathrm{fix}\,\alpha.\rho \qquad \Gamma, x:\rho[\mathrm{fix}\,\alpha.\rho/\alpha] \vdash N:\sigma}{\Gamma \vdash \mathrm{case}M\ \mathrm{of}\{\mathrm{In}(x)\to N\}:\sigma}$$

Note that due to these rules a term can have more than one type.

2.3 Domain-theoretic Semantics

Since we can solve the domain equations as described in Section 2.1, we now define a particular domain **D** by a recursive domain equation which we will use to interpret types and terms.

Definition 2.3.1 We define the Scott domain **D** by the recursive domain equation:

$$\mathbf{D} \simeq (\mathbf{1}+\mathbf{D}+\mathbf{D}+\mathbf{D}+\mathbf{D}\times\mathbf{D}+[\mathbf{D}\to\mathbf{D}])_\perp$$

Using the constructors of **LoR** as names for the injections into the sum, each element in **D** has exactly one of the following forms: $\perp$, Nil, Left(a), Right(a), In(a), Pair(a,b),

$\mathrm{Fun}(f)$, where a and b range over $\mathbf{D}$, and f ranges over continuous function from $\mathbf{D}$ to $\mathbf{D}$.

It will be convenient to use the continuous functions

$$\mathrm{case}_{C_1,\ldots,C_n} : \mathbf{D} \to [\mathbf{D}^{\mathrm{arity}(C_1)} \to \mathbf{D}] \to \ldots \to [\mathbf{D}^{\mathrm{arity}(C_n)} \to \mathbf{D}] \to \mathbf{D}$$

$$\mathrm{case}_{C_1,\ldots,C_n}\, a\, f_1 \ldots f_n := \begin{cases} f_i(\vec{b_i}) & \text{if } a = C_i(\vec{b_i}), \\ \bot & \text{otherwise.} \end{cases}$$

We also use an informal lambda-notation $\lambda a.f(a)$ and composition $f \circ g$ to define continuous functions on $\mathbf{D}$. We do not prove the continuity in each case since this follows from well-known fact about the category of Scott domains and continuous functions. We also let $\mathrm{LFP} : [\mathbf{D} \to \mathbf{D}] \to \mathbf{D}$ be the continuous least fixed point operator, which can be defined by $\mathrm{LFP}(f) = \bigsqcup_n f^n(\bot)$.

The following definition gives an unexpected interpretation of a type ρ as a finitary projection $\langle\rho\rangle$, but from this one can derive a more familiar definition as a set, namely the image (or, equivalently, set of fixed points) of $\langle\rho\rangle$. Also, $\langle\rho\rangle$ will be used later in Theorem 2.4.16, and act as a function (not only a type), providing a link between the two semantics.

Definition 2.3.2 (Semantics of types) Let $[\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}}$ be the set of type environments, i.e., functions from **TVar** to $[\mathbf{D} \to \mathbf{D}]$.

For every type ρ we define $\langle\rho\rangle : [[\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}} \to [\mathbf{D} \to \mathbf{D}]]$

$$\begin{aligned}
\langle \mathbf{1} \rangle \zeta(a) &= \mathrm{case}_{\mathrm{Nil}}\, a\, \mathrm{Nil} \qquad \left(= \begin{cases} \mathrm{Nil} & \text{if } a = \mathrm{Nil} \\ \bot & \text{otherwise} \end{cases}\right) \\
\langle \alpha \rangle \zeta(a) &= \zeta(\alpha)(a) \\
\langle \rho + \sigma \rangle \zeta(a) &= \mathrm{case}_{\mathrm{Left,Right}}\, a\, (\mathrm{Left} \circ \langle\rho\rangle\zeta)\, (\mathrm{Right} \circ \langle\sigma\rangle\zeta) \\
\langle \rho \times \sigma \rangle \zeta(a) &= \mathrm{case}_{\mathrm{Pair}}\, a\, (\lambda b_1 b_2.\mathrm{Pair}(\langle\rho\rangle\zeta(b_1), \langle\sigma\rangle\zeta(b_2))) \\
\langle \rho \to \sigma \rangle \zeta(a) &= \mathrm{case}_{\mathrm{Fun}}\, a\, (\lambda f.\mathrm{Fun}(\langle\sigma\rangle\zeta \circ f \circ \langle\rho\rangle\zeta)) \\
\langle \mathrm{fix}\, \alpha.\rho \rangle \zeta &= \mathrm{LFP}(\lambda p.\lambda a.\mathrm{case}_{\mathrm{In}}\, a\, (\lambda b.\mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](b))))
\end{aligned}$$

We set $[\![\rho]\!]\zeta := (\langle\rho\rangle\zeta)(\mathbf{D})$.

Note that Definition 2.3.2 is well-defined since the category of domains is Cartesian closed and a continuous function has a least fixed point.

We call $\zeta : [\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}}$ a finitary projection if $\zeta(\alpha)$ is a finitary projection for all $\alpha \in \mathbf{TVar}$. Our goal is to prove that if ζ is a finitary projection, then $\langle\rho\rangle\zeta$ is a finitary projection. To achieve this, we first prove the following auxiliary lemmas.

Lemma 2.3.3

$$a \in \mathbf{D}_c \Rightarrow \mathrm{In}(a) \in \mathbf{D}_c.$$

Proof. Assume $x \in \mathbf{D}_c$. We need to show $\mathrm{In}(x) \in \mathbf{D}_c$, i.e. to show $\forall A \subseteq \mathbf{D}$ directed $(\mathrm{In}(x) \sqsubseteq \sqcup A \Rightarrow \exists y \in A.\mathrm{In}(x) \sqsubseteq y)$.

Assume $\mathrm{In}(x) \sqsubseteq \sqcup A$ where $A \subseteq \mathbf{D}$ is directed. Let $A' := \{z \mid \mathrm{In}(z) \in A\}$. Then there are two cases.

case1 A contains no constructor element, i.e. $A = \{\bot\}$. Thus, we get $\sqcup A = \bot \not\sqsupseteq \mathrm{In}(x)$, which is impossible.

case2 A contains a constructor. Since we have $\mathrm{In}(x) \sqsubseteq \sqcup A$ and only elements in the form of the same constructor can be ordered, this constructor must be In. Hence, without loss of generality, we have $\bot \notin A$ and $A = \{\mathrm{In}(z) \mid z \in A'\}$.

Let $z, z' \in A'$. Then $\mathrm{In}(z), \mathrm{In}(z') \in A$. Since A is directed, we get $\exists y \in A$ s.t. $\mathrm{In}(z) \sqsubseteq y$ and $\mathrm{In}(z') \sqsubseteq y$. Thus there exists some z'' s.t. $y = \mathrm{In}(z'')$. So $z \sqsubseteq z'' \in A'$ and $z' \sqsubseteq z'' \in A'$ since $z_1 \sqsubseteq z_2$ iff $\mathrm{In}(z_1) \sqsubseteq \mathrm{In}(z_2)$. Therefore, A' is directed.

Now we want to show that $x \sqsubseteq \sqcup A'$.

From assumption $\mathrm{In}(x) \sqsubseteq \sqcup A$, we get there exists some y, s.t. $\sqcup A = \mathrm{In}(y)$ where $\forall \mathrm{In}(z) \in A(\mathrm{In}(z) \sqsubseteq \mathrm{In}(y)) \wedge \forall d \in \mathbf{D}(\forall \mathrm{In}(z) \in A(\mathrm{In}(z) \sqsubseteq d) \rightarrow \mathrm{In}(y) \sqsubseteq d)$, and d must be in the form of $\mathrm{In}(d')$. Thus, we get $\forall z \in A'(z \sqsubseteq y) \wedge \forall d' \in \mathbf{D}(\forall z \in A'(z \sqsubseteq d') \rightarrow y \sqsubseteq d')$, i.e. $y = \sqcup A'$. Since $\mathrm{In}(x) \sqsubseteq \sqcup A$ by assumption, we get $\mathrm{In}(x) \sqsubseteq \mathrm{In}(y)$. Hence, we get $x \sqsubseteq y = \sqcup A'$.

Since x is compact, we get $\exists z \in A'.x \sqsubseteq z$. And then we get $\mathrm{In}(x) \sqsubseteq \mathrm{In}(z)$ and $\mathrm{In}(z) \in A$. □

Lemma 2.3.4

$$\forall C \subseteq \mathbf{D}_c(\sqcup C \text{ exists} \Rightarrow \mathrm{In}(\sqcup C) = \sqcup \mathrm{In}(C)).$$

Proof. Assume $\sqcup C$ exists. We need to show $\mathrm{In}(\sqcup C) = \sqcup \mathrm{In}(C)$.

"$\sqsubseteq$": We need to show $\mathrm{In}(\sqcup C) \sqsubseteq \sqcup \mathrm{In}(C)$.

Let x be an upper bound of $\mathrm{In}(C)$. We need to show $\mathrm{In}(\sqcup C) \sqsubseteq x$.

We have $\forall y \in C(\mathrm{In}(y) \sqsubseteq x)$. Then $x = \mathrm{In}(x')$ s.t. $\forall y \in C(y \sqsubseteq x')$. Thus, $\sqcup C \sqsubseteq x'$. So $\mathrm{In}(\sqcup C) \sqsubseteq \mathrm{In}(x') = x$.

"$\sqsupseteq$": We need to show $\sqcup \mathrm{In}(C) \sqsubseteq \mathrm{In}(\sqcup C)$.

Since $y \sqsubseteq \sqcup C$ for all $y \in C$, we get $\mathrm{In}(y) \sqsubseteq \mathrm{In}(\sqcup C)$ for all $y \in C$, i.e. $\mathrm{In}(\sqcup C)$ is an upper bound of $\mathrm{In}(C)$. Therefore, $\sqcup \mathrm{In}(C) \sqsubseteq \mathrm{In}(\sqcup C)$. □

The following lemma is used in Lemma 2.3.6.

Lemma 2.3.5 (Subdomain in the term of In) If A is a subdomain of $\mathbf{D}$, then $\mathrm{In}(A) \cup \{\bot\}$ is a subdomain of $\mathbf{D}$.

Proof. Let $\widetilde{A} := \mathrm{In}(A) \cup \{\bot\}$.

Assume A is a subdomain of $\mathbf{D}$. We need to show $\widetilde{A}$ is a subdomain of $\mathbf{D}$. By Lemma 2.1.7 it is to show there exists a finitary projection $p' : \mathbf{D} \to \mathbf{D}$ such that $\widetilde{A} = p'(\mathbf{D})$.

From assumption, we get there exists a finitary projection $p : \mathbf{D} \to \mathbf{D}$ such that $A = p(\mathbf{D})$ by Lemma 2.1.7.

Let $p'(x) := \mathrm{case}\, x\, \mathrm{of}\{\mathrm{In}(y) \to \mathrm{In}(p(y))\}$ for all $x \in \mathbf{D}$ and $y \in A$.

Now we need to show p' is a finitary projection. By Lemma 2.1.9 (c) it suffices to show $\exists A' \subseteq \mathbf{D}_c.\forall x \in \mathbf{D}(p'(x) = \sqcup\{a \in A' \mid a \sqsubseteq x\})$.

By assumption, we get $\exists B \subseteq \mathbf{D}_c.\forall x \in \mathbf{D}(p(x) = \sqcup\{a \in B \mid a \sqsubseteq x\})$ by Lemma 2.1.9 (c).

Then we need to show $\widetilde{B} = A'$.

If $p'(x) = \bot$, we get $\bot = \sqcup\{a \in \widetilde{B} \mid a \sqsubseteq \bot\}$.

If $p'(x) = p'(\mathrm{In}(y))$, we get $p'(\mathrm{In}(y)) = \mathrm{In}(p(y)) = \mathrm{In}(\sqcup\{a \in B \mid a \sqsubseteq y\}) = \sqcup\{\mathrm{In}(a) \mid a \in B, a \sqsubseteq y\}$ (by Lemma 2.3.4) $= \sqcup\{b \in \widetilde{B} \mid b \sqsubseteq \mathrm{In}(y)\}$.

It remains to show $\widetilde{A} = p'(\mathbf{D})$. By definition of $p(x)$ we get $p'(\mathbf{D}) = \mathrm{In}(p(\mathbf{D})) \cup \{\bot\}$. Then by $A = p(\mathbf{D})$, we get $p'(\mathbf{D}) = \mathrm{In}(p(\mathbf{D})) \cup \{\bot\} = \mathrm{In}(A) \cup \{\bot\} = \widetilde{A}$. □

Lemma 2.3.6 If ζ is a finitary projection, then $\langle \rho \rangle \zeta$ is a finitary projection.

Proof. By induction on ρ using Lemma 2.1.7 and 2.1.9.

- $\rho \equiv \mathbf{1}$.
  1. We need to show $\langle \mathbf{1} \rangle \zeta$ is a projection, i.e. to verify clauses of Definition 2.1.5.
     (a) By definition of $\langle \mathbf{1} \rangle \zeta$, $\langle \mathbf{1} \rangle \zeta$ is continuous.
     (b) We need to show $\forall x \in \mathbf{D}.\langle \mathbf{1} \rangle \zeta(x) \sqsubseteq x$.
     If $x = \mathrm{Nil}$, $\langle \mathbf{1} \rangle \zeta(\mathrm{Nil}) = \mathrm{Nil}$. Thus, $\mathrm{Nil} \sqsubseteq \mathrm{Nil}$.
     If $x \neq \mathrm{Nil}$, $\langle \mathbf{1} \rangle \zeta(x) = \bot$. Thus, $\bot \sqsubseteq x$.
     Therefore, $\langle \mathbf{1} \rangle \zeta(x) \sqsubseteq x$ for all $x \in \mathbf{D}$.
     (c) We need to show $\langle \mathbf{1} \rangle \zeta \circ \langle \mathbf{1} \rangle \zeta = \langle \mathbf{1} \rangle \zeta$.
     If $x = \mathrm{Nil}$, $\langle \mathbf{1} \rangle \zeta(\langle \mathbf{1} \rangle \zeta(\mathrm{Nil})) = \langle \mathbf{1} \rangle \zeta(\mathrm{Nil})$.
     If $x \neq \mathrm{Nil}$, $\langle \mathbf{1} \rangle \zeta(\langle \mathbf{1} \rangle \zeta(x)) = \langle \mathbf{1} \rangle \zeta(\bot) = \bot = \langle \mathbf{1} \rangle \zeta(x)$.
  2. We need to show $\langle \rho \rangle \zeta$ is finitary. Then to show $\exists A \subseteq \mathbf{D}_c.\forall x \in \mathbf{D}(\langle \mathbf{1} \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$ by Lemma 2.1.9 (c).
     Let $A = \{\mathrm{Nil}\}$, we get $\forall x \in \mathbf{D}(\langle \mathbf{1} \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$.
- $\rho \equiv \alpha$.
  Since $\langle \alpha \rangle \zeta(a) = \zeta(\alpha)(a)$ for all $a \in \mathbf{D}$, and $\zeta(\alpha)$ is a finitary projection, we get $\langle \alpha \rangle \zeta$ is a finitary projection.
- $\rho \equiv \rho + \sigma$.

1. We need to show $\langle \rho + \sigma \rangle \zeta$ is a projection, i.e. to verify clauses of Definition 2.1.5.
   (a) By definition of $\langle \rho + \sigma \rangle \zeta$, $\langle \rho + \sigma \rangle \zeta$ is continuous.
   (b) We need to show $\forall x \in \mathbf{D}. \langle \rho + \sigma \rangle \zeta(x) \sqsubseteq x$.
   If $x = \text{Left}(b)$, $\langle \rho + \sigma \rangle \zeta(\text{Left}(b)) = \text{Left}(\langle \rho \rangle \zeta(b))$.
   By I.H. we get $\langle \rho \rangle \zeta(b) \sqsubseteq b$. Thus, $\text{Left}(\langle \rho \rangle \zeta(b)) \sqsubseteq \text{Left}(b)$.
   Therefore, $\langle \rho + \sigma \rangle \zeta(\text{Left}(b)) \sqsubseteq \text{Left}(b)$.
   If $x = \text{Right}(b)$, $\langle \rho + \sigma \rangle \zeta(\text{Right}(b)) = \text{Right}(\langle \rho \rangle \zeta(b))$.
   By I.H. we get $\langle \rho \rangle \zeta(b) \sqsubseteq b$. Thus, $\text{Right}(\langle \rho \rangle \zeta(b)) \sqsubseteq \text{Right}(b)$.
   Therefore, $\langle \rho + \sigma \rangle \zeta(\text{Right}(b)) \sqsubseteq \text{Right}(b)$.
   Otherwise, $\langle \rho + \sigma \rangle \zeta(x) = \bot \sqsubseteq x$.
   (c) We need to show $\langle \rho + \sigma \rangle \zeta \circ \langle \rho + \sigma \rangle \zeta = \langle \rho + \sigma \rangle \zeta$.
   If $x = \text{Left}(b)$,

$$\begin{aligned} & \langle \rho + \sigma \rangle \zeta(\langle \rho + \sigma \rangle \zeta(\text{Left}(b))) \\ = \; & \langle \rho + \sigma \rangle \zeta(\text{Left}(\langle \rho \rangle \zeta(b))) \\ = \; & \text{Left}(\langle \rho \rangle \zeta(\langle \rho \rangle \zeta(b))) \\ = \; & \text{Left}(\langle \rho \rangle \zeta(b)) \qquad \text{(by I.H.)} \\ = \; & \langle \rho + \sigma \rangle \zeta(\text{Left}(b)) \end{aligned}$$

   If $x = \text{Right}(b)$, similar to the case $x = \text{Left}(b)$.
   Otherwise, $\langle \rho + \sigma \rangle \zeta(\langle \rho + \sigma \rangle \zeta(x)) = \langle \rho + \sigma \rangle \zeta(\bot) = \bot = \langle \rho + \sigma \rangle \zeta(x)$.
2. We need to show $\langle \rho + \sigma \rangle \zeta$ is finitary.
   By Lemma 2.1.9 (c), it is to show

$$\exists A \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \rho + \sigma \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$$

   If $x = \text{Left}(b)$, we get $\langle \rho + \sigma \rangle \zeta(\text{Left}(b)) = \text{Left}(\langle \rho \rangle \zeta(b))$. By the induction hyposis we get

$$\exists A \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \rho \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$$

   If $x = \text{Right}(b)$, we get $\langle \rho + \sigma \rangle \zeta(\text{Right}(b)) = \text{Right}(\langle \sigma \rangle \zeta(b))$. By the induction hyposis we get

$$\exists A \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \sigma \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$$

   Otherwise, $A = \{\bot\}$.

- $\rho \equiv \rho \times \sigma$.
  Similar to the proof of $\rho + \sigma$.

- $\rho \equiv \rho \to \sigma$.

  1. We need to show $\langle \rho \to \sigma \rangle \zeta$ is a projection, i.e. to verify clauses of Definition 2.1.5.

     (a) By definition of $\langle \rho \to \sigma \rangle \zeta$, $\langle \rho \to \sigma \rangle \zeta$ is continuous.

     (b) We need to show $\forall x \in \mathbf{D}. \langle \rho \to \sigma \rangle \zeta(x) \sqsubseteq x$.
     If $a = \mathrm{Fun}(f)$, to show $\langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(f)) \sqsubseteq \mathrm{Fun}(f)$. By Definition 2.3.2, we get $\langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(f)) = \mathrm{Fun}(g)$. So it is to show $\mathrm{Fun}(g) \sqsubseteq \mathrm{Fun}(f)$, i.e. $g \sqsubseteq f$, i.e. $\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta \sqsubseteq f$.
     By I.H. we get $\langle \sigma \rangle \zeta(f(\langle \rho \rangle(a))) \sqsubseteq f(\langle \rho \rangle(a))$. By I.H. we get $\langle \rho \rangle \zeta(a) \sqsubseteq a$. Since f is monotone/continuous, we get $f(\langle \rho \rangle(a)) \sqsubseteq f(a)$.
     Otherwise, $\langle \rho \to \sigma \rangle \zeta(a) = \bot \sqsubseteq a$.

     (c) We need to show $\langle \rho \to \sigma \rangle \zeta \circ \langle \rho \to \sigma \rangle \zeta = \langle \rho \to \sigma \rangle \zeta$.
     If $x = \mathrm{Fun}(f)$,

     $$\begin{aligned} & \langle \rho \to \sigma \rangle \zeta(\langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(f))) \\ = \; & \langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(g)) \text{ where } g = \langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta \\ = \; & \mathrm{Fun}(h) \text{ where } h = \langle \sigma \rangle \zeta \circ g \circ \langle \rho \rangle \zeta \\ = \; & \mathrm{Fun}(h) \text{ where } h = \langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta \qquad \text{(by I.H.)} \\ = \; & \langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(f)) \end{aligned}$$

     Otherwise, $\langle \rho \to \sigma \rangle \zeta(\langle \rho \to \sigma \rangle \zeta(x)) = \langle \rho \to \sigma \rangle \zeta(\bot) = \bot = \langle \rho \to \sigma \rangle \zeta(x)$.

  2. We need to show $\langle \rho \to \sigma \rangle \zeta$ is finitary. By Lemma 2.1.9 (c), to show $\exists A \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \rho \to \sigma \rangle \zeta(x) = \sqcup\{a \in A \mid a \sqsubseteq x\})$.
     If $x = \mathrm{Fun}(f)$, we get $\langle \rho \to \sigma \rangle \zeta(\mathrm{Fun}(f)) = \mathrm{Fun}(\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta)$. By the induction hyposis we get

     $$\begin{aligned} & \exists B \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \rho \rangle \zeta(x) = \sqcup\{b \in B \mid b \sqsubseteq x\}) \\ & \exists C \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \sigma \rangle \zeta(x) = \sqcup\{c \in C \mid c \sqsubseteq x\}) \end{aligned}$$

     Then we get $f(\langle \rho \rangle \zeta(x)) = \sqcup f(\{b \in B \mid b \sqsubseteq x\})$. Therefore $\exists C \subseteq \mathbf{D}_c. \forall x \in \mathbf{D}(\langle \rho \to \sigma \rangle \zeta(x) = \sqcup\{a \in C \mid a \sqsubseteq x\})$.
     Otherwise, $A = \{\bot\}$.

- $\rho \equiv \mathrm{fix}\, \alpha.\rho$.

  Let $f := \lambda p. \lambda a. \mathrm{case}_{\mathrm{In}}\, a\, (\lambda b. \mathrm{In}(\langle \rho \rangle \zeta[\alpha := p](b)))$.

  Hence $\langle \mathrm{fix}\, \alpha.\rho \rangle \zeta = \mathrm{LFP}(f) = \sqcup_{n \in N} f^n(\bot)$. Therefore, by Lemma 2.1.10, it suffices to show that $\forall n. f^n(\bot)$ is a finitary projection.

We do a side induction on n.

$n = 0$: We need to show $f^0(\bot)$ is a finitary projection.

Since $f^0(\bot) = \bot$ and $\bot$ is a finitary projection, we get $f^0(\bot)$ is a finitary projection.

$n+1$: Assume $p := f^n(\bot)$ is a finitary projection. We need to show $f(p)$ is a finitary projection.

First to show $f(p)$ is a projection.

We have $f(p) = \lambda a.\mathrm{case}_{\mathrm{In}}\ a\ (\lambda b.\mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](b)))$.

By definition of $\mathrm{case}_{\mathrm{In}}$, $f(p)$ is a continuous function, since by I.H. we get $\langle\rho\rangle\zeta[\alpha := p]$ is a continuous function/finitary projection.

We need to show $f(p) \sqsubseteq \mathrm{id}$.

We have $f(p)(x) = \begin{cases} \mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](b)) & \text{if } x = \mathrm{In}(b) \\ \bot & \text{otherwise} \end{cases}$ for all $x \in \mathbf{D}$.

$\mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](b)) \sqsubseteq \mathrm{In}(b)$, since by I.H. $\langle\rho\rangle\zeta[\alpha := p]$ is a projection.

Therefore, $f(p)(x) \sqsubseteq x$.

We need to show $f(p)(f(p)(x)) = f(p)(x)$.

If $x = \mathrm{In}(b)$, since by I.H. $\langle\rho\rangle\zeta[\alpha := p]$ is idempotent, we have $f(p)(f(p)(x)) = \mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](\langle\rho\rangle\zeta[\alpha := p](b))) = \mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](b)) = f(p)(x)$.

Otherwise, we get $f(p)(f(p)(x)) = \bot = f(p)(x)$.

Therefore, $f(p)$ is a projection.

We need to show $f(p)(\mathbf{D})$ is a subdomain of $\mathbf{D}$.

Let $A := \{\langle\rho\rangle\zeta[\alpha := p](b) \mid b \in \mathbf{D}\} = \langle\rho\rangle\zeta[\alpha := p](\mathbf{D})$.

By I.H. $\langle\rho\rangle\zeta[\alpha := p]$ is a finitary projection. Thus, A is a subdomain of $\mathbf{D}$.

By the definition of f, $f(p)(\mathbf{D}) = \mathrm{In}(A) \cup \{\bot\}$. This follows by Lemma 2.3.5.

□

Now we are ready to define the semantics of **LoR**-terms. The leading idea in the definition of the value of a typed lambda-abstraction $\lambda x : \rho.M$ is that the domain of the resulting function is (the semantics of) ρ. Therefore, the incoming argument a is first projected down to ρ.

Definition 2.3.7 (Semantics of terms) For all environments $\zeta : [\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}}$, $\eta : \mathbf{D}^{\mathbf{Var}}$, and every **LoR** term M we define the value $[\![M]\!]^\zeta \eta \in \mathbf{D}$.

$$\begin{aligned}
[\![x]\!]^\zeta \eta &= \eta(x) \\
[\![C(M_1,\ldots,M_n)]\!]^\zeta \eta &= C([\![M_1]\!]^\zeta \eta,\ldots,[\![M_n]\!]^\zeta \eta) \\
[\![MN]\!]^\zeta \eta &= \mathrm{case}_{\mathrm{Fun}}\ ([\![M]\!]^\zeta \eta)\ (\lambda f.f([\![N]\!]^\zeta \eta)) \\
[\![\lambda x:\rho.M]\!]^\zeta \eta &= \mathrm{Fun}(\lambda a.[\![M]\!]^\zeta \eta[x := \langle\rho\rangle\zeta(a)]) \\
[\![\mathrm{rec}\, x:\tau.M]\!]^\zeta \eta &= \mathrm{LFP}(\lambda a.[\![M]\!]^\zeta \eta[x := \langle\tau\rangle\zeta(a)]) \\
[\![\mathrm{case}\, M \,\mathrm{of}\, \{C_i(\vec{x}_i) \to R_i\}_i]\!]^\zeta \eta &= \mathrm{case}_{C_1,\ldots,C_n}\ ([\![M]\!]^\zeta \eta)\ (\lambda \vec{a}.[\![R_i]\!]^\zeta \eta[\vec{x}_i := \vec{a}])_i
\end{aligned}$$

Note that in the recursion case, the least fixed point exists by the Cartesian closedness.

One can prove the following soundness theorem, stating that if from a context Γ we can derive **LoR** term M with type ρ, and for every variable $x \in \mathrm{dom}(\Gamma)$, $\eta(x)$ is an element of $[\![\Gamma(x)]\!]\zeta$ (we will write $\eta \in [\![\Gamma]\!]\zeta$ for this), then the value of term M is an element of the value of type ρ.

Theorem 2.3.8 (Soundness For **LoR** terms) Let ζ be a finitary projection. If $\Gamma \vdash M : \rho$ and $\eta \in [\![\Gamma]\!]\zeta$, then $[\![M]\!]^\zeta \eta \in [\![\rho]\!]\zeta$.

Proof. By induction on the definition of the relation $\Gamma \vdash M : \rho$.

1. $\Gamma \vdash \mathrm{Nil} : \mathbf{1}$.

   We need to show $[\![\mathrm{Nil}]\!]^\zeta \eta \in [\![\mathbf{1}]\!]\zeta$.

   By Definition 2.3.7, we have $[\![\mathrm{Nil}]\!]^\zeta \eta = \mathrm{Nil}$.

   By Definition 2.3.2, we have $[\![\mathbf{1}]\!]\zeta = \{\mathrm{Nil}, \bot\}$. Thus, $[\![\mathrm{Nil}]\!]^\zeta \eta \in [\![\mathbf{1}]\!]\zeta$.

2. $\Gamma, x:\rho \vdash x : \rho$.

   We need to show $[\![x]\!]^\zeta \eta \in [\![\rho]\!]\zeta$.

   By Definition 2.3.7, we have $[\![x]\!]^\zeta \eta = \eta(x)$. By assumption, we have $\eta(x) \in [\![\rho]\!]\zeta$. Thus, $[\![x]\!]^\zeta \eta \in [\![\rho]\!]\zeta$.

3. $\dfrac{\Gamma, x:\rho \vdash M : \sigma}{\Gamma \vdash \lambda x:\rho.M : \rho \to \sigma}$.

   We need to show $[\![\lambda x:\rho.M]\!]^\zeta \eta \in [\![\rho \to \sigma]\!]\zeta$.

   By Definition 2.3.7, we have $[\![\lambda x:\rho.M]\!]^\zeta \eta := \mathrm{Fun}(f)$ where $\forall a \in \mathbf{D}.f(a) = [\![M]\!]^\zeta \eta[x := \langle\rho\rangle\zeta(a)]$.

   Then it is to show $\mathrm{Fun}(f) \in [\![\rho \to \sigma]\!]\zeta$, i.e. $\mathrm{Fun}(f) = \langle\rho \to \sigma\rangle\zeta(\mathrm{Fun}(f))$, by Definition 2.3.2, it is to show $f = \langle\sigma\rangle\zeta \circ f \circ \langle\rho\rangle\zeta$.

Let arbitrary $a \in \mathbf{D}$. It is to show $f(a) = \langle\sigma\rangle\zeta(f(\langle\rho\rangle\zeta(a)))$. The right-hand side of above equation

$$\begin{aligned} & \langle\sigma\rangle\zeta(f(\langle\rho\rangle\zeta(a))) \\ = \; & \langle\sigma\rangle\zeta([\![M]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(\langle\rho\rangle\zeta(a))]) \\ = \; & \langle\sigma\rangle\zeta([\![M]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)]) \end{aligned}$$

We need to show $[\![M]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)] = \langle\sigma\rangle\zeta([\![M]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)])$.

Define $\eta_a := \eta[x := \langle\rho\rangle\zeta(a)]$. We have the following

$$\Gamma, x : \rho \vdash M : \sigma \wedge \eta_a \in [\![\Gamma, x : \rho]\!]\zeta \Rightarrow [\![M]\!]^{\zeta}\eta_a \in [\![\sigma]\!]\zeta. \qquad \text{(IH1)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH1, we need to show $\eta_a \in [\![\Gamma, x : \rho]\!]\zeta$, i.e. to show $\eta_a(x) \in [\![\rho]\!]\zeta$.

This follows by $\eta_a(x) = \langle\rho\rangle\zeta(a) \in \langle\rho\rangle\zeta(\mathbf{D}) = [\![\rho]\!]\zeta$.

By IH1, we have $[\![M]\!]^{\zeta}\eta_a \in [\![\sigma]\!]\zeta$.

Thus, $[\![M]\!]^{\zeta}\eta_a = \langle\sigma\rangle\zeta([\![M]\!]^{\zeta}\eta_a)$.

4. $\dfrac{\Gamma, x : \tau \vdash M : \tau}{\Gamma \vdash \mathrm{rec}\, x : \tau.M : \tau}$.

We need to show $[\![\mathrm{rec}\, x : \tau.M]\!]^{\zeta}\eta \in [\![\tau]\!]\zeta$.

By Definition 2.3.7, we have $[\![\mathrm{rec}\, x : \tau.M]\!]^{\zeta}\eta = \mathrm{LFP}(f)$ where $f(a) := [\![M]\!]^{\zeta}\eta[x := \langle\tau\rangle\zeta(a)]$.

Then it is to show $\mathrm{LFP}(f) \in [\![\tau]\!]\zeta$.

Define $\eta' := \eta[x := \langle\tau\rangle\zeta(b)]$ where $b = \mathrm{LFP}(f)$. We have the following

$$\Gamma, x : \tau \vdash M : \tau \wedge \eta' \in [\![\Gamma, x : \tau]\!]\zeta \Rightarrow [\![M]\!]^{\zeta}\eta' \in [\![\tau]\!]\zeta. \qquad \text{(IH1)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH1, we need to show $\eta' \in [\![\Gamma, x : \tau]\!]\zeta$, i.e. to show $\eta'(x) \in [\![\tau]\!]\zeta$.

This follows by $\eta'(x) = \langle\tau\rangle\zeta(b) \in \langle\tau\rangle\zeta(\mathbf{D}) = [\![\tau]\!]\zeta$.

By IH1, we have $[\![M]\!]^{\zeta}\eta' \in [\![\tau]\!]\zeta$, i.e. $[\![M]\!]^{\zeta}\eta[x := \langle\tau\rangle\zeta(b)] \in [\![\tau]\!]\zeta$, i.e. $f(b) \in [\![\tau]\!]\zeta$.

Since $b = f(b)$, we have $\mathrm{LFP}(f) \in [\![\tau]\!]\zeta$.

5. $\dfrac{\Gamma \vdash M : \rho \to \sigma \qquad \Gamma \vdash N : \rho}{\Gamma \vdash MN : \sigma}$.

We need to show $[\![MN]\!]^{\zeta}\eta \in [\![\sigma]\!]\zeta$.

By Definition 2.3.7, we have two cases.

(1) If $[\![M]\!]^\zeta\eta = \mathrm{Fun}(f)$, then $[\![MN]\!]^\zeta\eta = f([\![N]\!]^\zeta\eta)$.
We need to show $f([\![N]\!]^\zeta\eta) \in [\![\sigma]\!]\zeta$, i.e. $f([\![N]\!]^\zeta\eta) = \langle\sigma\rangle\zeta(f([\![N]\!]^\zeta\eta))$.
We have the following

$$[\![M]\!]^\zeta\eta \in [\![\rho \to \sigma]\!]\zeta, \tag{IH1}$$

$$[\![N]\!]^\zeta\eta \in [\![\rho]\!]\zeta. \tag{IH2}$$

By IH1, we have $[\![M]\!]^\zeta\eta = \langle\rho \to \sigma\rangle\zeta([\![M]\!]^\zeta\eta)$, i.e. $\mathrm{Fun}(f) = \langle\rho \to \sigma\rangle\zeta(\mathrm{Fun}(f))$. By Definition 2.3.2, we have $f = \langle\sigma\rangle\zeta \circ f \circ \langle\rho\rangle\zeta$,
i.e. $f([\![N]\!]^\zeta\eta) = \langle\sigma\rangle\zeta(f(\langle\rho\rangle\zeta([\![N]\!]^\zeta\eta)))$.
By IH2, we have $[\![N]\!]^\zeta\eta = \langle\rho\rangle\zeta([\![N]\!]^\zeta\eta)$.
Then we get $f([\![N]\!]^\zeta\eta) = \langle\sigma\rangle\zeta(f([\![N]\!]^\zeta\eta))$.

(2) Otherwise, $[\![MN]\!]^\zeta\eta = \bot$. We need to show $\bot \in [\![\sigma]\!]\zeta$.
This follows by Definition 2.3.2.

6. $\dfrac{\Gamma \vdash M : \rho \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash \mathrm{Pair}(M,N) : \rho \times \sigma}$.

We need to show $[\![\mathrm{Pair}(M,N)]\!]^\zeta\eta \in [\![\rho \times \sigma]\!]\zeta$.

By Definition 2.3.7, we have $[\![\mathrm{Pair}(M,N)]\!]^\zeta\eta = \mathrm{Pair}([\![M]\!]^\zeta\eta, [\![N]\!]^\zeta\eta)$.

By Definition 2.3.2, we have $[\![\rho \times \sigma]\!]\zeta = \langle\rho \times \sigma\rangle\zeta(\mathbf{D}) = \mathrm{Pair}([\![\rho]\!]\zeta, [\![\sigma]\!]\zeta) \cup \{\bot\}$.

Then it is to show $\mathrm{Pair}([\![M]\!]^\zeta\eta, [\![N]\!]^\zeta\eta) \in \mathrm{Pair}([\![\rho]\!]\zeta, [\![\sigma]\!]\zeta)$,

i.e. to show $[\![M]\!]^\zeta\eta \in [\![\rho]\!]\zeta$ and $[\![N]\!]^\zeta\eta \in [\![\sigma]\!]\zeta$.

This follows by

$$[\![M]\!]^\zeta\eta \in [\![\rho]\!]\zeta, \tag{IH1}$$

$$[\![N]\!]^\zeta\eta \in [\![\sigma]\!]\zeta. \tag{IH2}$$

7. $\dfrac{\Gamma \vdash M : \rho}{\Gamma \vdash \mathrm{Left}(M) : \rho + \sigma}$.

We need to show $[\![\mathrm{Left}(M)]\!]^\zeta\eta \in [\![\rho + \sigma]\!]\zeta$.

By Definition 2.3.7, we have $[\![\mathrm{Left}(M)]\!]^\zeta\eta = \mathrm{Left}([\![M]\!]^\zeta\eta)$.

By Definition 2.3.2, we have $[\![\rho + \sigma]\!]\zeta = \langle\rho + \sigma\rangle\zeta(\mathbf{D}) = \{\langle\rho + \sigma\rangle\zeta(a) \mid a = \mathrm{Left}(b), b \in \mathbf{D}\} \cup \{\langle\rho + \sigma\rangle\zeta(a) \mid a = \mathrm{Right}(b), b \in \mathbf{D}\} \cup \{\bot\}$, and $\langle\rho + \sigma\rangle\zeta(a) = \mathrm{Left}(\langle\rho\rangle\zeta(b))$ if $a = \mathrm{Left}(b)$.

We need to show $\mathrm{Left}([\![M]\!]^\zeta\eta) \in \{\mathrm{Left}(\langle\rho\rangle\zeta(b)) \mid b \in \mathbf{D}\} = \mathrm{Left}([\![\rho]\!]\zeta)$, i.e. to show $[\![M]\!]^\zeta\eta \in [\![\rho]\!]\zeta$.

This follows by

$$[\![M]\!]^{\zeta}\eta \in [\![\rho]\!]\zeta. \qquad \text{(IH1)}$$

8. $\dfrac{\Gamma \vdash M : \sigma}{\Gamma \vdash \mathrm{Right}(M) : \rho + \sigma}$.

We need to show $[\![\mathrm{Right}(M)]\!]^{\zeta}\eta \in [\![\rho + \sigma]\!]\zeta$.

Similar to the case Left.

9. $\dfrac{\Gamma \vdash M : \rho + \sigma \qquad \Gamma, x_1 : \rho \vdash L : \tau \qquad \Gamma, x_2 : \sigma \vdash R : \tau}{\Gamma \vdash \mathrm{case}\, M \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L; \mathrm{Right}(x_2) \to R\} : \tau}$.

We need to show $[\![\mathrm{case}\, M \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L; \mathrm{Right}(x_2) \to R\}]\!]^{\zeta}\eta \in [\![\tau]\!]\zeta$.

Let $[\![\mathrm{case}]\!]^{\zeta}\eta$ be $[\![\mathrm{case}\, M \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L; \mathrm{Right}(x_2) \to R\}]\!]^{\zeta}\eta$.

By Definition 2.3.7, we have three cases.

(1) If $[\![M]\!]^{\zeta}\eta = \mathrm{Left}(\langle\rho\rangle\zeta(a))$.
Then $[\![\mathrm{case}]\!]^{\zeta}\eta = [\![L]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)]$.
It is to show $[\![L]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)] \in [\![\tau]\!]\zeta$.
We have the following

$$\forall a \in \mathbf{D}.[\![L]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)] \in [\![\tau]\!]\zeta. \qquad \text{(IH1)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH1, we need to show $\forall y \in \mathrm{dom}(\Gamma, x : \rho).\ \eta[x := \langle\rho\rangle\zeta(a)](y) \in [\![(\Gamma, x : \rho)(y)]\!]\zeta$.
The checking is similar to the previous one.
Applying IH1, we have $[\![L]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a)] \in [\![\tau]\!]\zeta$.

(2) If $[\![M]\!]^{\zeta}\eta = \mathrm{Right}(\langle\sigma\rangle\zeta(a))$.
Then $[\![\mathrm{case}]\!]^{\zeta}\eta = [\![R]\!]^{\zeta}\eta[x := \langle\sigma\rangle\zeta(a)]$.
It is to show $[\![R]\!]^{\zeta}\eta[x := \langle\sigma\rangle\zeta(a)] \in [\![\tau]\!]\zeta$.
We have the following

$$\forall a \in \mathbf{D}.[\![R]\!]^{\zeta}\eta[x := \langle\sigma\rangle\zeta(a)] \in [\![\tau]\!]\zeta. \qquad \text{(IH2)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH2, we need to show $\forall y \in \mathrm{dom}(\Gamma, x : \sigma).\ \eta[x := \langle\sigma\rangle\zeta(a)](y) \in [\![(\Gamma, x : \sigma)(y)]\!]\zeta$.
The checking is similar to the previous one.
Applying IH2, we have $[\![R]\!]^{\zeta}\eta[x := \langle\sigma\rangle\zeta(a)] \in [\![\tau]\!]\zeta$.

(3) Otherwise, $[\![\mathrm{case}]\!]^{\zeta}\eta = \bot$. We need to show $\bot \in [\![\tau]\!]\zeta$.
This follows by Definition 2.3.2.

10. $\dfrac{\Gamma \vdash M : \rho \times \sigma \qquad \Gamma, x:\rho, y:\sigma \vdash N : \tau}{\Gamma \vdash \mathrm{case}\, M\, \mathrm{of}\, \{\mathrm{Pair}(x,y) \to N\} : \tau}$.

We need to show $[\![\mathrm{case}\, M\, \mathrm{of}\, \{\mathrm{Pair}(x,y) \to N\}]\!]^{\zeta}\eta \in [\![\tau]\!]\zeta$.

Let $[\![\mathrm{case}]\!]^{\zeta}\eta$ be $[\![\mathrm{case}\, M\, \mathrm{of}\, \{\mathrm{Pair}(x,y) \to N\}]\!]^{\zeta}\eta$.

By Definition 2.3.7, we have two cases.

(1) If $[\![M]\!]^{\zeta}\eta = \mathrm{Pair}(\langle\rho\rangle\zeta(a), \langle\sigma\rangle\zeta(b))$.
Then $[\![\mathrm{case}]\!]^{\zeta}\eta = [\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)]$.
It is to show $[\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \in [\![\tau]\!]\zeta$.
We have the following

$$\forall a, b \in \mathbf{D}.[\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \in [\![\tau]\!]\zeta. \qquad \text{(IH1)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH1, we need to show $\forall z \in \mathrm{dom}(\Gamma, x:\rho, y:\sigma).\ \eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)](z) \in [\![(\Gamma, x:\rho, y:\sigma)(z)]\!]\zeta$.
The checking is similar to the previous one.
Applying IH1, we have $[\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \in [\![\tau]\!]\zeta$.

(2) Otherwise, $[\![\mathrm{case}]\!]^{\zeta}\eta = \bot$. We need to show $\bot \in [\![\tau]\!]\zeta$.
This follows by Definition 2.3.2.

11. $\dfrac{\Gamma \vdash M : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]}{\Gamma \vdash \mathrm{In}(M) : \mathrm{fix}\,\alpha.\rho'}$.

We need to show $[\![\mathrm{In}(M)]\!]^{\zeta}\eta \in [\![\mathrm{fix}\,\alpha.\rho']\!]\zeta$.

By Definition 2.3.7, we have $[\![\mathrm{In}(M)]\!]^{\zeta}\eta = \mathrm{In}([\![M]\!]^{\zeta}\eta)$.

Then it is to show $\mathrm{In}([\![M]\!]^{\zeta}\eta) \in [\![\mathrm{fix}\,\alpha.\rho']\!]\zeta$.

By Definition 2.3.2, we have

$$\begin{aligned}
& [\![\mathrm{fix}\,\alpha.\rho']\!]\zeta \\
= \ & \mathrm{LFP}(\lambda p.\lambda a.\mathrm{case}_{\mathrm{In}}\ a\ (\lambda b.\mathrm{In}(\langle\rho'\rangle\zeta[\alpha := p](b))))(\mathbf{D}) \\
= \ & \{\mathrm{In}(\langle\rho'\rangle\zeta[\alpha := \mathrm{fix}\,\alpha.\rho'](b)) \mid b \in \mathbf{D}\} \cup \{\bot\}\ (\text{taking } p := \mathrm{fix}\,\alpha.\rho') \\
= \ & \mathrm{In}(\langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(\mathbf{D})) \cup \{\bot\} \\
= \ & \mathrm{In}([\![\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]]\!]\zeta) \cup \{\bot\}
\end{aligned}$$

We need to show $\mathrm{In}([\![M]\!]^{\zeta}\eta) \in \mathrm{In}([\![\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]]\!]\zeta)$.

This follows by

$$[\![M]\!]^{\zeta}\eta \in [\![\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]]\!]\zeta. \qquad \text{(IH1)}$$

12. $\dfrac{\Gamma \vdash M : \mathrm{fix}\,\alpha.\rho' \qquad \Gamma, x : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha] \vdash N : \sigma}{\Gamma \vdash \mathrm{case}\,M\,\mathrm{of}\,\{\mathrm{In}(x) \to N\} : \sigma}$.

We need to show $[\![\mathrm{case}\,M\,\mathrm{of}\,\{\mathrm{In}(x) \to N\}]\!]^{\zeta}\eta \in [\![\sigma]\!]\zeta$.

Let $[\![\mathrm{case}]\!]^{\zeta}\eta$ be $[\![\mathrm{case}\,M\,\mathrm{of}\,\{\mathrm{In}(x) \to N\}]\!]^{\zeta}\eta$.

By Definition 2.3.7, we have two cases.

(1) If $[\![M]\!]^{\zeta}\eta = \mathrm{In}(\langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a))$.
Then $[\![\mathrm{case}]\!]^{\zeta}\eta = [\![N]\!]^{\zeta}\eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)]$.
It is to show $[\![N]\!]^{\zeta}\eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)] \in [\![\sigma]\!]\zeta$.
We have the following

$$\forall a \in \mathbf{D}.[\![N]\!]^{\zeta}\eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)] \in [\![\sigma]\!]\zeta. \qquad \text{(IH1)}$$

Since we have the assumption $\eta \in [\![\Gamma]\!]\zeta$, in order to apply IH1, we need to show

$$\forall y \in \mathrm{dom}(\Gamma, x : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]).\ \eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)](y) \in [\![(\Gamma, x : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha])(y)]\!]\zeta$$

The checking is similar to the previous one.
Applying IH1, we have $[\![N]\!]^{\zeta}\eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)] \in [\![\sigma]\!]\zeta$.

(2) Otherwise, $[\![\mathrm{case}]\!]^{\zeta}\eta = \bot$. We need to show $\bot \in [\![\sigma]\!]\zeta$.
This follows by Definition 2.3.2.

□

2.4 Relating Typed and Untyped Terms

We now relate the semantics of typed terms with the semantics of untyped terms which are defined exactly as typed terms except that the type annotations for abstraction and recursion are omitted:

Definition 2.4.1 (Untyped terms)

$$\mathbf{LoR}^{-} \ni M, N, R_i ::= x \mid \lambda x.M \mid MN \mid \mathrm{rec}\,x.M \mid C(M_1, \ldots, M_n) \mid \mathrm{case}\,M\,\mathrm{of}\,\{C_i(\vec{x_i}) \to R_i\}_{i \in \{1,\ldots,n\}}$$

The same provisions made in Definition 2.2.2 for typed terms apply here.

The semantics of untyped terms is straightforward. It can be defined exactly as in the typed case except that the type environment $\zeta : [\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}}$ and finitary projections involved in typed abstraction and recursion are omitted.

Definition 2.4.2 (Semantics of untyped terms) For every environment $\eta : \mathbf{Var} \to \mathbf{D}$ and every $\mathbf{LoR}^-$ term M we define the value $[\![M]\!]\eta \in \mathbf{D}$.

$$\begin{aligned}
[\![x]\!]\eta &= \eta(x) \\
[\![C(M_1,\ldots,M_n)]\!]\eta &= C([\![M_1]\!]\eta,\ldots,[\![M_n]\!]\eta) \\
[\![MN]\!]\eta &= \mathrm{case}_{\mathrm{Fun}}\,([\![M]\!]\eta)\,(\lambda f.f([\![N]\!]\eta)) \\
[\![\lambda x.M]\!]\eta &= \mathrm{Fun}(\lambda a.[\![M]\!]\eta[x := a]) \\
[\![\mathrm{rec}\,x.M]\!]\eta &= \mathrm{LFP}(\lambda a.[\![M]\!]\eta[x := a]) \\
[\![\mathrm{case}\,M\,\mathrm{of}\,\{C_i(x_i) \to R_i\}_i]\!]\eta &= \mathrm{case}_{C_1,\ldots,C_n}\,([\![M]\!]\eta)\,(\lambda\vec{a}.[\![R_i]\!]\eta[\vec{x}_i := \vec{a}])_i
\end{aligned}$$

Our main result, the Coincidence Theorem 2.4.16, only applies to terms that are typed w.r.t. to a restricted notion of types where fixed point types $\mathrm{fix}\,\alpha.\rho$ are allowed only if ρ is positive in α.

Definition 2.4.3 (ρ positive/negative in α)
α is positive in α.
$\mathbf{1}$ is positive and negative in α.
$\rho \to \sigma$ is positive in α if ρ is negative in α and σ is positive in α.
$\rho \to \sigma$ is negative in α if ρ is positive in α and σ is negative in α.
$\rho + \sigma$ and $\rho \times \sigma$ are positive in α if ρ and σ are positive in α.
$\rho + \sigma$ and $\rho \times \sigma$ are negative in α if ρ and σ are negative in α.
$\mathrm{fix}\,\beta.\rho$ is positive in α if $\alpha = \beta$ or ρ is positive in α.
$\mathrm{fix}\,\beta.\rho$ is negative in α if $\alpha = \beta$ or ρ is negative in α.

Definition 2.4.4 (Regular types) We define *regular* types ρ as follows.
$\mathbf{1}$ is regular.
α is regular.
$\rho + \sigma$, $\rho \times \sigma$, $\rho \to \sigma$ are regular if ρ and σ are regular.
$\mathrm{fix}\,\alpha.\rho$ is regular if ρ is regular and ρ is positive in α.

Example 2.4.5 $\mathrm{fix}\,\alpha.(\alpha \to \alpha)$ is not regular, since $\alpha \to \alpha$ is not positive in α.

In the following all types are assumed to be **regular**.

The proof of our main result uses a logical relation based on the notion of *admissibility*. It has been used in [AJ94] and generalised in [Pit93], where it is used to prove properties of least fixed points.

Definition 2.4.6 (Admissible relation) A relation $r \subseteq \mathbf{D}^2$ is called *admissible* if it satisfies

1. $(\bot,\bot) \in r$.
2. If $(d_n,d'_n) \in r$ and $(d_n,d'_n) \sqsubseteq (d_{n+1},d'_{n+1})$ for all n, then $\sqcup_{n\in N}(d_n,d'_n) \in r$.

Note that a finite relation $r \subseteq \mathbf{D}^2$ with $(\bot,\bot) \in r$ is always admissible.
Let $\mathbf{Ad} := \{r \subseteq \mathbf{D}^2 \mid r \text{ is admissible}\}$.

Lemma 2.4.7 **Ad** is a complete lattice.

Proof. We know $(\mathscr{P}(\mathbf{D}^2), \subseteq)$ is a complete lattice, and if $X \subseteq \mathscr{P}(\mathbf{D}^2)$, then $\sqcup X = \cup X$ and $\sqcap X = \cap X$. Clearly, $\mathbf{Ad} \subseteq \mathscr{P}(\mathbf{D}^2)$.

We need to show **Ad** is a complete lattice. It suffices to show that either every subset has a least upper bound or every subset has a greatest lower bound. First, we show that if we have a set of admissible relations, the intersection is admissible again. And then it follows this intersection is the greatest lower bound in **Ad**.

Let $X \subseteq \mathbf{Ad}$. Then $\cap X = \{(x,y) \in \mathbf{D}^2 \mid \forall r \in X.(x,y) \in r\}$.
We need to show $\cap X$ is admissible, i.e. to show the following two statements hold.

- $(\bot,\bot) \in \cap X$.

  All relations $r \in X$ are admissible and every admissible r contains $(\bot,\bot)$.

  Therefore, $(\bot,\bot) \in \cap X$.

- If $(x_n,y_n) \in \cap X$ and $(x_n,y_n) \sqsubseteq (x_{n+1},y_{n+1})$ for all n, then $\sqcup_{n\in N}(x_n,y_n) \in \cap X$.

  Since **D** is a domain, $\sqcup_n x_n$ and $\sqcup_n y_n$ exist. Thus, $\sqcup_{n\in N}(x_n,y_n) = (\sqcup_n x_n, \sqcup_n y_n)$ exists.

  Let $r \in X$. We need to show $\sqcup_{n\in N}(x_n,y_n) \in r$, i.e. $(x,y) \in r$ where $x = \sqcup_n x_n$ and $y = \sqcup_n y_n$.

  Since $X \subseteq \mathbf{Ad}$, r is admissible.

  By assumption $(x_n,y_n) \in r$ for all n. Thus, $(x,y) \in r$.

  Therefore, $\sqcup_{n\in N}(x_n,y_n) \in \cap X$.

□

To prove our main result we define a logical relation $\sim_\rho^R \subseteq \mathbf{D} \times \mathbf{D}$ which can intuitively be understood as a notion of equivalence of elements of a *regular* type ρ. We use the informal (second-order) lambda abstraction $(\Lambda r \in \mathbf{Ad}.)$ to define functions on the set **Ad** of admissible relations on **D**.

Definition 2.4.8, Lemma 2.4.10 and Lemma 2.4.9 below should be considered simultaneously, in order to guarantee that the definition is well-defined. We use the fact that every monotone function f on a complete lattice L has a least fixed point $\mathrm{LFP}_L(f)$.

Definition 2.4.8 (Logical Relation) We define a relation $\sim_\rho^R \subseteq \mathbf{D} \times \mathbf{D}$ for every regular type ρ and every family of admissible relations $R \in \mathbf{Ad}^{\mathbf{TVar}}$ (i.e. $R(\alpha) \in \mathbf{Ad}$ for all

$\alpha \in \mathbf{TVar}$).

$$
\begin{aligned}
\sim_{\mathbf{1}}^{R} &:= \{(\bot,\bot),(\mathrm{Nil},\mathrm{Nil})\} \\
\sim_{\alpha}^{R} &:= R(\alpha) \\
\sim_{\rho_1\times\rho_2}^{R} &:= \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_{\rho_1}^{R} b_1, a_2 \sim_{\rho_2}^{R} b_2\} \\
\sim_{\rho_1+\rho_2}^{R} &:= \{(\bot,\bot)\} \cup \{(\mathrm{Left}(a_1),\mathrm{Left}(b_1)) \mid a_1 \sim_{\rho_1}^{R} b_1\} \\
&\qquad \cup \{(\mathrm{Right}(a_2),\mathrm{Right}(b_2)) \mid a_2 \sim_{\rho_2}^{R} b_2\} \\
\sim_{\rho\to\sigma}^{R} &:= \{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \\
&\qquad \forall a,b \in \mathbf{D}(a \sim_{\rho}^{R} b \Rightarrow f(a) \sim_{\sigma}^{R} g(b))\} \\
\sim_{\mathrm{fix}\,\alpha.\rho}^{R} &:= \mathrm{LFP}_{\mathbf{Ad}}(\Lambda r \in \mathbf{Ad}.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_{\rho}^{R[\alpha:=r]} b\})
\end{aligned}
$$

Remark Logical relations [Plo73] have been used successfully to prove properties of typed systems. Famous examples are the strong normalisation proofs by Tait and Girard using logical relations called computability predicates or reducibility candidates. The crucial feature of a logical relation is that it is a family of relations indexed by types and defined by induction on types such that all type constructors are interpreted by their logical interpretations, e.g. $\to$ is interpreted as logical implication.

Lemma 2.4.9

(1) If ρ is positive in α, then $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho}^{R[\alpha:=r]}$ is monotone in r.

(2) If ρ is negative in α, then $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho}^{R[\alpha:=r]}$ is anti-monotone in r.

Proof. By induction on ρ.

1. $\rho \equiv \mathbf{1}$.

   For (1), to show $\sim_{\mathbf{1}}^{R[\alpha:=r_1]} \subseteq \sim_{\mathbf{1}}^{R[\alpha:=r_2]}$ for $r_1 \subseteq r_2$.

   For (2), to show $\sim_{\mathbf{1}}^{R[\alpha:=r_2]} \subseteq \sim_{\mathbf{1}}^{R[\alpha:=r_1]}$ for $r_1 \subseteq r_2$.

   By Definition 2.4.8 we have $\sim_{\mathbf{1}}^{R[\alpha:=r_1]} = \sim_{\mathbf{1}}^{R[\alpha:=r_2]} = \{(\bot,\bot),(\mathrm{Nil},\mathrm{Nil})\}$.

   Proved by equality.

2. $\rho \equiv \alpha$.

   Since α is not negative in α, there is only one case.

   We need to show $\sim_{\alpha}^{R[\alpha:=r_1]} \subseteq \sim_{\alpha}^{R[\alpha:=r_2]}$ for $r_1 \subseteq r_2$.

   We get $\sim_{\alpha}^{R[\alpha:=r_1]} = R[\alpha := r_1](\alpha) = r_1$ and $\sim_{\alpha}^{R[\alpha:=r_2]} = R[\alpha := r_2](\alpha) = r_2$ by Definition 2.4.8.

   We get $\sim_{\alpha}^{R[\alpha:=r_1]} \subseteq \sim_{\alpha}^{R[\alpha:=r_2]}$ by $r_1 \subseteq r_2$.

3. $\rho \equiv \rho_1 \times \rho_2$.

   For (1), to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_1\times\rho_2}^{R[\alpha:=r]}$ is monotone, i.e. $\sim_{\rho_1\times\rho_2}^{R[\alpha:=r_1]} \subseteq \sim_{\rho_1\times\rho_2}^{R[\alpha:=r_2]}$ for $r_1 \subseteq r_2$.

   For (2), to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_1\times\rho_2}^{R[\alpha:=r]}$ is anti-monotone, i.e. $\sim_{\rho_1\times\rho_2}^{R[\alpha:=r_2]} \subseteq \sim_{\rho_1\times\rho_2}^{R[\alpha:=r_1]}$ for $r_1 \subseteq r_2$.

   We get $\sim_{\rho_1\times\rho_2}^{R[\alpha:=r_1]} = \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_{\rho_1}^{R[\alpha:=r_1]} b_1, a_2 \sim_{\rho_2}^{R[\alpha:=r_1]} b_2\}$, and $\sim_{\rho_1\times\rho_2}^{R[\alpha:=r_2]} = \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_{\rho_1}^{R[\alpha:=r_2]} b_1, a_2 \sim_{\rho_2}^{R[\alpha:=r_2]} b_2\}$ by Definition 2.4.8.

   We need to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_1}^{R[\alpha:=r]}$ and $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_2}^{R[\alpha:=r]}$ are monotone and anti-monotone.

   For (1), since $\rho_1 \times \rho_2$ is positive in α, we get ρ_1 and ρ_2 are positive in α. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_1}^{R[\alpha:=r]}$ and $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_2}^{R[\alpha:=r]}$ are monotone.

   For (2), since $\rho_1 \times \rho_2$ is negative in α, we get ρ_1 and ρ_2 are negative in α. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_1}^{R[\alpha:=r]}$ and $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho_2}^{R[\alpha:=r]}$ are anti-monotone.

4. $\rho \equiv \rho_1 + \rho_2$.

   Similar to the proof of $\rho_1 \times \rho_2$.

5. $\rho \equiv \rho \to \sigma$.

   For (1), to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho\to\sigma}^{R[\alpha:=r]}$ is monotone, i.e. $\sim_{\rho\to\sigma}^{R[\alpha:=r_1]} \subseteq \sim_{\rho\to\sigma}^{R[\alpha:=r_2]}$ for $r_1 \subseteq r_2$.

   For (2), to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho\to\sigma}^{R[\alpha:=r]}$ is anti-monotone, i.e. $\sim_{\rho\to\sigma}^{R[\alpha:=r_2]} \subseteq \sim_{\rho\to\sigma}^{R[\alpha:=r_1]}$ for $r_1 \subseteq r_2$.

   By Definition 2.4.8 we have $\sim_{\rho\to\sigma}^{R[\alpha:=r_1]} =$
   $\{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}(a \sim_{\rho}^{R[\alpha:=r_1]} b \Rightarrow f(a) \sim_{\sigma}^{R[\alpha:=r_1]} g(b))\}$,
   and $\sim_{\rho\to\sigma}^{R[\alpha:=r_2]} =$
   $\{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}(a \sim_{\rho}^{R[\alpha:=r_2]} b \Rightarrow f(a) \sim_{\sigma}^{R[\alpha:=r_2]} g(b))\}$.

   For (1), assume $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho}^{R[\alpha:=r]}$ is anti-monotone, to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\sigma}^{R[\alpha:=r]}$ is monotone.

   Since $\rho \to \sigma$ is positive in α, we get ρ is negative in α and σ is positive in α. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\sigma}^{R[\alpha:=r]}$ is monotone.

   For (2), assume $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\rho}^{R[\alpha:=r]}$ is monotone, to show $\Lambda r \subseteq \mathbf{D}^2.\ \sim_{\sigma}^{R[\alpha:=r]}$ is anti-monotone.

Since $\rho \to \sigma$ is negative in α, we get ρ is positive in α and σ is negative in α. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\ \sim_\sigma^{R[\alpha:=r]}$ is anti-monotone.

6. $\rho \equiv \mathrm{fix}\,\beta.\rho$.

   For (1), to show $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r]}$ is monotone, i.e. $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_1]} \subseteq \sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_2]}$ for $r_1 \subseteq r_2$.

   For (2), to show $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r]}$ is anti-monotone, i.e. $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_2]} \subseteq \sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_1]}$ for $r_1 \subseteq r_2$.

   By Definition 2.4.8 we have

   $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_1]} = \mathrm{LFP}(\Lambda r' \subseteq \mathbf{D}^2.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_\rho^{R[\alpha:=r_1][\beta:=r']} b\})$, and

   $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_2]} = \mathrm{LFP}(\Lambda r' \subseteq \mathbf{D}^2.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_\rho^{R[\alpha:=r_2][\beta:=r']} b\})$.

   Let $\sim_{\mathrm{fix}\,\beta.\rho}^{R[\alpha:=r_i]} = \mathrm{LFP}(\Phi_i)$ $(i = 1,2)$ where $\Phi_i : \mathscr{P}(\mathbf{D}^2) \to \mathscr{P}(\mathbf{D}^2)$ and

   $\Phi_i(r') = \{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_\rho^{R[\alpha:=r_i][\beta:=r']} b\}$.

   By Lemma 2.1.11 with setting $L := \mathscr{P}(\mathbf{D}^2)$, it is to show $\Phi_1 \subseteq \Phi_2$ or $\Phi_2 \subseteq \Phi_1$ respectively for (1) and (2), i.e. to show $\Lambda r \subseteq \mathbf{D}^2.\Lambda r' \subseteq \mathbf{D}^2.\ \sim_\rho^{R[\alpha:=r][\beta:=r']}$ are monotone and anti-monotone.

   For (1), since $\mathrm{fix}\,\beta.\rho$ is positive in α, we get ρ is positive in α and β. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\Lambda r' \subseteq \mathbf{D}^2.\ \sim_\rho^{R[\alpha:=r][\beta:=r']}$ is monotone.

   For (2), since $\mathrm{fix}\,\beta.\rho$ is negative in α, we get ρ is negative in α and β. By I.H. we get $\Lambda r \subseteq \mathbf{D}^2.\Lambda r' \subseteq \mathbf{D}^2.\ \sim_\rho^{R[\alpha:=r][\beta:=r']}$ is anti-monotone.

□

Lemma 2.4.10 We assume $R \in \mathbf{Ad}^{\mathbf{TVar}}$. Then $\sim_\rho^R$ is admissible.

Proof. By induction on ρ.

Trivially we have $(\bot,\bot)$ in all relations. So we now focus on proofs of the second clause of admissibility (Definition 2.4.6).

- $\rho \equiv \mathbf{1}$.

  Since $\sim_{\mathbf{1}}^R$ is a finite relation containing $(\bot,\bot)$, we get $\sim_{\mathbf{1}}^R$ is admissible.

- $\rho_1 \times \rho_2$.

  Let $(d_n)_{n\in N}$ be a chain in $\sim_{\rho_1\times\rho_2}^R$. Then $d_n = (\mathrm{Pair}(a_n,a'_n),\mathrm{Pair}(b_n,b'_n))$ where $a_n \sqsubseteq a_{n+1}$, $a'_n \sqsubseteq a'_{n+1}$, $b_n \sqsubseteq b_{n+1}$, $b'_n \sqsubseteq b'_{n+1}$ for all n. Hence, $(a_n,b_n)_{n\in N}$ is a chain in $\sim_{\rho_1}^R$ and $(a'_n,b'_n)_{n\in N}$ is a chain in $\sim_{\rho_2}^R$.

Since $\sim^R_{\rho_1}$ and $\sim^R_{\rho_2}$ are admissible, we get $\sqcup(a_n, b_n) = (\sqcup a_n, \sqcup b_n) \in \sim^R_{\rho_1}$ and $\sqcup(a'_n, b'_n) = (\sqcup a'_n, \sqcup b'_n) \in \sim^R_{\rho_2}$. So $(\mathrm{Pair}(\sqcup a_n, \sqcup b_n), \mathrm{Pair}(\sqcup a'_n, \sqcup b'_n)) \in \sim^R_{\rho_1 \times \rho_2}$.

Since $(\mathrm{Pair}(\sqcup a_n, \sqcup b_n), \mathrm{Pair}(\sqcup a'_n, \sqcup b'_n)) = \sqcup(\mathrm{Pair}(a_n, a'_n), \mathrm{Pair}(b_n, b'_n)) = \sqcup d_n$, we get $\sqcup d_n \in \sim^R_{\rho_1 \times \rho_2}$.

- $\rho \equiv \rho_1 + \rho_2$.

  Let $(d_n)_{n \in N}$ be a chain in $\sim^R_{\rho_1 + \rho_2}$. Since it is an increasing chain, then d_n could be $(\bot, \bot)$, or $(\mathrm{Left}(a_n), \mathrm{Left}(b_n))$, or $(\mathrm{Right}(a_n), \mathrm{Right}(b_n))$. w.l.g. we get $d_n = (\mathrm{Left}(a_n), \mathrm{Left}(b_n))$ and $a_n \sim^R_{\rho_1} b_n$. Since $d_n \sqsubseteq d_{n+1}$, then we have $a_n \sqsubseteq a_{n+1}$, $b_n \sqsubseteq b_{n+1}$ for all n. Hence $(a_n, b_n)_{n \in N}$ is a chain in $\sim^R_{\rho_1}$.

  Since $\sim^R_{\rho_1}$ is admissible, we get $\sqcup(a_n, b_n) = (\sqcup a_n, \sqcup b_n) \in \sim^R_{\rho_1}$. Hence, we have $(\mathrm{Left}(\sqcup a_n), \mathrm{Left}(\sqcup b_n)) \in \sim^R_{\rho_1 + \rho_2}$.

  Since $(\mathrm{Left}(\sqcup a_n), \mathrm{Left}(\sqcup b_n)) = \sqcup d_n$, we get $\sqcup d_n \in \sim^R_{\rho_1 + \rho_2}$.

- $\rho \equiv \rho \to \sigma$.

  Let $(d_n)_{n \in N}$ be a chain in $\sim^R_{\rho \to \sigma}$. Then $d_n = (\mathrm{Fun}(f_n), \mathrm{Fun}(g_n))$ where $f_n, g_n \in [\mathbf{D} \to \mathbf{D}]$ s.t. $\forall a, b \in \mathbf{D}((a,b) \in \sim^R_\rho \Rightarrow (f_n(a), g_n(b)) \in \sim^R_\sigma)$.

  Hence, $(f_n(a), g_n(b))_{n \in N}$ is a chain in $\sim^R_\sigma$. Since $\sim^R_\sigma$ is admissible, we get $(\sqcup f_n(a), \sqcup g_n(b)) \in \sim^R_\sigma$.

  Then we have $\sqcup d_n = (\mathrm{Fun}(f), \mathrm{Fun}(g))$ where $f = \sqcup f_n$ and $g = \sqcup g_n$, i.e. $f(a) = \sqcup(f_n(a))$ and $g(b) = \sqcup(g_n(b))$. Therefore, $\sqcup d_n \in \sim^R_{\rho \to \sigma}$.

- $\mathrm{fix}\,\alpha.\rho$.

  We have $\sim^R_{\mathrm{fix}\,\alpha.\rho} = \mathrm{LFP}_{\mathbf{Ad}}(\Phi)$ where

$$\Phi : \mathbf{Ad} \to \mathbf{Ad}$$
$$\Phi(r) = \{(\bot, \bot)\} \cup \{(\mathrm{In}(a), \mathrm{In}(b)) \mid a \sim^{R[\alpha:=r]}_\rho b\}.$$

  First we have to show that indeed $\Phi : \mathbf{Ad} \to \mathbf{Ad}$. Let $r \in \mathbf{Ad}$. We have to show $\Phi(r) \in \mathbf{Ad}$. The proof is similar to the proof of $\rho_1 + \rho_2$. By Lemma 2.4.9, we have $\sim^{R[\alpha:=r]}_\rho$ is monotone. Then the least fixed point exists, we have $\sim^R_{\mathrm{fix}\,\alpha.\rho}$ is admissible.

□

Definition 2.4.11 (Compatibility) Let $r \subseteq \mathbf{D}^2$ and $p \in [\mathbf{D} \to \mathbf{D}]$. We call r and p *compatible*, in symbols $r \approx p$, if

(i) $\forall a, b \in \mathbf{D}\,(r(a,b) \to p(a) = p(b))$.

(ii) $\forall a \in \mathbf{D}\, r(p(a), p(a))$.

(iii) $\forall a, b \in \mathbf{D}\, (r(a,b) \to r(p(a), b))$.

We call $R \in \mathscr{P}(\mathbf{D}^2)^{\mathbf{TVar}}$ and $\zeta \in [\mathbf{D} \to \mathbf{D}]^{\mathbf{TVar}}$ compatible, in symbols $R \approx \zeta$ if $R(\alpha) \approx \zeta(\alpha)$ holds for all $\alpha \in \mathbf{TVar}$.

To obtain an example of compatibility one may take any idempotent $p \in [\mathbf{D} \to \mathbf{D}]$ and define $r \subseteq \mathbf{D}^2$ by $r := \{(a,b) \in \mathbf{D}^2 \mid p(a) = p(b)\}$. Then, clearly, $r \approx p$.

Lemma 2.4.12 If R is admissible, ζ is a finitary projection, and $R \approx \zeta$, then $\sim_\rho^R \approx \langle\rho\rangle\zeta$.

Proof. We write $r \approx_{i,ii} p$ for the notion of compatibility obtained by deleting property (iii) in Definition 2.4.11. Similarly, $r \approx_{iii} p$ means compatibility where properties (i) and (ii) are deleted. The notions $R \approx_{i,ii} \zeta$ and $R \approx_{iii} \zeta$ are defined mutatis mutandis as in Definition 2.4.11.

We show that if R is admissible and $\zeta(\alpha)$ is a finitary projection, then:

(1) If $R \approx_{i,ii} \zeta$, then $\sim_\rho^R \approx_{i,ii} \langle\rho\rangle\zeta$.

(2) If $R \approx_{iii} \zeta$, then $\sim_\rho^R \approx_{iii} \langle\rho\rangle\zeta$.

Both statements are proved by induction on ρ.

First, we prove the statement (1).

Case 1:

(i) Assume $a \sim_1^R b$. We have to show $\langle\mathbf{1}\rangle\zeta(a) = \langle\mathbf{1}\rangle\zeta(b)$.

By Definition 2.4.8, if $a = b = \bot$, we have $\langle\mathbf{1}\rangle\zeta(a) = \bot = \langle\mathbf{1}\rangle\zeta(b)$ by Definition 2.3.2.

If $a = b = \mathrm{Nil}$, we have $\langle\mathbf{1}\rangle\zeta(a) = \mathrm{Nil} = \langle\mathbf{1}\rangle\zeta(b)$ by Definition 2.3.2.

(ii) Let $a \in \mathbf{D}$. We have to show $\langle\mathbf{1}\rangle\zeta(a) \sim_1^R \langle\mathbf{1}\rangle\zeta(a)$.

If $a = \mathrm{Nil}$. Then by Definition 2.3.2, we have $\langle\mathbf{1}\rangle\zeta(a) = \mathrm{Nil}$ and $\mathrm{Nil} \sim_1^R \mathrm{Nil}$.

If $a \neq \mathrm{Nil}$. Then by Definition 2.3.2, we have $\langle\mathbf{1}\rangle\zeta(a) = \bot$ and $\bot \sim_1^R \bot$.

Case α:

Let $R(\alpha) := \{(a_0, b_0) \mid \zeta(\alpha)(a_0) = \zeta(\alpha)(b_0)\}$.

(i) Assume $a \sim_\alpha^R b$. We have to show $\langle\alpha\rangle\zeta(a) = \langle\alpha\rangle\zeta(b)$. By Definition 2.3.2, it is to show $\zeta(\alpha)(a) = \zeta(\alpha)(b)$.

By Definition 2.4.8, we have $a \sim_\alpha^R b = R(\alpha)(a,b)$.

(ii) Let $a \in \mathbf{D}$. We have to show $\langle\alpha\rangle\zeta(a) \sim_\alpha^R \langle\alpha\rangle\zeta(a)$, which follows by $R \approx \zeta$.

Case $\rho + \sigma$:

(i) Assume $a \sim_{\rho+\sigma}^R b$. We have to show $\langle\rho+\sigma\rangle\zeta(a) = \langle\rho+\sigma\rangle\zeta(b)$.

By Definition 2.4.8, if $a = b = \bot$, we have $\langle\rho+\sigma\rangle\zeta(a) = \bot = \langle\rho+\sigma\rangle\zeta(b)$ by Definition 2.3.2.

If $a = \text{Left}(a_1)$ and $b = \text{Left}(b_1)$, by Definition 2.3.2 we have $\langle \rho + \sigma \rangle \zeta(\text{Left}(a_1)) = \text{Left}(\langle \rho \rangle \zeta(a_1))$ and $\langle \rho + \sigma \rangle \zeta(\text{Left}(b_1)) = \text{Left}(\langle \rho \rangle \zeta(b_1))$. By induction hypothesis (i) of ρ, we have $\langle \rho \rangle \zeta(a_1) = \langle \rho \rangle \zeta(b_1)$. Then we get $\text{Left}(\langle \rho \rangle \zeta(a_1)) = \text{Left}(\langle \rho \rangle \zeta(b_1))$.

If $a = \text{Right}(a_1)$ and $b = \text{Right}(b_1)$, we get $\langle \rho + \sigma \rangle \zeta(\text{Right}(a_1)) = \text{Right}(\langle \rho \rangle \zeta(a_1))$ and $\langle \rho + \sigma \rangle \zeta(\text{Right}(b_1)) = \text{Right}(\langle \rho \rangle \zeta(b_1))$ by Definition 2.3.2. Again from induction hypothesis (i) of ρ, we get $\text{Right}(\langle \rho \rangle \zeta(a_1)) = \text{Right}(\langle \rho \rangle \zeta(b_1))$.

(ii) Let $a \in \mathbf{D}$. We have to show $\langle \rho + \sigma \rangle \zeta(a) \sim^R_{\rho+\sigma} \langle \rho + \sigma \rangle \zeta(a)$.

If $a = \text{Left}(a_1)$, it is to show $\text{Left}(\langle \rho \rangle \zeta(a_1)) \sim^R_{\rho+\sigma} \text{Left}(\langle \rho \rangle \zeta(a_1))$ by Definition 2.3.2, then it is to show $\langle \rho \rangle \zeta(a_1) \sim^R_\rho \langle \rho \rangle \zeta(a_1)$. This follows by the induction hypothesis (ii) of ρ.

If $a = \text{Right}(b_1)$, similar to the proof of case $\text{Left}(a_1)$.

Case $\rho \times \sigma$:

(i) Assume $a \sim^R_{\rho\times\sigma} b$. We have to show $\langle \rho \times \sigma \rangle \zeta(a) = \langle \rho \times \sigma \rangle \zeta(b)$.

By Definition 2.4.8, if $a = b = \bot$, we have $\langle \rho \times \sigma \rangle \zeta(a) = \bot = \langle \rho \times \sigma \rangle \zeta(b)$ by Definition 2.3.2.

If $a = \text{Pair}(a_1, a_2)$, $b = \text{Pair}(b_1, b_2)$, we get $\langle \rho \times \sigma \rangle \zeta(\text{Pair}(a_1, a_2)) = \text{Pair}(\langle \rho \rangle \zeta(a_1), \langle \sigma \rangle \zeta(a_2))$ and $\langle \rho \times \sigma \rangle \zeta(\text{Pair}(b_1, b_2)) = \text{Pair}(\langle \rho \rangle \zeta(b_1), \langle \sigma \rangle \zeta(b_2))$ by Definition 2.3.2. By induction hypothesis (i) of ρ, we have $\langle \rho \rangle \zeta(a_1) = \langle \rho \rangle \zeta(b_1)$. By induction hypothesis (i) of σ, we have $\langle \sigma \rangle \zeta(a_2) = \langle \sigma \rangle \zeta(b_2)$.

Then we get $\text{Pair}(\langle \rho \rangle \zeta(a_1), \langle \sigma \rangle \zeta(a_2)) = \text{Pair}(\langle \rho \rangle \zeta(b_1), \langle \sigma \rangle \zeta(b_2))$.

(ii) Let $a \in \mathbf{D}$. We have to show $\langle \rho \times \sigma \rangle \zeta(a) \sim^R_{\rho\times\sigma} \langle \rho \times \sigma \rangle \zeta(a)$.

If $a = \text{Pair}(a_1, a_2)$, by Definition 2.3.2 it is to show $\text{Pair}(\langle \rho \rangle \zeta(a_1), \langle \sigma \rangle \zeta(a_2)) \sim^R_{\rho\times\sigma} \text{Pair}(\langle \rho \rangle \zeta(a_1), \langle \sigma \rangle \zeta(a_2))$, then it is to show $\langle \rho \rangle \zeta(a_1) \sim^R_\rho \langle \rho \rangle \zeta(a_1)$ and $\langle \sigma \rangle \zeta(a_2) \sim^R_\sigma \langle \sigma \rangle \zeta(a_2)$. This follows by the induction hypothesis (ii) of ρ and σ.

Case $\rho \to \sigma$:

(i) Assume $a \sim^R_{\rho\to\sigma} b$. We have to show $\langle \rho \to \sigma \rangle \zeta(a) = \langle \rho \to \sigma \rangle \zeta(b)$.

If $a = b = \bot$, then this is trivial.

If $a = \text{Fun}(f)$ and $b = \text{Fun}(g)$. By the definition of $\sim^R_{\rho\to\sigma}$ (Definition 2.4.8), we have

$$\forall c, d \in \mathbf{D}(c \sim^R_\rho d \Rightarrow f(c) \sim^R_\sigma g(d)) \qquad (*)$$

By Definition 2.3.2, we have $\langle \rho \to \sigma \rangle \zeta(f) = \text{Fun}(\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta)$ and $\langle \rho \to \sigma \rangle \zeta(g) = \text{Fun}(\langle \sigma \rangle \zeta \circ g \circ \langle \rho \rangle \zeta)$.

Let $c \in \mathbf{D}$. We need to show $\langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(c))) = \langle \sigma \rangle \zeta(g(\langle \rho \rangle \zeta(c)))$.

By induction hypothesis (ii) for ρ we have $\langle \rho \rangle \zeta(c) \sim^R_\rho \langle \rho \rangle \zeta(c)$.

Then by $(*)$ we have $f(\langle \rho \rangle \zeta(c)) \sim^R_\sigma g(\langle \rho \rangle \zeta(c))$.

Then by induction hypothesis (i) for σ we have

$$\langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(c))) = \langle \sigma \rangle \zeta(g(\langle \rho \rangle \zeta(c)))$$

(ii) Let $a \in \mathbf{D}$. We have to show $\langle \rho \to \sigma \rangle \zeta(a) \sim^R_{\rho\to\sigma} \langle \rho \to \sigma \rangle \zeta(a)$.

If $a \neq \mathrm{Fun}(f)$. Then $\langle \rho \to \sigma \rangle \zeta(a) = \bot$. Thus $\bot \sim^R_{\rho\to\sigma} \bot$.

If $a = \mathrm{Fun}(f)$, then, by Definition 2.3.2, we have $\langle \rho \to \sigma \rangle \zeta(a) = \mathrm{Fun}(\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta)$.

We need to show $\mathrm{Fun}(\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta) \sim^R_{\rho\to\sigma} \mathrm{Fun}(\langle \sigma \rangle \zeta \circ f \circ \langle \rho \rangle \zeta)$. By the definition of $\sim^R_{\rho\to\sigma}$ (Definition 2.4.8), it is to show

$$\forall c,d \in \mathbf{D}(c \sim^R_\rho d \Rightarrow \langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(c))) \sim^R_\sigma \langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(d)))).$$

Assume $c \sim^R_\rho d$. By induction hypothesis (i) for ρ, we have $\langle \rho \rangle \zeta(c) = \langle \rho \rangle \zeta(d)$. Then $f(\langle \rho \rangle \zeta(c)) = f(\langle \rho \rangle \zeta(d))$.

By induction hypothesis (ii) for σ, we have

$$\langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(c))) \sim^R_\sigma \langle \sigma \rangle \zeta(f(\langle \rho \rangle \zeta(d)))$$

Case $\mathrm{fix}\,\alpha.\rho$:

By Definition 2.3.2, we have $\langle \mathrm{fix}\,\alpha.\rho \rangle \zeta = \bigsqcup_n p_n$ where

$p_0 = \lambda a.\bot$, $p_{n+1} = \lambda a.\mathrm{case}_{\mathrm{In}}\ a\ (\lambda b.\mathrm{In}(\langle \rho \rangle \zeta[\alpha := p_n](b)))$.

We set $r := \sim^R_{\mathrm{fix}\,\alpha.\rho}$ and show $r \approx_{i,ii} p_n$ by a side induction on n. This will be sufficient, since, as one easily sees, because r is admissible (by Definition 2.4.8), the conditions (i) and (ii) of compatibility are closed under taking directed suprema of the right argument. Hence from $r \approx_{i,ii} p_n$ for all n it follows $r \approx_{i,ii} p$.

$n = 0$: (i) is trivial since p_0 is constant. (ii) holds since $p_0(a) = \bot$ and $(\bot, \bot) \in r$ since r is admissible, by Definition 2.4.8.

$n+1$: (i) Assume $r(a,b)$. We have to show $p_{n+1}(a) = p_{n+1}(b)$.

If $a = b = \bot$, the equation trivially holds.

Now assume $a = \mathrm{In}(c)$ and $b = \mathrm{In}(d)$.

Then $p_{n+1}(a) = \mathrm{In}(\langle \rho \rangle \zeta[\alpha := p_n](c))$ and $p_{n+1}(b) = \mathrm{In}(\langle \rho \rangle \zeta[\alpha := p_n](d))$, and we have $c \sim^{R[\alpha:=r]}_\rho d$ by the definition of r.

By the side induction hypothesis, $r \approx_{i,ii} p_n$.

Hence, by the main induction hypothesis, $\sim^{R[\alpha:=r]}_\rho \approx_{i,ii} \langle \rho \rangle \zeta[\alpha := p_n]$, since r is admissible and p_n is a finitary projection (see proof of Lemma 2.3.6).

It follows $\langle \rho \rangle \zeta[\alpha := p_n](c) = \langle \rho \rangle \zeta[\alpha := p_n](d)$ and therefore $p_{n+1}(a) = p_{n+1}(b)$.

(ii) Let $a \in D$. We have to show $r(p_{n+1}(a), p_{n+1}(a))$.

If $a = \bot$, then $p_{n+1}(a) = \bot$ and $r(\bot, \bot)$ holds by admissibility of r.

If $a = \mathrm{In}(c)$, then $p_{n+1}(a) = \mathrm{In}(\langle \rho \rangle \zeta[\alpha := p_n](c))$.

By the side induction hypothesis, $r \approx_{i,ii} p_n$.

Hence, by the main induction hypothesis, $\sim^{R[\alpha:=r]}_\rho \approx_{i,ii} \langle \rho \rangle \zeta[\alpha := p_n]$, since r is admissible and p_n is a finitary projection (see proof of Lemma 2.3.6).

It follows $\langle \rho \rangle \zeta[\alpha := p_n](c) \sim^{R[\alpha:=r]}_\rho \langle \rho \rangle \zeta[\alpha := p_n](c)$.

Therefore $p_{n+1}(a) \sim^R_{\mathrm{fix}\,\alpha.\rho} p_{n+1}(a)$, i.e. $r(p_{n+1}(a), p_{n+1}(a))$.

Second, for statement (2), we only show two interesting cases, $\rho \to \sigma$ and $\mathrm{fix}\,\alpha.\rho$.

Case $\rho \to \sigma$:

Assume $a \sim^R_{\rho\to\sigma} b$. We have to show $\langle \rho \to \sigma\rangle\zeta(a) \sim^R_{\rho\to\sigma} b$.

If $a = b = \bot$, then $\langle \rho \to \sigma\rangle\zeta(a) = \bot = b$. Thus $\bot \sim^R_{\rho\to\sigma} \bot$.

If $a = \mathrm{Fun}(f)$ and $b = \mathrm{Fun}(g)$, then by the definition of $\sim^R_{\rho\to\sigma}$, we have

$$\forall c,d \in \mathbf{D}(c \sim^R_\rho d \Rightarrow f(c) \sim^R_\sigma g(d)) \qquad (*)$$

By Definition 2.3.2, we have $\langle \rho \to \sigma\rangle\zeta(a) = \mathrm{Fun}(\langle\sigma\rangle\zeta \circ f \circ \langle\rho\rangle\zeta)$.

We need to show $\mathrm{Fun}(\langle\sigma\rangle\zeta \circ f \circ \langle\rho\rangle\zeta) \sim^R_{\rho\to\sigma} \mathrm{Fun}(g)$. By the definition of $\sim^R_{\rho\to\sigma}$, it is to show

$$\forall c,d \in \mathbf{D}(c \sim^R_\rho d \Rightarrow \langle\sigma\rangle\zeta(f(\langle\rho\rangle\zeta(c))) \sim^R_\sigma g(d)).$$

Assume $c \sim^R_\rho d$. By induction hypothesis for ρ, we have

$$\langle\rho\rangle\zeta(c) \sim^R_\rho d \qquad \text{(IH(iii))}$$

Then by $(*)$ we have $f(\langle\rho\rangle\zeta(c)) \sim^R_\sigma g(d)$, and by induction hypothesis for σ, we have

$$\langle\sigma\rangle\zeta(f(\langle\rho\rangle\zeta(c))) \sim^R_\sigma g(d)$$

Case $\mathrm{fix}\,\alpha.\rho$:

Set $r := \sim^R_{\mathrm{fix}\,\alpha.\rho}$ and $p := \langle \mathrm{fix}\,\alpha.\rho\rangle\zeta$. We have to show that $r(a,b)$ implies $r(p(a),b)$, i.e. $r \subseteq s$ where $s := \{(a,b) \mid r(p(a),b)\}$.

We verify that $r \cap s \approx_{iii} p$ holds. Indeed, if $(r\cap s)(a,b)$, then $r(p(a),b)$ since $s(a,b)$ holds, and hence, since by Lemma 2.3.6 p is idempotent, $r(p(p((a)),b)$ since $s(p(a),b)$ holds, i.e. $(r\cap s)(p(a),b)$.

Since r is the least fixed point of the operator $\Phi := \Lambda r.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim^{R[\alpha:=r]}_\rho b\}$ we can attempt to prove the inclusion $r \subseteq s$ by induction on Φ, i.e. $\Phi(s) \subseteq s$.

In fact we use the *strong induction principle* (see, for example, [Ber11]) according to which it suffices to show $\Phi(r\cap s) \subseteq s$ (instead of $\Phi(s) \subseteq s$). Clearly $(\bot,\bot) \in s$. Hence we assume $a \sim^{R[\alpha:=r\cap s]}_\rho b$ and have to show $r(p(\mathrm{In}(a)),\mathrm{In}(b))$. Since $p(\mathrm{In}(a)) = \mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](a))$ and $r = \Phi(r)$, we have to show $\Phi(r)(\mathrm{In}(\langle\rho\rangle\zeta[\alpha := p](a)),\mathrm{In}(b))$, i.e., $\langle\rho\rangle\zeta[\alpha := p](a) \sim^{R[\alpha:=r]}_\rho b$.

By induction hypothesis, and because $r\cap s \approx_{iii} p$, $\langle\rho\rangle\zeta[\alpha := p](a) \sim^{R[\alpha:=r\cap s]}_\rho b$ and hence $\langle\rho\rangle\zeta[\alpha := p](a) \sim^{R[\alpha:=r]}_\rho b$, by monotonicity (Lemma 2.4.9). □

The next theorem is the core of the proof of the Coincidence Theorem (Theorem 2.4.16). For its proof we need the following auxiliary lemmas.

Let X be a set of type variables. We define $R =_X R'$ as $\forall\alpha \in X(R(\alpha) = R'(\alpha))$. Thus, $R =_X R' \wedge Y \subseteq X \Rightarrow R =_Y R'$.

Lemma 2.4.13 $\sim_\tau^R = \sim_\tau^{R'}$ if R and R' coincide on all free type variables of τ, i.e. $R =_{\mathrm{FTV}(\tau)} R'$.

Proof. By induction on τ.

1. $\tau \equiv \mathbf{1}$.

   We need to show $\sim_{\mathbf{1}}^R = \sim_{\mathbf{1}}^{R'}$.

   This follows by $\sim_{\mathbf{1}}^R = \sim_{\mathbf{1}}^{R'} = \{(\bot,\bot),(\mathrm{Nil},\mathrm{Nil})\}$ (Definition 2.4.8).

2. $\tau \equiv \alpha$.

   We need to show $\sim_\alpha^R = \sim_\alpha^{R'}$.

   By Definition 2.4.8, we get $\sim_\alpha^R = R(\alpha)$, and $\sim_\alpha^{R'} = R'(\alpha)$.

   Since $R =_{\mathrm{FTV}(\tau)} R'$, we have $\sim_\alpha^R = \sim_\alpha^{R'}$.

3. $\tau \equiv \rho \times \sigma$.

   We need to show $\sim_{\rho\times\sigma}^R = \sim_{\rho\times\sigma}^{R'}$.

   By Definition 2.4.8, we have $\sim_{\rho\times\sigma}^R = \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_\rho^R b_1, a_2 \sim_\sigma^R b_2\}$, and $\sim_{\rho\times\sigma}^{R'} = \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_\rho^{R'} b_1, a_2 \sim_\sigma^{R'} b_2\}$.

   By I.H. (i.e. $\sim_\rho^R = \sim_\rho^{R'}$ and $\sim_\sigma^R = \sim_\sigma^{R'}$), we have $\sim_{\rho\times\sigma}^R = \sim_{\rho\times\sigma}^{R'}$.

4. $\tau \equiv \rho + \sigma$.

   Similar to the proof of $\rho \times \sigma$.

5. $\tau \equiv \rho \to \sigma$.

   We need to show $\sim_{\rho\to\sigma}^R = \sim_{\rho\to\sigma}^{R'}$.

   By Definition 2.4.8, we have $\sim_{\rho\to\sigma}^R = \{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}\,(a \sim_\rho^R b \Rightarrow f(a) \sim_\sigma^R g(b))\}$, and $\sim_{\rho\to\sigma}^{R'} = \{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}\,(a \sim_\rho^{R'} b \Rightarrow f(a) \sim_\sigma^{R'} g(b))\}$.

   By I.H. (i.e. $\sim_\rho^R = \sim_\rho^{R'}$, $\sim_\sigma^R = \sim_\sigma^{R'}$), $\sim_{\rho\to\sigma}^R = \sim_{\rho\to\sigma}^{R'}$.

6. $\tau \equiv \mathrm{fix}\,\alpha.\rho$.

   We need to show $\sim_{\mathrm{fix}\,\alpha.\rho}^R = \sim_{\mathrm{fix}\,\alpha.\rho}^{R'}$.

   By Definition 2.4.8, we get $\sim_{\mathrm{fix}\,\alpha.\rho}^R = \mathrm{LFP}(\Lambda r \subseteq \mathbf{D}^2.\{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_\rho^{R[\alpha:=r]} b\})$, and $\sim_{\mathrm{fix}\,\alpha.\rho}^{R'} = \mathrm{LFP}(\Lambda r \subseteq \mathbf{D}^2.\{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim_\rho^{R'[\alpha:=r]} b\})$.

By I.H. (i.e. $\sim_\rho^R = \sim_\rho^{R'}$ and $R =_{\mathrm{FTV}(\tau)} R'$), we have $\sim_{\mathrm{fix}\,\alpha.\rho}^R = \sim_{\mathrm{fix}\,\alpha.\rho}^{R'}$.

□

Lemma 2.4.14 (Substitution for $\sim_\rho^R$)

$$\sim_{\rho[\tau/\alpha]}^R = \sim_\rho^{R[\alpha:=\sim_\tau^R]}$$

Proof. By induction on ρ.

1. $\rho \equiv \mathbf{1}$.

   Trivial, $\sim_{\mathbf{1}}^R = \sim_{\mathbf{1}}^R$, since there is no variable α to be substituted.

2. $\rho \equiv \alpha$.

   We need to show $\sim_{\alpha[\tau/\alpha]}^R = \sim_\alpha^{R[\alpha:=\sim_\tau^R]}$.

   For the left-hand side of the equation we have $\sim_{\alpha[\tau/\alpha]}^R = \sim_\tau^R$.

   For the right-hand side we have $\sim_\alpha^{R[\alpha:=\sim_\tau^R]} = R[\alpha :=\sim_\tau^R](\alpha) = \sim_\tau^R$.

   By equality, we have $\sim_{\alpha[\tau/\alpha]}^R = \sim_\alpha^{R[\alpha:=\sim_\tau^R]}$.

3. $\rho \equiv \rho_1 \times \rho_2$.

   We need to show $\sim_{(\rho_1\times\rho_2)[\tau/\alpha]}^R = \sim_{\rho_1\times\rho_2}^{R[\alpha:=\sim_\tau^R]}$.

   For the left-hand side of the equation we have $\sim_{(\rho_1\times\rho_2)[\tau/\alpha]}^R = \sim_{\rho_1[\tau/\alpha]\times\rho_2[\tau/\alpha]}^R$
   $= \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_{\rho_1[\tau/\alpha]}^R b_1, a_2 \sim_{\rho_2[\tau/\alpha]}^R b_2\}$.

   We have $\sim_{\rho_1\times\rho_2}^{R[\alpha:=\sim_\tau^R]} = \{(\bot,\bot)\} \cup \{(\mathrm{Pair}(a_1,a_2),\mathrm{Pair}(b_1,b_2)) \mid a_1 \sim_{\rho_1}^{R[\alpha:=\sim_\tau^R]} b_1,$
   $a_2 \sim_{\rho_2}^{R[\alpha:=\sim_\tau^R]} b_2\}$ for the right-hand side.

   We have the following

   $$\sim_{\rho_1[\tau/\alpha]}^R = \sim_{\rho_1}^{R[\alpha:=\sim_\tau^R]} \qquad \text{(IH1)}$$

   $$\sim_{\rho_2[\tau/\alpha]}^R = \sim_{\rho_2}^{R[\alpha:=\sim_\tau^R]} \qquad \text{(IH2)}$$

   Applying IH1 and IH2, we have $\sim_{(\rho_1\times\rho_2)[\tau/\alpha]}^R = \sim_{\rho_1\times\rho_2}^{R[\alpha:=\sim_\tau^R]}$.

4. $\rho \equiv \rho_1 + \rho_2$.

   We need to show $\sim_{(\rho_1+\rho_2)[\tau/\alpha]}^R = \sim_{\rho_1+\rho_2}^{R[\alpha:=\sim_\tau^R]}$.

For the left-hand side of the equation, we have $\sim^R_{(\rho_1+\rho_2)[\tau/\alpha]} = \sim^R_{\rho_1[\tau/\alpha]+\rho_2[\tau/\alpha]}$ $= \{(\bot,\bot)\} \cup \{(\mathrm{Left}(a_1),\mathrm{Left}(b_1)) \mid a_1 \sim^R_{\rho_1[\tau/\alpha]} b_1\} \cup \{(\mathrm{Right}(a_2),\mathrm{Right}(b_2)) \mid a_2 \sim^R_{\rho_2[\tau/\alpha]} b_2\}$.

For the right-hand side we have $\sim^{R[\alpha:=\sim^R_\tau]}_{\rho_1\times\rho_2} = \{(\bot,\bot)\} \cup \{(\mathrm{Left}(a_1),\mathrm{Left}(b_1)) \mid a_1 \sim^{R[\alpha:=\sim^R_\tau]}_{\rho_1} b_1\} \cup \{(\mathrm{Right}(a_2),\mathrm{Right}(b_2)) \mid a_2 \sim^{R[\alpha:=\sim^R_\tau]}_{\rho_2} b_2\}$.

We have the following

$$\sim^R_{\rho_1[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho_1} \quad \text{(IH1)}$$

$$\sim^R_{\rho_2[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho_2} \quad \text{(IH2)}$$

Applying IH1 and IH2, we have $\sim^R_{(\rho_1+\rho_2)[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho_1+\rho_2}$.

5. $\rho \equiv \rho \to \sigma$.

We need to show $\sim^R_{(\rho\to\sigma)[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho\to\sigma}$.

For the left-hand side of the equation we get $\sim^R_{(\rho\to\sigma)[\tau/\alpha]} = \sim^R_{\rho[\tau/\alpha]\to\sigma[\tau/\alpha]} =$ $\{(\bot,\bot)\} \cup \{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}(a \sim^R_{\rho[\tau/\alpha]} b \Rightarrow f(a) \sim^R_{\sigma[\tau/\alpha]} g(b))\}$.

For the right-hand side we have $\sim^{R[\alpha:=\sim^R_\tau]}_{\rho\to\sigma} = \{(\bot,\bot)\}$

$\cup\{(\mathrm{Fun}(f),\mathrm{Fun}(g)) \mid \forall a,b \in \mathbf{D}(a \sim^{R[\alpha:=\sim^R_\tau]}_{\rho} b \Rightarrow f(a) \sim^{R[\alpha:=\sim^R_\tau]}_{\sigma} g(b))\}$.

We have the following

$$\sim^R_{\rho[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho} \quad \text{(IH1)}$$

$$\sim^R_{\sigma[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\sigma} \quad \text{(IH2)}$$

Applying IH1 and IH2, we have $\sim^R_{(\rho\to\sigma)[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\rho\to\sigma}$.

6. $\rho \equiv \mathrm{fix}\ \alpha'.\rho$.

We need to show $\sim^R_{(\mathrm{fix}\ \alpha'.\rho)[\tau/\alpha]} = \sim^{R[\alpha:=\sim^R_\tau]}_{\mathrm{fix}\ \alpha'.\rho}$.

For the left-hand side of the equation we have $\sim^R_{(\mathrm{fix}\ \alpha'.\rho)[\tau/\alpha]} = \sim^R_{\mathrm{fix}\ \alpha'.(\rho[\tau/\alpha])} =$ $\mathrm{LFP}(\Lambda r \subseteq \mathbf{D}^2.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim^{R[\alpha':=r]}_{\rho[\tau/\alpha]} b\})$.

For the right-hand side we have $\sim^{R[\alpha:=\sim^R_\tau]}_{\text{fix }\alpha'.\rho} =$

$\mathrm{LFP}(\Lambda r \subseteq \mathbf{D}^2.\{(\bot,\bot)\} \cup \{(\mathrm{In}(a),\mathrm{In}(b)) \mid a \sim^{R[\alpha:=\sim^R_\tau][\alpha':=r]}_\rho b\})$.

Let $R' := R[\alpha' := r]$.

We need to show $\sim^{R'}_{\rho[\tau/\alpha]} = \sim^{R'[\alpha:=\sim^R_\tau]}_\rho$.

We have the following

$$\sim^{R'}_{\rho[\tau/\alpha]} = \sim^{R'[\alpha:=\sim^{R'}_\tau]}_\rho \qquad \text{(IH1)}$$

Then it is to show $\sim^R_\tau = \sim^{R'}_\tau$. This follows by Lemma 2.4.13.

□

Let $\eta \sim^R_\Gamma \eta'$ denote the following: for all $x \in \mathrm{dom}(\Gamma)$, $\eta(x) \sim^R_{\Gamma(x)} \eta'(x)$.

Let $\Gamma \vdash^{\mathrm{reg}} M : \rho$ mean that $\Gamma \vdash M : \rho$ has been derived using regular types only.

M^- be the untyped term obtained from the Church-style term M by deleting the type information in lambda abstractions.

Theorem 2.4.15 Assume R is admissible, ζ is a finitary projection, and $R \approx \zeta$. If $\Gamma \vdash^{\mathrm{reg}} M : \rho$ and $\eta \sim^R_\Gamma \eta'$, then $[\![M]\!]^\zeta \eta \sim^R_\rho [\![M^-]\!]\eta'$.

Proof. By induction on the definition of the relation $\Gamma \vdash^{\mathrm{reg}} M : \rho$.

1. $\Gamma \vdash^{\mathrm{reg}} \mathrm{Nil} : \mathbf{1}$.

   We need to show $[\![\mathrm{Nil}]\!]^\zeta \eta \sim^R_1 [\![\mathrm{Nil}^-]\!]\eta'$.

   By Definition 2.3.7 and 2.4.2, we have $[\![\mathrm{Nil}]\!]^\zeta \eta = \mathrm{Nil}$ and $[\![\mathrm{Nil}^-]\!]\eta' = \mathrm{Nil}$.

   By Definition 2.4.8, we have $\mathrm{Nil} \sim^R_1 \mathrm{Nil}$. Thus, $[\![\mathrm{Nil}]\!]^\zeta \eta \sim^R_1 [\![\mathrm{Nil}^-]\!]\eta'$.

2. $\Gamma, x : \rho \vdash^{\mathrm{reg}} x : \rho$.

   We need to show $[\![x]\!]^\zeta \eta \sim^R_\rho [\![x^-]\!]\eta'$.

   By Definition 2.3.7 and 2.4.2, we have $[\![x]\!]^\zeta \eta = \eta(x)$ and $[\![x^-]\!]\eta' = \eta'(x)$.

   By assumption, we have $\eta(x) \sim^R_\rho \eta'(x)$. Thus, $[\![x]\!]^\zeta \eta \sim^R_\rho [\![x^-]\!]\eta'$.

3. $$\frac{\Gamma, x : \rho \vdash^{\mathrm{reg}} M : \sigma}{\Gamma \vdash^{\mathrm{reg}} \lambda x : \rho.M : \rho \to \sigma}.$$

   We have to show $[\![\lambda x : \rho.M]\!]^\zeta \eta \sim^R_{\rho\to\sigma} [\![\lambda x.M^-]\!]\eta'$.

   By Definition 2.3.7 and 2.4.2, we have

   $[\![\lambda x : \rho.M]\!]^\zeta \eta = \mathrm{Fun}(f)$ where $f(a) = [\![M]\!]^\zeta \eta[x := \langle\rho\rangle\zeta(a)]$,
   $[\![\lambda x.M^-]\!]\eta' = \mathrm{Fun}(g)$ where $g(b) = [\![M^-]\!]\eta'[x := b]$.

It is to show $\mathrm{Fun}(f) \sim^R_{\rho\to\sigma} \mathrm{Fun}(g)$. By definition of our logical relation (Definition 2.4.8), it is to show

$$\forall a,b \in \mathbf{D}(a \sim^R_\rho b \Rightarrow f(a) \sim^R_\sigma g(b)) \qquad (*)$$

By induction hypothesis for σ we have

$$\forall a,b \in \mathbf{D}(a \sim^R_\rho b \Rightarrow [\![M]\!]^\zeta \eta[x := a] \sim^R_\sigma [\![M^-]\!]\eta'[x := b]) \qquad \text{(IH)}$$

To prove $(*)$, assume $a \sim^R_\rho b$. By Lemma 2.4.12 (iii) we have $\langle\rho\rangle\zeta(a) \sim^R_\rho b$. Hence $[\![M]\!]^\zeta\eta[x := \langle\rho\rangle\zeta(a)] \sim^R_\sigma [\![M^-]\!]\eta'[x := b]$.

4. $\dfrac{\Gamma, x:\tau \vdash^{\mathrm{reg}} M:\tau}{\Gamma \vdash^{\mathrm{reg}} \mathrm{rec}\, x:\tau.M:\tau}$.

We have to show $[\![\mathrm{rec}x:\tau.M]\!]^\zeta\eta \sim^R_\tau [\![\mathrm{rec}x.M^-]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, we have

$[\![\mathrm{rec}x:\tau.M]\!]^\zeta\eta = \mathrm{LFP}(f)$ where $f(a) = [\![M]\!]^\zeta\eta[x := \langle\tau\rangle\zeta(a)]$,
$[\![\mathrm{rec}x.M^-]\!]\eta' = \mathrm{LFP}(g)$ where $g(b) = [\![M^-]\!]\eta'[x := b]$.

Now we have to show $\mathrm{LFP}(f) \sim^R_\tau \mathrm{LFP}(g)$.

By definition it is to show $\sqcup_n f^n(\bot) \sim^R_\tau \sqcup_n g^n(\bot)$.

Since, by Definition 2.4.8, $\sim^R_\tau$ is admissible, it suffices to show that $\forall n. f^n(\bot) \sim^R_\tau g^n(\bot)$.

We do a side induction on n.

$n = 0$: We need to show $f^0(\bot) \sim^R_\tau g^0(\bot)$, i.e. $\bot \sim^R_\tau \bot$. This holds by Definition 2.4.8.

$n+1$: Assume as side induction hypothesis, $f^n(\bot) \sim^R_\tau g^n(\bot)$, to show $f^{(n+1)}(\bot) \sim^R_\tau g^{(n+1)}(\bot)$. We have

$$f^{(n+1)}(\bot) = f(f^n(\bot)) = [\![M]\!]^\zeta\eta[x := \langle\tau\rangle\zeta(f^n(\bot))],$$
$$g^{(n+1)}(\bot) = g(g^n(\bot)) = [\![M^-]\!]\eta'[x := g^n(\bot)].$$

Then it is to show $[\![M]\!]^\zeta\eta[x := \langle\tau\rangle\zeta(f^n(\bot))] \sim^R_\tau [\![M^-]\!]\eta'[x := g^n(\bot)]$. By side induction hypothesis, we have $\langle\tau\rangle\zeta(f^n(\bot)) \sim^R_\tau g^n(\bot)$ by Lemma 2.4.12 (iii).

By main induction hypothesis we have

$[\![M]\!]^\zeta\eta[x := \langle\tau\rangle\zeta(f^n(\bot))] \sim^R_\tau [\![M^-]\!]\eta'[x := g^n(\bot)]$.

5. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M:\rho\to\sigma \qquad \Gamma \vdash^{\mathrm{reg}} N:\rho}{\Gamma \vdash^{\mathrm{reg}} MN:\sigma}$.

We need to show $[\![MN]\!]^\zeta\eta \sim^R_\sigma [\![M^-N^-]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, we get $[\![MN]\!]^\zeta\eta = f([\![N]\!]^\zeta\eta)$ if $[\![M]\!]^\zeta\eta = \mathrm{Fun}(f)$, and $[\![M^-N^-]\!]\eta' = g([\![N^-]\!]\eta')$ if $[\![M^-]\!]^\zeta\eta' = \mathrm{Fun}(g)$.

Then it is to show $f([\![N]\!]^\zeta\eta) \sim^R_\sigma g([\![N^-]\!]\eta')$.

We have the following

$$[\![M]\!]^\zeta\eta \sim^R_{\rho\to\sigma} [\![M^-]\!]\eta' \qquad \text{(IH1)}$$

$$[\![N]\!]^\zeta\eta \sim^R_\rho [\![N^-]\!]\eta' \qquad \text{(IH2)}$$

By IH1 and Definition 2.4.8, we get $[\![M]\!]^\zeta\eta = \mathrm{Fun}(f)$, $[\![M^-]\!]^\zeta\eta' = \mathrm{Fun}(g)$, and $\forall a,b \in \mathbf{D}(a \sim^R_\rho b \Rightarrow f(a) \sim^R_\sigma g(b))$. Applying IH2, we have $f([\![N]\!]^\zeta\eta) \sim^R_\sigma g([\![N^-]\!]\eta')$.

6. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \rho \qquad \Gamma \vdash^{\mathrm{reg}} N : \sigma}{\Gamma \vdash^{\mathrm{reg}} \mathrm{Pair}(M,N) : \rho \times \sigma}$.

We need to show $[\![\mathrm{Pair}(M,N)]\!]^\zeta\eta \sim^R_{\rho\times\sigma} [\![\mathrm{Pair}(M^-,N^-)]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, we have $[\![\mathrm{Pair}(M,N)]\!]^\zeta\eta = \mathrm{Pair}([\![M]\!]^\zeta\eta, [\![N]\!]^\zeta\eta)$ and $[\![\mathrm{Pair}(M^-,N^-)]\!]\eta' = \mathrm{Pair}([\![M^-]\!]\eta', [\![N^-]\!]\eta')$.

Then it is to show $\mathrm{Pair}([\![M]\!]^\zeta\eta, [\![N]\!]^\zeta\eta) \sim^R_{\rho\times\sigma} \mathrm{Pair}([\![M^-]\!]\eta', [\![N^-]\!]\eta')$.

By Definition 2.4.8, it is to show $[\![M]\!]^\zeta\eta \sim^R_\rho [\![M^-]\!]\eta'$ and $[\![N]\!]^\zeta\eta \sim^R_\sigma [\![N^-]\!]\eta'$.

This follows by

$$[\![M]\!]^\zeta\eta \sim^R_\rho [\![M^-]\!]\eta' \qquad \text{(IH1)}$$

$$[\![N]\!]^\zeta\eta \sim^R_\sigma [\![N^-]\!]\eta' \qquad \text{(IH2)}$$

7. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \rho}{\Gamma \vdash^{\mathrm{reg}} \mathrm{Left}(M) : \rho + \sigma}$.

We need to show $[\![\mathrm{Left}(M)]\!]^\zeta\eta \sim^R_{\rho+\sigma} [\![\mathrm{Left}(M^-)]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, we have $[\![\mathrm{Left}(M)]\!]^\zeta\eta = \mathrm{Left}([\![M]\!]^\zeta\eta)$ and $[\![\mathrm{Left}(M^-)]\!]\eta' = \mathrm{Left}([\![M^-]\!]\eta')$.

Then it is to show $\mathrm{Left}([\![M]\!]^\zeta\eta) \sim^R_{\rho+\sigma} \mathrm{Left}([\![M^-]\!]\eta')$.

By Definition 2.4.8, it is to show $[\![M]\!]^\zeta\eta \sim^R_\rho [\![M^-]\!]\eta'$.

This follows by

$$[\![M]\!]^\zeta\eta \sim^R_\rho [\![M^-]\!]\eta' \qquad \text{(IH1)}$$

8. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \sigma}{\Gamma \vdash^{\mathrm{reg}} \mathrm{Right}(M) : \rho + \sigma}$.

Similar to the proof of case Left.

9. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \rho + \sigma \qquad \Gamma, x_1 : \rho \vdash^{\mathrm{reg}} L : \tau \qquad \Gamma, x_2 : \sigma \vdash^{\mathrm{reg}} R : \tau}{\Gamma \vdash^{\mathrm{reg}} \mathrm{case} M \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L; \mathrm{Right}(x_2) \to R\} : \tau}$.

Let $[\![\mathrm{case} M]\!] = [\![\mathrm{case} M \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L; \mathrm{Right}(x_2) \to R\}]\!]$

and $[\![\mathrm{case} M^-]\!] = [\![\mathrm{case} M^- \,\mathrm{of}\, \{\mathrm{Left}(x_1) \to L^-; \mathrm{Right}(x_2) \to R^-\}]\!]$.

We need to show $[\![\mathrm{case} M]\!]^\zeta \eta \sim^R_\tau [\![\mathrm{case} M^-]\!]\eta'$.

case1 By Definition 2.3.7 and 2.4.2, we have

$$[\![\mathrm{case} M]\!]^\zeta \eta = [\![L]\!]^\zeta \eta[x_1 := \langle\rho\rangle\zeta(a)] \qquad \text{if } [\![M]\!]^\zeta \eta = \mathrm{Left}(x_1)$$
$$[\![\mathrm{case} M^-]\!]\eta' = [\![L^-]\!]\eta'[x_1 := b] \qquad \text{if } [\![M^-]\!]\eta' = \mathrm{Left}(x_1)$$

Then it is to show $[\![L]\!]^\zeta \eta[x_1 := \langle\rho\rangle\zeta(a)] \sim^R_\tau [\![L^-]\!]\eta'[x_1 := b]$.

We have the following

$$\forall a, b \in \mathbf{D}(a \sim^R_\rho b \Rightarrow [\![L]\!]^\zeta \eta[x_1 := a] \sim^R_\tau [\![L^-]\!]\eta'[x_1 := b]) \qquad \text{(IH1)}$$

Assume $a \sim^R_\rho b$. By Lemma 2.4.12 (iii) we have $\langle\rho\rangle\zeta(a) \sim^R_\rho b$.

Hence $[\![L]\!]^\zeta \eta[x_1 := \langle\rho\rangle\zeta(a)] \sim^R_\tau [\![L^-]\!]\eta'[x_1 := b]$.

case2 By Definition 2.3.7 and 2.4.2, we have

$$[\![\mathrm{case} M]\!]^\zeta \eta = [\![R]\!]^\zeta \eta[x_2 := \langle\sigma\rangle\zeta(a)] \qquad \text{if } [\![M]\!]^\zeta \eta = \mathrm{Right}(x_2)$$
$$[\![\mathrm{case} M^-]\!]\eta' = [\![R^-]\!]\eta'[x_2 := b] \qquad \text{if } [\![M^-]\!]\eta' = \mathrm{Right}(x_2)$$

Then it is to show $[\![R]\!]^\zeta \eta[x_2 := \langle\sigma\rangle\zeta(a)] \sim^R_\tau [\![R^-]\!]\eta'[x_2 := b]$.

We have the following

$$\forall a, b \in \mathbf{D}(a \sim^R_\sigma b \Rightarrow [\![R]\!]^\zeta \eta[x_2 := a] \sim^R_\tau [\![R^-]\!]\eta'[x_2 := b]) \qquad \text{(IH2)}$$

Assume $a \sim^R_\sigma b$. By Lemma 2.4.12 (iii) we have $\langle\sigma\rangle\zeta(a) \sim^R_\sigma b$.

Hence $[\![R]\!]^\zeta \eta[x_2 := \langle\sigma\rangle\zeta(a)] \sim^R_\tau [\![R^-]\!]\eta'[x_2 := b]$.

10. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \rho \times \sigma \qquad \Gamma, x : \rho, y : \sigma \vdash^{\mathrm{reg}} N : \tau}{\Gamma \vdash^{\mathrm{reg}} \mathrm{case} M \,\mathrm{of}\, \{\mathrm{Pair}(x, y) \to N\} : \tau}$.

Let $[\![\mathrm{case} M]\!] = [\![\mathrm{case} M \,\mathrm{of}\, \{\mathrm{Pair}(x, y) \to N\}]\!]$

and $[\![\mathrm{case} M^-]\!] = [\![\mathrm{case}\ M^- \,\mathrm{of}\, \{\mathrm{Pair}\ (x, y) \to N^-\}]\!]$.

We need to show $[\![\mathrm{case} M]\!]^\zeta \eta \sim^R_\tau [\![\mathrm{case}\ M^-]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, we have

$$[\![\mathrm{case}M]\!]^{\zeta}\eta = [\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \quad \text{if } [\![M]\!]^{\zeta}\eta = \mathrm{Pair}(x,y)$$
$$[\![\mathrm{case}M^-]\!]\eta' = [\![N^-]\!]\eta'[x := a', y := b'] \quad \text{if } [\![M^-]\!]\eta' = \mathrm{Pair}(x,y)$$

Then it is to show $[\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \sim^R_\tau [\![N^-]\!]\eta'[x := a', y := b']$.

We have the following that $\forall a, b, a', b' \in \mathbf{D}$

$$a \sim^R_\rho b \wedge a' \sim^R_\sigma b' \Rightarrow [\![N]\!]^{\zeta}\eta[x := a, y := a'] \sim^R_\tau [\![N^-]\!]\eta'[x := b, y := b'] \quad \text{(IH1)}$$

Assume $a \sim^R_\rho b \wedge a' \sim^R_\sigma b'$.

By Lemma 2.4.12 (iii) we get $\langle\rho\rangle\zeta(a) \sim^R_\rho b$, $\langle\sigma\rangle\zeta(a') \sim^R_\sigma b'$.

Hence $[\![N]\!]^{\zeta}\eta[x := \langle\rho\rangle\zeta(a), y := \langle\sigma\rangle\zeta(b)] \sim^R_\tau [\![N^-]\!]\eta'[x := a', y := b']$.

11. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]}{\Gamma \vdash^{\mathrm{reg}} \mathrm{In}(M) : \mathrm{fix}\,\alpha.\rho'}$.

We need to show $[\![\mathrm{In}(M)]\!]^{\zeta}\eta \sim^R_{\mathrm{fix}\,\alpha.\rho'} [\![\mathrm{In}(M^-)]\!]\eta'$.

That is, to show $\mathrm{In}([\![M]\!]^{\zeta}\eta) \sim^R_{\mathrm{fix}\,\alpha.\rho'} \mathrm{In}([\![M^-]\!]\eta')$ by Definition 2.3.7 and 2.4.2.

By definition of $\sim^R_{\mathrm{fix}\,\alpha.\rho'}$ (Definition 2.4.8), it is to show

$$[\![M]\!]^{\zeta}\eta \sim^{R[\alpha:=\sim^R_{\mathrm{fix}\,\alpha.\rho'}]}_{\rho'} [\![M^-]\!]\eta'$$

We have the following

$$[\![M]\!]^{\zeta}\eta \sim^R_{\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]} [\![M^-]\!]\eta' \quad \text{(IH1)}$$

We need to show $\sim^{R[\alpha:=\sim^R_{\mathrm{fix}\,\alpha.\rho'}]}_{\rho'} = \sim^R_{\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]}$.

This follows by Lemma 2.4.14.

12. $\dfrac{\Gamma \vdash^{\mathrm{reg}} M : \mathrm{fix}\,\alpha.\rho' \qquad \Gamma, x : \rho'[\mathrm{fix}\,\alpha.\rho'/\alpha] \vdash^{\mathrm{reg}} N : \sigma}{\Gamma \vdash^{\mathrm{reg}} \mathrm{case}M\ \mathrm{of}\{\mathrm{In}(x) \to N\} : \sigma}$.

We need to show $[\![\mathrm{case}M\ \mathrm{of}\{\mathrm{In}(x) \to N\}]\!]^{\zeta}\eta \sim^R_\sigma [\![\mathrm{case}M^-\ \mathrm{of}\{\mathrm{In}(x) \to N^-\}]\!]\eta'$.

By Definition 2.3.7 and 2.4.2, it is to show

$[\![N]\!]^{\zeta}\eta[x := \langle\rho'[\mathrm{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)] \sim^R_\sigma [\![N^-]\!]\eta'[x := b]$.

We have the following

$$\forall a,b(a \sim^R_\sigma b \Rightarrow [\![N]\!]^\zeta \eta[x := a] \sim^R_\sigma [\![N^-]\!]\eta'[x := b]) \qquad \text{(IH1)}$$

Assume $a \sim^R_\rho b$. By Lemma 2.4.12 (iii) we get $\langle \rho'[\text{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a) \sim^R_\rho b$.

Hence $[\![N]\!]^\zeta \eta[x := \langle \rho'[\text{fix}\,\alpha.\rho'/\alpha]\rangle\zeta(a)] \sim^R_\sigma [\![N^-]\!]\eta'[x := b]$.

□

The above theorem (Theorem 2.4.15) yields as an immediate consequence our main result that if from a context Γ we can derive a **LoR** term M with regular type ρ, and for every variable $x \in \text{dom}(\Gamma)$, $\eta(x)$ is an element of $[\![\Gamma(x)]\!]\zeta$, then the value of M and its corresponding untyped term M^- coincide up to the finitary projection $\langle\rho\rangle\zeta$.

Theorem 2.4.16 (Coincidence) If $\Gamma \vdash^{\text{reg}} M : \rho$ and $\eta \in [\![\Gamma]\!]\zeta$ where ζ is a finitary projection, then $[\![M]\!]^\zeta \eta = \langle\rho\rangle\zeta([\![M^-]\!]\eta)$.

Proof. Given a finitary projection ζ, we define $R(\alpha) := \{(a,b) \in \mathbf{D}^2 \mid \zeta(\alpha)(a) = \zeta(\alpha)(b)\}$. Then $R \approx \zeta$, as explained in the example following Definition 2.4.11. By Lemma 2.4.15, we then have $[\![M]\!]^\zeta \eta \sim^R_\rho [\![M^-]\!]\eta$. By Lemma 2.4.12 (i), we get $\langle\rho\rangle\zeta([\![M]\!]^\zeta \eta) = \langle\rho\rangle\zeta([\![M^-]\!]\eta)$. Then, by Soundness Theorem 2.3.8 and the definition of $\langle\rho\rangle\zeta(\mathbf{D})$, we have $[\![M]\!]^\zeta \eta = \langle\rho\rangle\zeta([\![M]\!]^\zeta \eta)$. Thus, $[\![M]\!]^\zeta \eta = \langle\rho\rangle\zeta([\![M^-]\!]\eta)$. □

If $\Gamma = \varnothing$ and ρ is closed in Theorem 2.4.16, we have the following corollary.

Corollary 2.4.17 If $\vdash^{\text{reg}} M : \rho$, then $[\![M]\!] = \langle\rho\rangle[\![M^-]\!]$.

2.5 Conclusion

We have studied a domain-theoretic semantics for Church-style system **LoR** of typed lambda terms and proved that, when restricted to regular types, it is closely related to its untyped counterpart. The reason for studying this domain-theoretic semantics is that it allows for very simply and elegant proofs of computational adequacy, and hence the correctness of program extraction.

Our results could be easily extended to also include full second-order polymorphism $\forall\alpha.\rho$, $\exists\alpha.\rho$ as in [ABL86], but for our application, simple parametric and recursive types are sufficient.

The problem of relating typed and untyped realisability was also studied by Longley [Lon00]. He used a condition called (constructive) logical full abstraction to connect realisability over typed and untyped structures by means of partial combinatory algebra.

As future work we intend to investigate whether the requirement of regularity is indeed necessary for our result to hold. Furthermore, we plan to compare the Church-style system with a corresponding Curry-style system.

Chapter 3

Church's Simple Theory of Types

Contents

Type theories date back in general to the 20th century. *Ramified Theory of Types* was first introduced by Russell [Rus08], and was elaborated in the momentous *Principia Mathematica* [WR13]. The original idea of Russell's type theory is to exclude the set-theoretic paradoxes, which could be conducted from Frege's *Begriffschrift* [Fre79], by postulating the *vicious circle principle*. This principle states that no collection can be introduced by a definition that depends on that collection itself, and is implemented by the use of a hierarchy of levels of types. Unfortunately, this theory was too weak to justify classical mathematics. Russell had to introduce the Axiom of Reducibility to crush the orders down to one. However, Russell's type theory was never completely formalised, i.e. there were no formal definitions of the fundamentals types such as type, proposition or logic formula.

In order to deal with Russell's paradoxes without the requirement of types, Alonzo Church introduced what is nowadays called the lambda calculus in [Chu33]. Later Kleene and Rosser found that the logical system of this lambda calculus was inconsistent [KR35]. To avoid the Kleene-Rosser paradox, Church introduced the *simply typed lambda calculus*. The simple type theory of Church basically is the simply typed lambda calculus with the type of individuals and the type of truth values, providing a notion of logical formulas and a notion of provability.

Building rigorous reasoning systems for hardware and software verifications is a very demanding endeavour. Church's type theory equipped with some modifications

and enhancements has been incorporated into theorem proving systems for specifying and verifying the correctness of mathematical proofs. Notable developments in this direction include HOL [Gor88, GM93], IMPS [FGT93], Isabelle [Pau86, Pau89, Pau94], PVS [ORR+96] and TPS [AINP90]. The logic of HOL is based on Church's type theory extended with equality, implication and a higher-order version of Hilbert's choice operator. As types consist of type constants, type variables, and function types, HOL allows polymorphic types and inference rules for definitions. The IMPS system is an implementation of a logic based on a partial-functions-version of Church's type theory, due to the fact that mathematics focus on the axiomatic method and mainly on functions including partial functions. Isabelle supports ML-style type inference with unification, and uses a polymorphic version of Church's type theory as its logic language. The syntax of the logic of PVS is encoded in Church's type theory with parametric theories and predicate subtyping, since PVS is designed for the specification and proof of digital systems. TPS is the earliest automated theorem provers based on Church's type theory with mating aiming to find an expansion proof [Mil87]. More recently, LEO-II [BPTF08], whose logic is built on Church's type theory, implements an extensional higher-order resolution calculus. Satallax [BB11, Bro12], which extends Church's type theory with extensionality and choice operators, is a system based on higher-order extensional tableaux.

In this chapter, we start with the Simply Typed Lambda Calculus (STLC), which is constructed freely from type atoms. Next, we introduce the notion of an interpretation from one instance of STLC to another. And then we describe the logic and semantics of our version of Church's Simple Type Theory.

3.1 Simply Typed Lambda Calculus

Although the pure typed lambda calculus has only a rather restricted collection of terms and types, its powerful relatively simple extensions have been widely used, and hence build up the theoretical foundations of some theorem provers. The simply typed lambda calculus (STLC), constructed freely from type atoms, will be presented in this section. We will describe the basic logic of the STLC and a substitution operating on typed terms. Later, this STLC will be enriched with more types and other useful constructs in Section 3.3.

Syntax

The set $\mathscr{T}$ of types is generated from a set of base types $\mathscr{B}$ using the function and product type constructors, respectively, $\to$ and $\times$.

Definition 3.1.1 (Types)

$$\mathscr{T} \ni \rho, \sigma ::= b \mid \rho \to \sigma \mid \rho \times \sigma$$

where b ranges over $\mathscr{B}$.

The basic expressions of STLC are called *terms*, given by the following definition.

Definition 3.1.2 (Terms) We define *terms* inductively by the following grammar.

$$\begin{aligned} \Sigma \ni M, N ::= {} & x \mid c \mid \lambda x : \rho . M \mid MN \mid \\ & \langle M, N \rangle \mid \pi_0(M) \mid \pi_1(M) \end{aligned}$$

where x ranges over a set **Var** of variables, and c ranges over a set $\mathscr{C}$ of constants.

Definition 3.1.3 (Free Variables) Let M be a term. Then the set $\mathrm{FV}(M) \subseteq$ **Var** of *free variables* of M is defined recursively as follows based on the structure of terms.

- $\mathrm{FV}(x) := \{x\}$
- $\mathrm{FV}(c) := \varnothing$
- $\mathrm{FV}(\lambda x : \rho . M) := \mathrm{FV}(M) / \{x\}$
- $\mathrm{FV}(MN) := \mathrm{FV}(M) \cup \mathrm{FV}(N)$
- $\mathrm{FV}(\langle M, N \rangle) := \mathrm{FV}(M) \cup \mathrm{FV}(N)$
- $\mathrm{FV}(\pi_0(M)) := \mathrm{FV}(M)$
- $\mathrm{FV}(\pi_1(M)) := \mathrm{FV}(M)$

Typing Rules

A *type context*, also called *environment*, is a set of pairs $\Gamma := x_1 : \rho_1, \ldots, x_n : \rho_n$ (for notational convenience we omit the curly braces) where ρ_i are types and x_i are distinct variables. The set of variables $\{x_1, \ldots, x_n\}$ (which may be empty) is denoted by $\mathrm{dom}(\Gamma)$. Every type ρ_i in the context Γ is denoted by $\Gamma(x_i)$. Thus, in a context, each variable occurs at most once, i.e. if $(x, \rho_1) \in \Gamma$ and $(x, \rho_2) \in \Gamma$, then $\rho_1 = \rho_2$. Also we denote $\Gamma \cup \{x : \rho\}$ as $(\Gamma, x : \rho)$ which does exclude that $x : \rho$ is already in the context Γ.

In order to define the type of a term in a context we need a *typing of constants*, i.e., a set $\mathscr{C}$ of pairs $c : \rho$ such that if $c : \rho$ and $c : \rho'$, then $\rho = \rho'$.

The relation $\Gamma \vdash M : \rho$ (M is a term of type ρ in context Γ) is inductively defined as follows.

$$\Gamma, x:\rho \vdash x:\rho \qquad \Gamma \vdash c:\rho \quad \text{if } c:\rho$$

$$\frac{\Gamma, x:\rho \vdash M:\sigma}{\Gamma \vdash \lambda x:\rho.M:\rho\to\sigma} \qquad \frac{\Gamma \vdash M:\rho\to\sigma \qquad \Gamma \vdash N:\rho}{\Gamma \vdash MN:\sigma}$$

$$\frac{\Gamma \vdash M:\rho \qquad \Gamma \vdash N:\sigma}{\Gamma \vdash \langle M,N\rangle:\rho\times\sigma} \qquad \frac{\Gamma \vdash M:\rho\times\sigma}{\Gamma \vdash \pi_0(M):\rho} \qquad \frac{\Gamma \vdash M:\rho\times\sigma}{\Gamma \vdash \pi_1(M):\sigma}$$

Lemma 3.1.4 If $\Gamma \vdash M : \rho$, then $\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$.

Proof. By induction on M.

- $M \equiv x$.

  We need to show $\mathrm{FV}(x) \subseteq \mathrm{dom}(\Gamma)$.

  We get $\mathrm{FV}(x) = \{x\}$ by Definition 3.1.3, and by the typing rule for variables, we have $\{x\} \subseteq \mathrm{dom}(\Gamma)$.

- $M \equiv c$.

  We need to show $\mathrm{FV}(c) \subseteq \mathrm{dom}(\Gamma)$.

  Since $\mathrm{FV}(c) = \varnothing$ by Definition 3.1.3, we get $\mathrm{FV}(c) \subseteq \mathrm{dom}(\Gamma)$.

- $M \equiv \lambda x:\rho.M$.

  We need to show $\mathrm{FV}(\lambda x:\rho.M) \subseteq \mathrm{dom}(\Gamma)$.

  We get $\mathrm{FV}(\lambda x:\rho.M) = \mathrm{FV}(M)/\{x\}$ by Definition 3.1.3.

  Applying induction hypothesis $\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma, x:\rho)$, we get

  $\mathrm{FV}(M)/\{x\} \subseteq \mathrm{dom}(\Gamma, x:\rho)/\{x\} = (\mathrm{dom}(\Gamma) \cup \{x\})/\{x\} = \mathrm{dom}(\Gamma)$.

- $M \equiv MN$.

  We need to show $\mathrm{FV}(MN) \subseteq \mathrm{dom}(\Gamma)$.

  We get $\mathrm{FV}(MN) = \mathrm{FV}(M) \cup \mathrm{FV}(N)$ by Definition 3.1.3.

  We have the following induction hypothesis

  $$\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma) \qquad \text{(IH1)}$$
  $$\mathrm{FV}(N) \subseteq \mathrm{dom}(\Gamma) \qquad \text{(IH2)}$$

  Applying IH1 and IH2, we get $\mathrm{FV}(M) \cup \mathrm{FV}(N) \subseteq \mathrm{dom}(\Gamma)$.

- $M \equiv \langle M, N \rangle$.

  Similar to the proof of above case.

- $M \equiv \pi_0(M)$.

  We need to show $\mathrm{FV}(\pi_0(M)) \subseteq \mathrm{dom}(\Gamma)$.

  We get $\mathrm{FV}(\pi_0(M)) = \mathrm{FV}(M)$ by Definition 3.1.3. Then it is to show $\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$. This follows immediately by the induction hypothesis.

- $M \equiv \pi_1(M)$.

  Similar to the proof of above case.

□

Lemma 3.1.5 (Uniqueness) If $\Gamma \vdash M : \rho_1$ and $\Gamma \vdash M : \rho_2$, then these derivations are identical, i.e. $\rho_1 = \rho_2$.

Proof. By induction on M.

The typing rules are syntax directed. Hence for a given term M, a context Γ and a type ρ, there is at most one typing rule that derives $\Gamma \vdash M : \rho$.

- $M \equiv x$.

  Since $(x, \rho_1) \in \Gamma$ and $(x, \rho_2) \in \Gamma$, we get $\rho_1 = \rho_2$.

- $M \equiv c$.

  Since c has only one type, we get $\rho_1 = \rho_2$.

- $M \equiv \lambda x : \rho . M$.

  We need to show $\rho \to \sigma_1 = \rho \to \sigma_2$.

  By induction hypothesis, applied to $\Gamma, x : \rho \vdash M : \sigma_1$ and $\Gamma, x : \rho \vdash M : \sigma_2$, we get $\sigma_1 = \sigma_2$.

- $M \equiv MN$.

  We need to show $\sigma_1 = \sigma_2$.

  By induction hypothesis, applied to $\Gamma \vdash M : \rho_1 \to \sigma_1$, $\Gamma \vdash M : \rho_2 \to \sigma_2$, $\Gamma \vdash N : \rho_1$ and $\Gamma \vdash N : \rho_2$, we get $\rho_1 \to \sigma_1 = \rho_2 \to \sigma_2$ and $\rho_1 = \rho_2$. Hence, we have $\sigma_1 = \sigma_2$.

- $M \equiv \langle M, N \rangle$.

  Similar to the proof of above case.

- $M \equiv \pi_0(M)$.

  We need to show $\rho_1 = \rho_2$.

  By induction hypothesis, applied to $\Gamma \vdash M : \rho_1 \times \sigma_1$ and $\Gamma \vdash M : \rho_2 \times \sigma_2$, we get $\rho_1 \times \sigma_1 = \rho_2 \times \sigma_2$. Hence, we have $\rho_1 = \rho_2$.

- $M \equiv \pi_1(M)$.

  Similar to the proof of above case.

□

Substitution

Simplifying or evaluating a term involving lambda expressions requires us to encompass operations of function application. Since free variable substitution may cause variable capture, we need a mechanism to ensure that no bound variable has the same name as a free variable in the term being substituted. An elementary simplification step, called *β-conversion*, involves the substitution of a free variable of a term by another term.

The *α-equivalence* is defined as usual. Intuitively, α-equivalence, $T_1 \equiv_\alpha T_2$ means that T_2 is obtained from T_1 by renaming the bound variables in T_1.

We state without proof the following lemma:

Lemma 3.1.6 α-equivalence is a congruence on lambda expressions and we have that if $M \equiv_\alpha N$, then $\mathrm{FV}(M) = \mathrm{FV}(N)$.

Definition 3.1.7 Let $X, Y \subseteq \mathbf{Var}$, and $\Sigma(X) := \{M \in \Sigma \mid \mathrm{FV}(M) \subseteq X\}$.

A *variable substitution* is a function $\eta : X \to \Sigma(Y)$. Every variable substitution η can be extended to a function $\overline{\eta} : \Sigma(X) \to \Sigma(Y)$ by

$$
\begin{aligned}
&\overline{\eta}(x) := \eta(x)\\
&\overline{\eta}(c) := c\\
&\overline{\eta}(MN) := \overline{\eta}(M)\overline{\eta}(N)\\
&\overline{\eta}(\langle M,N\rangle) := \langle\overline{\eta}(M),\overline{\eta}(N)\rangle\\
&\overline{\eta}(\pi_0(M)) := \pi_0(\overline{\eta}(M))\\
&\overline{\eta}(\pi_1(M)) := \pi_1(\overline{\eta}(M))\\
&\overline{\eta}(\lambda x:\rho.M) := \lambda x' : \rho.\overline{\eta'}(M)
\end{aligned}
$$

where $x' \notin \cup\{\mathrm{FV}(\eta(y)) \mid y \in \mathrm{FV}(\lambda x:\rho.M)\} \cup \mathrm{FV}(\lambda x:\rho.M)$ and for $\eta' : X \to \Sigma(Y \cup \{x'\})$, $\eta'(x) = x'$ and $\eta'(y) = \eta(y)$ for $y \neq x$.

We write $M[N/x]$ for $\overline{\eta}(M)$ where $\eta(x) = N$ and $\eta(y) = y$ for $y \neq x$.

The meaning of $\eta \equiv_\alpha \eta'$ is that for all $x \in X$, $\eta(x) \equiv_\alpha \eta'(x)$.

Lemma 3.1.8 If $M, N \in \Sigma(X)$, $M \equiv_\alpha N$, and $\eta \equiv_\alpha \eta'$, then $\overline{\eta}(M) \equiv_\alpha \overline{\eta'}(N)$.

Proof. By induction on the definition of $\overline{\eta}(M)$. □

Definition 3.1.9 We define *β-equivalence* as the least congruence relation $\equiv_\beta$ on terms such that

$$(\lambda x : \rho.M)N \equiv_\beta M[N/x]$$

We write $\Gamma' \vdash \eta : \Gamma$ if $\Gamma' \vdash \eta(x) : \rho$ for all $x : \rho \in \Gamma$.

Lemma 3.1.10 If $\Gamma \vdash M : \rho$ and $\Gamma' \vdash \eta : \Gamma$, then $\Gamma' \vdash \overline{\eta}(M) : \rho$.

Proof. By induction on the derivation of $\Gamma \vdash M : \rho$. □

3.2 Interpretation of STLC

Let a formal system be a set of operations and axioms in a formal language. An interpretation of a formal system T_1 in a formal system T_2 is a mapping from the expressions of T_1 to the expressions of T_2 which preserves the validity of axioms. Roughly speaking, interpretation is to represent T_1 in T_2. Interpretations are a fundamental logical tool to prove properties about formal systems and support mathematical reasonings. In this section, we will introduce the notion of interpretation in the setting of two instances of STLC. The results can be applied to arbitrary instances of STLC.

Let S_i ($i \in \{1,2\}$) be two instances of a simply typed lambda calculus given by base type sets $\mathscr{B}_i$, constant sets $\mathscr{C}_i$, and typings $:_i$ of the constants. Let $\mathscr{T}_i$ and Σ_i be the corresponding sets of types and terms.

We assume that for every variable x in S_1 we have fixed in a one-to-one fashion a variable $\widetilde{x}$ in S_2. More precisely, let $\widetilde{(\_)}$ be an injective function mapping from **Var** to **Var**, and $\widetilde{X} := \{\widetilde{x} \mid x \in X\}$ where $X \subseteq$ **Var**.

Definition 3.2.1 A *base type substitution* from S_1 to S_2, written as $\xi : S_1 \to S_2$, is a function $\xi : \mathscr{B}_1 \to \mathscr{T}_2$. Any base type substitution ξ can be extended to a function $\xi : \mathscr{T}_1 \to \mathscr{T}_2$ by setting

$$\xi(\rho \to \sigma) := \xi(\rho) \to \xi(\sigma)$$
$$\xi(\rho \times \sigma) := \xi(\rho) \times \xi(\sigma)$$

We also extend base type substitutions to contexts by setting $\xi(\Gamma) := \{\widetilde{x} : \xi(\rho) \mid x : \rho \in \Gamma\}$.

Definition 3.2.2 A *constant substitution* from S_1 to S_2, written as $\theta : S_1 \to S_2$, is a function $\theta : \mathscr{C}_1 \to \Sigma_2(\varnothing)$ s.t. $\theta(c)$ is a closed S_2 term for all S_1 constants c. Any constant substitution θ from S_1 to S_2, together with a base type substitution ξ from S_1 to S_2 determines a function $\theta_\xi : \Sigma_1(X) \to \Sigma_2(\widetilde{X})$ as follows.

$$\begin{array}{ll} \theta_\xi(x) := \widetilde{x} & \theta_\xi(c) := \theta(c) \\ \theta_\xi(\lambda x : \rho.M) := \lambda\widetilde{x} : \xi(\rho).\theta_\xi(M) & \theta_\xi(MN) := \theta_\xi(M)\theta_\xi(N) \\ \theta_\xi(\langle M,N\rangle) := \langle \theta_\xi(M), \theta_\xi(N)\rangle & \theta_\xi(\pi_0(M)) := \pi_0(\theta_\xi(M)) \\ \theta_\xi(\pi_1(M)) := \pi_1(\theta_\xi(M)) & \end{array}$$

Lemma 3.2.3 $\mathrm{FV}(\theta_\xi(M)) = \widetilde{\mathrm{FV}}(M)$ where $\widetilde{\mathrm{FV}}(M) = \{\widetilde{x} \mid x \in \mathrm{FV}(M)\}$.

Proof. By induction on M.

- $M \equiv x$.

  We need to show $\mathrm{FV}(\theta_\xi(x)) = \widetilde{\mathrm{FV}}(x)$.

  From the left-hand side of the equation, we have $\mathrm{FV}(\theta_\xi(x)) = \mathrm{FV}(\widetilde{x})$ by Definition 3.2.2. Then it is equal to $\{\widetilde{x}\}$ by Definition 3.1.3.

  From the right-hand side, since $\mathrm{FV}(x) = \{x\}$, we have $\widetilde{\mathrm{FV}}(x) = \{\widetilde{x}\}$.

  Hence, we get $\{\widetilde{x}\} = \{\widetilde{x}\}$.

- $M \equiv c$.

  We need to show $\mathrm{FV}(\theta_\xi(c)) = \widetilde{\mathrm{FV}}(c)$.

  From the left-hand side of the equation, we have $\mathrm{FV}(\theta_\xi(c)) = \mathrm{FV}(\theta(c)) = \varnothing$ by Definition 3.2.2, since $\theta(c)$ is closed for all constants c.

  From the right-hand side, we have $\widetilde{\mathrm{FV}}(c) = \varnothing$.

  Hence, we get $\varnothing = \varnothing$.

- $M \equiv \lambda x : \rho.M$.

  We need to show $\mathrm{FV}(\theta_\xi(\lambda x : \rho.M)) = \widetilde{\mathrm{FV}}(\lambda x : \rho.M)$.

  From the left-hand side of the equation, we have $\mathrm{FV}(\theta_\xi(\lambda x : \rho.M)) = \mathrm{FV}(\lambda\widetilde{x} : \xi(\rho).\theta_\xi(M))$ by Definition 3.2.2.

  By Definition 3.1.3, we have $\mathrm{FV}(\lambda\widetilde{x} : \xi(\rho).\theta_\xi(M)) = \mathrm{FV}(\theta_\xi(M))/\{\widetilde{x}\}$, and $\mathrm{FV}(\lambda x : \rho.M) = \mathrm{FV}(M)/\{x\}$.

  Then from the right-hand side, we have $\widetilde{\mathrm{FV}}(\lambda x : \rho.M) = \widetilde{\mathrm{FV}}(M)/\{\widetilde{x}\}$.

  Now it is to show $\mathrm{FV}(\theta_\xi(M))/\{\widetilde{x}\} = \widetilde{\mathrm{FV}}(M)/\{\widetilde{x}\}$, i.e. $\mathrm{FV}(\theta_\xi(M)) = \widetilde{\mathrm{FV}}(M)$.

  This follows immediately by the induction hypothesis applied to M.

- $M \equiv MN$.

  We need to show $\mathrm{FV}(\theta_\xi(MN)) = \widetilde{\mathrm{FV}}(MN)$.

  From the left-hand side of the equation, by Definition 3.2.2, we get $\mathrm{FV}(\theta_\xi(MN)) = \mathrm{FV}(\theta_\xi(M)\theta_\xi(N))$.

  We have $\mathrm{FV}(\theta_\xi(M)\theta_\xi(N)) = \mathrm{FV}(\theta_\xi(M)) \cup \mathrm{FV}(\theta_\xi(N))$ and $\mathrm{FV}(MN) = \mathrm{FV}(M) \cup \mathrm{FV}(N)$ by Definition 3.1.3.

  Then from the right-hand side, we have $\widetilde{\mathrm{FV}}(MN) = \widetilde{\mathrm{FV}}(M) \cup \widetilde{\mathrm{FV}}(N)$.

  We need to show $\mathrm{FV}(\theta_\xi(M)) \cup \mathrm{FV}(\theta_\xi(N)) = \widetilde{\mathrm{FV}}(M) \cup \widetilde{\mathrm{FV}}(N)$. Then it is to show $\mathrm{FV}(\theta_\xi(M)) = \widetilde{\mathrm{FV}}(M)$ and $\mathrm{FV}(\theta_\xi(N)) = \widetilde{\mathrm{FV}}(N)$.

  This follows immediately by the induction hypothesis applied to M and N.

- $M \equiv \langle M, N \rangle$.

  Similar to the proof of above case.

- $M \equiv \pi_0(M)$.

  We need to show $\mathrm{FV}(\theta_\xi(\pi_0(M))) = \widetilde{\mathrm{FV}}(\pi_0(M))$.

  From the left-hand side of the equation by Definition 3.2.2 we get $\mathrm{FV}(\theta_\xi(\pi_0(M))) = \mathrm{FV}(\pi_0(\theta_\xi(M)))$.

  We have $\mathrm{FV}(\pi_0(\theta_\xi(M))) = \mathrm{FV}(\theta_\xi(M))$ and $\mathrm{FV}(\pi_0(M)) = \mathrm{FV}(M)$ by Definition 3.1.3. Then from the right-hand side, we have $\widetilde{\mathrm{FV}}(\pi_0(M)) = \widetilde{\mathrm{FV}}(M)$.

  We need to show $\mathrm{FV}(\theta_\xi(M)) = \widetilde{\mathrm{FV}}(M)$.

  This follows immediately by the induction hypothesis applied to M.

- $M \equiv \pi_1(M)$.

  Similar to the proof of above case.

□

Let $\eta : X \to \Sigma_2(Y)$ and $\eta' : \widetilde{X} \to \Sigma_2(Y)$, for every variable substitution η of S_1, we set $\eta'(\widetilde{x}) = \eta(x)$.

The mapping θ_ξ commutes with variable substitutions in the following sense.

$$\begin{array}{ccc} \Sigma_1(X) & \xrightarrow{\overline{\eta}} & \Sigma_2(Y) \\ {\scriptstyle \theta_\xi}\downarrow & \circlearrowleft & {\scriptstyle \theta_\xi}\downarrow \\ \Sigma_1(\widetilde{X}) & \xrightarrow[\overline{(\theta_\xi \circ \eta')}]{} & \Sigma_2(\widetilde{Y}) \end{array}$$

Lemma 3.2.4 $\theta_\xi \circ \overline{\eta} = \overline{(\theta_\xi \circ \eta')} \circ \theta_\xi$.

Proof. By induction on M.

- $M \equiv x$.

  We need to show $\theta_\xi(\overline{\eta}(x)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(x))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(x)) = \theta_\xi(\eta(x)) = \theta_\xi(\eta'(\widetilde{x})) = (\theta_\xi \circ \eta')(\widetilde{x})$.

  From the right-hand side, we have $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(x)) = \overline{(\theta_\xi \circ \eta')}(\widetilde{x})$ by Definition 3.2.2. Then by Definition 3.1.7, we have $\overline{(\theta_\xi \circ \eta')}(\widetilde{x}) = (\theta_\xi \circ \eta')(\widetilde{x})$.

  Hence, we get $(\theta_\xi \circ \eta')(\widetilde{x}) = (\theta_\xi \circ \eta')(\widetilde{x})$.

- $M \equiv c$.

  We need to show $\theta_\xi(\overline{\eta}(c)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(c))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(c)) = \theta_\xi(c) = \theta(c)$.

  From the right-hand side, we have $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(c)) = \overline{(\theta_\xi \circ \eta')}(\theta(c))$ by Definition 3.2.2. We have $\overline{(\theta_\xi \circ \eta')}(\theta(c)) = \theta(c)$ since $\theta(c)$ is closed for all constants c. Hence, we get $\theta(c) = \theta(c)$.

- $M \equiv \lambda x : \rho.M$.

  We need to show $\theta_\xi(\overline{\eta}(\lambda x : \rho.M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(\lambda x : \rho.M))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(\lambda x : \rho.M)) = \theta_\xi(\lambda x' : \rho.\overline{\eta'}(M)) = \lambda\widetilde{x'} : \xi(\rho).\theta_\xi(\overline{\eta'}(M))$.

  From the right-hand side, by Definition 3.1.7 and 3.2.2 we get $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(\lambda x : \rho.M)) = \overline{(\theta_\xi \circ \eta')}(\lambda\widetilde{x} : \xi(\rho).\theta_\xi(M)) = \lambda\widetilde{x'} : \xi(\rho).\overline{(\theta_\xi \circ \eta')'}(\theta_\xi(M)))$.

  Now it is to show $\lambda\widetilde{x'} : \xi(\rho).\theta_\xi(\overline{\eta'}(M)) = \lambda\widetilde{x'} : \xi(\rho).\overline{(\theta_\xi \circ \eta')'}(\theta_\xi(M)))$, i.e. to show $\theta_\xi(\overline{\eta'}(M)) = \overline{(\theta_\xi \circ \eta')'}(\theta_\xi(M)))$.

  This follows by the induction hypothesis

  $$\theta_\xi(\overline{\eta'}(M)) = \overline{(\theta_\xi \circ \eta'')}(\theta_\xi(M)) \qquad \text{(IH1)}$$

- $M \equiv MN$.

  We need to show $\theta_\xi(\overline{\eta}(MN)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(MN))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(MN)) = \theta_\xi(\overline{\eta}(M)\overline{\eta}(N)) = \theta_\xi(\overline{\eta}(M))\theta_\xi(\overline{\eta}(N))$.

From the right-hand side, we have $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(MN)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)\theta_\xi(N))$ $= (\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)))$ by Definition 3.1.7 and 3.2.2.

We need to show $\theta_\xi(\overline{\eta}(M))\theta_\xi(\overline{\eta}(N)) = (\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)))$, i.e. to show $\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))$ and

$\theta_\xi(\overline{\eta}(N)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N))$.

These follow by the induction hypotheses

$$\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)) \quad \text{(IH1)}$$

$$\theta_\xi(\overline{\eta}(N)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)) \quad \text{(IH2)}$$

- $M \equiv \langle M,N \rangle$.

  We need to show $\theta_\xi(\overline{\eta}(\langle M,N \rangle)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(\langle M,N \rangle))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(\langle M,N \rangle)) = \theta_\xi(\langle \overline{\eta}(M), \overline{\eta}(N) \rangle) = \langle \theta_\xi(\overline{\eta}(M)), \theta_\xi(\overline{\eta}(N)) \rangle$.

  From the right-hand side, we get $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(\langle M, N \rangle)) = \overline{(\theta_\xi \circ \eta')}(\langle \theta_\xi(M),$ $\theta_\xi(N) \rangle) = \langle \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)), \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)) \rangle$ by Definition 3.1.7 and 3.2.2.

  Now it is to show

  $\langle \theta_\xi(\overline{\eta}(M)), \theta_\xi(\overline{\eta}(N)) \rangle = \langle \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)), \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)) \rangle$,

  i.e. $\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))$ and $\theta_\xi(\overline{\eta}(N)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N))$.

  These follow by the induction hypotheses

  $$\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)) \quad \text{(IH1)}$$

  $$\theta_\xi(\overline{\eta}(N)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(N)) \quad \text{(IH2)}$$

- $M \equiv \pi_0(M)$.

  We need to show $\theta_\xi(\overline{\eta}(\pi_0(M))) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(\pi_0(M)))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(\pi_0(M))) = \theta_\xi(\pi_0(\overline{\eta}(M))) = \pi_0(\theta_\xi(\overline{\eta}(M)))$.

  From the right-hand side, by Definition 3.1.7 and 3.2.2 we have

  $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(\pi_0(M))) = \overline{(\theta_\xi \circ \eta')}(\pi_0(\theta_\xi(M))) = \pi_0(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)))$.

  Now it is to show $\pi_0(\theta_\xi(\overline{\eta}(M))) = \pi_0(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)))$, i.e. it is to show $\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))$.

  This follows by the induction hypothesis

  $$\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)) \quad \text{(IH1)}$$

- $M \equiv \pi_1(M)$.

  We need to show $\theta_\xi(\overline{\eta}(\pi_1(M))) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(\pi_1(M)))$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.2.2, we have $\theta_\xi(\overline{\eta}(\pi_1(M))) = \theta_\xi(\pi_1(\overline{\eta}(M))) = \pi_1(\theta_\xi(\overline{\eta}(M)))$.

  From the right-hand side, by Definition 3.1.7 and 3.2.2 we have $\overline{(\theta_\xi \circ \eta')}(\theta_\xi(\pi_1(M))) = \overline{(\theta_\xi \circ \eta')}(\pi_1(\theta_\xi(M))) = \pi_1(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)))$.

  Now it is to show $\pi_1(\theta_\xi(\overline{\eta}(M))) = \pi_1(\overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)))$, i.e. it is to show $\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M))$.

  This follows by the induction hypothesis

$$\theta_\xi(\overline{\eta}(M)) = \overline{(\theta_\xi \circ \eta')}(\theta_\xi(M)) \qquad \text{(IH1)}$$

□

Definition 3.2.5 (Interpretation) An *interpretation* of S_1 in S_2, written $(\xi, \theta) : S_1 \to S_2$ consists of a base type substitution $\xi : S_1 \to S_2$ and a constant substitution $\theta : S_1 \to S_2$ such that whenever $c :_1 \rho$, then $\vdash_2 \theta(c) : \xi(\rho)$.

Theorem 3.2.6 (Interpretation) Assume (ξ, θ) is an interpretation of S_1 in S_2. If $\Gamma \vdash_1 M : \rho$, then $\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\rho)$.

Proof. By induction on the derivation of $\Gamma \vdash_1 M : \rho$.

- $\Gamma, x : \rho \vdash_1 x : \rho$

  We need to show $\xi(\Gamma, x : \rho) \vdash_2 \theta_\xi(x) : \xi(\rho)$.

  By Definition 3.2.1 we have $\xi(\Gamma, x : \rho) = \xi(\Gamma) \cup \{\widetilde{x} : \xi(\rho)\}$.

  By Definition 3.2.2 we have $\theta_\xi(x) : \xi(\rho) = \widetilde{x} : \xi(\rho)$.

  Hence we get $\xi(\Gamma, x : \rho) \vdash_2 \theta_\xi(x) : \xi(\rho)$.

- $\Gamma \vdash_1 c : \rho$ if $c : \rho$

  We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(c) : \xi(\rho)$. This follows from Definition 3.2.5.

- $$\frac{\Gamma, x : \rho \vdash_1 M : \sigma}{\Gamma \vdash_1 \lambda x : \rho . M : \rho \to \sigma}$$

  We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(\lambda x : \rho . M) : \xi(\rho \to \sigma)$.

  We have $\theta_\xi(\lambda x : \rho . M) : \xi(\rho \to \sigma) = \lambda \widetilde{x} : \xi(\rho) . \theta_\xi(M) : \xi(\rho \to \sigma) = \lambda \widetilde{x} : \xi(\rho) . \theta_\xi(M) : \xi(\rho) \to \xi(\sigma)$ by Definition 3.2.2 and 3.2.1.

  We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\sigma)$.

This follows by the induction hypothesis

$$\xi(\Gamma, x:\rho) \vdash_2 \theta_\xi(M) : \xi(\sigma) \qquad \text{(IH1)}$$

- $\dfrac{\Gamma \vdash_1 M : \rho \to \sigma \qquad \Gamma \vdash_1 N : \rho}{\Gamma \vdash_1 MN : \sigma}$

We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(MN) : \xi(\sigma)$.

By Definition 3.2.2 it is to show $\xi(\Gamma) \vdash_2 \theta_\xi(M)\theta_\xi(N) : \xi(\sigma)$.

We have the following induction hypothesis

$$\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\rho \to \sigma) \qquad \text{(IH1)}$$
$$\xi(\Gamma) \vdash_2 \theta_\xi(N) : \xi(\rho) \qquad \text{(IH2)}$$

By IH1, IH2 and Definition 3.2.1, we have $\xi(\Gamma) \vdash_2 \theta_\xi(M)\theta_\xi(N) : \xi(\sigma)$.

- $\dfrac{\Gamma \vdash_1 M : \rho \qquad \Gamma \vdash_1 N : \sigma}{\Gamma \vdash_1 \langle M, N\rangle : \rho \times \sigma}$

We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(\langle M, N\rangle) : \xi(\rho \times \sigma)$.

We have $\theta_\xi(\langle M, N\rangle) : \xi(\rho \times \sigma) = \langle \theta_\xi(M), \theta_\xi(N)\rangle : \xi(\rho \times \sigma) = \langle \theta_\xi(M), \theta_\xi(N)\rangle : \xi(\rho) \times \xi(\sigma)$ by Definition 3.2.2 and 3.2.1.

We have the following induction hypothesis

$$\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\rho) \qquad \text{(IH1)}$$
$$\xi(\Gamma) \vdash_2 \theta_\xi(N) : \xi(\sigma) \qquad \text{(IH2)}$$

By IH1 and IH2, we have $\xi(\Gamma) \vdash_2 \theta_\xi(\langle M, N\rangle) : \xi(\rho \times \sigma)$.

- $\dfrac{\Gamma \vdash_1 M : \rho \times \sigma}{\Gamma \vdash_1 \pi_0(M) : \rho}$

We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(\pi_0(M)) : \xi(\rho)$.

By Definition 3.2.2 we have $\theta_\xi(\pi_0(M)) : \xi(\rho) = \pi_0(\theta_\xi(M)) : \xi(\rho)$.

We have the following induction hypothesis

$$\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\rho \times \sigma) \qquad \text{(IH1)}$$

By IH1 and Definition 3.2.1, we have $\xi(\Gamma) \vdash_2 \theta_\xi(\pi_0(M)) : \xi(\rho)$.

- $\dfrac{\Gamma \vdash_1 M : \rho \times \sigma}{\Gamma \vdash_1 \pi_1(M) : \sigma}$

We need to show $\xi(\Gamma) \vdash_2 \theta_\xi(\pi_1(M)) : \xi(\rho)$.

By Definition 3.2.2 we have $\theta_\xi(\pi_1(M)) : \xi(\rho) = \pi_1(\theta_\xi(M)) : \xi(\rho)$.

We have the following induction hypothesis

$$\xi(\Gamma) \vdash_2 \theta_\xi(M) : \xi(\rho \times \sigma) \tag{IH1}$$

By IH1 and Definition 3.2.1, we have $\xi(\Gamma) \vdash_2 \theta_\xi(\pi_1(M)) : \xi(\rho)$.

□

One important aspect of an interpretation is that it preserves full β-equality. To this end, we start with a definition of congruence.

Definition 3.2.7 Let $\sim_1$ be a binary relation on S_1 terms. We inductively defined a congruence $\approx_1$ on S_1 generated by $\sim_1$ as follows.

- If $M \sim_1 N$, then $M \approx_1 N$.
- If $M \approx_1 N$, then $\lambda x : \rho.M \approx_1 \lambda x : \rho.N$.
- If $M \approx_1 M'$ and $N \approx_1 N'$, then $MN \approx_1 M'N'$.
- If $M \approx_1 M'$ and $N \approx_1 N'$, then $\langle M,N \rangle \approx_1 \langle M',N' \rangle$.
- If $M \approx_1 N$, then $\pi_0(M) \approx_1 \pi_0(N)$.
- If $M \approx_1 N$, then $\pi_1(M) \approx_1 \pi_1(N)$.

The next two lemmas explain why our interpretation respects β-equality.

Lemma 3.2.8 Let $\approx_2$ be a congruence on S_2 terms. Let $(\xi, \theta) : S_1 \to S_2$ be an interpretation.

If θ_ξ respects $\sim_1$ and $\approx_2$, i.e. for all M, N if $M \sim_1 N$, then $\theta_\xi(M) \approx_2 \theta_\xi(N)$, then θ_ξ respects $\approx_1$ and $\approx_2$, i.e. for all M, N if $M \approx_1 N$ then $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

Proof. We assume θ_ξ respects $\sim_1$ and $\approx_2$, to show for all M, N if $M \approx_1 N$, then $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

By induction on $\approx_1$.

- $M \approx_1 N$.

  We need to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$. This follows by Definition 3.2.7 and the assumption that if $M \sim_1 N$, then $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

- $\lambda x : \rho.M \approx_1 \lambda x : \rho.N$.

  We need to show $\theta_\xi(\lambda x : \rho.M) \approx_2 \theta_\xi(\lambda x : \rho.N)$.

By Definition 3.2.2, it is to show $\lambda\widetilde{x}:\xi(\rho).\theta_\xi(M) \approx_2 \lambda\widetilde{x}:\xi(\rho).\theta_\xi(N)$. By Definition 3.2.7, it is to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

This follows by the induction hypothesis

$$\theta_\xi(M) \approx_2 \theta_\xi(N) \tag{IH1}$$

- $MN \approx_1 M'N'$.

  We need to show $\theta_\xi(MN) \approx_2 \theta_\xi(M'N')$.

  By Definition 3.2.2, it is to show $\theta_\xi(M)\theta_\xi(N) \approx_2 \theta_\xi(M')\theta_\xi(N')$. By Definition 3.2.7, it is to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$ and $\theta_\xi(M') \approx_2 \theta_\xi(N')$.

  These follow by the induction hypotheses

$$\theta_\xi(M) \approx_2 \theta_\xi(N) \tag{IH1}$$
$$\theta_\xi(M') \approx_2 \theta_\xi(N') \tag{IH2}$$

- $\langle M,N\rangle \approx_1 \langle M',N'\rangle$.

  We need to show $\theta_\xi(\langle M,N\rangle) \approx_2 \theta_\xi(\langle M',N'\rangle)$.

  By Definition 3.2.2, it is to show $\langle\theta_\xi(M),\theta_\xi(N)\rangle \approx_2 \langle\theta_\xi(M'),\theta_\xi(N')\rangle$. By Definition 3.2.7, it is to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$ and $\theta_\xi(M') \approx_2 \theta_\xi(N')$.

  These follow by the induction hypotheses

$$\theta_\xi(M) \approx_2 \theta_\xi(N) \tag{IH1}$$
$$\theta_\xi(M') \approx_2 \theta_\xi(N') \tag{IH2}$$

- $\pi_0(M) \approx_1 \pi_0(N)$.

  We need to show $\theta_\xi(\pi_0(M)) \approx_2 \theta_\xi(\pi_0(N))$.

  By Definition 3.2.2, it is to show $\pi_0(\theta_\xi(M)) \approx_2 \pi_0(\theta_\xi(N))$. By Definition 3.2.7, it is to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

  This follows by the induction hypothesis

$$\theta_\xi(M) \approx_2 \theta_\xi(N) \tag{IH1}$$

- $\pi_1(M) \approx_1 \pi_1(N)$.

  We need to show $\theta_\xi(\pi_1(M)) \approx_2 \theta_\xi(\pi_1(N))$.

  By Definition 3.2.2, it is to show $\pi_1(\theta_\xi(M)) \approx_2 \pi_1(\theta_\xi(N))$. By Definition 3.2.7, it is to show $\theta_\xi(M) \approx_2 \theta_\xi(N)$.

  This follows by the induction hypothesis

$$\theta_\xi(M) \approx_2 \theta_\xi(N) \tag{IH1}$$

□

Lemma 3.2.9 Every interpretation $\theta_\xi : S_1 \to S_2$ respects β-equality.

Proof. By Lemma 3.2.8 it suffices to show that θ_ξ respects the relation generating $\equiv_\beta$.

Hence, we first have to show $\theta_\xi((\lambda x : \rho.M)N) \equiv_\beta \theta_\xi(M[N/x])$. The left hand side is equal to $(\lambda \widetilde{x} : \xi(\rho).\theta_\xi(M))\theta_\xi(N)$ by Definition 3.2.2. By Definition 3.1.7 and Lemma 3.2.4, the right hand side is equal to $\theta_\xi(M)[\theta_\xi(N)/\widetilde{x}]$. Hence both sides are β-equality. □

3.3 Church's Simple Theory of Types (CST)

While in a general STLC there is no restriction on base types and constants, Church's type theory uses particular logical constants such as negation, disjunction, universal quantifier and a description or selection operator. From these constants, one can define the other logical operators. Church also introduced propositional, quantificational, functional extensionality, infinity, descriptions and choice axioms. Church remarked that one may introduce an axiom of Boolean extensionality ($p \equiv q \supset p = q$), although it was excluded in his type theory.

In this section, CST, a particular instance of Simple Theory of Types according to Church, is presented. We assume two kinds of base types: a set of base types for individuals and the type of propositions (or truth values). We also assume a specific set of constants and their typings. The rest is the same as in STLC.

Basic Logic

The types are those of STLC, but with the restriction that the set $\mathscr{B}$ of base types contains a set $\mathscr{I}$ of base types for individuals and the proposition type o. They are given by the following grammar.

Definition 3.3.1 (Types)

$$\mathscr{T} \ni \rho, \sigma ::= b \mid \rho \to \sigma \mid \rho \times \sigma$$

where

$$\mathscr{B} \ni b ::= \iota \in \mathscr{I} \mid o$$

Predicate types are types that are canonically (in any ccc) isomorphic to a finite product of types of the form $\rho \to o$. The inductive definition is below.

Definition 3.3.2 We define *predicate types* inductively as follows.

- o is a predicate type.

- if ρ and σ are predicate types, then $\rho \times \sigma$ is a predicate type.
- if ρ is arbitrary and σ is a predicate type, then $\rho \to \sigma$ is a predicate type.

The set of constants contains at least logical conjunction, disjunction, implication, universal and existential quantifiers, equality, the least and greatest fixed points.

Definition 3.3.3 (Constants) We assume a set $\mathscr{C}$ of typed constants $c : \rho$ that requires logical constants $\wedge$, $\vee$, $\to$, $\forall_\rho$, $\exists_\rho$, $=_\rho$, μ_ρ and ν_ρ as a minimum.

$$\begin{aligned} \wedge, \vee, \to &: o \to o \to o \\ \forall_\rho, \exists_\rho &: (\rho \to o) \to o \\ =_\rho &: \rho \to \rho \to o \\ \mu_\rho, \nu_\rho &: (\rho \to \rho) \to \rho \end{aligned}$$

In $\forall_\rho$, $\exists_\rho$ and $=_\rho$ the type ρ is arbitrary while for μ_ρ and ν_ρ, ρ is restricted to predicate types.

Note the overloading of the symbol $\to$ which we use at the same time as the function type constructor and the constant for implication.

The constants μ_ρ and ν_ρ will be interpreted as least and greatest fixed point operators. In order to precisely express their properties we need higher-order versions of inclusion between predicates, which can be declared meaningfully only by predicate types.

Definition 3.3.4 For every predicate type ρ we define the *inclusion operator* $\subseteq_\rho : \rho \to \rho \to o$ inductively as follows.

$$\begin{aligned} \subseteq_o &:= \lambda x, y : o.\, x \to y \\ \subseteq_{\rho \times \sigma} &:= \lambda x, y : \rho \times \sigma.\, \pi_0(x) \subseteq_\rho \pi_0(y) \wedge \pi_1(x) \subseteq_\sigma \pi_1(y) \\ \subseteq_{\rho \to \sigma} &:= \lambda x, y : \rho \to \sigma. \forall u : \rho.\, xu \subseteq_\sigma yu \end{aligned}$$

Definition 3.3.5 For every predicate type ρ define a *monotonicity* predicate.

$$\begin{aligned} &\mathrm{mono}_\rho : (\rho \to \rho) \to o \\ &\mathrm{mono}_\rho := \lambda \Phi : \rho \to \rho\, .\, \forall x, y : \rho.\, x \subseteq_\rho y \to \Phi x \subseteq_\rho \Phi y \end{aligned}$$

Church introduced a proof calculus which we will also do, but later. We first look at the semantics in domain-theoretical style.

Models of CST

For CST there are two obvious choices of models: the classical set-theoretic model and an interpretation into constructive type theory. In both cases the models can be

extended to the realisability interpretation of CST, which will be discussed in detail later (Chapter 4). We will extend CST to the realisability interpretation of CST by adding an extra base type δ for realisers and extra constants nil, in_L, in_R, pr_L, pr_R, pair, app, fun, case, rec with the following typing

$$\begin{aligned}
\mathrm{nil} &: \delta \\
\mathrm{in}_L, \mathrm{in}_R, \mathrm{pr}_L, \mathrm{pr}_R &: \delta \to \delta \\
\mathrm{pair}, \mathrm{app} &: \delta \to \delta \to \delta \\
\mathrm{fun} &: (\delta \to \delta) \to \delta \\
\mathrm{case} &: \delta \to (\delta \to \delta) \to (\delta \to \delta) \to \delta \\
\mathrm{rec} &: (\delta \to \delta) \to \delta
\end{aligned}$$

We sketch the type-theoretic interpretation only very briefly since, although it is useful for the constructivist to know that the systems have an entirely constructive interpretation, the point of realisability is that it smoothly links a fragment of classical mathematics to computation.

In the type-theoretic interpretation the types of CST are interpreted as types in the type theory. The base type o is interpreted as the type of propositions and the base types $\iota \in \mathscr{I}$ are interpreted as some inhabited types. Product and function types are interpreted type-theoretically. Constants are interpreted as inhabitants of their types. The propositional constants $\to$, $\wedge$ and $\vee$ are interpreted type-theoretically as function, product and sum types. The quantifiers $\forall_\rho$ and $\exists_\rho$ are interpreted as dependent product and dependent sum (Π- and Σ-types). The extension of the type-theoretic model to realisability would be difficult, since it requires to interpret the constant fun by δ as a type that contains its own function space.

In the classical model types are interpreted as sets in the following way: the base type o is interpreted as the set $\{0,1\}$ of boolean values (0 stands for false and 1 for true); the base types $\iota \in \mathscr{I}$ are interpreted as some nonempty set. Product and function types are interpreted as Cartesian product and set-theoretic function space. Constants are interpreted as elements of their types. The propositional constants $\to$, $\wedge$ and $\vee$ are interpreted as the corresponding boolean functions of implication, conjunction and disjunction. The quantifiers $\forall_\rho$ and $\exists_\rho$ are interpreted as minimum and maximum functions. Hence, if ρ is interpreted as an infinite set, quantifiers of type ρ are in no reasonable sense computable.

The classical model of CST can be extended to a model of the realisability interpretation of CST by interpreting the extra base type δ as an element of Scott domain D obtained as the solution to the recursive domain equation (as mentioned in Section 2.3)

$$\mathbf{D} \simeq (\mathbf{1} + \mathbf{D} + \mathbf{D} + \mathbf{D} + \mathbf{D} \times \mathbf{D} + [\mathbf{D} \to \mathbf{D}])_\perp$$

where $+$ and $\times$ denote the domain-theoretic separated sum and product, and $[D \to D]$ denotes the continuous function space.

There seems to be a mismatch with the interpretation of types in CST. First, the type δ is interpreted as a domain while the types in CST are interpreted as plain sets. Second, the type of fun is $(\delta \to \delta) \to \delta$, which requires fun to map all set-theoretic functions $f : D \to D$ to an element $\mathrm{fun}(f) \in D$, while $\mathrm{fun}(f)$ is defined only for continuous f.

In order to solve this problem we slightly generalise the notion of a classical model by interpreting types as objects in the cartesian closed category of quasi-domains and continuous functions. By a *quasi-domain* we mean a topological space whose T_0-collapse is a Scott domain. Quasi-domains may also be defined order-theoretically as those quasi-orders whose order-collapse (identifying points x and y such that $x \leq y$ and $y \leq x$) is a Scott domain. It is easy to see that the continuous function space with the pointwise topology is the exponential in the category of quasi-domains. Every classical model of CST in the previous sense can be viewed as a model in the new sense by endowing plain nonempty sets with the trivial topology. Note that nonempty trivial topological spaces have the one-point domain as T_0 collapse and all set-theoretic functions between such spaces are continuous.

Semantics

We will begin by defining a model in order to give semantics for CST.

Definition 3.3.6 A pre-model $\mathscr{M}_p := (\mathbf{D}_b)_{b \in \mathscr{B}}$ where for each base type $b \in \mathscr{B}$, $\mathbf{D}_b$ is a quasi-domain. We require additionally that $D_o = \{0,1\}$ (0, 1 representing falsity and truth) considered as a quasi-domain with the trivial topology.

Definition 3.3.7 (Semantics of types) Given pre-model $\mathscr{M}_p$ we inductively define a quasi-domain $[\![\rho]\!]$ for every type ρ

$$\begin{aligned} [\![b]\!] &:= \mathbf{D}_b \\ [\![\rho \to \sigma]\!] &:= [[\![\rho]\!] \to [\![\sigma]\!]] \\ [\![\rho \times \sigma]\!] &:= [\![\rho]\!] \times [\![\sigma]\!] \end{aligned}$$

Note that a quasi-domain is given as the meaning for base types, and the meaning of function types is provided by a continuous function space.

Definition 3.3.8 A model $\mathscr{M}$ is a pre-model $\mathscr{M}_p$ together with a value $[\![c]\!] \in [\![\rho]\!]$ for every constant $c : \rho$.

In the following, we consider a model $\mathscr{M}$ with the constant values defined as follows.

For $[\![\wedge]\!], [\![\vee]\!], [\![\to]\!] : [\{0,1\} \to \{0,1\} \to \{0,1\}]$, $[\![\forall_\rho]\!], [\![\exists_\rho]\!] : [([\![\rho]\!] \to \{0,1\}) \to$

$\{0,1\}]$ and $[\![=_\rho]\!] : [[\![\rho]\!] \to [\![\rho]\!] \to \{0,1\}]$, we define

$$([\![\wedge]\!](a))(b) := \begin{cases} 1 & \text{if } a = 1 \text{ and } b = 1 \\ 0 & \text{o.w.} \end{cases}$$

$$([\![\vee]\!](a))(b) := \begin{cases} 0 & \text{if } a = 0 \text{ and } b = 0 \\ 1 & \text{o.w.} \end{cases}$$

$$([\![\to]\!](a))(b) := \begin{cases} 1 & \text{if } a = 1 \text{ and } b = 1 \\ 0 & \text{o.w.} \end{cases}$$

$$[\![\forall_\rho]\!](p) := \begin{cases} 1 & \text{if } \forall m \in [\![\rho]\!].\, p(m) = 1 \\ 0 & \text{o.w.} \end{cases}$$

$$:= min\{p(m) \mid m \in [\![\rho]\!]\}$$

$$[\![\exists_\rho]\!](p) := \begin{cases} 1 & \text{if } \exists m \in [\![\rho]\!].\, p(m) = 1 \\ 0 & \text{o.w.} \end{cases}$$

$$:= max\{p(m) \mid m \in [\![\rho]\!]\}$$

$$([\![=_\rho]\!](m))(n) := m = n$$

In order to give a definition of $[\![\mu_\rho]\!]$ and $[\![\nu_\rho]\!]$, we first define predicates $\subseteq_\rho : [\![\rho]\!] \to [\![\rho]\!] \to \{0,1\}$ inductively as follows for every predicate type ρ

$$\subseteq_o := \lambda x, y \in \{0,1\}.\, x\, [\![\to]\!]\, y$$

$$\subseteq_{\rho\times\sigma} := \lambda x, y \in [\![\rho \times \sigma]\!].\, \pi_0(x) \subseteq_\rho \pi_0(y)\, [\![\wedge]\!]\, \pi_1(x) \subseteq_\sigma \pi_1(y)$$

$$\subseteq_{\rho\to\sigma} := \lambda x, y \in [\![\rho \to \sigma]\!].\, [\![\forall_\rho]\!](\lambda u \in [\![\rho]\!].\, x(u) \subseteq_\sigma y(u))$$

It is easy to see that $[\![\subseteq_\rho]\!] = \subseteq_\rho$, and $\subseteq_\rho$ is a partial order.

Obviously we have that $\subseteq_\rho$ is a complete lattice, where the supremum $\cup_i x_i$ and infimum $\cap_i x_i$ of any family objects exist ($x_i \in [\![\rho]\!]$). Therefore by Tarski's theorem [Tar55] for a $\subseteq_\rho$-monotone function $g : [\![\rho]\!] \to [\![\rho]\!]$, the least and greatest fixed points LFP(g) and GFP(g) respectively exist, and

$$\text{LFP}(g) := \bigcap\{x \mid x \in [\![\rho]\!], gx \subseteq_\rho x\} \quad \text{GFP}(g) := \bigcup\{x \mid x \in [\![\rho]\!], x \subseteq_\rho gx\}$$

Now we define $[\![\mu_\rho]\!] : [[\![\rho]\!] \to [\![\rho]\!]] \to [\![\rho]\!]$ and $[\![\nu_\rho]\!] : [[\![\rho]\!] \to [\![\rho]\!]] \to [\![\rho]\!]$ as follows.

$$[\![\mu_\rho]\!](f) := \text{LFP}(\lambda x \in [\![\rho]\!].\, \bigcup_{y \subseteq_\rho x} f(y))$$

$$[\![\nu_\rho]\!](f) := \text{GFP}(\lambda x \in [\![\rho]\!].\, \bigcap_{x \subseteq_\rho y} f(y))$$

Note that $\lambda x \in [\![\rho]\!].\, \bigcup_{y \subseteq_\rho x} f(y)$ is monotone, since if x increases then y increases because $\subseteq_\rho$ is a partial order. Then the union becomes bigger. Similarly, $\lambda x \in$

$[\![\rho]\!].\bigcap_{x\subseteq_\rho y} f(y)$ is monotone, since if x increases then y decreases. Then the intersection becomes bigger. In addition, every function from $[\![\rho]\!]$ to $[\![\rho]\!]$ is continuous, since ρ ranges over predicate types which means they all end up with o, then the topology is trivial.

Given a context Γ, let η be an environment mapping from $\mathrm{dom}(\Gamma)$ to $[\![\Gamma(x)]\!]$. We write $\eta \models \Gamma$ if for every i, $\eta(x_i) \in [\![\rho_i]\!]$ where $x_i \in \mathrm{dom}(\Gamma)$.

Definition 3.3.9 (Semantics of terms) For every derivation $\Gamma \vdash M : \rho$, if $\eta \models \Gamma$, then we define the value $[\![M]\!]^\Gamma \eta \in [\![\rho]\!]$.

$$\begin{aligned}
[\![x]\!]^\Gamma \eta &:= \eta(x) \\
[\![c]\!]^\Gamma \eta &:= [\![c]\!] \\
[\![\lambda x:\rho.M]\!]^\Gamma \eta &:= \lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma,x:\rho}\eta[x:=a] \\
[\![MN]\!]^\Gamma \eta &:= [\![M]\!]^\Gamma \eta([\![N]\!]^\Gamma \eta) \\
[\![\langle M,N\rangle]\!]^\Gamma \eta &:= \langle [\![M]\!]^\Gamma \eta, [\![N]\!]^\Gamma \eta\rangle \\
[\![\pi_0(M)]\!]^\Gamma \eta &:= \pi_0([\![M]\!]^\Gamma \eta) \\
[\![\pi_1(M)]\!]^\Gamma \eta &:= \pi_1([\![M]\!]^\Gamma \eta)
\end{aligned}$$

where $[\![c]\!] \in [\![\rho]\!]$ for each constant $c : \rho$.

A term of type o in a context Γ is called Γ-*formula*. The formula is true if its value is 1.

Let Γ be $\{x_1 : \rho_1, \ldots, x_n : \rho_n\}$, and the *proof context* Δ a finite set of terms A_i such that $\Gamma \vdash A_i : o$.

Definition 3.3.10 We define

- $\mathscr{M}, \eta \models_\Gamma A := [\![A]\!]^\Gamma \eta = 1$
- $\Delta \models_\Gamma A := \forall \mathscr{M}, \eta\,((\mathscr{M}, \eta \models \Gamma \wedge \mathscr{M}, \eta \models_\Gamma \Delta) \Rightarrow \mathscr{M}, \eta \models_\Gamma A)$

We can omit the model $\mathscr{M}$ if it is clear from the context.

Lemma 3.3.11 If $\Gamma \vdash M : \rho$, $\eta \models \Gamma$, and $M \equiv_\alpha M'$, then $[\![M]\!]^\Gamma \eta = [\![M']\!]^\Gamma \eta$.

Proof. By induction on M. □

Semantically, the significance of the free variables of a term is that they delineate the only part of the variable substitution on which the value of the term depends. This is captured by the following lemma.

Lemma 3.3.12 (Coincidence) If $\Gamma \vdash M : \rho$, $\eta \models \Gamma$, $\eta' \models \Gamma'$, and $\forall x \in \mathrm{FV}(M)$, $\eta(x) = \eta'(x) \wedge \Gamma(x) = \Gamma'(x)$, then $[\![M]\!]^\Gamma \eta = [\![M]\!]^{\Gamma'} \eta'$.

Proof. By induction on the derivation of $\Gamma \vdash M : \rho$.

- $\Gamma, x : \rho \vdash x : \rho$

  We need to show $[\![x]\!]^{\Gamma}\eta = [\![x]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 it is to show $\eta(x) = \eta'(x)$. This follows by the assumption.

- $\Gamma \vdash c : \rho \quad \text{if } c : \rho$

  We need to show $[\![c]\!]^{\Gamma}\eta = [\![c]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 both left- and right-hand side of above equation are equal to $[\![c]\!]$.

- $$\frac{\Gamma, x : \rho \vdash M : \sigma}{\Gamma \vdash \lambda x : \rho . M : \rho \to \sigma}$$

  We need to show $[\![\lambda x : \rho . M]\!]^{\Gamma}\eta = [\![\lambda x : \rho . M]\!]^{\Gamma'}\eta'$.

  It is to show $\lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma, x:\rho}\eta[x := a] = \lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma', x:\rho}\eta'[x := a]$ by Definition 3.3.9.

  We have the following induction hypothesis

  $$[\![M]\!]^{\Gamma, x:\rho}\eta_0 = [\![M]\!]^{\Gamma', x:\rho}\eta_0' \tag{IH1}$$

  Let $\eta_0 := \eta[x := a]$ and $\eta_0' := \eta'[x := a]$.

  Before applying IH1, we need to show 1) $\eta[x := a] \models (\Gamma, x : \rho)$, 2) $\eta'[x := a] \models (\Gamma', x : \rho)$, and 3) $\forall y \in \mathrm{FV}(M)$, $\eta[x := a](y) = \eta'[x := a](y) \wedge (\Gamma, x : \rho)(y) = (\Gamma', x : \rho)(y)$. Let $a \in [\![\rho]\!]$.

  For 1), it is to show $\forall i. \eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$ where $x_i \in \mathrm{dom}(\Gamma, x : \rho)$. If $x_i \in \mathrm{dom}(\Gamma)$, then $\eta[x := a](x_i) = \eta(x_i) \in [\![\Gamma(x_i)]\!]$ by the assumption $\eta \models \Gamma$. If $x_i \equiv x$, then $\eta[x := a](x) = a \in [\![\rho]\!]$. Thus, $\eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$.

  For 2), similar to the proof of 1).

  For 3), clearly $x \in \mathrm{FV}(M)$. Hence if $y \equiv x$, then we have $\eta[x := a](x) = a = \eta'[x := a](x)$ and $(\Gamma, x : \rho)(x) = \rho = (\Gamma', x : \rho)(x)$. If $y \neq x$, then by assumptions we have $\eta[x := a](y) = \eta(y) = \eta'(y) = \eta'[x := a](y)$ and $(\Gamma, x : \rho)(y) = \Gamma(y) = \Gamma'(y) = (\Gamma', x : \rho)(y)$.

  By IH1, we have $\lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma, x:\rho}\eta[x := a] = \lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma', x:\rho}\eta'[x := a]$.

- $$\frac{\Gamma \vdash M : \rho \to \sigma \qquad \Gamma \vdash N : \rho}{\Gamma \vdash MN : \sigma}$$

  We need to show $[\![MN]\!]^{\Gamma}\eta = [\![MN]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 it is to show $[\![M]\!]^{\Gamma}\eta([\![N]\!]^{\Gamma}\eta) = [\![M]\!]^{\Gamma'}\eta'([\![N]\!]^{\Gamma'}\eta')$.

This follows by the induction hypotheses

$$[\![M]\!]^{\Gamma}\eta = [\![M]\!]^{\Gamma'}\eta' \tag{IH1}$$
$$[\![N]\!]^{\Gamma}\eta = [\![N]\!]^{\Gamma'}\eta' \tag{IH2}$$

- $\dfrac{\Gamma \vdash M : \rho \qquad \Gamma \vdash N : \sigma}{\Gamma \vdash \langle M,N\rangle : \rho \times \sigma}$

  We need to show $[\![\langle M,N\rangle]\!]^{\Gamma}\eta = [\![\langle M,N\rangle]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 it is to show $\langle [\![M]\!]^{\Gamma}\eta, [\![N]\!]^{\Gamma}\eta\rangle = \langle [\![M]\!]^{\Gamma'}\eta', [\![N]\!]^{\Gamma'}\eta'\rangle$.

  This follows by the induction hypotheses

$$[\![M]\!]^{\Gamma}\eta = [\![M]\!]^{\Gamma'}\eta' \tag{IH1}$$
$$[\![N]\!]^{\Gamma}\eta = [\![N]\!]^{\Gamma'}\eta' \tag{IH2}$$

- $\dfrac{\Gamma \vdash M : \rho \times \sigma}{\Gamma \vdash \pi_0(M) : \rho}$

  We need to show $[\![\pi_0(M)]\!]^{\Gamma}\eta = [\![\pi_0(M)]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 it is to show $\pi_0([\![M]\!]^{\Gamma}\eta) = \pi_0([\![M]\!]^{\Gamma'}\eta')$.

  This follows by the induction hypothesis

$$[\![M]\!]^{\Gamma}\eta = [\![M]\!]^{\Gamma'}\eta' \tag{IH1}$$

- $\dfrac{\Gamma \vdash M : \rho \times \sigma}{\Gamma \vdash \pi_1(M) : \sigma}$

  We need to show $[\![\pi_1(M)]\!]^{\Gamma}\eta = [\![\pi_1(M)]\!]^{\Gamma'}\eta'$.

  By Definition 3.3.9 it is to show $\pi_1([\![M]\!]^{\Gamma}\eta) = \pi_1([\![M]\!]^{\Gamma'}\eta')$.

  This follows by the induction hypothesis

$$[\![M]\!]^{\Gamma}\eta = [\![M]\!]^{\Gamma'}\eta' \tag{IH1}$$

□

We write $\Gamma[x : \rho]$ for extending the context Γ by $x : \rho$, and if $x \in \mathrm{dom}(\Gamma)$, replace the type of x in Γ by ρ.

The next two lemmas show that if two terms are β-equivalent, then the values of these two terms are identical.

Lemma 3.3.13 If $\Gamma[x:\sigma] \vdash M:\rho$, $\Gamma \vdash N:\sigma$, and $\eta \models \Gamma$, then $[\![M[N/x]]\!]^{\Gamma}\eta = [\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

Proof. By induction on M.

- $M \equiv x$.

  We need to show $[\![x[N/x]]\!]^{\Gamma}\eta = [\![x]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  From the left-hand side of the equation we get $[\![x[N/x]]\!]^{\Gamma}\eta = [\![N]\!]^{\Gamma}\eta$ by Definition 3.1.7.

  From the right-hand side we get $[\![x]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] = \eta[x := [\![N]\!]^{\Gamma}\eta](x) = [\![N]\!]^{\Gamma}\eta$ by Definition 3.3.9.

  Hence, we have $[\![N]\!]^{\Gamma}\eta = [\![N]\!]^{\Gamma}\eta$.

- $M \equiv y$ where $x \neq y$.

  We need to show $[\![y[N/x]]\!]^{\Gamma}\eta = [\![y]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  By Definition 3.1.7 and 3.3.9, it is to show $\eta(y) = \eta(y)$. This follows by equality.

- $M \equiv c$.

  We need to show $[\![c[N/x]]\!]^{\Gamma}\eta = [\![c]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  By Definition 3.1.7 and 3.3.9, both left- and right-hand side of above equation are equal to $[\![c]\!]^{\Gamma}\eta$.

- $M \equiv \lambda z:\rho.M'$.

  We can always find a term $\lambda y:\rho.M$ such that $\lambda y:\rho.M =_{\alpha} \lambda z:\rho.M'$ where $x \neq y$ and $y \notin \mathrm{FV}(N)$.

  Then to show $[\![(\lambda y:\rho.M)[N/x]]\!]^{\Gamma}\eta = [\![\lambda y:\rho.M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  We get $[\![(\lambda y:\rho.M)[N/x]]\!]^{\Gamma}\eta = \lambda a \in [\![\rho]\!].[\![M[N/x]]\!]^{\Gamma,y:\rho}\eta[y := a]$ by Definition 3.1.7 and 3.3.9 from the left-hand side of the equation.

  From the right-hand side, by Definition 3.3.9 we get $[\![\lambda y:\rho.M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] = \lambda a \in [\![\rho]\!].[\![M]\!]^{\Gamma[x:\sigma],y:\rho}\eta[x := [\![N]\!]^{\Gamma}\eta][y := a]$.

  By Lemma 3.3.12, we get $[\![N]\!]^{\Gamma}\eta = [\![N]\!]^{\Gamma,y:\rho}\eta[y := a]$.

  Now it is to show $[\![M[N/x]]\!]^{\Gamma,y:\rho}\eta[y := a] = [\![M]\!]^{\Gamma[x:\sigma],y:\rho}\eta[x := [\![N]\!]^{\Gamma,y:\rho}\eta[y := a]][y := a]$.

  We have the following induction hypothesis

$$[\![M[N/x]]\!]^{\Gamma_0}\eta_0 = [\![M]\!]^{\Gamma_0[x:\sigma]}\eta_0[x := [\![N]\!]^{\Gamma_0}\eta_0] \qquad \text{(IH1)}$$

Let $\Gamma_0 := \Gamma, y:\rho$ and $\eta_0 := \eta[y := a]$.

By IH1, we have $[\![M[N/x]]\!]^{\Gamma,y:\rho}\eta[y := a] = [\![M]\!]^{\Gamma[x:\sigma],y:\rho}\eta[x := [\![N]\!]^{\Gamma,y:\rho}\eta[y := a]][y := a]$.

- $M \equiv M_1M_2$.

  We need to show $[\![(M_1M_2)[N/x]]\!]^{\Gamma}\eta = [\![M_1M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.3.9, we get $[\![(M_1M_2)[N/x]]\!]^{\Gamma}\eta = [\![M_1[N/x](M_2[N/x])]\!]^{\Gamma}\eta = [\![M_1[N/x]]\!]^{\Gamma}\eta([\![M_2[N/x]]\!]^{\Gamma}\eta)$.

  We get $[\![M_1M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] = [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]([\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta])$ from the right-hand side, by Definition 3.3.9.

  Then it is to show that $[\![M_1[N/x]]\!]^{\Gamma}\eta([\![M_2[N/x]]\!]^{\Gamma}\eta) = [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]([\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta])$.

  This follows by the induction hypotheses

$$[\![M_1[N/x]]\!]^{\Gamma}\eta = [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] \quad \text{(IH1)}$$
$$[\![M_2[N/x]]\!]^{\Gamma}\eta = [\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] \quad \text{(IH2)}$$

- $M \equiv \langle M_1, M_2\rangle$.

  We need to show $[\![\langle M_1,M_2\rangle[N/x]]\!]^{\Gamma}\eta = [\![\langle M_1,M_2\rangle]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  From the left-hand side of the equation we get $[\![\langle M_1,M_2\rangle[N/x]]\!]^{\Gamma}\eta = [\![\langle M_1[N/x], M_2[N/x]\rangle]\!]^{\Gamma}\eta = \langle [\![M_1[N/x]]\!]^{\Gamma}\eta, [\![M_2[N/x]]\!]^{\Gamma}\eta\rangle$ by Definition 3.1.7 and 3.3.9.

  From the right-hand side, by Definition 3.3.9, we get $[\![\langle M_1,M_2\rangle]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] = \langle [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta], [\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]\rangle$.

  Now it is to show $\langle [\![M_1[N/x]]\!]^{\Gamma}\eta, [\![M_2[N/x]]\!]^{\Gamma}\eta\rangle = \langle [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta], [\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]\rangle$.

  This follows by the induction hypotheses

$$[\![M_1[N/x]]\!]^{\Gamma}\eta = [\![M_1]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] \quad \text{(IH1)}$$
$$[\![M_2[N/x]]\!]^{\Gamma}\eta = [\![M_2]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] \quad \text{(IH2)}$$

- $M \equiv \pi_0(M)$.

  We need to show $[\![\pi_0(M)[N/x]]\!]^{\Gamma}\eta = [\![\pi_0(M)]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta]$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.3.9, we get $[\![\pi_0(M)[N/x]]\!]^{\Gamma}\eta = [\![\pi_0(M[N/x])]\!]^{\Gamma}\eta = \pi_0([\![M[N/x]]\!]^{\Gamma}\eta)$.

  From the right-hand side we get $[\![\pi_0(M)]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta] = \pi_0([\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^{\Gamma}\eta])$ by Definition 3.3.9.

We need to show $\pi_0([\![M[N/x]]\!]^\Gamma\eta) = \pi_0([\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta])$.

This follows by the induction hypothesis

$$[\![M[N/x]]\!]^\Gamma\eta = [\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta] \qquad \text{(IH1)}$$

- $M \equiv \pi_1(M)$.

  We need to show $[\![\pi_1(M)[N/x]]\!]^\Gamma\eta = [\![\pi_1(M)]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta]$.

  From the left-hand side of the equation, by Definition 3.1.7 and 3.3.9, we get $[\![\pi_1(M)[N/x]]\!]^\Gamma\eta = [\![\pi_1(M[N/x])]\!]^\Gamma\eta = \pi_1([\![M[N/x]]\!]^\Gamma\eta)$.

  From the right-hand side we get $[\![\pi_1(M)]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta] = \pi_1([\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta])$ by Definition 3.3.9.

  We need to show $\pi_1([\![M[N/x]]\!]^\Gamma\eta) = \pi_1([\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta])$.

  This follows by the induction hypothesis

$$[\![M[N/x]]\!]^\Gamma\eta = [\![M]\!]^{\Gamma[x:\sigma]}\eta[x := [\![N]\!]^\Gamma\eta] \qquad \text{(IH1)}$$

□

Lemma 3.3.14 If $\Gamma \vdash M,N : \rho$, then $M \equiv_\beta N \Rightarrow [\![M]\!]^\Gamma\eta = [\![N]\!]^\Gamma\eta$.

Proof. By induction on the definition of β-equivalence. We only look at the interesting case, namely β-conversion.

Let M be $(\lambda x : \rho.T_1)T_2$, and N be $T_1[T_2/x]$, where T_2 has the type ρ.

Then it is to show $[\![(\lambda x : \rho.T_1)T_2]\!]^\Gamma\eta = [\![T_1[T_2/x]]\!]^\Gamma\eta$.

From the left-hand side of the equation by Definition 3.3.9 we get $[\![(\lambda x : \rho.T_1)T_2]\!]^\Gamma\eta = [\![\lambda x : \rho.T_1]\!]^\Gamma\eta([\![T_2]\!]^\Gamma\eta) = (\lambda a \in [\![\rho]\!].[\![T_1]\!]^{\Gamma,x:\rho}\eta[x := a])([\![T_2]\!]^\Gamma\eta) = [\![T_1]\!]^{\Gamma,x:\rho}\eta[x := [\![T_2]\!]^\Gamma\eta]$.

On the right-hand side, by Lemma 3.3.13 we have $[\![T_1[T_2/x]]\!]^\Gamma\eta = [\![T_1]\!]^{\Gamma[x:\rho]}\eta[x := [\![T_2]\!]^\Gamma\eta]$.

Thus, $[\![T_1]\!]^{\Gamma,x:\rho}\eta[x := [\![T_2]\!]^\Gamma\eta] = [\![T_1]\!]^{\Gamma[x:\rho]}\eta[x := [\![T_2]\!]^\Gamma\eta]$ since $\Gamma,x : \rho = \Gamma[x : \rho]$. □

Proof Calculus and Soundness

Denotational semantics is concerned with abstracting away from irrelevant details in order to focus on the aspects that are of interest to the user of the semantics. Therefore, it is natural to ensure a semantics is reasonably or sufficiently abstract for the need of the user. Soundness of a deductive system is the property that whenever a formula A of the language Γ upon which the deductive system is based is derivable from a set Δ

of formulas of that language, the formula is a consequence of that set. In other words, any model making all formulas in Δ true makes A true as well.

We start with constructing a deductive system. An intuitionistic proof calculus for CST, which is suitable for program extraction, is given in Table 3.1. It derives sequents of the form $\Delta \vdash_\Gamma A$, where Δ is a finite set of Γ-formulas and A is a Γ-formula, where by a Γ-*formula* we mean a term M such that $\Gamma \vdash M : o$.

Remarks. 1. In order to enable a realisability interpretation we had to deviate from Church's calculus in several aspects: first our system is intuitionistic while Church's is classical. Secondly, we dropped the choice operator since it does not appear to admit a realisability interpretation. There are other realisability interpretations where versions of choice are realisable, e.g. Krivine's countable choice in classical second-order logic [Kri03], Raffalli and Ruyer's choice axiom in higher-order logic [RR08], Oliva and Streicher study the connection between Krivine's work and modified realisability [OS08].

2. From a logical point of view, the constants $\to$ and $\forall_\rho$ would suffice to define all other constants. For the logical constants including equality this was already observed by Church [Chu40]. Furthermore, we could have defined equality, following Church, as

$$(x =_\rho y) \quad := \quad \forall p : \rho \to o.\, px \to py.$$

More formally

$$=_\rho \quad := \quad \lambda x, y : \rho. \forall p : \rho \to o.\, px \to py.$$

$\mu_\rho \Phi$ can be defined as the infimum of all $x : \rho$ such that $\Phi x \subseteq_\rho x$, and $\nu_\rho \Phi$ can be defined similarly. The reason why we prefer this richer set of constants is that they can be given simpler realisers as the ones extracted from the impredicative definitions above. For example, Tatsuta [Tat98] (in his theory of $\mathrm{TID}_{\nu 2}$) extracted the realiser from the impredicative definition of ν_ρ. However, the realiser of the coclosure rule for coinduction is simply the identity in our case, but Tatsuta has a more complicated λ-term $(\lambda \vec{x} r. r(\lambda p. m(\lambda \vec{x} r s. s(p_0, r))\vec{x}(p_0 \vec{x} p_1)))$ on page 353). More details will be given in the next chapter (Lemma 4.2.7).

3. In CST, the rules for μ_ρ and ν_ρ are restricted to monotone operators. In the next chapter, we will consider an extension of CST that has rules for $\mu_\rho \Phi$ and $\nu_\rho \Phi$ for arbitrary operators Φ.

4. In the following we will tacitly use the following derived rules that state that β-equality implies equality and equals can be replaced by equals in any context.

$$\frac{\Gamma \vdash A, B : \rho}{\Delta \vdash_\Gamma A =_\rho B}\, A \equiv_\beta B \qquad \frac{\Delta \vdash_\Gamma A =_\rho B \quad \Delta \vdash_\Gamma M[A/x]}{\Delta \vdash_\Gamma M[B/x]}$$

Table 3.1: Proof Calculus for CST

$$\frac{\Gamma \vdash \Delta, A : o}{\Delta, A \vdash_\Gamma A}\ \text{use} \qquad \frac{\Delta \vdash_\Gamma A \quad \Gamma \vdash B : o}{\Delta \vdash_\Gamma B}\ \beta \quad A \equiv_\beta B$$

$$\frac{\Delta \vdash_\Gamma A \quad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \wedge B}\ \wedge^+ \qquad \frac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma A}\ \wedge_l^- \qquad \frac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma B}\ \wedge_r^-$$

$$\frac{\Delta \vdash_\Gamma A \quad \Gamma \vdash B : o}{\Delta \vdash_\Gamma A \vee B}\ \vee_l^+ \qquad \frac{\Gamma \vdash A : o \quad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \vee B}\ \vee_r^+$$

$$\frac{\Delta \vdash_\Gamma A \vee B \quad \Delta, A \vdash_\Gamma C \quad \Delta, B \vdash_\Gamma C}{\Delta \vdash_\Gamma C}\ \vee^-$$

$$\frac{\Delta, A \vdash_\Gamma B}{\Delta \vdash_\Gamma A \to B}\ \to^+ \qquad \frac{\Delta \vdash_\Gamma A \to B \quad \Delta \vdash_\Gamma A}{\Delta \vdash_\Gamma B}\ \to^-$$

$$\frac{\Delta \vdash_{\Gamma, x:\rho} Mx}{\Delta \vdash_\Gamma \forall_\rho M}\ \forall_\rho^+\ x \text{ fresh} \qquad \frac{\Delta \vdash_\Gamma \forall_\rho M \quad \Gamma \vdash N : \rho}{\Delta \vdash_\Gamma MN}\ \forall_\rho^-$$

$$\frac{\Delta \vdash_\Gamma MN \quad \Gamma \vdash N : \rho}{\Delta \vdash_\Gamma \exists_\rho M}\ \exists_\rho^+ \qquad \frac{\Delta \vdash_\Gamma \exists_\rho M \quad \Delta, Mx \vdash_{\Gamma, x:\rho} C}{\Delta \vdash_\Gamma C}\ \exists_\rho^-\ x \text{ fresh}$$

$$\frac{\Delta, pA \vdash_{\Gamma, p:\rho\to o} pB}{\Delta \vdash_\Gamma A =_\rho B}\ =^+\ p \text{ fresh} \qquad \frac{\Delta \vdash_\Gamma A =_\rho B \quad \Delta \vdash_\Gamma PA}{\Delta \vdash_\Gamma PB}\ =^-$$

$$\frac{\Delta \vdash_{\Gamma, x:\rho} Ax =_\sigma Bx}{\Delta \vdash_\Gamma A =_{\rho\to\sigma} B}\ \text{ext}\quad x \text{ fresh}$$

$$\frac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi}\ \text{Cl}_\rho \qquad \frac{\Delta \vdash_\Gamma \text{mono}_\rho(\Phi) \quad \Delta \vdash_\Gamma \Phi(M) \subseteq_\rho M}{\Delta \vdash_\Gamma \mu_\rho \Phi \subseteq_\rho M}\ \text{Ind}_\rho$$

$$\frac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \nu_\rho \Phi \subseteq_\rho \Phi(\nu_\rho \Phi)}\ \text{Cocl}_\rho \qquad \frac{\Delta \vdash_\Gamma \text{mono}_\rho(\Phi) \quad \Delta \vdash_\Gamma M \subseteq_\rho \Phi(M)}{\Delta \vdash_\Gamma M \subseteq_\rho \nu_\rho(\Phi)}\ \text{Coi}_\rho$$

Now we give two examples to demonstrate inductive and coinductive definitions. Later in the next chapter we will give the realisability interpretation of these two examples.

Example 3.3.15 (Inductive Natural Numbers)

Consider the typing context $\Gamma_+ := \{0,1 : \iota, + : \iota \to \iota \to \iota, - : \iota \to \iota\}$ and let Δ_+ consist of the formulas stating that $(+,-,0)$ is an Abelian group. Formally, the terms $0,1,+,-$ are variables, but we like to view them as constants and the assumptions Δ_+ as axioms. We will write $M-N$ as a shorthand for $M+(-N)$.

We define the set of natural numbers as an inductive predicate. Set $\Phi := \lambda p : \iota \to o.\lambda x : \iota.x =_\iota 0 \vee p(x-1)$, and define $\mathbb{N} := \mu_{\iota\to o}\Phi$. A more readable notation for the definition of $\mathbb{N}$ would be

$$\mathbb{N}x \overset{\mu}{=} x =_\iota 0 \vee \mathbb{N}(x-1).$$

Example 3.3.16 (Coinductive Fibonacci Numbers)

Continuing the previous example, we give a definition of a coinductive predicate $\mathrm{FIB} := \nu_{\iota\to\iota\to o}\Psi$ where

$$\Psi := \lambda q : \iota \to \iota \to o.\lambda x,y : \iota.\mathbb{N}x \wedge qy(x+y).$$

This becomes more readable if a similar notation as for $\mathbb{N}$ is used:

$$\mathrm{FIB}xy \overset{\nu}{=} \mathbb{N}x \wedge \mathrm{FIB}y(x+y)$$

Informally, $\mathrm{FIB}xy$ states that there exists a Fibonacci sequence of natural numbers starting with x,y.

Lemma 3.3.17 If $\Delta \vdash_\Gamma A$, then $\Gamma \vdash A : o$ and $\forall B \in \Delta(\Gamma \vdash B : o)$.

Proof. By induction on the structure of the relation $\Delta \vdash_\Gamma A$.

- $\dfrac{\Gamma \vdash \Delta, A : o}{\Delta, A \vdash_\Gamma A}$ use

  We have $\Gamma \vdash \Delta, A : o$, which means $\Gamma \vdash A : o \wedge \forall B \in (\Delta, A)(\Gamma \vdash B : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma B}\ \beta \quad A \equiv_\beta B$

  We need to show $\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  Since we have $\Gamma \vdash B : o$, this follows by the induction hypothesis

  $$\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \wedge B}\ \wedge^+$

  We need to show $\Gamma \vdash A \wedge B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$
  $$\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH2)}$$

By the uniqueness of derivations, from $\Gamma \vdash A : o$ and $\Gamma \vdash B : o$, we have $\Gamma \vdash A \wedge B : o$. Then by IH1, we have $\Gamma \vdash A \wedge B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma A} \wedge_l^-$

  We need to show $\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A \wedge B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

  By the uniqueness of derivations, from $\Gamma \vdash A \wedge B : o$, we have $\Gamma \vdash A : o$. Then by IH1, we have $\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma B} \wedge_r^-$

  We need to show $\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A \wedge B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

  By the uniqueness of derivations, from $\Gamma \vdash A \wedge B : o$, we have $\Gamma \vdash B : o$. Then by IH1, we have $\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma A \vee B} \vee_l^+$

  We need to show $\Gamma \vdash A \vee B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

  By the uniqueness of derivations, from $\Gamma \vdash A : o$, we have $\Gamma \vdash A \vee B : o$. Then by IH1, we have $\Gamma \vdash A \vee B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Gamma \vdash A : o \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \vee B} \vee_r^+$

  We need to show $\Gamma \vdash A \vee B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

  By the uniqueness of derivations, from $\Gamma \vdash B : o$, we have $\Gamma \vdash A \vee B : o$. Then by IH1, we have $\Gamma \vdash A \vee B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \vee B \qquad \Delta, A \vdash_\Gamma C \qquad \Delta, B \vdash_\Gamma C}{\Delta \vdash_\Gamma C} \vee^-$

  We need to show $\Gamma \vdash C : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A \vee B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \quad \text{(IH1)}$$
  $$\Gamma \vdash C : o \wedge \forall B' \in (\Delta, A)(\Gamma \vdash B' : o) \quad \text{(IH2)}$$
  $$\Gamma \vdash C : o \wedge \forall B' \in (\Delta, B)(\Gamma \vdash B' : o) \quad \text{(IH3)}$$

  By the weakening rule, it is safe to apply IH2 and IH3. Then by IH2, we have $\Gamma \vdash C : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta, A \vdash_\Gamma B}{\Delta \vdash_\Gamma A \to B} \to^+$

  We need to show $\Gamma \vdash A \to B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash B : o \wedge \forall B' \in (\Delta, A)(\Gamma \vdash B' : o) \quad \text{(IH1)}$$

  Now we need to show $\Gamma \vdash A \to B : o$.

  By the uniqueness of derivations, it is to show $\Gamma, A \vdash B : o$. It follows by the weakening rule and IH1. By IH1, we have $\forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A \to B \qquad \Delta \vdash_\Gamma A}{\Delta \vdash_\Gamma B} \to^-$

  We need to show $\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma \vdash A \to B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \quad \text{(IH1)}$$
  $$\Gamma \vdash A : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \quad \text{(IH2)}$$

  By the uniqueness of derivations, from $\Gamma \vdash A \to B : o$ and $\Gamma \vdash A : o$, we have $\Gamma \vdash B : o$. Then by IH1, we have $\Gamma \vdash B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_{\Gamma, x:\rho} Mx}{\Delta \vdash_\Gamma \forall_\rho M} \forall_\rho^+$

  We need to show $\Gamma \vdash \forall_\rho M : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

  We have the following induction hypothesis

  $$\Gamma, x : \rho \vdash Mx : o \wedge \forall B' \in \Delta(\Gamma, x : \rho \vdash B' : o) \quad \text{(IH1)}$$

From $\forall_\rho M$ we know there exists some x that has the type ρ. Thus from Γ we can derive $\Gamma, x:\rho$. So it is safe to apply IH1. By the uniqueness of derivations and the weakening rule, from $\Gamma, x:\rho \vdash Mx : o$, we have $\Gamma \vdash \forall_\rho M : o$.

Since the x in $(\Gamma, x:\rho)$ is generated from the application with M, it has no effects on Δ. Then by IH1, from $\forall B' \in \Delta(\Gamma, x:\rho \vdash B' : o)$, we have $\forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma \forall_\rho M \qquad \Gamma \vdash N : \rho}{\Delta \vdash_\Gamma MN}\ \forall_\rho^-$

We need to show $\Gamma \vdash MN : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma \vdash \forall_\rho M : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

By the uniqueness of derivations, from $\Gamma \vdash \forall_\rho M : o$ and $\Gamma \vdash N : \rho$, we have $\Gamma \vdash MN : o$. Then by IH1, we have $\Gamma \vdash MN : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma MN \qquad \Gamma \vdash N : \rho}{\Delta \vdash_\Gamma \exists_\rho M}\ \exists_\rho^+$

We need to show $\Gamma \vdash \exists_\rho M : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma \vdash MN : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$

By the uniqueness of derivations, from $\Gamma \vdash MN : o$ and $\Gamma \vdash N : \rho$, we have $\Gamma \vdash \forall_\rho M : o$. Then by IH1, we have $\Gamma \vdash \exists_\rho M : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma \exists_\rho M \qquad \Delta, Mx \vdash_{\Gamma, x:\rho} C}{\Delta \vdash_\Gamma C}\ \exists_\rho^-$

We need to show $\Gamma \vdash C : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma \vdash \exists_\rho M : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$
$$\Gamma, x:\rho \vdash C : o \wedge \forall B' \in (\Delta, Mx)(\Gamma, x:\rho \vdash B' : o) \qquad \text{(IH2)}$$

By the weakening rule it is safe to apply IH2. By the uniqueness of derivations, from $\Gamma \vdash \exists_\rho M : o$ and $\Gamma, x:\rho \vdash C : o$, we have $\Gamma \vdash C : o$. Then by IH1, we have $\Gamma \vdash C : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta, pA \vdash_{\Gamma, p:\rho\to o} pB}{\Delta \vdash_\Gamma A =_\rho B}\ =^+ \quad p \text{ fresh}$

We need to show $\Gamma \vdash A =_\rho B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma, p : \rho \to o \vdash pB : o \wedge \forall B' \in (\Delta, pA)(\Gamma, p : \rho \to o \vdash B' : o) \qquad \text{(IH1)}$$

By the weakening rule it is safe to apply IH1. By the uniqueness of derivations, from $\Gamma, p : \rho \to o \vdash pB : o$, we have $\Gamma \vdash A =_\rho B : o$. Since p has no effects on Δ, by IH1, we have $\forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_\Gamma A =_\rho B \qquad \Delta \vdash_\Gamma PA}{\Delta \vdash_\Gamma PB} =^-$

We need to show $\Gamma \vdash PB : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma \vdash A =_\rho B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH1)}$$
$$\Gamma \vdash PA : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o) \qquad \text{(IH2)}$$

By the uniqueness of derivations, from $\Gamma \vdash A =_\rho B : o$ and $\Gamma \vdash PA : o$, we have $\Gamma \vdash PB : o$. Then by IH1, we have $\Gamma \vdash PB : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Delta \vdash_{\Gamma, x:\rho} Ax =_\sigma Bx}{\Delta \vdash_\Gamma A =_{\rho\to\sigma} B}$ ext x fresh

We need to show $\Gamma \vdash A =_{\rho\to\sigma} B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

We have the following induction hypothesis

$$\Gamma, x : \rho \vdash Ax =_\sigma Bx : o \wedge \forall B' \in \Delta(\Gamma, x : \rho \vdash B' : o) \qquad \text{(IH1)}$$

Since x is generated for applications with A and B, it has no effects on Δ. Thus it is safe to apply IH1. By the uniqueness of derivations, from $\Gamma, x : \rho \vdash Ax =_\sigma Bx : o$, we have $\Gamma \vdash A =_{\rho\to\sigma} B : o$.

Then by IH1, we have $\Gamma \vdash A =_{\rho\to\sigma} B : o \wedge \forall B' \in \Delta(\Gamma \vdash B' : o)$.

- $\dfrac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi}$ Cl_ρ

We need to show $\Gamma \vdash \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

This follows by the assumptions and the definitions of μ_ρ and $\subseteq_\rho$.

- $\dfrac{\Delta \vdash_\Gamma \mathrm{mono}_\rho(\Phi) \quad \Delta \vdash_\Gamma \Phi(M) \subseteq_\rho M}{\Delta \vdash_\Gamma \mu_\rho \Phi \subseteq_\rho M}$ Ind_ρ

We need to show $\Gamma \vdash \mu_\rho \Phi \subseteq_\rho M : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

We have the following induction hypothesis

$$\Gamma \vdash \mathrm{mono}_\rho(\Phi) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o) \quad \text{(IH1)}$$
$$\Gamma \vdash \Phi(M) \subseteq_\rho M : o \wedge \forall B \in \Delta(\Gamma \vdash B : o) \quad \text{(IH2)}$$

By the uniqueness of derivations, we have $\Gamma \vdash \mu_\rho \Phi \subseteq_\rho M : o$. Then by IH1, we have $\Gamma \vdash \mu_\rho \Phi \subseteq_\rho M : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

- $\dfrac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \nu_\rho \Phi \subseteq_\rho \Phi(\nu_\rho \Phi)}\ \mathrm{Cocl}_\rho$

  We need to show $\Gamma \vdash \nu_\rho \Phi \subseteq_\rho \Phi(\nu_\rho \Phi) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

  This follows by the assumptions and the definitions of ν_ρ and $\subseteq_\rho$.

- $\dfrac{\Delta \vdash_\Gamma \mathrm{mono}_\rho(\Phi) \qquad \Delta \vdash_\Gamma M \subseteq_\rho \Phi(M)}{\Delta \vdash_\Gamma M \subseteq_\rho \nu_\rho(\Phi)}\ \mathrm{Coi}_\rho$

  We need to show $\Gamma \vdash M \subseteq_\rho \nu_\rho(\Phi) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

  We have the following induction hypothesis

$$\Gamma \vdash \mathrm{mono}_\rho(\Phi) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o) \quad \text{(IH1)}$$
$$\Gamma \vdash M \subseteq_\rho \Phi(M) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o) \quad \text{(IH2)}$$

  By the uniqueness of derivations, we have $\Gamma \vdash M \subseteq_\rho \nu_\rho(\Phi) : o$. Then by IH1, we have $\Gamma \vdash M \subseteq_\rho \nu_\rho(\Phi) : o \wedge \forall B \in \Delta(\Gamma \vdash B : o)$.

□

Theorem 3.3.18 (Soundness) If $\Delta \vdash_\Gamma A$, then $\Delta \models_\Gamma A$.

Proof. By induction on the structure of the relation $\Delta \vdash_\Gamma A$.

- $\dfrac{\Gamma \vdash \Delta, A : o}{\Delta, A \vdash_\Gamma A}\ \mathrm{use}$

  We need to show $\Delta, A \models_\Gamma A$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma (\Delta, A)$, to show $\mathscr{M}, \eta \models_\Gamma A$.

  From the assumption $\mathscr{M}, \eta \models_\Gamma (\Delta, A)$ we get $\mathscr{M}, \eta \models_\Gamma A$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma B}\ \beta \quad A \equiv_\beta B$

  We need to show $\Delta \models_\Gamma B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma B$, i.e. to show $[\![B]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \tag{IH1}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma A$.

That is, $[\![A]\!]^\Gamma \eta = 1$. By Lemma 3.3.14 we get $[\![B]\!]^\Gamma \eta = 1$ since $A \equiv_\beta B$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \wedge B} \wedge^+$

We need to show $\Delta \models_\Gamma A \wedge B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A \wedge B$, i.e. to show $[\![A \wedge B]\!]^\Gamma \eta = 1$. That is to show $[\![A]\!]^\Gamma \eta = 1$ and $[\![B]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \tag{IH1}$$
$$\Delta \models_\Gamma B \tag{IH2}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma A$ and $\mathscr{M}, \eta \models_\Gamma B$.

That is, $[\![A]\!]^\Gamma \eta = 1$ and $[\![B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma A} \wedge_l^-$

We need to show $\Delta \models_\Gamma A$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A$, i.e. to show $[\![A]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \wedge B \tag{IH1}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma A \wedge B$.

That is, $[\![A \wedge B]\!]^\Gamma \eta = 1$, i.e. $[\![A]\!]^\Gamma \eta = 1$ and $[\![B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma B} \wedge_r^-$

We need to show $\Delta \models_\Gamma B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma B$, i.e. to show $[\![B]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \wedge B \tag{IH1}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma A \wedge B$.

That is, $[\![A \wedge B]\!]^\Gamma \eta = 1$, i.e. $[\![A]\!]^\Gamma \eta = 1$ and $[\![B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma A \vee B} \vee_l^+$

We need to show $\Delta \models_\Gamma A \vee B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A \vee B$, i.e. to show $[\![A \vee B]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \tag{IH1}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma A$.

That is, $[\![A]\!]^\Gamma \eta = 1$. Thus, $[\![A \vee B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Gamma \vdash A : o \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \vee B} \vee_r^+$

We need to show $\Delta \models_\Gamma A \vee B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A \vee B$, i.e. to show $[\![A \vee B]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma B \tag{IH1}$$

By Definition 3.3.10 we have $\mathscr{M}, \eta \models_\Gamma B$.

That is, $[\![B]\!]^\Gamma \eta = 1$. Thus, $[\![A \vee B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_\Gamma A \vee B \qquad \Delta, A \vdash_\Gamma C \qquad \Delta, B \vdash_\Gamma C}{\Delta \vdash_\Gamma C} \vee^-$

We need to show $\Delta \models_\Gamma C$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma C$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A \vee B \tag{IH1}$$
$$\Delta, A \models_\Gamma C \tag{IH2}$$
$$\Delta, B \models_\Gamma C \tag{IH3}$$

By Definition 3.3.10 we have

$$(\eta \models \Gamma \wedge \mathscr{M}, \eta \models_\Gamma \Delta) \Rightarrow \mathscr{M}, \eta \models_\Gamma A \vee B$$
$$(\eta \models \Gamma \wedge \mathscr{M}, \eta \models_\Gamma (\Delta, A)) \Rightarrow \mathscr{M}, \eta \models_\Gamma C$$
$$(\eta \models \Gamma \wedge \mathscr{M}, \eta \models_\Gamma (\Delta, B)) \Rightarrow \mathscr{M}, \eta \models_\Gamma C$$

Before we apply IH2 or IH3, we need to show $\mathscr{M}, \eta \models_\Gamma (\Delta, A)$ or $\mathscr{M}, \eta \models_\Gamma (\Delta, B)$. That is to show $\mathscr{M}, \eta \models_\Gamma A$ or $\mathscr{M}, \eta \models_\Gamma B$, i.e. $[\![A]\!]^\Gamma \eta = 1$ or $[\![B]\!]^\Gamma \eta = 1$. Then to show $[\![A \vee B]\!]^\Gamma \eta = 1$. This follows by $\mathscr{M}, \eta \models_\Gamma A \vee B$.

By IH2, we have $\mathscr{M}, \eta \models_\Gamma C$.

- $\dfrac{\Delta, A \vdash_\Gamma B}{\Delta \vdash_\Gamma A \to B} \to^+$

  We need to show $\Delta \models_\Gamma A \to B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A \to B$, i.e. to show $[\![A \to B]\!]^\Gamma \eta = 1$. That is, assume $[\![A]\!]^\Gamma \eta = 1$, to show $[\![B]\!]^\Gamma \eta = 1$.

  We have the following induction hypothesis

  $$\Delta, A \models_\Gamma B \qquad \text{(IH1)}$$

  Before applying IH1, we need to show $\mathscr{M}, \eta \models_\Gamma (\Delta, A)$ first. We have $\mathscr{M}, \eta \models_\Gamma \Delta$ by assumption, then it is to show $\mathscr{M}, \eta \models_\Gamma A$, i.e. $[\![A]\!]^\Gamma \eta = 1$. This follows by the assumption.

  By IH1, by Definition 3.3.10 we have $(\eta \models \Gamma \wedge \mathscr{M}, \eta \models_\Gamma (\Delta, A)) \Rightarrow \mathscr{M}, \eta \models_\Gamma B$. That is, $[\![B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_\Gamma A \to B \qquad \Delta \vdash_\Gamma A}{\Delta \vdash_\Gamma B} \to^-$

  We need to show $\Delta \models_\Gamma B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma B$, i.e. to show $[\![B]\!]^\Gamma \eta = 1$.

  We have the following induction hypothesis

  $$\Delta \models_\Gamma A \to B \qquad \text{(IH1)}$$
  $$\Delta \models_\Gamma A \qquad \text{(IH2)}$$

  By Definition 3.3.10, from IH1, we have $\mathscr{M}, \eta \models_\Gamma A \to B$. That is, $[\![A \to B]\!]^\Gamma \eta = 1$, i.e. if $[\![A]\!]^\Gamma \eta = 1$, then $[\![B]\!]^\Gamma \eta = 1$. $[\![A]\!]^\Gamma \eta = 1$ holds by IH2. Thus, $[\![B]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta \vdash_{\Gamma, x:\rho} Mx}{\Delta \vdash_\Gamma \forall_\rho M} \forall_\rho^+ \quad x \text{ fresh}$

  We need to show $\Delta \models_\Gamma \forall_\rho M$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma \forall_\rho M$, i.e. to show $[\![\forall_\rho M]\!]^\Gamma \eta = 1$. That is, $[\![\forall_\rho]\!][\![M]\!]^\Gamma \eta = 1$, i.e. to show $\forall a \in [\![\rho]\!].[\![M]\!]^\Gamma \eta(a) = 1$.

  We have the following induction hypothesis

  $$\Delta \models_{\Gamma, x:\rho} Mx \qquad \text{(IH1)}$$

  Before applying IH1, we need to show a) $\eta[x := a] \models (\Gamma, x : \rho)$ and b) $\mathscr{M}, \eta[x := a] \models_{\Gamma, x:\rho} \Delta$ first. Let $a \in [\![\rho]\!]$.

  For a), it is to show $\forall i. \eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$ where $x_i \in \mathrm{dom}(\Gamma, x : \rho)$. If $x_i \in \mathrm{dom}(\Gamma)$, then $\eta[x := a](x_i) = \eta(x_i) \in [\![\Gamma(x_i)]\!]$ by $\eta \models \Gamma$. If $x_i \equiv x$, then $\eta[x := a](x) = a \in [\![\rho]\!]$. Thus, $\eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$.

For b), it is to show $\forall A \in \Delta(\mathscr{M}, \eta[x := a] \models_{\Gamma,x:\rho} A)$, i.e. $[\![A]\!]^{\Gamma,x:\rho}\eta[x := a] = 1$. By Lemma 3.3.13, it is to show $[\![A[A'/x]]\!]^{\Gamma}\eta = 1$ where $[\![A']\!]^{\Gamma}\eta = a$. Since x is fresh, it is to show $[\![A]\!]^{\Gamma}\eta = 1$ for all $A \in \Delta$. This follows by the assumption $\mathscr{M}, \eta \models_{\Gamma} \Delta$.

From IH1, we have $(\eta[x := a] \models (\Gamma, x:\rho) \wedge \mathscr{M}, \eta[x := a] \models_{\Gamma,x:\rho} \Delta) \Rightarrow \mathscr{M}, \eta[x := a] \models_{\Gamma,x:\rho} Mx$ by Definition 3.3.10. That is, we have

$$
\begin{aligned}
& [\![Mx]\!]^{\Gamma,x:\rho}\eta[x := a] = 1 \\
\Rightarrow & [\![M]\!]^{\Gamma,x:\rho}\eta[x := a]([\![x]\!]^{\Gamma,x:\rho}\eta[x := a]) = 1 && \text{(by Definition 3.3.9)} \\
\Rightarrow & [\![M]\!]^{\Gamma,x:\rho}\eta[x := a](a) = 1 && \text{(by Definition 3.3.9)} \\
\Rightarrow & [\![M[A'/x]]\!]^{\Gamma}\eta(a) = 1 && \text{(by Lemma 3.3.13)} \\
\Rightarrow & [\![M]\!]^{\Gamma}\eta(a) = 1 && \text{(by } x \text{ fresh)}
\end{aligned}
$$

- $\dfrac{\Delta \vdash_{\Gamma} \forall_\rho M \qquad \Gamma \vdash N : \rho}{\Delta \vdash_{\Gamma} MN}\ \forall_\rho^-$

We need to show $\Delta \models_{\Gamma} MN$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_{\Gamma} \Delta$, to show $\mathscr{M}, \eta \models_{\Gamma} MN$, i.e. to show $[\![MN]\!]^{\Gamma}\eta = 1$.

We have the following induction hypothesis

$$\Delta \models_{\Gamma} \forall_\rho M \qquad \text{(IH1)}$$

From IH1, we get $\mathscr{M}, \eta \models_{\Gamma} \forall_\rho M$ by Definition 3.3.10. That is $[\![\forall_\rho M]\!]^{\Gamma}\eta = 1$, i.e. $\forall a \in [\![\rho]\!].[\![M]\!]^{\Gamma}\eta(a) = 1$. Then by $\Gamma \vdash N : \rho$ and $\eta \models \Gamma$, we get $[\![N]\!]^{\Gamma}\eta \in [\![\rho]\!]$. Thus, we get $[\![M]\!]^{\Gamma}\eta([\![N]\!]^{\Gamma}\eta) = 1$, i.e. $[\![MN]\!]^{\Gamma}\eta = 1$ by Definition 3.3.9.

- $\dfrac{\Delta \vdash_{\Gamma} MN \qquad \Gamma \vdash N : \rho}{\Delta \vdash_{\Gamma} \exists_\rho M}\ \exists_\rho^+$

We need to show $\Delta \models_{\Gamma} \exists_\rho M$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_{\Gamma} \Delta$, to show $\mathscr{M}, \eta \models_{\Gamma} \exists_\rho M$, i.e. to show $[\![\exists_\rho M]\!]^{\Gamma}\eta = 1$. That is, $[\![\exists_\rho]\!][\![M]\!]^{\Gamma}\eta = 1$, i.e. to show $\exists a \in [\![\rho]\!].[\![M]\!]^{\Gamma}\eta(a) = 1$.

We have the following induction hypothesis

$$\Delta \models_{\Gamma} MN \qquad \text{(IH1)}$$

Then we get $\mathscr{M}, \eta \models_{\Gamma} MN$ by Definition 3.3.10. That is, $[\![MN]\!]^{\Gamma}\eta = 1$, i.e. $[\![M]\!]^{\Gamma}\eta([\![N]\!]^{\Gamma}\eta) = 1$. Then by Definition 3.3.9, from $\Gamma \vdash N : \rho$ and $\eta \models \Gamma$, we get $[\![N]\!]^{\Gamma}\eta \in [\![\rho]\!]$. Thus, we get $\exists a \in [\![\rho]\!].[\![M]\!]^{\Gamma}\eta(a) = 1$.

- $\dfrac{\Delta \vdash_\Gamma \exists_\rho M \qquad \Delta, Mx \vdash_{\Gamma,x:\rho} C}{\Delta \vdash_\Gamma C}\ \exists_\rho^-$ $\quad x$ fresh

We need to show $\Delta \models_\Gamma C$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma C$, i.e. to show $[\![C]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma \exists_\rho M \tag{IH1}$$
$$\Delta, Mx \models_{\Gamma,x:\rho} C \tag{IH2}$$

Then by Definition 3.3.10 we have

$$\forall \eta'(\eta' \models \Gamma \wedge \mathscr{M}, \eta' \models_\Gamma \Delta) \Rightarrow \mathscr{M}, \eta' \models_\Gamma \exists_\rho M$$
$$\forall \eta''(\eta'' \models (\Gamma, x:\rho) \wedge \mathscr{M}, \eta'' \models_{\Gamma,x:\rho} (\Delta, Mx)) \Rightarrow \mathscr{M}, \eta'' \models_{\Gamma,x:\rho} C$$

Let $\eta' := \eta$, and $\eta'' := \eta[x := a]$.

From $\mathscr{M}, \eta \models_\Gamma \exists_\rho M$, we get $[\![\exists_\rho M]\!]^\Gamma \eta = 1$, i.e. $[\![\exists_\rho]\!][\![M]\!]^\Gamma \eta = 1$, i.e. $\exists a \in [\![\rho]\!].[\![M]\!]^\Gamma \eta\ (a) = 1$.

Before applying IH2, we need to show a) $\eta[x := a] \models (\Gamma, x:\rho)$ and b)$\mathscr{M}, \eta[x := a] \models_{\Gamma,x:\rho} (\Delta, Mx)$ first. Let $a \in [\![\rho]\!]$.

For a), it is to show $\forall i.\eta[x := a](x_i) \in [\![(\Gamma, x:\rho)(x_i)]\!]$ where $x_i \in \mathrm{dom}(\Gamma, x:\rho)$. If $x_i \in \mathrm{dom}(\Gamma)$, then $\eta[x := a](x_i) = \eta(x_i) \in [\![\Gamma(x_i)]\!]$ by the assumption $\eta \models \Gamma$. If $x_i \equiv x$, then $\eta[x := a](x) = a \in [\![\rho]\!]$. Thus, $\eta[x := a](x_i) \in [\![(\Gamma, x:\rho)(x_i)]\!]$.

For b), it is to show $[\![\Delta, Mx]\!]^{\Gamma,x:\rho} \eta[x := a] = 1$, i.e. $[\![\Delta]\!]^{\Gamma,x:\rho} \eta[x := a] = 1$ and $[\![Mx]\!]^{\Gamma,x:\rho} \eta[x := a] = 1$ which is equal to $[\![M]\!]^{\Gamma,x:\rho} \eta[x := a]([\![x]\!]^{\Gamma,x:\rho} \eta[x := a]) = 1$, i.e. $[\![M]\!]^{\Gamma,x:\rho} \eta[x := a](\eta[x := a](x)) = 1$, i.e. $[\![M]\!]^{\Gamma,x:\rho} \eta[x := a](a) = 1$. Since x is fresh, then by Lemma 3.3.13 it is to show $[\![\Delta]\!]^\Gamma \eta = 1$ and $[\![M]\!]^\Gamma \eta(a) = 1$. It follows by the assumption $\mathscr{M}, \eta \models_\Gamma \Delta$ and IH1.

From $\mathscr{M}, \eta[x := a] \models_{\Gamma,x:\rho} C$ (from IH2), we get $[\![C]\!]^{\Gamma,x:\rho} \eta[x := a] = 1$. Since x is fresh, by Lemma 3.3.13, we get $[\![C]\!]^\Gamma \eta = 1$.

- $\dfrac{\Delta, pA \vdash_{\Gamma,p:\rho\to o} pB}{\Delta \vdash_\Gamma A =_\rho B}\ =^+$ $\quad p$ fresh

We need to show $\Delta \models_\Gamma A =_\rho B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A =_\rho B$, i.e. to show $[\![A =_\rho B]\!]^\Gamma \eta = 1$. That is, for every $p \in [\![\rho \to o]\!]$, assume $[\![pA]\!]^\Gamma \eta = 1$, to show $[\![pB]\!]^\Gamma \eta = 1$.

We have the following induction hypothesis

$$\Delta, pA \models_{\Gamma,p:\rho\to o} pB \tag{IH1}$$

Then by Definition 3.3.10 we have

$$\forall \eta'(\eta' \models \Gamma' \wedge \mathscr{M}, \eta' \models_{\Gamma'} (\Delta, pA)) \Rightarrow \mathscr{M}, \eta' \models_{\Gamma'} pB$$

Let $\eta' := \eta[p := q]$ and $\Gamma' := \Gamma, p : \rho \to o$.

Before applying IH1, we need to show a) $\eta[p := q] \models (\Gamma, p : \rho \to o)$ and b) $\mathscr{M}, \eta[p := q] \models_{\Gamma, p:\rho\to o} (\Delta, pA)$ first. Let $q \in [\![\rho \to o]\!]$.

For a), it is to show $\forall i. \eta[p := q](p_i) \in [\![(\Gamma, p : \rho \to o)(p_i)]\!]$ where $p_i \in \mathrm{dom}(\Gamma, p : \rho \to o)$. If $p_i \in \mathrm{dom}(\Gamma)$, then $\eta[p := q](p_i) = \eta(p_i) \in [\![\Gamma(p_i)]\!]$ by the assumption $\eta \models \Gamma$. If $p_i \equiv p$, then $\eta[p := q](p) = q \in [\![\rho \to o]\!]$. Thus, $\eta[p := q](p_i) \in [\![(\Gamma, p : \rho \to o)(p_i)]\!]$.

For b), it is to show $[\![\Delta, pA]\!]^{\Gamma, p:\rho\to o}\eta[p := q] = 1$, i.e. $[\![\Delta]\!]^{\Gamma, p:\rho\to o}\eta[p := q] = 1$ and $[\![pA]\!]^{\Gamma, p:\rho\to o}\eta[p := q] = 1$. Since p is fresh, then by Lemma 3.3.13 it is to show $[\![\Delta]\!]^{\Gamma}\eta = 1$ and $[\![pA]\!]^{\Gamma}\eta = 1$. It follows by the assumptions $\mathscr{M}, \eta \models_{\Gamma} \Delta$ and $[\![pA]\!]^{\Gamma}\eta = 1$.

From $\mathscr{M}, \eta[p := q] \models_{\Gamma, p:\rho\to o} pB$ (from IH1), we get $[\![pB]\!]^{\Gamma, p:\rho\to o}\eta[p := q] = 1$. Since p is fresh, by Lemma 3.3.13, we get $[\![pB]\!]^{\Gamma}\eta = 1$.

- $$\frac{\Delta \vdash_\Gamma A =_\rho B \qquad \Delta \vdash_\Gamma PA}{\Delta \vdash_\Gamma PB} =^-$$

We need to show $\Delta \models_\Gamma PB$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma PB$, i.e. to show $[\![PB]\!]^{\Gamma}\eta = 1$.

We have the following induction hypothesis

$$\Delta \models_\Gamma A =_\rho B \qquad \text{(IH1)}$$
$$\Delta \models_\Gamma PA \qquad \text{(IH2)}$$

By Definition 3.3.10, from IH1, we have $\mathscr{M}, \eta \models_\Gamma A =_\rho B$. That is, $[\![A =_\rho B]\!]^{\Gamma}\eta = 1$, i.e. if $[\![PA]\!]^{\Gamma}\eta = 1$, then $[\![PB]\!]^{\Gamma}\eta = 1$ for every $P \in [\![\rho \to o]\!]$. $[\![PA]\!]^{\Gamma}\eta = 1$ holds by IH2. Thus, $[\![PB]\!]^{\Gamma}\eta = 1$.

- $$\frac{\Delta \vdash_{\Gamma, x:\rho} Ax =_\sigma Bx}{\Delta \vdash_\Gamma A =_{\rho\to\sigma} B}\ \text{ext} \quad x \text{ fresh}$$

We need to show $\Delta \models_\Gamma A =_{\rho\to\sigma} B$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma A =_{\rho\to\sigma} B$, i.e. to show $[\![A =_{\rho\to\sigma} B]\!]^{\Gamma}\eta = 1$. That is, for every $p \in (\rho \to \sigma) \to o$, assume $[\![pA]\!]^{\Gamma}\eta = 1$, to show $[\![pB]\!]^{\Gamma}\eta = 1$.

We have the following induction hypothesis

$$\Delta \models_{\Gamma, x:\rho} Ax =_\sigma Bx \qquad \text{(IH1)}$$

Then by Definition 3.3.10 we have

$$\forall \eta'(\eta' \models \Gamma' \wedge \mathscr{M}, \eta' \models_{\Gamma'} \Delta) \Rightarrow \mathscr{M}, \eta' \models_{\Gamma'} Ax =_\sigma Bx$$

Let $\eta' := \eta[x := a]$ and $\Gamma' := \Gamma, x : \rho$.

Before applying IH1, we need to show a) $\eta[x := a] \models (\Gamma, x : \rho)$ and b)$\mathscr{M}, \eta[x := a] \models_{\Gamma, x:\rho} \Delta$ first. Let $a \in [\![\rho]\!]$.

For a), it is to show $\forall i.\eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$ where $x_i \in \mathrm{dom}(\Gamma, x : \rho)$. If $x_i \in \mathrm{dom}(\Gamma)$, then $\eta[x := a](x_i) = \eta(x_i) \in [\![\Gamma(x_i)]\!]$ by the assumption $\eta \models \Gamma$. If $x_i \equiv x$, then $\eta[x := a](x) = a \in [\![\rho]\!]$. Thus, $\eta[x := a](x_i) \in [\![(\Gamma, x : \rho)(x_i)]\!]$.

For b), it is to show $[\![\Delta]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$. Since x is fresh, then by Lemma 3.3.13 it is to show $[\![\Delta]\!]^{\Gamma}\eta = 1$. It follows by the assumptions $\mathscr{M}, \eta \models_\Gamma \Delta$.

From $\mathscr{M}, \eta[x := a] \models_{\Gamma, x:\rho} Ax =_\sigma Bx$ (from IH1), we get $[\![Ax =_\sigma Bx]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$. That is, for every $q \in \sigma \to o$, if $[\![qAx]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$, then $[\![qAx]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$. It equals to for every $p \in (\rho \to \sigma) \to o$, if $[\![pA]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$, then $[\![pB]\!]^{\Gamma, x:\rho}\eta[x := a] = 1$. Since x is fresh, by Lemma 3.3.13, it equals to if $[\![pA]\!]^{\Gamma}\eta = 1$, then $[\![pB]\!]^{\Gamma}\eta = 1$.

- $$\frac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi}\ \mathrm{Cl}_\rho$$

  We need to show $\Delta \models_\Gamma \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma \Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi$, i.e. to show $[\![\Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi]\!]^{\Gamma}\eta = 1$.

  By the definition of $[\![\mu_\rho]\!]$, we have $[\![\Phi(\mu_\rho \Phi)]\!]^{\Gamma}\eta = [\![\mu_\rho \Phi]\!]^{\Gamma}\eta$. Then since $[\![\subseteq_\rho]\!]$ is a partial order, we have $[\![\Phi(\mu_\rho \Phi)]\!]^{\Gamma}\eta\ [\![\subseteq_\rho]\!]\ [\![\mu_\rho \Phi]\!]^{\Gamma}\eta$, i.e. $[\![\Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi]\!]^{\Gamma}\eta$ holds.

- $$\frac{\Delta \vdash_\Gamma \mathrm{mono}_\rho(\Phi) \quad \Delta \vdash_\Gamma \Phi(M) \subseteq_\rho M}{\Delta \vdash_\Gamma \mu_\rho \Phi \subseteq_\rho M}\ \mathrm{Ind}_\rho$$

  We need to show $\Delta \models_\Gamma \mu_\rho \Phi \subseteq_\rho M$. Then by Definition 3.3.10, that is, assume $\eta \models \Gamma$ and $\mathscr{M}, \eta \models_\Gamma \Delta$, to show $\mathscr{M}, \eta \models_\Gamma \mu_\rho \Phi \subseteq_\rho M$, i.e. to show $[\![\mu_\rho \Phi \subseteq_\rho M]\!]^{\Gamma}\eta = 1$.

  This follows by the Tarski's theorem.

- $$\frac{\Gamma \vdash \Phi : \rho \to \rho \quad \Gamma \vdash \Delta : o}{\Delta \vdash_\Gamma \nu_\rho \Phi \subseteq_\rho \Phi(\nu_\rho \Phi)}\ \mathrm{Cocl}_\rho$$

  The proof is dual to the Cl_ρ case.

- $$\frac{\Delta \vdash_\Gamma \mathrm{mono}_\rho(\Phi) \quad \Delta \vdash_\Gamma M \subseteq_\rho \Phi(M)}{\Delta \vdash_\Gamma M \subseteq_\rho \nu_\rho(\Phi)}\ \mathrm{Coi}_\rho$$

  The proof is dual to the Ind_ρ case.

□

3.4 Conclusion

Firstly we have studied the simply typed lambda calculus (with products), and introduced the notion of an interpretation from one instance of the simply typed lambda calculus to another. Every interpretation respects β-equality.

And then we have described a particular instance of the simply typed lambda calculus, called CST, where the set of base types contains a set $\mathscr{I}$ of base types for individuals and the proposition type o, and the constant set consists of the constants $\wedge, \vee, \rightarrow, \forall_\rho, \exists_\rho, =_\rho, \mu_\rho$, and ν_ρ.

Our deductive system of CST is different from Church's original one in several aspects: first, we used natural deduction while Church used a Hilbert style calculus; second, our calculus is intuitionistic while Church's is classical; third, we dropped the choice operator in order to enable a realisability interpretation.

Chapter 4

Program Extraction via Realisability

Contents

One method of program extraction is to employ a realisability interpretation. In 1945 Kleene [Kle45] first introduced the concept of realisability with the idea of defining a relation "*n realises A*" between natural numbers *n* and logical sentences *A*. Later many other notions on realisability were introduced. In particular Gödel's functional or Dialectica Interpretation [Goe58, Goe90] and Kreisel's modified realisability [Kre59] have a profound impact. Originally intended to be a contribution to Hilbert's program, the Dialectica interpretation showed that the consistency of Peano Arithmetic is reduced to that of Gödel's system T, a quantifier-free theory based on finite types. In contrast to the Dialectica Interpretation, modified realisability also established a correspondence between Peano Arithmetic and System T, but treated logical implication differently. In addition, Kleene and Vesley [KV65, Tro73] introduced a notion of function realisability for the purpose of providing a classically understood model for Brouwerian Intuitionism. The possibility of effectively obtaining a program and its verification proof is based on a sound realisability interpretation. A historical account of realisability is presented in [VO02].

After nearly three decades of research, program extraction constitutes a fruitful area with an established theory and incorporations in several proof assistants. We give a brief overview of some proof systems supporting program extractions. In Section 4.3, we will elaborate more details on this. The first theorem provers that support program extraction are Nuprl [CAA+86] which is based on Martin-Löf type

theory, and the PX system [HN87], based on Feferman's theory of functions and classes [Fef79], which provides an optimising method of extracting untyped LISP programs from proofs of specifications. An interesting method of program extraction is implemented in PhoX [Raf04] which uses second-order formulas to define data types. The program extraction facilities of the Coq system [PM89b, PMW93, Let03], based on a variety of type theory, Calculus of Inductive Constructions [CPM90] can transform Coq proofs and functions into OCaml, Haskell and Scheme. A distinguish characterisation of Minlog [BBS+98] is that the logical language in use is minimal first order logic. There has also been some work on program extraction in the Isabelle system [Ber03c] which is based on simply-typed, minimal higher order logic. Additionally, based on the Curry-Howard correspondence, some constructively typed languages, e.g. Agda [BDN09, FS11] and Epigram [MM04], allow program extractions in a way. Their type checkers ensure the correctness of extracted programs at compile time. Chuang in his PhD thesis gives a concrete example of applying Agda's program extraction machinery to solve exact real number computation problem in [Chu11].

In this chapter in order to implement program extractions, CST described in Chapter 3 is extended by some extra datatype and new constants. Hence, the realisability is applied to the full CST, and it is untyped.

4.1 Realisability Interpretation

We will begin our exploration of realisability by extending CST to a system RCST that contains an untyped rudimentary functional programming language. We will give a realisability interpretation of CST in RCST and prove a soundness result.

As mentioned previously in Section 3.3, we extend CST to RCST by adding an extra base type δ for realisers and extra constants nil, in_L, in_R, pr_L, pr_R, pair, app, fun, case, rec with the following typing

$$\begin{aligned}
\mathrm{nil} &: \delta \\
\mathrm{in}_L, \mathrm{in}_R, \mathrm{pr}_L, \mathrm{pr}_R &: \delta \to \delta \\
\mathrm{pair}, \mathrm{app} &: \delta \to \delta \to \delta \\
\mathrm{fun} &: (\delta \to \delta) \to \delta \\
\mathrm{case} &: \delta \to (\delta \to \delta) \to (\delta \to \delta) \to \delta \\
\mathrm{rec} &: (\delta \to \delta) \to \delta
\end{aligned}$$

where δ is interpreted as a Scott domain D obtained as the solution to the recursive domain equation

$$\mathbf{D} \simeq (\mathbf{1} + \mathbf{D} + \mathbf{D} + \mathbf{D} + \mathbf{D} \times \mathbf{D} + [\mathbf{D} \to \mathbf{D}])_\perp$$

We also extend the ranges of the parameters of the constants $\forall_\rho$, $\exists_\rho$, $=_\rho$ and μ_ρ, ν_ρ to all types respectively predicate types ρ of RCST.

We define $\equiv_\delta$ as the least congruence relation on terms such that

$$\begin{aligned}
\mathrm{case}\,(\mathrm{in}_L M)\,M_1\,M_2 &\equiv_\delta M_1 M\\
\mathrm{case}\,(\mathrm{in}_R M)\,M_1\,M_2 &\equiv_\delta M_2 M\\
\mathrm{pr}_L\,(\mathrm{pair} M_1\,M_2) &\equiv_\delta M_1\\
\mathrm{pr}_R\,(\mathrm{pair} M_1\,M_2) &\equiv_\delta M_2\\
\mathrm{app}\,(\mathrm{fun} M)\,N &\equiv_\delta MN\\
\mathrm{app} M(\mathrm{rec} M) &\equiv_\delta \mathrm{rec}\,M
\end{aligned}$$

and define $\equiv$ as the least congruence relation containing $\equiv_\beta$ and $\equiv_\delta$.

Note that the case expression here is a special instance (with two clauses) of the case expression $(\mathrm{case} M \,\mathrm{of}\,\{C_i(\vec{x}_i) \to R_i\}_{i\in\{1,\ldots,n\}})$ in Chapter 2.

For better readability we use the notations

$$\begin{aligned}
\mathrm{case} M\,\mathrm{of}\,\{(\mathrm{in}_L x) \to M_1;\,(\mathrm{in}_R x) \to M_2\} &:= \mathrm{case} M\,(\lambda x:\delta.M_1)\,(\lambda x:\delta.M_2)\\
\lambda_\delta x.M &:= \mathrm{fun}\,(\lambda x:\delta.M)\\
M\cdot N &:= \mathrm{app} MN
\end{aligned}$$

Hence

$$\begin{aligned}
\mathrm{case}\,(\mathrm{in}_L M)\,\mathrm{of}\,\{(\mathrm{in}_L x) \to M_1;\,(\mathrm{in}_R x) \to M_2\} &\equiv M_1[M/x]\\
\mathrm{case}\,(\mathrm{in}_R M)\,\mathrm{of}\,\{(\mathrm{in}_L x) \to M_1;\,(\mathrm{in}_R x) \to M_2\} &\equiv M_2[M/x]\\
(\lambda_\delta x.M)\cdot N &\equiv M[N/x]
\end{aligned}$$

The logical rules of RCST are the same as those of CST (with ρ ranging over RCST types of course), except for the rule β and the rules for μ_ρ and ν_ρ.

The β rule is replaced by the rule

$$\frac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma B}\,\beta\delta \quad A \equiv B.$$

We replace the rules for monotone induction, Cl_ρ and Ind_ρ, by rules expressing that $\mu_\rho\Phi$ is the least fixed point of the operator $\check{\Phi}(X) := \bigcup^\rho_{Y\subseteq_\rho X}\Phi(Y)$. The operator $\check{\Phi}$ (which could be easily formally defined in RCST) is monotone for arbitrary $\Phi : \rho \to \rho$, hence the least fixed point exists. If Φ is monotone, then Φ is the same as $\check{\Phi}$. Dually, the rules for coinduction are replaced by rules expressing that $\nu_\rho\Phi$ is the greatest fixed point of the operator $\hat{\Phi}(X) := \bigcap^\rho_{Y\supseteq_\rho X}\Phi(Y)$. This general (co)induction is a generali-

sation to higher-order logic of Mendler-style (co)induction [Men91].

$$\frac{\Delta \vdash_\Gamma M \subseteq_\rho \mu_\rho \Phi}{\Delta \vdash_\Gamma \Phi(M) \subseteq_\rho \mu_\rho \Phi}\ \mathrm{ClG}_\rho \qquad \frac{\Delta, Y \subseteq_\rho M \vdash_\Gamma \Phi(Y) \subseteq_\rho M}{\Delta \vdash_\Gamma \mu_\rho \Phi \subseteq_\rho M}\ \mathrm{IndG}_\rho\ Y \text{ fresh}$$

$$\frac{\Delta \vdash_\Gamma \nu_\rho \Phi \subseteq_\rho M}{\Delta \vdash_\Gamma \nu_\rho \Phi \subseteq_\rho \Phi(M)}\ \mathrm{CoclG}_\rho \qquad \frac{\Delta, M \subseteq_\rho Y \vdash M \subseteq_\rho \Phi(Y)}{\Delta \vdash_\Gamma M \subseteq_\rho \nu_\rho \Phi}\ \mathrm{CoiG}_\rho\ Y \text{ fresh}$$

Definition 4.1.1 (Realisability Interpretation) We define an interpretation $(\mathbf{r}, \mathbf{R})$ of CST in RCST where the base type substitution

$$\mathbf{r}(o) := \delta \to o$$
$$\mathbf{r}(\iota) := \iota$$

and the constant substitution (for the sake of readability we write $\forall x : \rho.A$ for $\forall_\rho(\lambda x : \rho.A)$ and $\exists x : \rho.A$ for $\exists_\rho(\lambda x : \rho.A)$)

$$\mathbf{R}(\to) := \lambda \widetilde{A}, \widetilde{B} : \delta \to o.\lambda d : \delta.\forall a : \delta.\widetilde{A}a \to \widetilde{B}(d \cdot a)$$
$$\mathbf{R}(\wedge) := \lambda \widetilde{A}, \widetilde{B} : \delta \to o.\lambda d : \delta.\exists a, b : \delta.d =_\delta \mathrm{pair} ab \wedge \widetilde{A}a \wedge \widetilde{B}b$$
$$\mathbf{R}(\vee) := \lambda \widetilde{A}, \widetilde{B} : \delta \to o.\lambda d : \delta.\exists a : \delta.(d =_\delta \mathrm{in}_L a \wedge \widetilde{A}a) \vee (d =_\delta \mathrm{in}_R a \wedge \widetilde{B}a)$$
$$\mathbf{R}(\forall_\rho) := \lambda \widetilde{A} : \mathbf{r}(\rho) \to \delta \to o.\lambda d : \delta.\forall \widetilde{x} : \mathbf{r}(\rho).\widetilde{A}\widetilde{x}d$$
$$\mathbf{R}(\exists_\rho) := \lambda \widetilde{A} : \mathbf{r}(\rho) \to \delta \to o.\lambda d : \delta.\exists \widetilde{x} : \mathbf{r}(\rho).\widetilde{A}\widetilde{x}d$$
$$\mathbf{R}(=_\rho) := \lambda \widetilde{A}, \widetilde{B} : \mathbf{r}(\rho).\lambda d : \delta.d =_\delta \mathrm{nil} \wedge \widetilde{A} =_{\mathbf{r}(\rho)} \widetilde{B}$$
$$\mathbf{R}(\mu_\rho) := \mu_{\mathbf{r}(\rho)}$$
$$\mathbf{R}(\nu_\rho) := \nu_{\mathbf{r}(\rho)}$$

According to Definition 3.2.2 the type substitution $\mathbf{r}$ and the constant substitution $\mathbf{R}$ induce a mapping $\mathbf{R}_\mathbf{r}$ from CST terms to RCST terms. For notational simplicity we denote this mapping again by $\mathbf{R}$.

Remark The definition of $\mathbf{R}(c)$ becomes more comprehensible when we apply it to

arguments.

$$\begin{aligned}
\mathbf{R}(A \to B)\, d &\equiv \forall a : \delta.\, \mathbf{R}(A)\, a \to \mathbf{R}(B)(d \cdot a) \\
\mathbf{R}(A \wedge B)\, d &\equiv \exists a, b : \delta.\, d =_\delta \mathrm{pair}\, a\, b \wedge \mathbf{R}(A)\, a \wedge \mathbf{R}(B)\, b \\
\mathbf{R}(A \vee B)\, d &\equiv (\exists a : \delta.\, d =_\delta \mathrm{in}_L\, a \wedge \mathbf{R}(A)\, a) \vee (\exists b : \delta.\, d =_\delta \mathrm{in}_R\, b \wedge \mathbf{R}(B)\, b) \\
\mathbf{R}(\forall x : \rho.A)\, d &\equiv \forall \widetilde{x} : \mathbf{r}(\rho).\, \mathbf{R}(A)\, d \\
\mathbf{R}(\exists x : \rho.A)\, d &\equiv \exists \widetilde{x} : \mathbf{r}(\rho).\, \mathbf{R}(A)\, d \\
\mathbf{R}(A =_\rho B)\, d &\equiv d =_\delta \mathrm{nil} \wedge \mathbf{R}(A) =_{\mathbf{r}(\rho)} \mathbf{R}(B) \\
\mathbf{R}(\mu_\rho\, \Phi x)\, d &\equiv \mu_{\mathbf{r}(\rho)} \widetilde{\Phi} \widetilde{x} d \\
\mathbf{R}(\nu_\rho\, \Phi x)\, d &\equiv \nu_{\mathbf{r}(\rho)} \widetilde{\Phi} \widetilde{x} d
\end{aligned}$$

Example 4.1.2 (Realisers of natural numbers)

Let us continue with Example 3.3.15.

Recall that we define the set of natural numbers as an inductive predicate. Set $\Phi := \lambda p : \iota \to o.\, \lambda x : \iota.\, x =_\iota 0 \vee p(x-1)$, and define $\mathbb{N} := \mu_{\iota \to o} \Phi$. A more readable notation for the definition of $\mathbb{N}$ would be

$$\mathbb{N}x \stackrel{\mu}{=} x =_\iota 0 \vee \mathbb{N}(x-1).$$

As an example of a proof by induction we show that the natural numbers are closed under addition:

$$\Delta_+ \vdash_{\Gamma_+} \forall x, y : \iota.\, \mathbb{N}x \wedge \mathbb{N}y \to \mathbb{N}(x+y)$$

Setting $P(x) := \lambda y : \iota.\, \mathbb{N}(x+y)$, the formula to be proven is equivalent to $\forall x : \iota.\, \mathbb{N}x \to \mathbb{N} \subseteq_{\iota \to o} P(x)$. Hence, it suffices to show $\Phi P(x) \subseteq_{\iota \to o} P(x)$ under the extra assumption $\mathbb{N}x$. Unfolding the definition of Φ and using proof by cases ($\vee^-$), this amounts to proving $\mathbb{N}(x+0)$ and $\forall y : \iota.\, \mathbb{N}(x+(y-1)) \to \mathbb{N}(x+y)$, which is easy, given the assumptions Δ_+ and $\mathbb{N}x$.

The realisability interpretation of natural numbers is

$$\mathbf{R}(\mathbb{N}x)\, d \equiv \mu_{\iota \to \delta \to o} \mathbf{R}(\Phi)\, x\, d$$

where $\mathbf{R}(\Phi) \equiv \lambda \widetilde{p} : \iota \to \delta \to o.\, \lambda x : \iota.\, \lambda d : \delta.\, (d =_\delta \mathrm{in}_L \mathrm{nil} \wedge x =_\iota 0) \vee (\exists b : \delta.\, d =_\delta \mathrm{in}_R\, b \wedge \widetilde{p}(x-1)\, b)$.

In a more readable notation:

$$\mathbf{R}(\mathbb{N}x)\, d \stackrel{\mu}{=} (d =_\delta \mathrm{in}_L \mathrm{nil} \wedge x = 0) \vee (\exists b : \delta.\, d =_\delta \mathrm{in}_R\, b \wedge \mathbf{R}(\mathbb{N}(x-1))\, b).$$

Hence, a natural number $n : \iota$ is realised by the numeral $\underline{n} : \delta$, where $\underline{0} := \mathrm{in}_L \mathrm{nil}$, $\underline{n+1} := \mathrm{in}_R \underline{n}$.

An element $d : \delta$ realises the closure of natural numbers under addition, i.e. the formula $\forall x,y : \iota.\mathbb{N}x \wedge \mathbb{N}y \to \mathbb{N}(x+y)$, if for all $x,y : \iota$ and all $a,b : \delta$

$$\mathbf{R}(\mathbb{N}x)a \wedge \mathbf{R}(\mathbb{N}y)b \to \mathbf{R}(\mathbb{N}(x+y))(d \cdot (\mathrm{pair}\, a\, b)).$$

which says that d adds natural numbers in unary notation.

Example 4.1.3 (Interpretation for Fibonacci Numbers)
Let us continue with Example 3.3.16.
Recall that we define a coinductive predicate FIB $:= \nu_{\iota\to\iota\to o}\Psi$ where

$$\Psi := \lambda q : \iota \to \iota \to o.\lambda x,y : \iota.\mathbb{N}x \wedge qy(x+y).$$

A more readable notation would be

$$\mathrm{FIB}\, x\, y \stackrel{\nu}{=} \mathbb{N}x \wedge \mathrm{FIB}\, y\, (x+y)$$

As an example of a proof that uses coinduction we show

$$\Delta_+ \vdash_{\Gamma_+} \mathrm{FIB}\, 1\, 1.$$

We show more generally, that FIB holds for any two natural numbers, i.e. $Q \subseteq_{\iota\to\iota\to o} \mathrm{FIB}$ where $Q := \lambda x,y : \iota.\mathbb{N}x \wedge \mathbb{N}y$. By coinduction, it suffices to show $Q \subseteq_{\iota\to\iota\to o} \Psi Q$, which is easily done using the previously proven fact that natural numbers are closed under addition.

The interpretation of Fibonacci numbers is

$$\mathbf{R}(\mathrm{FIB}\, x\, y)\, d \equiv \nu_{\iota\to\iota\to\delta\to o}\mathbf{R}(\Psi)\, x\, y\, d$$

where $\mathbf{R}(\Psi) \equiv \lambda\widetilde{q} : \iota \to \iota \to \delta \to o.\lambda x,y : \iota.\lambda d : \delta.\exists a,b : \delta.d =_\delta \mathrm{pair}\, a\, b \wedge \mathbf{R}(\mathbb{N}x)a \wedge \widetilde{q}\, y\, (x+y)\, b$.

In more readable notation

$$\mathbf{R}(\mathrm{FIB}\, x\, y)\, d \stackrel{\nu}{=} \mathbf{R}(\mathbb{N}x)(\mathrm{pr}_L d) \wedge \mathbf{R}(\mathrm{FIB}\, y\, (x+y))(\mathrm{pr}_R d).$$

which says that d is a stream of natural numbers in unary notation where the head realises $\mathbb{N}x$ and the tail realises $\mathrm{FIB}\, y\, (x+y)$.

Let us compare the given realisability interpretation of equality with the realisability interpretation of Leibniz equality.

Definition 4.1.4

$$\equiv^{\mathrm{L}}_\rho := \lambda A,B : \rho.\forall p : \rho \to o.\, pA \to pB$$

Lemma 4.1.5

(a) $A \equiv^{\mathrm{L}}_{\rho} B \leftrightarrow A =_{\rho} B$

(b) $\mathbf{R}(\equiv^{\mathrm{L}}_{\rho})\widetilde{A}\widetilde{B}f \leftrightarrow \forall \widetilde{q} : \mathbf{r}(\rho) \to \delta \to o.\forall a : \delta.\widetilde{q}\widetilde{A}a \to \widetilde{q}\widetilde{B}(f \cdot a)$

(c) $(\exists f : \delta.\mathbf{R}(\equiv^{\mathrm{L}}_{\rho})\widetilde{A}\widetilde{B}f) \leftrightarrow \widetilde{A} \equiv^{\mathrm{L}}_{\mathbf{r}(\rho)} \widetilde{B}$

Proof. (a) By the rules $=^{+}$ and $=^{-}$.

(b) By the definition of realisability.

(c) $\Rightarrow$:

Assume $\mathbf{R}(\equiv^{\mathrm{L}}_{\rho})\widetilde{A}\widetilde{B}f$ and $p : \mathbf{r}(\rho) \to o$, we need to show $p\widetilde{A} \to p\widetilde{B}$.

Then by (b) we get $\forall q : \mathbf{r}(\rho) \to \delta \to o.\forall a : \delta.q\widetilde{A}a \to q\widetilde{B}(f \cdot a)$.

Let $q := \lambda x : \mathbf{r}(\rho)\lambda a : \delta.px$. Then we get $p\widetilde{A} \to p\widetilde{B}$.

$\Leftarrow$:

Assume $\widetilde{A} \equiv^{\mathrm{L}}_{r(\rho)} \widetilde{B}$. It is easy to see that for $f := \mathrm{Fun}(\lambda a : \delta.a)$ we have $R(\equiv^{\mathrm{L}}_{\rho})\widetilde{A}\widetilde{B}f$.

□

Definition 4.1.6 (Realisability for sequents) Realisability for a sequent $\Delta \vdash_{\Gamma} A$ is defined as follows.

Let $\Delta := \{B_1, \ldots, B_n\}$ and $\Gamma := \{x_1 : \rho_1, \ldots, x_k : \rho_k\}$. Then $a : \delta$ realises $\Delta \vdash_{\Gamma} A$ if a realises the formula $\forall x_1 : \rho_1 \ldots \forall x_k : \rho_k.B_1 \wedge B_2 \wedge \ldots \wedge B_n \to A$.

Hence,

$$\mathbf{R}(\Delta \vdash_{\Gamma} A)a \equiv \forall \widetilde{x_1} : \mathbf{r}(\rho_1) \ldots \forall \widetilde{x_k} : \mathbf{r}(\rho_k).\forall b : \delta.\mathbf{R}(B_1 \wedge \ldots \wedge B_n)b \to \mathbf{R}(A)(a \cdot b).$$

We write $\Delta \vdash^{\mathbf{r}}_{\Gamma} A$ for a derivable sequent in RCST.

Note that the statement that a term M realises a sequent $\Delta \vdash_{\Gamma} A$ is equivalent to the derivability of $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)\,(M \cdot \mathrm{pair}_n(b_1, \ldots, b_n))$ and $\Gamma' \vdash M : \delta$, where $\Gamma' := \{a_i : \delta \mid i \in \{1, \ldots, n\}\}$, $\mathbf{r}(\Gamma) = \{\widetilde{x_1} : \mathbf{r}(\rho_1), \ldots, \widetilde{x_k} : \mathbf{r}(\rho_k)\}$, $\mathbf{R}(\Delta) := \{\mathbf{R}(B_i)\,b_i \mid i \in \{1, \ldots, n\}\}$ with fresh variables b_i, and pair_n is defined as $\mathrm{pair}_2(b_1, b_2) = \mathrm{pair}\,b_1\,b_2$, $\mathrm{pair}_n(b_1, \ldots, b_n) = \mathrm{pair}\,(\mathrm{pair}_{n-1}(b_1, \ldots, b_{n-1}))\,b_n$.

The following Soundness Theorem for realisability interpretation states that from a proof of a formula one can extract a program provably realising it.

Theorem 4.1.7 (Soundness) If $\Delta \vdash_{\Gamma} A$, then there exists some term M of type δ such that $\mathbf{R}(\Delta \vdash_{\Gamma} A)M$ is provable.

Proof. By induction on the derivation of $\Delta \vdash_{\Gamma} A$.

It suffices to show that each rule is realisable, i.e., for a rule

$$\frac{\Delta_1 \vdash_{\Gamma_1} A_1 \quad \ldots \quad \Delta_n \vdash_{\Gamma_n} A_n}{\Delta \vdash_{\Gamma} A}$$

to show that there is a term $e : \delta$ such that for all $a_1, \ldots, a_n : \delta$ realising the premises, $e \cdot a_1 \cdot \ldots \cdot a_n$ realises the conclusion.

Note that if the proof contexts Δ and Δ_i, and the typing contexts Γ and Γ_i are all the same respectively, it suffices to find a realiser of the formula $A_1 \wedge \ldots \wedge A_n \to A$.

Non-logical rules (equality, induction and coinduction) will be proved separately in later theorems (Theorem 4.1.8, Theorem 4.2.10 and Theorem 4.2.11).

- $\dfrac{\Gamma \vdash \Delta, A : o}{\Delta, A \vdash_\Gamma A}$ use

  We need to show $\mathbf{R}(\Delta, A) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A) M$.

  By Definition 4.1.6, we have $\mathbf{R}(\Delta, A) = \mathbf{R}(\Delta) \cup \{\mathbf{R}(A)\widetilde{x}\}$. We set $M := \widetilde{x}$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma B}\ \beta\delta \quad A \equiv B$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B) M$.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A) M_0 \tag{IH1}$$

  By Lemma 3.2.9, we have $\mathbf{R}(A) \equiv \mathbf{R}(B)$.

  Hence, we have $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B) M_0$, by $\beta\delta$ rule, so we set $M := M_0$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \wedge B}\ \wedge^+$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \wedge B) M$.

  It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)(\mathrm{pr}_L M) \wedge \mathbf{R}(B)(\mathrm{pr}_R M)$ by Definition 4.1.1.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A) M_1 \tag{IH1}$$
  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B) M_2 \tag{IH2}$$

  We set $M := \mathrm{pair} M_1 M_2$. Thus, $M_1 \equiv \mathrm{pr}_L M$ and $M_2 \equiv \mathrm{pr}_R M$.

  By IH1, IH2 and Rule $\wedge^+$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)(\mathrm{pr}_L M) \wedge \mathbf{R}(B)(\mathrm{pr}_R M)$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma A}\ \wedge_l^-$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A) M$.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \wedge B) M_0 \tag{IH1}$$

We have $\mathbf{R}(A \wedge B)M_0 \equiv \mathbf{R}(A)(\mathrm{pr}_L M_0) \wedge \mathbf{R}(B)(\mathrm{pr}_R M_0)$ by Definition 4.1.1.

We set $M := \mathrm{pr}_L M_0$. By Rule $\wedge_l^-$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)M$.

- $\dfrac{\Delta \vdash_\Gamma A \wedge B}{\Delta \vdash_\Gamma B}\ \wedge_r^-$

We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B)M$.

We have the following induction hypothesis

$$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \wedge B)M_0 \qquad \text{(IH1)}$$

We have $\mathbf{R}(A \wedge B)M_0 \equiv \mathbf{R}(A)(\mathrm{pr}_L M_0) \wedge \mathbf{R}(B)(\mathrm{pr}_R M_0)$ by Definition 4.1.1.

We set $M := \mathrm{pr}_R M_0$. By Rule $\wedge_r^-$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B)M$.

- $\dfrac{\Delta \vdash_\Gamma A \qquad \Gamma \vdash B : o}{\Delta \vdash_\Gamma A \vee B}\ \vee_l^+$

We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \vee B)M$.

It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} (\exists a : \delta.M =_\delta \mathrm{in}_L a \wedge \mathbf{R}(A)a) \vee (\exists b : \delta.M =_\delta \mathrm{in}_R b \wedge \mathbf{R}(B)b)$ by Definition 4.1.1.

We have the following induction hypothesis

$$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)M_1 \qquad \text{(IH1)}$$

We set $M := \mathrm{in}_L M_1$. By Rule $\vee_l^+$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} (\exists a : \delta.M =_\delta \mathrm{in}_L a \wedge \mathbf{R}(A)a) \vee (\exists b : \delta.M =_\delta \mathrm{in}_R b \wedge \mathbf{R}(B)b)$.

- $\dfrac{\Gamma \vdash A : o \qquad \Delta \vdash_\Gamma B}{\Delta \vdash_\Gamma A \vee B}\ \vee_r^+$

We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \vee B)M$.

It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} (\exists a : \delta.M =_\delta \mathrm{in}_L a \wedge \mathbf{R}(A)a) \vee (\exists b : \delta.M =_\delta \mathrm{in}_R b \wedge \mathbf{R}(B)b)$ by Definition 4.1.1.

We have the following induction hypothesis

$$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B)M_2 \qquad \text{(IH1)}$$

We set $M := \mathrm{in}_R M_2$. By Rule $\vee_r^+$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} (\exists a : \delta.M =_\delta \mathrm{in}_L a \wedge \mathbf{R}(A)a) \vee (\exists b : \delta.M =_\delta \mathrm{in}_R b \wedge \mathbf{R}(B)b)$.

- $\dfrac{\Delta \vdash_\Gamma A \vee B \qquad \Delta, A \vdash_\Gamma C \qquad \Delta, B \vdash_\Gamma C}{\Delta \vdash_\Gamma C}\ \vee^-$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(C)M$.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \vee B)M_0 \tag{IH1}$$
  $$\mathbf{R}(\Delta), \mathbf{R}(A)\widetilde{x} \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma', \widetilde{x}:\delta} \mathbf{R}(C)M_1 \tag{IH2}$$
  $$\mathbf{R}(\Delta), \mathbf{R}(B)\widetilde{y} \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma', \widetilde{y}:\delta} \mathbf{R}(C)M_2 \tag{IH3}$$

  From IH1, we have $\mathbf{R}(A \vee B)M_0 \equiv \exists a : \delta.(M_0 =_\delta \mathrm{in}_L a \wedge \mathbf{R}(A)a) \vee (M_0 =_\delta \mathrm{in}_R a \wedge \mathbf{R}(B)a)$ by Definition 4.1.1.

  We set $M := \mathrm{case} M_0 \,\mathrm{of}\, \{(\mathrm{in}_L \widetilde{x}) \to M_1; (\mathrm{in}_R \widetilde{x}) \to M_2\}$. Without loss of generality, we assume $M_0 := \mathrm{in}_L a$ for some $a : \delta$ such that $\mathbf{R}(A)a$. Then $M \equiv M_1[a/\widetilde{x}]$. Hence, by IH2, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(C)M$.

- $\dfrac{\Delta, A \vdash_\Gamma B}{\Delta \vdash_\Gamma A \to B}\ \to^+$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \to B)M$.

  It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \forall a : \delta.\mathbf{R}(A)a \to \mathbf{R}(B)(M \cdot a)$ by Definition 4.1.1.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta), \mathbf{R}(A)\widetilde{x} \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma', \widetilde{x}:\delta} \mathbf{R}(B)M_0 \tag{IH1}$$

  We set $M := \lambda_\delta \widetilde{x}.M_0$. Thus $M \cdot a \equiv M_0[a/\widetilde{x}]$. By Rule $\to^+$ and $\forall^+$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \forall a : \delta.\mathbf{R}(A)a \to \mathbf{R}(B)(M \cdot a)$.

- $\dfrac{\Delta \vdash_\Gamma A \to B \qquad \Delta \vdash_\Gamma A}{\Delta \vdash_\Gamma B}\ \to^-$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B)M$.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A \to B)M_0 \tag{IH1}$$
  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(A)M_1 \tag{IH2}$$

  From IH1 we get $\mathbf{R}(A \to B) \equiv \forall a : \delta.\mathbf{R}(A)a \to \mathbf{R}(B)(M_0 \cdot a)$ by Definition 4.1.1.

  We set $M := M_0 \cdot M_1$. By Rule $\forall^-$ and $\to^-$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(B)M$.

- $\dfrac{\Delta \vdash_{\Gamma, x:\rho} Mx}{\Delta \vdash_{\Gamma} \forall_\rho M}\ \forall^+_\rho$ x fresh

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(\forall_\rho A)\, M$.

  It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \forall \widetilde{x}\colon \mathbf{r}(\rho).\mathbf{R}(A)\,\widetilde{x} M$ by Definition 4.1.1.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma), \widetilde{x}:\mathbf{r}(\rho)\cup\Gamma'} \mathbf{R}(Ax)\, M_0 \qquad \text{(IH1)}$$

  We set $M := M_0$. Since $\mathbf{R}(Ax) = \mathbf{R}(A)\widetilde{x}$, by Rule $\forall^+_{\mathbf{r}(\rho)}$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \forall \widetilde{x}\colon \mathbf{r}(\rho).\mathbf{R}(A)\,\widetilde{x} M$.

- $\dfrac{\Delta \vdash_{\Gamma} \forall_\rho M \qquad \Gamma \vdash N : \rho}{\Delta \vdash_{\Gamma} MN}\ \forall^-_\rho$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(MN)\, K$.

  It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(M)\mathbf{R}(N)K$ by Definition 3.2.2.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(\forall_\rho M)\, K_0 \qquad \text{(IH1)}$$

  We have $\mathbf{R}(\forall_\rho M)\, K_0 \equiv \forall \widetilde{x}\colon \mathbf{r}(\rho).\mathbf{R}(M)\,\widetilde{x} K_0$ by Definition 4.1.1.

  We set $K := K_0$, by Rule $\forall^-_{\mathbf{r}(\rho)}$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(M)\mathbf{R}(N)K$.

- $\dfrac{\Delta \vdash_{\Gamma} MN \qquad \Gamma \vdash N : \rho}{\Delta \vdash_{\Gamma} \exists_\rho M}\ \exists^+_\rho$

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(\exists_\rho M)\, K$.

  It is to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \exists \widetilde{x}\colon \mathbf{r}(\rho).\mathbf{R}(M)\,\widetilde{x} K$ by Definition 4.1.1.

  We have the following induction hypothesis

  $$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(MN)\, K_0 \qquad \text{(IH1)}$$

  We have $\mathbf{R}(MN)K_0 = \mathbf{R}(M)\mathbf{R}(N)K_0$ by Definition 3.2.2.

  We set $K := K_0$, by Rule $\exists^+_{\mathbf{r}(\rho)}$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \exists \widetilde{x}\colon \mathbf{r}(\rho).\mathbf{R}(M)\,\widetilde{x} K$.

- $\dfrac{\Delta \vdash_{\Gamma} \exists_\rho M \qquad \Delta, Mx \vdash_{\Gamma, x:\rho} C}{\Delta \vdash_{\Gamma} C}\ \exists^-_\rho$ x fresh

  We need to show $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(C)\, K$.

We have the following induction hypothesis

$$\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(\exists_\rho M)\, K_1 \tag{IH1}$$

$$\mathbf{R}(\Delta), \mathbf{R}(Mx)\widetilde{y} \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma),\widetilde{x}:\delta\cup\Gamma',\widetilde{y}:\delta} \mathbf{R}(C)\, K_2 \tag{IH2}$$

We have $\mathbf{R}(\exists_\rho M)\, K_1 \equiv \exists\widetilde{x}:\mathbf{r}(\rho).\mathbf{R}(M)\,\widetilde{x}K_1$ by Definition 4.1.1, and $\mathbf{R}(Mx)\widetilde{y} = \mathbf{R}(M)\,\widetilde{x}\widetilde{y}$ by Definition 3.2.2.

We set $K := K_0$, by Rule $\exists^-_{\mathbf{r}(\rho)}$, we get $\mathbf{R}(\Delta) \vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)\cup\Gamma'} \mathbf{R}(C)\, K$.

□

Theorem 4.1.8 The rules for equality are realisable.

Proof. $(=^+)$: We show $e := \lambda a:\delta.\lambda b:\delta.\mathrm{nil}$ is a realiser.

Assume f realises $\Delta, pA \vdash_{\Gamma,p:\rho\to o} pB$, that is, (1) $\forall\widetilde{p}:\mathbf{r}(\rho)\to\delta\to o.\forall\vec{a},b$ $(\mathbf{R}(\Delta)\,\vec{a} \wedge \mathbf{R}(pA)\,b \to \mathbf{R}(pB)(f\cdot\vec{a}\cdot b))$, to show $\forall\vec{a}(\mathbf{R}(\Delta)\,\vec{a} \to \mathbf{R}(A =_\rho B)(e\cdot f\cdot\vec{a}))$.

Hence, assume $\mathbf{R}(\Delta)\,\vec{a}$, to show $\mathbf{R}(A =_\rho B)\mathrm{nil}$. By the definition of $\mathbf{R}(=_\rho)$, it is to show $\mathbf{R}(A) =_{\mathbf{r}(\rho)} \mathbf{R}(B)$.

Let $\widetilde{p} := \lambda\widetilde{x}:\mathbf{r}(\rho).\lambda a:\delta.\mathbf{R}(A) =_{\mathbf{r}(\rho)} \widetilde{x}$. Then we get $\mathbf{R}(pA)\,b \equiv \mathbf{R}(A) =_{\mathbf{r}(\rho)} \mathbf{R}(A)$. Hence, the premise of (1) holds. Therefore, $\mathbf{R}(pB)(f\cdot\vec{a}\cdot b) \equiv \mathbf{R}(A) =_{\mathbf{r}(\rho)} \mathbf{R}(B)$.

$(=^-)$: We show $e := \lambda f_1, f_2:\delta.f_2$ is a realiser.

Assume $f_1 := \lambda a:\delta.\mathrm{nil}$ realises $\Delta \vdash_\Gamma A =_\rho B$ and f_2 realises $\Delta \vdash_\Gamma PA$, that is, for all $\vec{a}$, (1) $\mathbf{R}(\Delta)\vec{a} \to \mathbf{R}(A =_\rho B)(f_1\cdot\vec{a})$ and (2) $\mathbf{R}(\Delta)\vec{a} \to \mathbf{R}(PA)(f_2\cdot\vec{a})$, it is to show $\forall\vec{a}.\mathbf{R}(\Delta)\vec{a} \to \mathbf{R}(PB)(e\cdot f_1\cdot f_2\cdot\vec{a})$.

Hence, assume $\mathbf{R}(\Delta)\,\vec{a}$, to show $\mathbf{R}(PB)(e\cdot f_1\cdot f_2\cdot\vec{a})$.

That is to show $\mathbf{R}(P)\mathbf{R}(B)(f_2\cdot\vec{a})$.

From (1) we have $\mathbf{R}(A =_\rho B)\mathrm{nil}$, i.e., $\mathbf{R}(A) =_{\mathbf{r}(\rho)} \mathbf{R}(B)$. From (2) we have $\mathbf{R}(PA)(f_2\cdot\vec{a})$, i.e. $\mathbf{R}(P)\mathbf{R}(A)(f_2\cdot\vec{a})$.

By Rule $=^-$, we obtain $\mathbf{R}(P)\mathbf{R}(B)(f_2\cdot\vec{a})$ as required since $\mathbf{R}(P)\mathbf{R}(A)(f_2\cdot\vec{a}) \equiv_\beta (\lambda\widetilde{p}:\mathbf{r}(\rho).\mathbf{R}(P)\widetilde{p}(f_2\cdot\vec{a}))\mathbf{R}(A)$.

(ext) : We show $e := \lambda a:\delta.\lambda b:\delta.\mathrm{nil}$ is a realiser.

Assume $f := \lambda a:\delta.\mathrm{nil}$ realises $\Delta \vdash_{\Gamma,x:\rho} Ax =_\sigma Bx$.

That is, (1) $\forall x:\rho.\forall\vec{a}\ (\mathbf{R}(\Delta)\,\vec{a} \to \mathbf{R}(Ax =_\sigma Bx)(f\cdot\vec{a}))$, to show $\forall\vec{a}(\mathbf{R}(\Delta)\,\vec{a} \to \mathbf{R}(A =_{\rho\to\sigma} B)(e\cdot f\cdot\vec{a}))$. Hence, assume $\mathbf{R}(\Delta)\,\vec{a}$, to show $\mathbf{R}(A =_{\rho\to\sigma} B)\mathrm{nil}$.

By the definition of $\mathbf{R}(=_{\rho\to\sigma})$, it is to show $\mathbf{R}(A) =_{\mathbf{r}(\rho\to\sigma)} \mathbf{R}(B)$, i.e. to show $\mathbf{R}(A) =_{\mathbf{r}(\rho)\to\mathbf{r}(\sigma)} \mathbf{R}(B)$.

From (1) we have $\forall x:\rho.\mathbf{R}(Ax =_\sigma Bx)\mathrm{nil}$, i.e. $\forall \widetilde{x}:\mathbf{r}(\rho).\mathbf{R}(A)\widetilde{x} =_{\mathbf{r}(\sigma)} \mathbf{R}(B)\widetilde{x}$.

Therefore, by Rule (ext) at type $\mathbf{r}(\rho)$, we have $\mathbf{R}(A) =_{\mathbf{r}(\rho)\to\mathbf{r}(\sigma)} \mathbf{R}(B)$.

□

4.2 Induction and Coinduction

The proof of the soundness for induction and coinduction hinges on the fact that the realisability of an inclusion can be expressed by an inclusion. To this end we introduce image and inverse-image operations. The realisability interpretation of higher type inclusion $\subseteq_\rho$ can then be expressed as a composition of $\subseteq_{\mathbf{r}(\rho)}$ and these image and inverse-image operations.

Definition 4.2.1 For every predicate type ρ we define terms $\mathrm{Im}_\rho, \mathrm{Im}^-_\rho : \delta \to \mathbf{r}(\rho) \to \mathbf{r}(\rho)$ as follows.

$$\begin{aligned}
\mathrm{Im}_o &:= \lambda d:\delta.\lambda p:\delta\to o.\lambda b:\delta.\exists a:\delta.\, pa \wedge d\cdot a =_\delta b\\
\mathrm{Im}_{\rho\to\sigma} &:= \lambda d:\delta.\lambda p:\mathbf{r}(\rho)\to\mathbf{r}(\sigma).\lambda x:\mathbf{r}(\rho).\,\mathrm{Im}_\sigma\, d\,(px)\\
\mathrm{Im}_{\rho\times\sigma} &:= \lambda d:\delta.\lambda p:\mathbf{r}(\rho)\times\mathbf{r}(\sigma).\langle \mathrm{Im}_\rho\,(\mathrm{pr}_L d)\,\pi_0(p), \mathrm{Im}_\sigma\,(\mathrm{pr}_R d)\,\pi_1(p)\rangle\\
\mathrm{Im}^-_o &:= \lambda d:\delta.\lambda p:\delta\to o.\lambda a:\delta.\, p(d\cdot a)\\
\mathrm{Im}^-_{\rho\to\sigma} &:= \lambda d:\delta.\lambda p:\mathbf{r}(\rho)\to\mathbf{r}(\sigma).\lambda x:\mathbf{r}(\rho).\,\mathrm{Im}^-_\sigma\, d\,(px)\\
\mathrm{Im}^-_{\rho\times\sigma} &:= \lambda d:\delta.\lambda p:\mathbf{r}(\rho)\times\mathbf{r}(\sigma).\langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L d)\,\pi_0(p), \mathrm{Im}^-_\sigma\,(\mathrm{pr}_R d)\,\pi_1(p)\rangle
\end{aligned}$$

Lemma 4.2.2 For every predicate type ρ, provably in RCST the following are equivalent for all $\widetilde{A},\widetilde{B}:\mathbf{r}(\rho)$ and $d:\delta$

(a) $\mathbf{R}(A \subseteq_\rho B)\,d$

(b) $\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}^-_\rho\, d\,\widetilde{B}$

(c) $\mathrm{Im}_\rho\, d\,\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \widetilde{B}$

Proof. By induction on ρ.

- $\rho \equiv o$.

First, to show $\mathbf{R}(A \subseteq_o B)d \leftrightarrow \mathrm{Im}_o\, d\widetilde{A} \subseteq_{\mathbf{r}(o)} \widetilde{B}$.

$$
\begin{aligned}
&\mathbf{R}(A \subseteq_o B)d \\
&=_o \mathbf{R}(A \to B)d && \text{(By Definition 3.3.4)} \\
&=_o \mathbf{R}(\to)\mathbf{R}(A)\mathbf{R}(B)d \\
&=_o \forall a : \delta.\widetilde{A}a \to \widetilde{B}(d \cdot a) && \text{(By Definition 4.1.1)} \\
&=_o \forall u : \delta.(\exists a : \delta.\widetilde{A}a \wedge d \cdot a =_o u) \to \widetilde{B}u && \text{(Let } u := d \cdot a\text{)} \\
&=_o \forall u : \delta.\mathrm{Im}_o\, d\widetilde{A}u \to \widetilde{B}u && \text{(By Definition 4.2.1)} \\
&=_o \mathrm{Im}_o\, d\widetilde{A} \subseteq_{\delta \to o} \widetilde{B} && \text{(By Definition 3.3.4)} \\
&=_o \mathrm{Im}_o\, d\widetilde{A} \subseteq_{\mathbf{r}(o)} \widetilde{B} && \text{(By Definition 4.1.1)}
\end{aligned}
$$

Then to show $\mathbf{R}(A \subseteq_o B)d =_o \widetilde{A} \subseteq_{\mathbf{r}(o)} \mathrm{Im}_o^-\, d\widetilde{B}$.

$$
\begin{aligned}
&\mathbf{R}(A \subseteq_o B)d \\
&=_o \mathbf{R}(A \to B)d && \text{(By Definition 3.3.4)} \\
&=_o \mathbf{R}(\to)\mathbf{R}(A)\mathbf{R}(B)d \\
&=_o \forall a : \delta.\widetilde{A}a \to \widetilde{B}(d \cdot a) && \text{(By Definition 4.1.1)} \\
&=_o \forall a : \delta.\widetilde{A}a \to \mathrm{Im}_o^-\, d\widetilde{B}a && \text{(By Definition 4.2.1)} \\
&=_o \widetilde{A} \subseteq_{\delta \to o} \mathrm{Im}_o^-\, d\widetilde{B} && \text{(By Definition 3.3.4)} \\
&=_o \widetilde{A} \subseteq_{\mathbf{r}(o)} \mathrm{Im}_o^-\, d\widetilde{B} && \text{(By Definition 4.1.1)}
\end{aligned}
$$

- $\rho \equiv \rho \to \sigma$.

  First, to show $\mathbf{R}(A \subseteq_{\rho \to \sigma} B)d \leftrightarrow \mathrm{Im}_{\rho \to \sigma}\, d\widetilde{A} \subseteq_{\mathbf{r}(\rho \to \sigma)} \widetilde{B}$.

$$
\begin{aligned}
&\mathbf{R}(A \subseteq_{\rho \to \sigma} B)d \\
&=_o \mathbf{R}(\forall x : \rho.Ax \subseteq_\sigma Bx)d && \text{(By Definition 3.3.4)} \\
&=_o \mathbf{R}(\forall_\rho)\mathbf{R}(\lambda x : \rho.Ax \subseteq_\sigma Bx)d \\
&=_o \mathbf{R}(\forall_\rho)(\lambda \widetilde{x} : \mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx))d \\
&=_o \forall u : \mathbf{r}(\rho).(\lambda \widetilde{x} : \mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx))ud && \text{(By Definition 4.1.1)} \\
&=_o \forall u : \mathbf{r}(\rho).\mathbf{R}(\subseteq_\sigma)(\widetilde{A}\widetilde{u})(\widetilde{B}\widetilde{u})d && \text{(By } \equiv_\beta\text{)} \\
&=_o \forall u : \mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx)d \\
&=_o \forall u : \mathbf{r}(\rho).\mathrm{Im}_\sigma\, d(\widetilde{A}u) \subseteq_{\mathbf{r}(\sigma)} (\widetilde{B}u) && \text{(By I.H.)} \\
&=_o \forall u : \mathbf{r}(\rho).(\mathrm{Im}_{\rho \to \sigma}\, d\widetilde{A})u \subseteq_{\mathbf{r}(\sigma)} \widetilde{B}u && \text{(By Definition 4.2.1)} \\
&=_o (\mathrm{Im}_{\rho \to \sigma}\, d\widetilde{A}) \subseteq_{\mathbf{r}(\rho \to \sigma)} \widetilde{B} && \text{(By Definition 3.3.4)}
\end{aligned}
$$

Then to show $\mathbf{R}(A \subseteq_{\rho\to\sigma} B)d =_o \widetilde{A} \subseteq_{\mathbf{r}(\rho\to\sigma)} \mathrm{Im}^{-}_{\rho\to\sigma} d\widetilde{B}$.

$$
\begin{array}{ll}
& \mathbf{R}(A \subseteq_{\rho\to\sigma} B)d \\
=_o & \mathbf{R}(\forall x:\rho. Ax \subseteq_\sigma Bx)d & \text{(By Definition 3.3.4)} \\
=_o & \mathbf{R}(\forall_\rho)\mathbf{R}(\lambda x:\rho. Ax \subseteq_\sigma Bx)d \\
=_o & \mathbf{R}(\forall_\rho)(\lambda\widetilde{x}:\mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx))d \\
=_o & \forall u:\mathbf{r}(\rho).(\lambda\widetilde{x}:\mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx))ud & \text{(By Definition 4.1.1)} \\
=_o & \forall u:\mathbf{r}(\rho).\mathbf{R}(\subseteq_\sigma)(\widetilde{A}u)(\widetilde{B}u)d & \text{(By } \equiv_\beta) \\
=_o & \forall u:\mathbf{r}(\rho).\mathbf{R}(Ax \subseteq_\sigma Bx)d \\
=_o & \forall u:\mathbf{r}(\rho).\widetilde{A}u \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}^{-}_{\sigma} d(\widetilde{B}u) & \text{(By I.H.)} \\
=_o & \forall u:\mathbf{r}(\rho).\widetilde{A}u \subseteq_{\mathbf{r}(\sigma)} (\mathrm{Im}^{-}_{\rho\to\sigma} d\widetilde{B})u & \text{(By Definition 4.2.1)} \\
=_o & \widetilde{A} \subseteq_{\mathbf{r}(\rho\to\sigma)} \mathrm{Im}^{-}_{\rho\to\sigma} d\widetilde{B} & \text{(By Definition 3.3.4)}
\end{array}
$$

- $\rho \equiv \rho \times \sigma$.

First, to show $\mathbf{R}(A \subseteq_{\rho\times\sigma} B)d \leftrightarrow \mathrm{Im}_{\rho\times\sigma} d\widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \widetilde{B}$.

$$
\begin{array}{ll}
& \mathbf{R}(A \subseteq_{\rho\times\sigma} B)d \\
=_o & \mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B) \wedge \pi_1(A) \subseteq_\sigma \pi_1(B))d & \text{(By Definition 3.3.4)} \\
=_o & \mathbf{R}(\wedge)\mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B))\mathbf{R}(\pi_1(A) \subseteq_\sigma \pi_1(B))d \\
=_o & \mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B))(\mathrm{pr}_L d) \\
& \quad \wedge \mathbf{R}(\pi_1(A) \subseteq_\sigma \pi_1(B))(\mathrm{pr}_R d) & \text{(By Definition 4.1.1)} \\
=_o & \mathrm{Im}_\rho(\mathrm{pr}_L d)\pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\widetilde{B}) \\
& \quad \wedge \mathrm{Im}_\sigma(\mathrm{pr}_R d)\pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\widetilde{B}) & \text{(By I.H.)} \\
=_o & \pi_0(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\pi_0(\widetilde{A}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\pi_1(\widetilde{A})\rangle) \subseteq_{\mathbf{r}(\rho)} \pi_0(\widetilde{B}) \\
& \quad \wedge \pi_1(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\pi_0(\widetilde{A}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\pi_1(\widetilde{A})\rangle) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\widetilde{B}) & \text{(By } \equiv_\beta) \\
=_o & \pi_0(\mathrm{Im}_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\widetilde{B}) \\
& \quad \wedge \pi_1(\mathrm{Im}_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\widetilde{B}) & \text{(By Definition 4.2.1)} \\
=_o & (\mathrm{Im}_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\rho\times\sigma)} \widetilde{B} & \text{(By Definition 3.3.4)}
\end{array}
$$

Then to show $\mathbf{R}(A \subseteq_{\rho\times\sigma} B)d =_o \widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \mathrm{Im}^-_{\rho\times\sigma} d\widetilde{B}$.

$$
\begin{aligned}
&\mathbf{R}(A \subseteq_{\rho\times\sigma} B)d \\
&=_o \mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B) \wedge \pi_1(A) \subseteq_\sigma \pi_1(B))d && \text{(By Definition 3.3.4)} \\
&=_o \mathbf{R}(\wedge)\mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B))\mathbf{R}(\pi_1(A) \subseteq_\sigma \pi_1(B))d \\
&=_o \mathbf{R}(\pi_0(A) \subseteq_\rho \pi_0(B))\,(\mathrm{pr}_L d) \\
&\quad \wedge \mathbf{R}(\pi_1(A) \subseteq_\sigma \pi_1(B))\,(\mathrm{pr}_R d) && \text{(By Definition 4.1.1)} \\
&=_o \pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}^-_\rho\,(\mathrm{pr}_L d)\,\pi_0(\widetilde{B}) \\
&\quad \wedge \pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}^-_\sigma\,(\mathrm{pr}_R d)\,\pi_1(\widetilde{B}) && \text{(By I.H.)} \\
&=_o \widetilde{A} \subseteq_{\mathbf{r}(\rho\to\sigma)} \mathrm{Im}^-_{\rho\to\sigma} d\widetilde{B} && \text{(By Definition 3.3.4)}
\end{aligned}
$$

□

Lemma 4.2.3 For every predicate type ρ and every $d : \delta$, $\mathrm{Im}_\rho d$ and $\mathrm{Im}^-_\rho d$ are monotone, i.e.

(a) $\vdash^{\mathbf{r}} \forall d : \delta.\forall \widetilde{A},\widetilde{B} : \mathbf{r}(\rho).\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \widetilde{B} \to \mathrm{Im}_\rho d\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho d\widetilde{B}$

(b) $\vdash^{\mathbf{r}} \forall d : \delta.\forall \widetilde{A},\widetilde{B} : \mathbf{r}(\rho).\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \widetilde{B} \to \mathrm{Im}^-_\rho d\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}^-_\rho d\widetilde{B}$

Proof. By induction on ρ.

- $\rho \equiv o$.

(a) $\widetilde{A} \subseteq_{\mathbf{r}(o)} \widetilde{B}$

$$
\begin{aligned}
&\Rightarrow \widetilde{A} \subseteq_{\delta\to o} \widetilde{B} && \text{(By Definition 4.1.1)} \\
&\Rightarrow \forall u : \delta.\widetilde{A}u \subseteq_o \widetilde{B}u && \text{(By Definition 3.3.4)} \\
&\Rightarrow \forall u : \delta.\widetilde{A}u \to \widetilde{B}u && \text{(By Definition 3.3.4)} \\
&\Rightarrow \forall u : \delta.(\exists a : \delta.\widetilde{A}a \wedge d\cdot a =_\delta u) \to \\
&\qquad (\exists b : \delta.\widetilde{B}b \wedge d\cdot b =_\delta u) && \text{(Chosen } u := a \text{ and } b := a\text{)} \\
&\Rightarrow \forall u : \delta.\mathrm{Im}_o d\widetilde{A}u \to \mathrm{Im}_o d\widetilde{B}u && \text{(By Definition 4.2.1)} \\
&\Rightarrow \forall u : \delta.\mathrm{Im}_o d\widetilde{A}u \subseteq_o \mathrm{Im}_o d\widetilde{B}u && \text{(By Definition 3.3.4)} \\
&\Rightarrow \mathrm{Im}_o d\widetilde{A} \subseteq_{\delta\to o} \mathrm{Im}_o d\widetilde{B} && \text{(By Definition 3.3.4)} \\
&\Rightarrow \mathrm{Im}_o d\widetilde{A} \subseteq_{\mathbf{r}(o)} \mathrm{Im}_o d\widetilde{B} && \text{(By Definition 4.1.1)}
\end{aligned}
$$

(*b*) $\widetilde{A} \subseteq_{\mathbf{r}(o)} \widetilde{B}$

$\Rightarrow \widetilde{A} \subseteq_{\delta \to o} \widetilde{B}$ (By Definition 4.1.1)

$\Rightarrow \forall a : \delta . \widetilde{A}\, a \subseteq_o \widetilde{B} a$ (By Definition 3.3.4)

$\Rightarrow \forall a : \delta . \widetilde{A}\, a \to \widetilde{B} a$ (By Definition 3.3.4)

$\Rightarrow \forall u : \delta . \widetilde{A}(d \cdot u) \to \widetilde{B}(d \cdot u)$ (Chosen $a := d \cdot u$)

$\Rightarrow \forall u : \delta . \mathrm{Im}_o^- \, d\widetilde{A}\, u \to \mathrm{Im}_o^- \, d\widetilde{B} u$ (By Definition 4.2.1)

$\Rightarrow \forall u : \delta . \mathrm{Im}_o^- \, d\widetilde{A}\, u \subseteq_o \mathrm{Im}_o^- \, d\widetilde{B} u$ (By Definition 3.3.4)

$\Rightarrow \mathrm{Im}_o^- \, d\widetilde{A} \subseteq_{\delta \to o} \mathrm{Im}_o^- \, d\widetilde{B}$ (By Definition 3.3.4)

$\Rightarrow \mathrm{Im}_o^- \, d\widetilde{A} \subseteq_{\mathbf{r}(o)} \mathrm{Im}_o^- \, d\widetilde{B}$ (By Definition 4.1.1)

- $\rho \equiv \rho \to \sigma$.

(*a*) $\widetilde{A} \subseteq_{\mathbf{r}(\rho \to \sigma)} \widetilde{B}$

$\Rightarrow \widetilde{A} \subseteq_{\mathbf{r}(\rho) \to \mathbf{r}(\sigma)} \widetilde{B}$

$\Rightarrow \forall u : \mathbf{r}(\rho) . \widetilde{A}\, u \subseteq_{\mathbf{r}(\sigma)} \widetilde{B} u$ (By Definition 3.3.4)

$\Rightarrow \forall u : \mathbf{r}(\rho) . \mathrm{Im}_\sigma \, d\widetilde{A}\, u \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}_\sigma \, d\widetilde{B} u$ (By I.H.)

$\Rightarrow \mathrm{Im}_{\rho \to \sigma} \, d\widetilde{A} \subseteq_{\mathbf{r}(\rho \to \sigma)} \mathrm{Im}_{\rho \to \sigma} \, d\widetilde{B}$ (By Definition 3.3.4)

(*b*) $\widetilde{A} \subseteq_{\mathbf{r}(\rho \to \sigma)} \widetilde{B}$

$\Rightarrow \widetilde{A} \subseteq_{\mathbf{r}(\rho) \to \mathbf{r}(\sigma)} \widetilde{B}$

$\Rightarrow \forall u : \mathbf{r}(\rho) . \widetilde{A}\, u \subseteq_{\mathbf{r}(\sigma)} \widetilde{B} u$ (By Definition 3.3.4)

$\Rightarrow \forall u : \mathbf{r}(\rho) . \mathrm{Im}_\sigma^- \, d\widetilde{A}\, u \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}_\sigma^- \, d\widetilde{B} u$ (By I.H.)

$\Rightarrow \mathrm{Im}_{\rho \to \sigma}^- \, d\widetilde{A} \subseteq_{\mathbf{r}(\rho \to \sigma)} \mathrm{Im}_{\rho \to \sigma}^- \, d\widetilde{B}$ (By Definition 3.3.4)

- $\rho \equiv \rho \times \sigma$.

  (a) $\widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \widetilde{B}$

  $\Rightarrow \pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\widetilde{B}) \wedge \pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\widetilde{B})$ (By Definition 3.3.4)

  $\Rightarrow \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B})$

  $\wedge \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})$ (By I.H.)

  $\Rightarrow \pi_0(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A})\rangle)$

  $\subseteq_{\mathbf{r}(\rho)} \pi_0(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})\rangle)$

  $\wedge \pi_1(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A})\rangle)$

  $\subseteq_{\mathbf{r}(\sigma)} \pi_1(\langle \mathrm{Im}_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B}), \mathrm{Im}_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})\rangle)$ (By $\equiv_\beta$)

  $\Rightarrow \pi_0(\mathrm{Im}_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\mathrm{Im}_{\rho\times\sigma} d\widetilde{B})$

  $\pi_1(\mathrm{Im}_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\mathrm{Im}_{\rho\times\sigma} d\widetilde{B})$ (By Definition 4.2.1)

  $\Rightarrow \mathrm{Im}_{\rho\times\sigma} d\widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \mathrm{Im}_{\rho\times\sigma} d\widetilde{B}$ (By Definition 3.3.4)

  (b) $\widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \widetilde{B}$

  $\Rightarrow \pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\widetilde{B}) \wedge \pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\widetilde{B})$ (By Definition 3.3.4)

  $\Rightarrow \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B})$

  $\wedge \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})$ (By I.H.)

  $\Rightarrow \pi_0(\langle \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}), \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A})\rangle)$

  $\subseteq_{\mathbf{r}(\rho)} \pi_0(\langle \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B}), \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})\rangle)$

  $\wedge \pi_1(\langle \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{A}), \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{A})\rangle)$

  $\subseteq_{\mathbf{r}(\sigma)} \pi_1(\langle \mathrm{Im}^-_\rho(\mathrm{pr}_L d)\,\pi_0(\widetilde{B}), \mathrm{Im}^-_\sigma(\mathrm{pr}_R d)\,\pi_1(\widetilde{B})\rangle)$ (By $\equiv_\beta$)

  $\Rightarrow \pi_0(\mathrm{Im}^-_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \pi_0(\mathrm{Im}^-_{\rho\times\sigma} d\widetilde{B})$

  $\pi_1(\mathrm{Im}^-_{\rho\times\sigma} d\widetilde{A}) \subseteq_{\mathbf{r}(\sigma)} \pi_1(\mathrm{Im}^-_{\rho\times\sigma} d\widetilde{B})$ (By Definition 4.2.1)

  $\Rightarrow \mathrm{Im}^-_{\rho\times\sigma} d\widetilde{A} \subseteq_{\mathbf{r}(\rho\times\sigma)} \mathrm{Im}^-_{\rho\times\sigma} d\widetilde{B}$ (By Definition 3.3.4)

□

Definition 4.2.4 For every predicate type ρ we define a closed term id_ρ of type δ such that

$$\mathrm{id}_o := \mathrm{fun}(\lambda a : \delta.\, a)$$
$$\mathrm{id}_{\rho\to\sigma} := \mathrm{id}_\sigma$$
$$\mathrm{id}_{\rho\times\sigma} := \mathrm{pair}(\mathrm{id}_\rho, \mathrm{id}_\sigma)$$

Intuitively, id_ρ is just a name for what will be used later as realisers of the closure and coclosure rules.

Lemma 4.2.5 For every $p : \mathbf{r}(\rho)$, $\mathrm{Im}^-_\rho\, \mathrm{id}_\rho\, p =_{\mathbf{r}(\rho)} p$ is provable in RCST for every predicate type ρ.

Proof. By induction on ρ.

- $\rho \equiv o$.

$$\begin{aligned}
&\mathrm{Im}^-_o\, \mathrm{id}_o\, p \\
&=_{\mathbf{r}(o)} \lambda a : \delta.\, p\,(\mathrm{id}_o \cdot a) && \text{(By Definition 4.2.1)} \\
&=_{\mathbf{r}(o)} \lambda a : \delta.\, pa && \text{(By Definition 4.2.4 and } \equiv_\beta) \\
&=_{\mathbf{r}(o)} p
\end{aligned}$$

- $\rho \equiv \rho \to \sigma$.

$$\begin{aligned}
&\mathrm{Im}^-_{\rho\to\sigma}\, \mathrm{id}_{\rho\to\sigma}\, p \\
&=_{\mathbf{r}(\rho\to\sigma)} \lambda x : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, \mathrm{id}_{\rho\to\sigma}\,(px) && \text{(By Definition 4.2.1)} \\
&=_{\mathbf{r}(\rho\to\sigma)} \lambda x : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, \mathrm{id}_\sigma\,(px) && \text{(By Definition 4.2.4)} \\
&=_{\mathbf{r}(\rho\to\sigma)} \lambda x : \mathbf{r}(\rho).\, px && \text{(By I.H.)} \\
&=_{\mathbf{r}(\rho\to\sigma)} p
\end{aligned}$$

- $\rho \equiv \rho \times \sigma$.

$$\begin{aligned}
&\mathrm{Im}^-_{\rho\times\sigma}\, \mathrm{id}_{\rho\times\sigma}\, p \\
&=_{\mathbf{r}(\rho\to\sigma)} \langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L\, \mathrm{id}_{\rho\times\sigma})\, \pi_0(p), \\
&\qquad\quad \mathrm{Im}^-_\sigma\,(\mathrm{pr}_R\, \mathrm{id}_{\rho\times\sigma})\, \pi_1(p)\rangle && \text{(By Definition 4.2.1)} \\
&=_{\mathbf{r}(\rho\to\sigma)} \langle \mathrm{Im}^-_\rho\, \mathrm{id}_\rho\, \pi_0(p), \mathrm{Im}^-_\sigma\, \mathrm{id}_\sigma\, \pi_1(p)\rangle && \text{(By Definition 4.2.4 and } \equiv_\beta) \\
&=_{\mathbf{r}(\rho\to\sigma)} \langle \pi_0(p), \pi_1(p)\rangle && \text{(By I.H.)} \\
&=_{\mathbf{r}(\rho\to\sigma)} p
\end{aligned}$$

□

Lemma 4.2.6 If $\Gamma \vdash A, B : \rho$, then $\vdash^{\mathbf{r}}_\Gamma \mathbf{R}(A \subseteq_\rho B)\mathrm{id}_\rho \leftrightarrow \mathbf{R}(A) \subseteq_{\mathbf{r}(\rho)} \mathbf{R}(B)$.

Proof. Immediate by Lemma 4.2.2 and 4.2.5. □

The following lemma is a part of the soundness theorem for induction and coinduction. It shows that the closure and coclosure rules are realisable.

Lemma 4.2.7 If $\Gamma \vdash \Phi : \rho \to \rho$ where ρ is a predicate type, then

(a) $\vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)} \mathbf{R}(\Phi(\mu_\rho \Phi) \subseteq_\rho \mu_\rho \Phi)\mathrm{id}_\rho$

(b) $\vdash^{\mathbf{r}}_{\mathbf{r}(\Gamma)} \mathbf{R}(\nu_\rho \Phi \subseteq_\rho \Phi(\nu_\rho \Phi))\mathrm{id}_\rho$

Proof. Assume $\Gamma \vdash \Phi : \rho \to \rho$, where ρ is a predicate type. Clearly $\mathbf{r}(\rho)$ is a predicate type as well. Furthermore, by Theorem 3.2.6, $\mathbf{r}(\Gamma) \vdash \mathbf{R}(\Phi) : \mathbf{r}(\rho) \to \mathbf{r}(\rho)$.

(a) By Lemma 4.2.6, it is to show $\mathbf{R}(\Phi)(\mu_{\mathbf{r}(\rho)}\mathbf{R}(\Phi)) \subseteq_{\mathbf{r}(\rho)} \mu_{\mathbf{r}(\rho)}\mathbf{R}(\Phi)$.

This follows by applying the closure rule $\mathrm{Cl}_{\mathbf{r}(\rho)}$ to $\mathbf{R}(\Phi)$.

(b) By Lemma 4.2.6, it is to show $\nu_{\mathbf{r}(\rho)}\mathbf{R}(\Phi) \subseteq_{\mathbf{r}(\rho)} \mathbf{R}(\Phi)(\nu_{\mathbf{r}(\rho)}\mathbf{R}(\Phi))$.

This follows by applying the coclosure rule $\mathrm{Cocl}_{\mathbf{r}(\rho)}$ to $\mathbf{R}(\Phi)$.

□

Note that this lemma states that closure and coclosure rules are realised by the identity. As pointed out in Section 3.3, the realiser of the coclosure rule given in Tatsuta's theory of $\mathrm{TID}_{\nu 2}$ [Tat98] is more complicated ($\lambda \vec{x} r.r(\lambda p.m(\lambda \vec{x} r s.s(p_0, r))\vec{x}(p_0 \vec{x} p_1))$ on page 353).

We give the definition of composition operations which will be used later in the soundness proof for induction and coinduction.

Definition 4.2.8 We define *composition operations* $\mathbf{o}_\rho : \delta \to \delta \to \delta$ for every predicate type ρ by

$$\begin{aligned}
&\mathbf{o}_o := \lambda d, e : \delta.\mathrm{fun}(\lambda a : \delta.d \cdot (e \cdot a)) \\
&\mathbf{o}_{\rho \to \sigma} := \mathbf{o}_\sigma \\
&\mathbf{o}_{\rho \times \sigma} := \lambda d, e : \delta.\mathrm{pair}((\mathrm{pr}_L d)\,\mathbf{o}_\rho\,(\mathrm{pr}_L e), (\mathrm{pr}_R d)\,\mathbf{o}_\sigma\,(\mathrm{pr}_R e))
\end{aligned}$$

Note that we use infix notation for representing composition operations.
We also set $\approx_\rho := \lambda x, y : \rho.x \subseteq_\rho y \wedge y \subseteq_\rho x$.

Lemma 4.2.9 In RCST the following are provable:

(a) $\mathrm{Im}_\rho\, d\,(\mathrm{Im}_\rho\, e x) \approx_{\mathbf{r}(\rho)} \mathrm{Im}_\rho\,(d\,\mathbf{o}_\rho\, e)x$

(b) $\mathrm{Im}^-_\rho\, d\,(\mathrm{Im}^-_\rho\, e x) =_{\mathbf{r}(\rho)} \mathrm{Im}^-_\rho\,(e\,\mathbf{o}_\rho\, d)x$

Proof. By induction on ρ.

- $\rho \equiv o$.

  (*a*) $\mathrm{Im}_o\, d\, (\mathrm{Im}_o\, e\, x)$

  $=_{\mathbf{r}(o)} \lambda c : \delta.\exists b : \delta.\, (\mathrm{Im}_o\, e\, x)\, b \wedge d \cdot b =_\delta c$ (By Definition 4.2.1)

  $=_{\mathbf{r}(o)} \lambda c : \delta.\exists b : \delta.\exists a : \delta.\, x\, a \wedge e \cdot a =_\delta b \wedge d \cdot b =_\delta c$ (By Definition 4.2.1)

  $\approx_{\mathbf{r}(o)} \lambda c : \delta.\exists b : \delta.\exists a : \delta.\, x\, a \wedge d \cdot (e \cdot a) =_\delta c$ (By $\equiv_\delta$)

  $=_{\mathbf{r}(o)} \lambda c : \delta.\exists b : \delta.\exists a : \delta.\, x\, a \wedge (\mathrm{fun}(\lambda a : \delta.\, d \cdot (e \cdot a))) \cdot a =_\delta c$ (By $\equiv_\beta$)

  $=_{\mathbf{r}(o)} \lambda c : \delta.\exists b : \delta.\exists a : \delta.\, x\, a \wedge (d\, \mathbf{o}_o\, e) \cdot a =_\delta c$ (By Definition 4.2.8)

  $=_{\mathbf{r}(o)} \mathrm{Im}_o\, (d\, \mathbf{o}_o\, e)\, x$ (By Definition 4.2.1)

  (*b*) $\mathrm{Im}^-_o\, d\, (\mathrm{Im}^-_o\, e\, x)$

  $=_{\mathbf{r}(o)} \lambda a : \delta.\, \mathrm{Im}^-_o\, e\, x\, (d \cdot a)$ (By Definition 4.2.1)

  $=_{\mathbf{r}(o)} \lambda a : \delta.\, x\, (e \cdot (d \cdot a))$ (By Definition 4.2.1)

  $=_{\mathbf{r}(o)} \lambda a : \delta.\, x\, (\mathrm{fun}(\lambda a : \delta.\, e \cdot (d \cdot a))) \cdot a$ (By $\equiv_\beta$)

  $=_{\mathbf{r}(o)} \lambda a : \delta.\, x\, ((e\, \mathbf{o}_o\, d) \cdot a)$ (By Definition 4.2.8)

  $=_{\mathbf{r}(o)} \mathrm{Im}^-_o\, (e\, \mathbf{o}_o\, d)\, x$ (By Definition 4.2.1)

- $\rho \equiv \rho \to \sigma$.

  (*a*) $\mathrm{Im}_{\rho\to\sigma}\, d\, (\mathrm{Im}_{\rho\to\sigma}\, e\, x)$

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}_\sigma\, d\, ((\mathrm{Im}_{\rho\to\sigma}\, e\, x)\, y)$ (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}_\sigma\, d\, (\mathrm{Im}_\sigma\, e\, (x\, y))$ (By Definition 4.2.1)

  $\approx_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}_\sigma\, (d\, \mathbf{o}_\sigma\, e)\, (x\, y)$ (By I.H.)

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}_\sigma\, (d\, \mathbf{o}_{\rho\to\sigma}\, e)\, (x\, y)$ (By Definition 4.2.8)

  $=_{\mathbf{r}(\rho\to\sigma)} \mathrm{Im}_{\rho\to\sigma}\, (d\, \mathbf{o}_{\rho\to\sigma}\, e)\, x$ (By Definition 4.2.1)

  (*b*) $\mathrm{Im}^-_{\rho\to\sigma}\, d\, (\mathrm{Im}^-_{\rho\to\sigma}\, e\, x)$

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, d\, ((\mathrm{Im}^-_{\rho\to\sigma}\, e\, x)\, y)$ (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, d\, (\mathrm{Im}^-_\sigma\, e\, (x\, y))$ (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, (e\, \mathbf{o}_\sigma\, d)\, (x\, y)$ (By I.H.)

  $=_{\mathbf{r}(\rho\to\sigma)} \lambda y : \mathbf{r}(\rho).\, \mathrm{Im}^-_\sigma\, (e\, \mathbf{o}_{\rho\to\sigma}\, d)\, (x\, y)$ (By Definition 4.2.8)

  $=_{\mathbf{r}(\rho\to\sigma)} \mathrm{Im}^-_{\rho\to\sigma}\, (e\, \mathbf{o}_{\rho\to\sigma}\, d)\, x$ (By Definition 4.2.1)

- $\rho \equiv \rho \times \sigma$.

  (*a*) $\mathrm{Im}_{\rho\times\sigma}\, d\,(\mathrm{Im}_{\rho\times\sigma}\, e x)$

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}_\rho\,(\mathrm{pr}_L d)\,\pi_0(\mathrm{Im}_{\rho\times\sigma}\, e x),$
  $\mathrm{Im}_\sigma\,(\mathrm{pr}_R d)\,\pi_1(\mathrm{Im}_{\rho\times\sigma}\, e x)\rangle$ (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}_\rho\,(\mathrm{pr}_L d)\,\pi_0(\langle \mathrm{Im}_\rho\,(\mathrm{pr}_L e)\,\pi_0(x), \mathrm{Im}_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x)\rangle),$
  $\mathrm{Im}_\sigma\,(\mathrm{pr}_R d)\,\pi_1(\langle \mathrm{Im}_\rho\,(\mathrm{pr}_L e)\,\pi_0(x), \mathrm{Im}_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x)\rangle)\rangle$
  (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}_\rho\,(\mathrm{pr}_L d)\,(\mathrm{Im}_\rho\,(\mathrm{pr}_L e)\,\pi_0(x)),$
  $\mathrm{Im}_\sigma\,(\mathrm{pr}_R d)\,(\mathrm{Im}_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x))\rangle$ (By $\equiv_\beta$)

  $\approx_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}_\rho\,((\mathrm{pr}_L d)\,\mathbf{o}_\rho\,(\mathrm{pr}_L e))\,\pi_0(x),$
  $\mathrm{Im}_\sigma\,((\mathrm{pr}_R d)\,\mathbf{o}_\sigma\,(\mathrm{pr}_R e))\,\pi_1(x)\rangle$ (By I.H.)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}_\rho\,\mathrm{pr}_L(d\,\mathbf{o}_{\rho\times\sigma}\, e)\,\pi_0(x),$
  $\mathrm{Im}_\sigma\,\mathrm{pr}_R(d\,\mathbf{o}_{\rho\times\sigma}\, e)\,\pi_1(x)\rangle$ (By Definition 4.2.8 and $\equiv_\beta$)

  $=_{\mathbf{r}(\rho\times\sigma)} \mathrm{Im}_{\rho\times\sigma}\,(d\,\mathbf{o}_{\rho\times\sigma}\, e)\,x$ (By Definition 4.2.1)

  (*b*) $\mathrm{Im}^-_{\rho\times\sigma}\, d\,(\mathrm{Im}^-_{\rho\times\sigma}\, e x)$

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L d)\,\pi_0(\mathrm{Im}^-_{\rho\times\sigma}\, e x),$
  $\mathrm{Im}^-_\sigma\,(\mathrm{pr}_R d)\,\pi_1(\mathrm{Im}^-_{\rho\times\sigma}\, e x)\rangle$ (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L d)\,\pi_0(\langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L e)\,\pi_0(x), \mathrm{Im}^-_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x)\rangle),$
  $\mathrm{Im}^-_\sigma\,(\mathrm{pr}_R d)\,\pi_1(\langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L e)\,\pi_0(x), \mathrm{Im}^-_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x)\rangle)\rangle$
  (By Definition 4.2.1)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}^-_\rho\,(\mathrm{pr}_L d)\,(\mathrm{Im}^-_\rho\,(\mathrm{pr}_L e)\,\pi_0(x)),$
  $\mathrm{Im}^-_\sigma\,(\mathrm{pr}_R d)\,(\mathrm{Im}^-_\sigma\,(\mathrm{pr}_R e)\,\pi_1(x))\rangle$ (By $\equiv_\beta$)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}^-_\rho\,((\mathrm{pr}_L e)\,\mathbf{o}_\rho\,(\mathrm{pr}_L d))\,\pi_0(x),$
  $\mathrm{Im}^-_\sigma\,((\mathrm{pr}_R e)\,\mathbf{o}_\sigma\,(\mathrm{pr}_R d))\,\pi_1(x)\rangle$ (By I.H.)

  $=_{\mathbf{r}(\rho\times\sigma)} \langle \mathrm{Im}^-_\rho\,\mathrm{pr}_L(e\,\mathbf{o}_{\rho\times\sigma}\, d)\,\pi_0(x),$
  $\mathrm{Im}^-_\sigma\,\mathrm{pr}_R(e\,\mathbf{o}_{\rho\times\sigma}\, d)\,\pi_1(x)\rangle$ (By Definition 4.2.8 and $\equiv_\beta$)

  $=_{\mathbf{r}(\rho\times\sigma)} \mathrm{Im}^-_{\rho\times\sigma}\,(e\,\mathbf{o}_{\rho\times\sigma}\, d)\,x$ (By Definition 4.2.1)

  □

Theorem 4.2.10 Monotone induction is realised by the term

$$\mathbf{I}_\rho := \lambda m, s : \delta.\mathrm{rec}(\lambda f : \delta.s\,\mathbf{o}_\rho\,(m \cdot f))$$

for every predicate type ρ.

Proof. It suffices to show that the closed term $\mathbf{I}_\rho$ realises the formula

$$\forall \Phi : \mathbf{r}(\rho) \to \mathbf{r}(\rho).\forall m : \delta.\mathbf{R}(\mathrm{mono}_\rho \Phi)m \to$$
$$\forall \widetilde{M} : \mathbf{r}(\rho).\forall s : \delta.\mathbf{R}(\Phi(M) \subseteq_\rho M)s \to \mathbf{R}(\mu_\rho \Phi \subseteq_\rho M)(\mathbf{I}_\rho \cdot m \cdot s).$$

Hence, we assume $\mathbf{R}(\mathrm{mono}_\rho \Phi)m$ and $\mathbf{R}(\Phi(M) \subseteq_\rho M)s$.

By Lemma 4.2.2, from the assumptions we have

(a) $\forall \widetilde{A}, \widetilde{B} : \mathbf{r}(\rho).\forall a : \delta.\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- a \widetilde{B} \to \Phi(\widetilde{A}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- (m \cdot a)(\Phi(\widetilde{B}))$

(b) $\Phi(\widetilde{M}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- s \widetilde{M}$

Let $f := \mathrm{rec}(\lambda f : \delta.s\,\mathbf{o}_\rho\,(m \cdot f))$, by Lemma 4.2.2, it is to show $\mu_\rho \Phi \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- f \widetilde{M}$.

We will use the general induction rule IndG_ρ to prove, so we assume

(c) $\widetilde{Y} \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- f \widetilde{M}$.

We have to show $\Phi(\widetilde{Y}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- f \widetilde{M}$.

From assumptions (c) and (a) we have $\Phi(\widetilde{Y}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- (m \cdot f)(\Phi(\widetilde{M}))$.

By assumption (b) and Lemma 4.2.3 (b) we get

$$\mathrm{Im}_\rho^- (m \cdot f)(\Phi(\widetilde{M})) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- (m \cdot f)(\mathrm{Im}_\rho^- s \widetilde{M}).$$

By Lemma 4.2.9 (b) we get

$$\mathrm{Im}_\rho^- (m \cdot f)(\mathrm{Im}_\rho^- s \widetilde{M}) =_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- (s\,\mathbf{o}_\rho\,(m \cdot f))\widetilde{M}.$$

Applying transitivity of $\subseteq_{\mathbf{r}(\rho)}$, we have

$$\Phi(\widetilde{Y}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- (s\,\mathbf{o}_\rho\,(m \cdot f))\widetilde{M}.$$

Since $f =_\delta s\,\mathbf{o}_\rho\,(m \cdot f)$, we have $\Phi(\widetilde{Y}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho^- f \widetilde{M}$. □

Theorem 4.2.11 Monotone coinduction is realised by the term

$$\mathbf{I}_\rho^{\mathbf{co}} := \lambda m, s : \delta.\mathrm{rec}(\lambda f : \delta.(m \cdot f)\,\mathbf{o}_\rho\,s)$$

for every predicate type ρ.

Proof. It suffices to show that the closed term $\mathbf{I}_\rho^{\mathbf{co}}$ realises the formula

$$\forall \Phi : \mathbf{r}(\rho) \to \mathbf{r}(\rho).\forall m : \delta.\mathbf{R}(\mathrm{mono}_\rho \Phi)m \to$$
$$\forall \widetilde{M} : \mathbf{r}(\rho).\forall s : \delta.\mathbf{R}(M \subseteq_\rho \Phi(M))s \to \mathbf{R}(M \subseteq_\rho \nu_\rho \Phi)(\mathbf{I}_\rho^{\mathbf{co}} \cdot m \cdot s)$$

Hence, we assume $\mathbf{R}(\mathrm{mono}_\rho \Phi)m$ and $\mathbf{R}(M \subseteq_\rho \Phi(M))s$.

By Lemma 4.2.2, from the assumptions we have

(a) $\forall \widetilde{A},\widetilde{B} : \mathbf{r}(\rho).\forall a : \delta.\mathrm{Im}_\rho\, a\widetilde{A} \subseteq_{\mathbf{r}(\rho)} \widetilde{B} \to \mathrm{Im}_\rho\,(m\cdot a)\,(\Phi(\widetilde{B})) \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{A})$

(b) $\mathrm{Im}_\rho\, s\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{M})$

Let $f := \mathbf{I}^{\mathbf{co}}_\rho \cdot m \cdot s$, by Lemma 4.2.2, it is to show $\mathrm{Im}_\rho\, f\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \nu_\rho\Phi$.

We will use the general coinduction rule CoiG_ρ to prove, so we assume

(c) $\mathrm{Im}_\rho\, f\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \widetilde{Y}$

We have to show $\mathrm{Im}_\rho\, f\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{Y})$.

From assumptions (c) and (a) we have $\mathrm{Im}_\rho\,(m\cdot f)\,(\Phi(\widetilde{M})) \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{Y})$.

By assumption (b) and Lemma 4.2.3 (a) we get

$$\mathrm{Im}_\rho\,(m\cdot f)\,(\mathrm{Im}_\rho\, s\widetilde{M}) \subseteq_{\mathbf{r}(\rho)} \mathrm{Im}_\rho\,(m\cdot f)\,(\Phi(\widetilde{M}))$$

By Lemma 4.2.9 (a) we get

$$\mathrm{Im}_\rho\,(m\cdot f)\,(\mathrm{Im}_\rho\, s\widetilde{M}) \approx_{\mathbf{r}(\rho)} \mathrm{Im}_\rho\,((m\cdot f)\,\mathbf{o}_\rho\, s)\widetilde{M}$$

Applying transitivity of $\subseteq_{\mathbf{r}(\rho)}$, we have

$$\mathrm{Im}_\rho\,((m\cdot f)\,\mathbf{o}_\rho\, s)\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{Y})$$

Since $f =_\delta (m\cdot f)\,\mathbf{o}_\rho\, s$, we have $\mathrm{Im}_\rho\, f\widetilde{M} \subseteq_{\mathbf{r}(\rho)} \Phi(\widetilde{Y})$. □

In the following two examples, for better reading, we re-state the details of previous examples.

Example 4.2.12 (Extracting Program for Addition)

Consider the typing context $\Gamma_+ := \{0,1 : \iota, + : \iota \to \iota \to \iota, - : \iota \to \iota\}$ and let Δ_+ consist of the formulas stating that $(+,-,0)$ is an Abelian group. Formally, the terms $0,1,+,-$ are variables, but we like to view them as constants and the assumptions Δ_+ as axioms. We will write $M-N$ as a shorthand for $M+(-N)$.

We define the set of natural numbers as an inductive predicate. Set $\Phi := \lambda p : \iota \to o.\lambda x : \iota.x =_\iota 0 \vee p(x-1)$, and define $\mathbb{N} := \mu_{\iota\to o}\Phi$. A more readable notation for the definition of $\mathbb{N}$ would be

$$\mathbb{N}x \overset{\mu}{=} x =_\iota 0 \vee \mathbb{N}(x-1).$$

As an example of a proof by induction we show that the natural numbers are closed under addition:

$$\Delta_+ \vdash_{\Gamma_+} \forall x,y : \iota.\mathbb{N}x \wedge \mathbb{N}y \to \mathbb{N}(x+y)$$

Setting $P(x) := \lambda y : \iota.\mathbb{N}(x+y)$, the formula to be proven is equivalent to $\forall x : \iota.\mathbb{N}x \to \mathbb{N} \subseteq_{\iota\to o} P(x)$. Hence, it suffices to show $\Phi P(x) \subseteq_{\iota\to o} P(x)$ under the extra assumption $\mathbb{N}x$. Unfolding the definition of Φ and using proof by cases ($\vee^-$), this amounts

to proving $\mathbb{N}(x+0)$ and $\forall y:\iota.\mathbb{N}(x+(y-1)) \to \mathbb{N}(x+y)$, which is easy, given the assumptions Δ_+ and $\mathbb{N}x$.

The realisability interpretation of natural numbers is

$$\mathbf{R}(\mathbb{N}x)d \equiv \mu_{\iota\to\delta\to o}\mathbf{R}(\Phi)xd$$

where $\mathbf{R}(\Phi) \equiv \lambda\widetilde{p}:\iota\to\delta\to o.\lambda x:\iota.\lambda d:\delta.(d =_\delta \mathrm{in}_L\mathrm{nil}\wedge x =_\iota 0)\vee(\exists b:\delta.d =_\delta \mathrm{in}_R b\wedge\widetilde{p}(x-1)b)$.

In a more readable notation:

$$\mathbf{R}(\mathbb{N}x)d \stackrel{\mu}{=} (d =_\delta \mathrm{in}_L\mathrm{nil}\wedge x=0)\vee(\exists b:\delta.d =_\delta \mathrm{in}_R b\wedge\mathbf{R}(\mathbb{N}(x-1))b).$$

Hence, a natural number $n:\iota$ is realised by the numeral $\underline{n}:\delta$, where $\underline{0} := \mathrm{in}_L\mathrm{nil}$, $\underline{n+1} := \mathrm{in}_R\underline{n}$.

An element $d:\delta$ realises the closure of natural numbers under addition, i.e. the formula $\forall x,y:\iota.\mathbb{N}x\wedge\mathbb{N}y\to\mathbb{N}(x+y)$, if for all $x,y:\iota$ and all $a,b:\delta$

$$\mathbf{R}(\mathbb{N}x)a\wedge\mathbf{R}(\mathbb{N}y)b\to\mathbf{R}(\mathbb{N}(x+y))(d\cdot(\mathrm{pair}ab)).$$

which says that d adds natural numbers in unary notation.

We define adding n for a fixed n as

$$\mathrm{add}[n] := \mathrm{rec}(\lambda_\delta f.\lambda_\delta m.\mathrm{case}m(\lambda_\delta a.n)(\lambda_\delta m'.\mathrm{in}_R(f\cdot m'))).$$

Thus, the base and step cases are $\mathrm{add}[n](\mathrm{in}_L\mathrm{nil}) \equiv n$ and $\mathrm{add}[n](\mathrm{in}_R m) \equiv \mathrm{in}_R(\mathrm{add}[n]m)$ respectively. Therefore, the addition program (the realiser) is

$$\mathrm{add} := \lambda_\delta p.\mathrm{add}[\mathrm{pr}_L p]\cdot(\mathrm{pr}_R p).$$

Example 4.2.13 (Extracting Program for Fibonacci Numbers)

Continuing the previous example, we give a definition of a coinductive predicate $\mathrm{FIB} := \nu_{\iota\to\iota\to o}\Psi$ where

$$\Psi := \lambda q:\iota\to\iota\to o.\lambda x,y:\iota.\mathbb{N}x\wedge qy(x+y).$$

This becomes more readable if a similar notation as for $\mathbb{N}$ is used:

$$\mathrm{FIB}xy \stackrel{\nu}{=} \mathbb{N}x\wedge\mathrm{FIB}y(x+y)$$

Informally, $\mathrm{FIB}xy$ states that there exists a Fibonacci sequence of natural numbers starting with x,y.

As an example of a proof that uses coinduction we show

$$\Delta_+ \vdash_{\Gamma_+} \mathrm{FIB}\,1\,1.$$

We show more generally, that FIB holds for any two natural numbers, i.e. $Q \subseteq_{\iota\to\iota\to o}$ FIB where $Q := \lambda x,y:\iota.\mathbb{N}x \wedge \mathbb{N}y$. By coinduction, it suffiices to show $Q \subseteq_{\iota\to\iota\to o} \Psi Q$, which is easily done using the previously proven fact thatt natural numbers are closed under addition.

The interpretation of Fibonacci numbers is

$$\mathbf{R}(\mathrm{FIB}\,x y)\,d \equiv \nu_{\iota\to\iota\to\delta\to o}\mathbf{R}(\Psi)\,x\,y\,d$$

where $\mathbf{R}(\Psi) \equiv \lambda\widetilde{q}:\iota\to\iota\to\delta\to o.\lambda x,y:\iota.\lambda d:\delta.\exists a,b:\delta.d =_\delta \mathrm{pair}\,a\,b \wedge \mathbf{R}(\mathbb{N}x)\,a \wedge \widetilde{q}\,y\,(x+y)\,b$.

In more readable notation

$$\mathbf{R}(\mathrm{FIB}\,x y)\,d \overset{\nu}{=} \mathbf{R}(\mathbb{N}x)(\mathrm{pr}_L\,d) \wedge \mathbf{R}(\mathrm{FIB}\,y\,(x+y))(\mathrm{pr}_R\,d).$$

which says that d is a stream of natural numbers in unary notation where the head realises $\mathbb{N}x$ and the tail realises $\mathrm{FIB}\,y\,(x+y)$.

The Fibonacci program realising that FIB contains all pairs of natural numbers is

$$\mathrm{Fib} := \mathrm{rec}(\lambda_\delta f.\lambda_\delta p.\mathrm{pair}\,(\mathrm{pr}_L\,p)\,(f\cdot(\mathrm{pair}\,(\mathrm{pr}_R\,p)\,(\mathrm{add}\,p)))).$$

4.3 Related Work

Coq is a proof assistant with the program extraction mechanism for higher-order logic. In 1985, Coquand introduced the first version of a logical system, called the *Calculus of Constructions* [CH88, Coq85]. There is one drawback to this approach: it is impossible to give direct inductive definitions. Later in 1989 Coquand and Paulin-Mohring extended it with primitive inductive definitions. As a result, the *Calculus of Inductive Constructions*, a higher-order typed lambda calculus with dependent types, is used by Coq as the logical language. It is in fact a pure type system with subtyping, and has Martin-Löf-style inductive definitions. Coq 8 is based on a weaker calculus called the *Predicative Calculus of Inductive Constructions* [BCHPM04] by making the universe (or sort) `Set` become predicative in order to be compatible with classical choice.

Paulin-Mohring [PM89b, PM89a] provided a realisability interpretation, in the sense of Kleene, for the Calculus of Constructions and proved the correctness of extracted programs. As the system grew up and several limitations popped up, Letouzey completely redesigned Coq's extraction mechanism to hamdle any Coq terms, ensure the correctness of the extraction, and guarantee that the extracted terms produced are typable. It assigns different types to terms representing data and terms representing properties of the data. Since the latter cannot be used in the definition of data, a.k.a. *computationally irrelevant*, there is no need to extract them. It is very important to remove the logical parts in proofs with regard to the size of the extracted program and the

speed of implementing the extraction. Miquell [Miq07] presented another extraction mechanism of Coq based on Krivine's realisability model of classical second-order arithmetic.

Following Paulin-Mohring's work, based on Krivine's realisability theory [Kri93, KP90] and the framework of Pure Type System [Bar91, Bar92], Bernardy and Lasson [BL11] constructed a logic from the programming language of its realisers with syntactic definitions of parametricity and realisability.

Berghofer [Ber03b] presented a generic framework for program extraction and instantiated it to Isabelle/HOL, aiming at demonstrating that Isabelle is suitable as a basis for program extraction. He also showed applications of a program extraction framework in Isabelle/HOL to realistic examples including the induction principle for natural numbers. However, as Isabelle is based on a classical logic, only the constructive parts of terms are considered in the extraction.

The underlying logic of the Minlog system is the Theory of Computable Functionals (TCF) [Sch93, SW12]. The difference between TCF and other type theories lies in that TCF emphasises partial continuous functionals. Minlog treats computable functionals as constants, and infinitary free algebras as base types. Not only ML-style type parameters, but also predicative, predicate variables for comprehension terms are allowed. In Minlog, terms with same normal forms are identical. Therefore, Minlog supports inductive and coinductive definitions and program extractions from classical and constructive proofs. Hou [Hou06] implemented a coinductive proof for the correctness of a corecursive program for the average function with regard to the signed digit stream representation in Minlog. Berger, Miyamoto, Schwichtenberg and Seisenberger gave an overview of Minlog and explained its program extraction mechanism in [BMSS11]. The program extraction procedure of Minlog is based on Kreisel's modified realisability, which from every constructive proof M of a formula A with computational content a term $[\![M]\!]$ can be extracted to "realise" A. Other program extraction methods like Dialectica Interpretation have been implemented in Minlog. Applications of program extraction from proofs via reallisability, including exact real numbers, are presented in [Ber09] and [BS12]. In addition, Miyamoto, Nordvall Forsberg and Schwichtenberg extended Minlog to support strictly positive nested inductive and coinductive definitions in [MNFS13].

Since the Nuprl system is based on constructive type theory, a proven correct program can be extracted from a proof of its specification (theorem statements). Nuprl can formalise and verify induction principles as lemmas within the type theory that yield extracts that are recursion schemes optimised for both readability and efficiency. Caldwell presented the extraction of "efficient" recursion schemes from proofs of well-founded induction principles in [Cal02].

4.4 Conclusion

We have studied a realisability interpretation of an intuitionistic version of Church's Simple Theory of Types described in Chapter 3.

The Soundness Theorem of realisability is split into three steps: First one shows that from a formal proof of a formula A one can extract a lambda-term and a formal proof that it realises A. Then one shows that formally proven formulas are true in a domain-theoretic model. Finally, one shows that for so-called Σ-formulas A, which are formulas whose potential realisers contain observable information (essentially finite lists, or trees), if it holds in the domain-theoretic model that a lambda-term M realises A, then M reduces w.r.t. a lazy operational semantics to a canonical term (e.g. numeral) that realises A (computational adequacy).

In this dissertation we have carried out only the first two steps. The last step will be an interesting future work. We will sketch how our realisability interpretation can be used to extract programs from proofs. Since most of the methods and results shown later can be taken over from [Ber10], we will be rather brief and will only comment in detail on the changes and additions necessary.

First, we define two classes of formulas.

Definition 4.4.1 (Non-computational formula) A Γ-formula A is called *non-computational* if it does not contain the constants $\vee$, ν_ρ, μ_ρ, and for all occurrences of $\forall_\rho$, $\exists_\rho$ in A and $x:\rho \in \Gamma$, ρ is an *object-type*, that is, ρ contains only base types from $\mathscr{S}$.

Definition 4.4.2 (Σ-formula) A Γ-formula A is called *Σ-formula* if (a) it does not contain the constants $\rightarrow$ and ν_ρ, (b) for all occurrences of $\forall_\rho$, $\exists_\rho$, in A and $x:\rho \in \Gamma$, ρ is an *object-type*, and (c) for all occurrences of μ_ρ, ρ is of the form $\sigma \rightarrow o$ where σ is an object type.

The pathway to extracted programs is now as follows:

(1) Given a proof of a sequent $\Delta \vdash_\Gamma A$ in CST we extract, using the Soundness Theorem (Theorem 4.1.7), a closed term M provably realising the sequent.

(2) If the formulas in Δ are non-computational, then the formula $\bigwedge \Delta$ is equivalent to the assertion that it is realised by some trivial term nil' built from nil by pairing and (dummy) lambda-abstraction. Hence we can derive $\Delta \vdash_\Gamma \mathbf{R}(M \cdot \mathrm{nil}')A$.

(3) Let P be the β-normal form of $M \cdot \mathrm{nil}'$. P will be closed and contain only program constants. Let us call such a term P a *program*.

(4) If the assumptions Δ are true in some standard classical model (see Section 3.3), then in that model the value $[\![P]\!]$ of P will realise A.

(5) If A is a Σ-formula, then $[\![P]\!]$ will be a data, that is a finite combinatorial object built from nil by left and right injections and pairing (numerals are examples of data). We can identify a data with the canonical program built from $\mathrm{nil}, \mathrm{in}_L, \mathrm{in}_R$, pair defining it.

(6) Now we can employ a *Computational Adequacy Theorem* ([Ber10] Theorem 11) according to which a program P denoting data reduces to that data in a suitable lazy big-step semantics.

(7) Since the big-step semantics is denotationally correct we know that the data $[\![P]\!]$ realises A and can be computed from P.

The steps (1-7) amount to a proof of the following theorem:

Theorem 4.4.3 (Program extraction for data) Let $\Gamma \vdash \Delta : o$ be a finite set of non-computational formulas. Then from a proof of $\Delta \vdash_\Gamma A$, where A is a Σ-formula, one can extract a program P with the property that P reduces to some data realising A. Furthermore, a proof that Δ implies that d realises A is extracted.

Theorem 4.4.3 can be easily generalised to the situation where the proven formula is an implication between Σ-formulas:

Theorem 4.4.4 (Program extraction for data functions) Let Δ be a finite set of non-computational formulas. Then from a proof of $\Delta \vdash_\Gamma A \to B$, where A and B are Σ-formulas, one can extract a program P with the property that for any data d realising A, $P \cdot d$ reduces to some data realising B. Furthermore, a proof of $\Delta, \mathrm{Data}\, d \vdash^r_{\Gamma, d:\delta} \mathbf{R}(A)\, d \to \mathbf{R}(B)\,(P \cdot d)$ is extracted, where $\mathrm{Data} : \delta \to o$ is a closed term defining of the property of being data.

Remark on applications. Theorems 4.4.3 and 4.4.4 cover most applications of program extraction, but they can be generalised to considerably larger classes of formulas. This will be the subject of further work.

Example 3.3.15 is covered by Theorem 4.4.4 since $\mathbb{N}x \wedge \mathbb{N}y \to \mathbb{N}(x+y)$ is an implication between Σ-formulas.

The program extracted from Example 3.3.16 is an infinite stream (see Example 4.2.13). In order to compute with it we need to extract finite data from that stream. For example, we can access its n-th element. To this end we define inductively (using slightly informal notation)

$$\mathrm{FIB}'\, x\, y\, z \stackrel{\mu}{=} (z =_\iota 0 \wedge \mathbb{N}x) \vee \mathrm{FIB}'\, y\, (x+y)\, (z-1)$$

and prove $\forall z : \iota\,.\, \mathbb{N}z \to \forall x, y : \iota\,.\, \mathrm{FIB}\, x\, y \to \mathrm{FIB}'\, x\, y\, z$ by induction on $\mathbb{N}$. Combining this with Example 3.3.16 we obtain $\forall z : \iota\,.\, \mathbb{N}z \to \mathrm{FIB}'\, 1\, 1\, z$, which is covered by Theorem 4.4.4. The extracted program would, given a natural number n in unary notation, read off the nth element of the infinite stream of Fibonacci numbers. Note that memoization takes place here: if we compute first the 100th Fibonacci number, then 100 additions of natural numbers will be carried out, but if we later compute the 99th number, the (partially computed) stream will only be looked up without performing any additions. We could of course prove directly (without detour through the coinduc-