Tel-Aviv University
Raymond and Beverly Sackler Faculty of Exact Sciences
School of Mathematical Sciences
Department of Pure Mathematics

Formalizing Mathematics via Predicative and Constructive Approaches

Thesis submitted
for the degree of Doctor of Philosophy

by

Liron Cohen

This work was carried out under the supervision of
Prof. Arnon Avron
and
Prof. Yoram Hirshfeld

Submitted to the Senate of Tel-Aviv University
June 2016

Acknowledgments

First and foremost, I wish to express my sincere gratitude to my mentor and supervisor Prof. Arnon Avron. Thank you for your uncompromising work standards, and for the endless hours during week days and weekends of reading and then rereading this thesis. Thank you for teaching me how to think, how to criticize my work, and how to keep finding new ideas, while constantly doubting myself in the quest to fight good with excellent. Your support along the course of this research is not taken for granted. I was proud to be your student, and I will take with me many of your unsettling principles into my future academic career.

This thesis is also a result of the work under the secondary supervision of Prof. Yoram Hirshfeld. Thank you Yoram for your insightful comments. I would also like to thank Prof. Nachum Dershowitz and Prof. Alexander Rabinovich for valuable feedback and many fruitful suggestions during the logic seminar and beyond it.

I am also thankful to Prof. Robert Lee Constable, a joint work with whom has led to some results which appear in this thesis, and to many other interesting research directions. I am grateful for your constant support and encouragement and for your kind guidance. Working with you was tremendously enriching, as well as a great delight.

I want to also thank my office-mates and my colleagues in teaching for their support, advice and friendship.

Finally, I would like to thank my dear family. I am truly grateful to my parents, Gavriel and Ester, for their constant support and encouragement during my studies. Your endless endeavors to provide me the best form of education, even in my early childhood, is what has gotten me this far. I would like to especially thank Barak, my husband, for always encouraging me to look ahead during challenging times, and for celebrating with me each and every milestone, as small as it may have been. Thank you for believing in me and being there for me, always. This thesis could not have happened without your endless support. Last but not least, I want to thank my daughter Gai, for simultaneously slowing me down and keeping me going.

This thesis is dedicated to my grandmother, Supti.

Abstract

Formalized mathematics and mathematical knowledge management (MKM) are extremely fruitful and quickly expanding fields of research at the intersection of mathematics and computer science. The declared goal of these fields is to develop computerized systems that effectively represent all important mathematical knowledge and techniques, while conforming to the highest standards of mathematical rigor. The use of proof assistants or interactive theorem provers has seen tremendous growth in recent years because of its record in assisting with the development and verification of formal proofs by human-machine collaboration. It has also found numerous high value applications in different research areas, including cyber security, cyber physical systems, correct-by-construction programming, and advanced programming language design. This rapidly developing field is bound to ultimately have a huge impact on the culture of mathematical practice and education.

Recent years, however, have seen an estrangement between the informal mathematical practice and the mainstream work in the field of MKM. Thus, while set theory is viewed by most mathematicians as the foundation of the mathematics they practice, this is not reflected in most extant proof assistants. We believe that the use for MKM of a set-theoretical frameworks could help overcome this increasing rift between what most mathematicians consider to be the basis of mathematics, and what existing formal-reasoning systems actually implement. Accordingly, this thesis aims at a formalization of applicable mathematics on the basis of a convenient formal set-theoretical framework which is suitable for mechanization and reflects real mathematical practice as presented in ordinary (or computationally-oriented) mathematical discourse. Another related goal of this thesis, motivated also by philosophical considerations, is to identify the minimal ontological commitments required for the above-mentioned task of the formalization of applicable mathematics. These (different, but related) goals can be simultaneously pursued by exploiting the modularity of the framework employed, which enables the use of different logics and set theories of different strength. Therefore, in this work we consider the following variations of the framework and of the theories developed within it:

- The underlying logic can be classical logic, or, in cases where the computational power of a theory should be enhanced, intuitionistic (constructive) logic.

- We take ancestral logic (the logic obtained by adding to first-order logic a transitive closure operator) to be the minimal underlying logic which is sufficient for the development of mathematics. We present and investigate from a proof theoretical point of view a very natural Gentzen-style proof system for ancestral logic, and prove its completeness with respect to a Henkin-style semantics. Then, motivated by the fact that this logic is also fundamental in many areas of computer science, we develop a constructive version of it which includes a natural proof system as well as a corresponding realizability semantics. The formal system is proven to be strongly sound with respect to this semantics in the sense that provable formulas are uniformly realizable. Then we turn to some applications of this logic in computer science, most notably Kleene Algebras with tests and program schemes.

- We adopt the predicative approach to mathematics in order to identify the most basic set theories which are truly indispensable for various levels of mathematical practice. This approach is arguably the most appropriate framework for developing computationally-oriented mathematics, while still being sufficient for scientifically applicable mathematics. A key property of the basic predicative theories we develop in this work is that they are *definitional*. That is, each of these theories has a minimal model, every element of which is definable by some closed term of the corresponding language. This allows for a very concrete, computationally-oriented interpretation.

Finally, we demonstrate the usefulness of the framework for the formalization of mathematics by developing in it large portions of scientifically applicable mathematics, focusing on analysis. We show that even on the first-order level, most of classical analysis can be carried out already within the minimal system of the framework and its minimal model. However, this development of mathematics involves coding, as well as treating the real line as a proper class. In contrast, using the minimal framework which is based on ancestral logic allows for a more natural and straightforward development of classical analysis.

Contents

1 Introduction **1**

1.1 Foundations and Formalizations of Mathematics 1

1.1.1 The Indispensable Logic 3

1.1.2 The Indispensable Set Theory 6

1.2 Thesis Outline 8

1.3 Notations 10

2 General Background **11**

2.1 The Intuitionistic Approach 11

2.2 The Predicative Approach 14

3 Ancestral Logic **19**

3.1 The Languages and their Semantics 20

3.2 Proof systems for AL 22

3.3 Arithmetics in AL 29

3.4 Henkin-Style Completeness 30

4 Constructive (Intuitionistic) Ancestral Logic **37**

4.1 Formal System for $iFOL$ 38

4.1.1 Realizability semantics for $iFOL$ 38

4.1.2 The Proof System $iFOL_S$ 39

4.1.3 Constructive (Intuitionistic) Metatheory 42

4.2 Formal System for iAL 44

4.2.1 Realizability Semantics for iAL 44

4.2.2 The Proof System iAL_S 45

4.2.3 Soundness for iAL_S 48

4.3 Applications of iAL . . . 49
4.3.1 Kleene Algebra . . . 49
4.3.2 Kleene Algebra with Tests . . . 52

5 The Predicative Framework **57**
5.1 Safety Relations . . . 58
5.2 The Languages and the Systems . . . 61
5.2.1 Languages . . . 61
5.2.2 Logics . . . 63
5.2.3 Axioms and Systems . . . 63
5.3 Universes . . . 68
5.4 Basic Set Theoretical Notions . . . 74
5.4.1 Standard Set Notations . . . 74
5.4.2 Classes . . . 77
5.4.3 Relations and Functions . . . 81
5.5 The Natural Numbers . . . 85
5.5.1 The Natural Numbers in RST_C^{FOL} . . . 85
5.5.2 The Natural Numbers in RST^{AL} . . . 89

6 Formalizing Analysis in the Minimal Frameworks **91**
6.1 The Minimal First-Order Framework . . . 92
6.1.1 The Minimal Model . . . 93
6.1.2 Real Analysis . . . 94
6.1.2.1 The Construction of the Real Line . . . 94
6.1.2.2 The Topology of the Reals . . . 97
6.1.2.3 Real Functions . . . 103
6.1.3 Going Beyond the Minimal Framework . . . 109
6.2 The Minimal AL Framework . . . 111

7 Summary and Further Work **115**

Bibliography **121**

Chapter 1

Introduction

1.1 Foundations and Formalizations of Mathematics

Formalized mathematics and mathematical knowledge management (MKM) are bound to ultimately have a huge impact on the culture of mathematical practice and education (e.g., correctness of proofs will be unquestionable, powerful results will be shown using huge non-human-readable proofs, etc.). These fields of study are very diverse and have many different (somewhat conflicting) approaches. The global approach aims at creating a unified framework which is sufficient for the formalization of the full-spectrum of mathematics, while the local approach deals with the implementation of concrete fractions of mathematics (sometimes in a rather ad-hoc manner). Unfortunately, the past decades have seen an increasing estrangement between the reality of informal mathematical practice and computer-implemented theorem proving. On one hand, type-free set theory is viewed by most mathematicians as the foundation of the mathematics they practice, and as such it is the most natural framework for MKM, especially for goals like those of the AUTOMATH project ([33, 84, 121]) and the QED manifesto ([17, 118]). On the other hand, most of the current work in the field of formalized mathematics and MKM is devoted to approaches and systems that are rather different from the set-theoretical ones. It either employs sophisticated type theories, with different notions of constructibility and computation that move more and more away from the common ground of "standard" mathematics and its

standard first-order foundations (like in Coq [18, 19, 20] or Nuprl [29, 30]), or it uses various fragments of higher-order logic (like in Isabelle/HOL [86]).[1]

In this work we address the above-mentioned anomaly by employing a user-friendly formal *set-theoretical* framework for formalizing mathematics which is suitable for mechanization and better reflects the way rigorous mathematics is (informally) presented in textbooks. The framework also settles to a great extent the existing tension between the global and the local approaches to the formalization of mathematics. This is because while it enables the use of set theories of different strength, it is very flexible and allows the use of it at different levels which are suitable for specific goals.

This work is also motivated by a philosophical approach to the foundations of mathematics. The well-known indispensability argument (propounded by Quine and Putnam, see e.g. [90, 91]) states that the indispensability of mathematics to empirical science gives us good reason to believe in the existence of mathematical entities. According to this line of argument, reference to mathematical entities, such as sets, numbers and functions, is indispensable to our best scientific theories, and so we ought to have ontological commitment to all these mathematical entities. This raises the following two questions:

- what exactly are those mathematical entities which are indispensable to scientific theories?
- what logical principles governing those mathematical entities are truly needed?

In keeping with this philosophical stance, as well as for practical reasons, this work identifies what we believe to be the minimal ontological commitments required for the formalization of applicable mathematics and uses the modularity of the framework to base the formal systems on them. This includes finding the minimal underlying logic which is sufficient for the development of mathematics, as well as recognizing the weakest possible set theories which are truly indispensable for various levels of mathematical practice. The minimality requirements naturally leads to a concrete variant of the system which is very suitable for mechanization. Nevertheless, it is clear that in order for the framework to be convenient for mechanical manipulations each of these minimal fragments (i.e. the underlying logic and the set theory) should also contain some computational power.

[1] Two notable exceptions are Mizar [96, 111] and Metamath [81].

1.1.1 The Indispensable Logic

Officially, most mathematicians accept that some first-order axiomatization of set theory (like Zermelo-Fraenkel set theory, ZF, or ZF with choice, ZFC) provides the standard foundations of mathematics. However, while first-order logic (FOL) clearly has many advantages (it is well studied and understood and it has many natural complete proof systems which are convenient for mechanization), it still seems to be unsuitable and too weak for this task. Thus, one cannot even give in it a categorical characterization of the most basic concept of mathematics - the natural numbers. Though FOL is too weak, we believe that choosing to use high-order logics (HOL) has many disadvantages too. First, they are based on debatable ontological commitments which are not universally accepted. Thus it does not seem satisfactory that dealing with basic notions (such as the natural numbers) requires the use of the strong notions involved in HOL, such as quantifying over all subsets of infinite sets. In fact, as claimed by Quine [93] with justice, higher-order "logics" are actually not *logics* at all, but set theories in disguise.[2] In addition, even second-order logic (SOL) is frequently too strong and very difficult to mechanically work with.

The above considerations imply that the most suitable framework for mechanizing mathematical reasoning should be provided by some logic which is intermediate between FOL and SOL. In search for the fundamental, indispensable principles that FOL is lacking, it is clear that the main feature one should seek to include is an appropriate induction principle, something which is not provided directly in FOL. We believe that the most prominent choice of such logic is *ancestral logic (AL)*[3] — the logic obtained by augmenting FOL with the concept of transitive closure of a given relation [4, 61, 76, 77, 82, 102, 105]. It is clear that any system designed for capturing the ability to do mathematics must provide the means to create the transitive closure of a relation and to make appropriate inferences regarding it (for instance, if a certain property is hereditary between objects in a given relation, then it will also be hereditary between objects which are related by the transitive closure of that relation). Thus, it was argued for in [4] that AL provides the most natural framework for the formalization and mechanization of mathematics.

[2] The precise quotation is "set-theory in sheep's clothing".

[3] Ancestral logic is also known in the literature as Transitive Closure Logic.

Several other logics which are intermediate between FOL and SOL have been suggested in the literature, such as: weak second-order logic, ω-logic, logics with a "cardinality quantifier" (i.e. "there are infinitely many"), and logics with Henkin quantifiers. In [102] it is proven that all of the above mentioned logics have the same expressive power as AL, in the sense that they all define the same class of *infinite* structures. Still, AL seems to us to be the most promising choice because there are several reasons to prefer it over the others: it is very useful also in the finite case; it seems to be the easiest choice from a *proof-theoretic* point of view; and it enters very naturally in computer science applications (see below). Another important advantage of AL is simply the simplicity of the notion of transitive closure. Thus any person, even with no mathematical background whatsoever, can easily grasp the concept of the ancestor of a given person (or, in other words, the idea of the transitive closure of a certain binary relation).

In addition to its use in the formalization of mathematics, logic has also played a major role in computer science since the early days of the field. In the famous paper with the telling name "On the Unusual Effectiveness of Logic in Computer Science" [52], it is forcefully noted that "at present concepts and methods of logic occupy a central place in computer science, insomuch that logic has been called 'the calculus of computer science' [75]". To demonstrate this claim, the paper then studies an impressive (but explicitly non-exhausting) list of applications of logics in different areas of computer science: descriptive complexity; database query languages; applications of constructive type theories; reasoning about knowledge; program verification and model checking. But again: *what* logic has such effectiveness? While it is quite usual to identify 'logic' with FOL[4], a check of the above list of applications from [52] reveals that first-order logic is sufficient for none of them. The only exception might seem to be its use for database query languages: [52] mentions only query languages which are directly based on FOL, like SQL. However, such languages cannot express important queries like "find all the people who will be in risk of being infected by Ebola if some infected passenger is found on flight 001". This is the reason why the

[4]Actually, at this point we are only referring to the formal *languages* used in the applications, ignoring (for the time being) other essential components of the notion of a 'logic', like the corresponding consequence relation. We note that such overloading in terminology is quite common in the literature.

SQL 3 (1999) standard added a WITH RECURSIVE construct which allows transitive closures to be computed inside the query processor, and by now such a construct is implemented also in IBM DB2, Microsoft SQL Server, and PostgreSQL.[5] Similar extensions of FOL are needed in all the other examples given in [52]:

- The characterization of complexity classes which is done in descriptive complexity always uses logics that are more expressive than FOL, like SOL, or logics which are intermediate between FOL and SOL like FOL enriched with a transitive closure operator, or the strongly related fix-point logics.

- Not only do type theories obviously go beyond FOL, but even their presentation and description cannot be done in FOL, since their introduction makes a massive use of *inductive definitions* of typing judgments. However, they can be done in FOL enriched with a transitive closure (TC) operator.

- A crucial notion for reasoning about knowledge is that of *common knowledge*. This notion is *inductively defined* in terms of the basic knowledge operators. However, this definition is not expressible in FOL (though it is again easily defined if a TC operator is added to FOL), and so in the language described in [52] it is introduced by brute force, as an independent operator.

- Verification of programs obviously involve *inductive arguments*, but such arguments are not a part of the logical machinery of FOL. Similarly, it is noted in [119] that "In all interesting applications of model-checking, reachability properties have to be checked, which are not expressible in the FOL-signature of labeled graphs (transition systems)". Again the problem is FOL's failure to define transitive closures of relations.

All these examples (as well as many others[6]) indicate that the crucial shortcoming of FOL is its inability to provide inductive definitions in general, and the notion of the transitive closure of a given binary relation in particular. In fact, because of this inability FOL cannot even serve as its own metalogic, since all its basic syntactic categories (such as terms, formulas, and formal derivations of formulas) are introduced

[5] Datalog too implements transitive closure computations.

[6] To give just one more example from computer science: in [71] the authors write: "Some appeal to second-order logic appears necessary here because transitive closure is not first-order definable". In fact, transitive closure is the only "second-order" feature they need.

via inductive definitions. Actually, the latter is true for a*ny* formal system and logic (see the example above of type theories). Hence only some extension of FOL which allows to introduce finitary inductive definitions ([41]) can be used as a framework for introducing and studying formal systems. In [4] it was shown (on the basis of Feferman's characterization in [41] of finitary inductive definitions and the system FS_0 introduced there for that purpose) that the minimal framework that can serve for this goal is AL. We strongly believe that just as in the case of mathematics, AL, rather than FOL, should be taken as the basic logic which underlies most applications of logic to Computer Science.

In recent years a great deal of attention has been given to AL in the area of finite model theory, and in related areas of computer science, like complexity classes (see [36, 73]). However, not much has been done so far about it in the context of *arbitrary* structures, or from a proof theoretical point-of-view.

According to the above-mentioned considerations, one goal of this thesis is to develop the theory of AL to the point it can serve as a reasonable (and in many cases, better) substitute for the use of FOL or HOL in the formalization of mathematics, as well as in different areas of computer science. We shall also demonstrate the usefulness of AL by exploring several applications of it.

1.1.2 The Indispensable Set Theory

The main principle of naive set theory is the comprehension schema:

$$\exists z(\forall x.x \in z \leftrightarrow \varphi)$$

where φ is a formula in which z is not free (but may contain other parameters). This basically states that the set $\{x \mid \varphi\}$ exists for any φ. Unfortunately, it is well known that allowing this for any formula φ would lead to paradoxes. Thus, any formal axiom system for set theory must replace the general comprehension schema by some particular ("safer") instances of it. The strength of the different formal set theories is determined by the variants of these instances.

The standard axiomatization of set theory is given by Zermelo-Fraenkel set theory, ZF. This system is based on classical logic, and its language, $\mathcal{L}_{ZF}$, is a first-order

language with equality which includes (besides $=$) only one binary predicate symbol: $\in$. The main instance of the general comprehension schema in ZF is the Restricted Comprehension Axiom Scheme:

$$\forall y \exists z \forall x \, (x \in z \leftrightarrow (x \in y \wedge \varphi))$$

Zermelo, in his original formulation of set theory [120], called properties for which the restricted comprehension is allowed "definite". However, he gave no characterization of formulas which define such "definite" properties. Later, Skolem identified Zermelo's concept of "definite" with first-order definability in $\mathcal{L}_{ZF}$ (see [43] for more details). This identification became a major component of the formal system ZF. In other words, the restricted comprehension is an axiom of ZF for every formula φ in $\mathcal{L}_{ZF}$. This makes ZF a rather strong system whose ontological assumptions maybe questioned. In particular, it allows what is known as "impredicative" definitions of sets (we return to this concept in the sequel).

Despite the success of ZF as the "official" formulation of set theory, in order to use it for the formalization of applicable mathematics it is necessary to overcome the following serious gaps that exist between working formally within ZF and actual mathematical practice:

- The language of ZF is very remote from real mathematical practice. Thus, almost all textbooks define set theories in languages in which variables (and perhaps a couple of constants) are the only terms which are directly provided. This feature makes these formalizations almost useless from a computational point of view. In contrast, all modern texts in all areas of mathematics (including set theory itself) employ much richer and more convenient languages. In particular: they make extensive use of terms for denoting sets, like abstraction terms of the form $\{x \mid \varphi\}$.

- ZF treats all the mathematical objects on a par, and so hides the computational significance of many of them. Thus although certain functions are first-class citizens in many programming languages, in set theory they are just "infinite sets", and ZF in its usual presentation is an extremely poor framework for computing with such sets (or handling them in a constructive way).

- Full ZF is far too strong for core mathematics, which practically deals only with a small fraction of the set-theoretical "universe". It is obvious that much weaker systems, corresponding to universes which are smaller, more effective, and better suited for computations, would do (presumably, such weaker systems will also be easier to mechanize).

The framework employed in this work (based on [5, 8]) aims to tackle all above-mentioned problems. It reflects real mathematical practice in making an extensive use of statically defined abstract set terms, in the same way they are used in ordinary mathematical discourse. Moreover, the framework will be provided with strong direct definitional power (akin to that used in informal mathematical texts), as well as computational power. This is due to the fact that its set of closed terms suffices for denoting every concrete set (including infinite ones) that might be needed in applications, as well as for computations with sets. The computational power of the framework can also be enhanced by using a constructive variant of it (See Section 2.1). Also, while the framework is to be based on set theory, it does not entail that the entire set-theoretical universe is indispensable. The flexibility of the framework allows us to invoke only the minimal set-theoretical ontological commitments and axioms required for that part of mathematics which is indispensable to (current) scientific theories (See Section 2.2).

1.2 Thesis Outline

The structure of this thesis is as follows:

Chapter 2 is a short overview of what we believe to be the two main alternatives to the standard approach to mathematics: the constructive one, and the predicative one.

Chapter 3 is devoted to presenting Ancestral Logic and exploring some of its basic properties. First, the formal definitions of (variants of) ancestral logic are provided and their most important model-theoretic properties are explored. Then we turn to a proof theoretical investigation of ancestral logic, and Gentzen-style proof systems for it are presented. Despite the fact that these systems cannot be complete with respect to the standard semantics, in Section 3.4 we provide a natural Henkin-style

semantics for ancestral logic and prove that the system for classical ancestral logic is complete with respect to it.

In Chapter 4 we focus on a constructive (intuitionistic) version of ancestral logic. We present a natural realizability semantics for this logic and provide a proof system for it which is strongly sound with respect to this semantics in the sense that provable formulas are uniformly realizable. This is followed by an investigation of some applications of intuitionistic ancestral logic for computer science. We show that this logic subsumes Kleene Algebras with Tests and thus serves as a natural programming logic for specifying, developing and reasoning about programs.

In Chapter 5 the general formal framework we use for formalizing predicative mathematics is presented. The framework makes it possible to employ in a natural way all the usual set notations and constructs as found in textbooks on naive or axiomatic set theory. A main feature of the framework is that the introduction of these set constructs is done statically, in a purely syntactic way. We present several variations of the formal system: a first-order one, and one that is based on the language of ancestral logic. Each of them also has two versions in accordance to the underlying logic taken: classical or intuitionistic. Then possible models for the various systems are explored. Next we expand the basic languages in order to introduce fundamental set theoretic concepts such as classes, relations and functions. This extension is done statically in keeping with the static nature of our framework. We further show how the natural numbers are incorporated into our framework both on the first-order level and by using the power of ancestral logic.

In Chapter 6 we demonstrate the usefulness of the framework described in Chapter 5, by developing in it large portions of applicable mathematics. We show that even on the first-order level, most of classical analysis can be carried out already within the minimal system of the framework and its minimal model. However, the restriction to this minimal first-order framework has its price: the development of mathematics involves coding, as well as treating the real line as a proper class. In contrast, using the minimal framework which is based on ancestral logic allows for a more natural development of classical analysis.

Finally, in Chapter 7 we conclude with a discussion of some directions for further research.

1.3 Notations

Below are some useful notations we shall use in this thesis.

- $Fv(exp)$ denotes the set of free variables occurring in exp.
- $\varphi\left\{\frac{t_1}{x_1}, ..., \frac{t_n}{x_n}\right\}$ denotes the result of simultaneously substituting t_i for the free occurrences of x_i in φ $(i = 1, ..., n)$. For simplicity, when there is no chance of confusion, we sometimes write $\varphi(t_1, ..., t_n)$ instead of $\varphi\left\{\frac{t_1}{x_1}, ..., \frac{t_n}{x_n}\right\}$.
- We sometimes write $\varphi(x)$ to indicate that x is free in φ.
- Let σ be some first-order signature, and let $\mathcal{L}$ be the corresponding language based on σ.
  - A structure for $\mathcal{L}$ is an ordered pair $M = \langle D, I\rangle$, where D is a non-empty set of elements (the domain) and I is an interpretation function on σ.
  - Let v be an assignment in M and T is a set of formulas in $\mathcal{L}$. We write $M, v \models T$ for the pair $\langle M, v\rangle$ satisfies T. In case T is satisfied under all assignments in M we write $M \models T$.
  - $v[x := a]$ denotes the x-variant of v which assigns to x the element a in D.

Chapter 2

General Background

As noted in the Introduction, the framework employed in this work is flexible and allows for different levels of formalization. A main feature of the framework is that it opens the door to making explicit and using the computational content which already exists implicitly in the foundation of set theory. However, in order to create a natural, user-friendly system for formalizing applicable mathematics which is suitable for mechanization it would be better to enhance the computational power of the system by invoking a more constructive approach. An overview of such constructive approach is given in Section 2.1 below. Moreover, in order to identify the minimal set theoretical universe which is indeed indispensable to applicable mathematics it might be useful to look at a different philosophical approach than the platonic one. One such alternative approach is the predicative approach to mathematics, which is reviewed in Section 2.2 below. These two alternative approaches, the constructive one and the predicative one, arose during the period of what was felt to be a foundational crisis in the early 20th century. Each approach rejected some essential logical aspects of classical mathematics. Thus, intuitionism (constructivism) criticized the unrestricted use of the Law of Excluded Middle, while predicativism objected to the use of certain type of definitions (which are known as "impredicative").

2.1 The Intuitionistic Approach

The intuitionistic approach to mathematics was founded about the year 1907 by Brouwer (see [56]). The intuitionistic philosophy is based on the idea that the truth

of a mathematical statement can only be conceived via a construction that proves it. As a result, Brouwer rejected the use of the Law of Excluded Middle, $A \vee \neg A$ for any formula A, which had been a pillar of classical logic for more than 2000 years. In particular, he objected to indirect existence proofs in mathematics.[1]

Brouwer developed a rich informal model of computation in terms of which he could interpret most concepts and theorems of mathematics, including set theory [113]. This intuitive interpretation has come to be known among logicians as Brouwer, Heyting, Kolmogorov (BHK) semantics when applied to formal intuitionistic logical calculi, as first done by Heyting [55] and Kolmogorov [65].[2] In 1945 Kleene [35, 63] invented his realizability semantics for intuitionistic number theory in order to connect Brouwer's informal notion of computability to the precise theory of partial recursive functions. By 1982 Martin-Löf [78, 79], building on the work of Kleene, refined the BHK approach and raised it to the level of a formal semantic method for constructive logics. The BHK semantics is strongly related, and one might even say equivalent, to Curry-Howard Isomorphism[3]/realizability semantics/propositions as types/proofs as terms/proofs as programs. This semantics plays an important role in building correct-by-construction software and in the semantics of stronger constructive typed systems, such as Computational Type Theory (CTT) [29], Intuitionistic Type Theory (ITT) [79], Intensional-ITT [16, 87], the Calculus of Inductive Constructions (CIC) [14], and Logical Frameworks such as the Edinburgh LF [53].[4] It is important to note that unlike classical logic, there are several different approaches (see, e.g., [110]) for providing precise semantics to intuitionistic logic. We believe that the most fruitful and faithful to the intuitionistic conception of knowledge is the BHK semantics.

The basic idea of the BHK semantics is to define what an intuitionistic (constructive) proof should consist of, by indicating how the connectives and the quantifiers of the language should be interpreted. Thus, the meaning of a proposition, say P,

[1] While Brouwer's motivation for intuitionism was a philosophical one, Bishop [15] advocated the constructive approach because it supports a computational view of mathematics.

[2] The name "BHK" was coined by Troelstra [109], where "K" initially stood for "Kreisel", and only later for "Kolmogorov".

[3] It should be noted that the Curry-Howard Isomorphism is not an isomorphism, nor was it invented by either Curry or Howard. In [107] several contributors to the principle are mentioned, the main ones being Brouwer, Heyting, Kolmogorov, Kleene, deBruijn, Curry, Howard, Girard, and Martin-Löf.

[4] All of these logics have been implemented by proof assistants such as Agda, Coq, Nuprl, and Twelf.

is given by a type whose elements can be understood as evidence for "knowing" P. Given an intended constructive domain, a standard description of the BHK semantics is as follows:

- $\bot$ has no proof.
- a proof of $A \wedge B$ is given by presenting a proof of A and a proof of B.
- a proof of $A \vee B$ is given by presenting either a proof of A or a proof of B, as well as stating which is proved.
- a proof of $A \to B$ is a construction which transforms any proof of A into a proof of B.
- a proof of $\exists x A(x)$ is given by providing an element d of the domain, and a proof of $A(d)$.
- a proof of $\forall x A(x)$ is a construction which transforms every proof that d belongs to the domain into a proof of $A(d)$.

A construction p that is a proof of a formula in the sense of the BHK semantics is often called a *proof term*, or an *evidence term*, or a *realizer*.

The negation $\neg A$ of a formula A is proven once it has been shown that there cannot exist a proof of A, which means providing a construction that derives $\bot$ (i.e. falsum) from any possible proof of A. Thus, the standard definition of $\neg A$ is as $A \to \bot$.

The constructive character of intuitionistic logic becomes particularly clear in the BHK semantics due to the correspondence between derivations in the logic and terms in simply typed λ-calculus, i.e., between proofs and computations. This correspondence preserves structure in that reduction of terms correspond to normalization of proofs. However, the correspondence given for implication and universal quantification are notoriously imprecise because the notion of function is left undefined. One way to make the BHK semantics precise is by requiring functions to be computable (recursive). Note also that the above interpretation (except for $\bot$) does not address the case of atomic formulas. Nevertheless, in practice these rules seem to suffice for codifying the constructive arguments of mathematicians.[5]

[5]Note that already on the informal level of the BHK semantics, one is forced to reject the Law of Excluded Middle.

The semantic tradition of realizability is grounded in precise knowledge of the underlying computation system. Proof terms are part of a computation system, and they can be evaluated. For constructive (intuitionistic) logics, a proof term can be considered as a program or data that reveal the implicit computational content of the proven propositions and make it explicit. Proof assistants based on computational type theory or other constructive theories (such as Nuprl and Coq) can evaluate these proof terms and thus extract their computational content.[6]

The axiomatic set theories, IZF [13] and CZF [2], are based on intuitionistic first-order logic and their sets of axioms are similar to the axioms of ZF (they are both formulated in the same language as ZF). By adding to either of them the Law of Excluded Middle we get full classical ZF.[7] The main difference between CZF and IZF is that the latter is impredicative (see the next section for an overview of predicativism), while CZF is a predicative subsystem of IZF.

2.2 The Predicative Approach

As noted above, it seems that restricted variants of classical set theory would suffice for ordinary mathematical practice (and may even be more compatible with it). However, the restriction imposed by the intuitionistic approach to mathematics, according to which simple number-theoretic statements may have no definite truth value, is rather strict and seems to violate the typical working mathematician's intuition. Predicativism offers an intermediate foundational stance for ordinary mathematical practice. This approach admits very little, if any, of the classical set-theoretic universe beyond what is used in applicable mathematics. It is for that reason that it seems to us to indeed be the minimal set theoretical fragment needed for the formalization of mainstream mathematics.

The predicativist program for the foundations of mathematics, initiated by Poincaré [88, 89] and first seriously developed by Weyl [116], seeks to establish certainty in mathematics without revolutionizing it (as the intuitionistic program does).

[6] Note that classical logic can also be given a BHK-style semantics, where only some proof terms are computable. This can be done by introducing "oracle" for treating the excluded middle [27], or by using "virtual evidence" [26, 28].

[7] Another well-known formalization of intuitionistic set theory is CST [83], which provides a formal foundation for Bishop's program of constructive mathematics.

The program (as is usually conceived nowadays) is based on the following two basic principles:

(I) Predicative Definitions: In predicative mathematics higher order constructs, such as sets or functions, are acceptable only when introduced through acceptable definitions. A major principle for distinguishing between definitions which are acceptable and those which are not is the Vicious Circle Principle (VCP). The VCP, which was put forward in response to the set-theoretic paradoxes, states that no object may be introduced by a definition which refers to a totality to which this object is suppose to belong (definitions that violate the VCP are called impredicative). Hence in defining a new construct one can only refer to constructs which were introduced by previous definitions. Thus in the predicative approach one cannot assume that there is a complete totality of all definable objects of a certain kind; rather, each one comes into existence through a definition in terms of previously defined objects.
The fundamental source of impredicativity in ZF is the Restricted Comprehension Axiom Scheme, which asserts the existence of the set $\{x \in z \,|\, \varphi\}$ for every previously formed collection z and every formula φ in $\mathcal{L}_{ZF}$. However, since φ may contain quantifiers ranging over the totality of all sets, this is impredicative according to the VCP. Therefore, predicative set theory must weaken the Restricted Comprehension Scheme.

(II) The Natural Numbers: In the standard predicative approaches to mathematics it is accepted that the sequence of natural numbers is a basic intuitively well-understood mathematical concept, and as a totality it constitutes a set.[8] The reason is that the construction of the natural numbers is done by iteration of a very simple, concrete step. In contrast, the concept of *arbitrary* subsets of the natural numbers is not immediately given by mathematical intuition, since we have no way of building up $P(\mathbb{N})$ (the power set of $\mathbb{N}$) in a step-by-step manner. Accordingly, predicativism does not sanction power sets of infinite sets. Therefore, predicative set theory cannot adopt ZF's Power Set Axiom. Note that the illegitimacy of infinite power sets has the consequence that in predicative mathematics apparently all sets are countable.

[8] [85] is a notable exception.

The first principle, **(I)**,was interpreted by Russell according to his philosophical views of logic [97, 98], and incorporated in Principia Mathematica as the ramified type theory (RTT) [117]. In RTT objects are divided into types, and each higher-order type is further divided into levels. However, the use of levels makes it impossible to develop mathematics in RTT, and so Russell had to add a special axiom of reducibility, which practically destroyed the predicative nature of his system [94]. Principle **(I)** was then taken again by Weyl in [116], but instead of Russell's ramified hierarchy, Weyl adopted principle **(II)** which goes back to Poincaré. Weyl's predicativist program was later extensively pursued by Feferman, who in a series of papers (e.g. [37, 38, 39, 40]) developed proof systems for predicative mathematics. Feferman's systems are less complex than RTT, and he has shown that a very large part of classical analysis can be developed within them. He further conjectured that predicative mathematics in fact suffices for developing all the mathematics that is actually indispensable to present-day natural sciences. Despite this success, Feferman's systems failed to receive in the mathematical community the interest they deserve. Unlike constructive mathematics, they were also almost totally ignored in the computer science community. The main reason for this seems to be the fact that, on the one hand, Feferman's systems are not "revolutionary" (since they allow the use of classical logic), but on the other hand they are still rather complicated in comparison to the impredicative formal set theory ZF, which provides the standard foundations and framework for developing mathematics. In particular: Feferman's systems still use complicated systems of types, and both functions and classes are taken in them as independent primitives. Therefore, working within these systems is somewhat complicated for someone used to the standard ZF (or something similar).

Another recent attempt to develop predicative mathematics was made by Weaver [114, 115], who demonstrated how core mathematics, particularly abstract analysis, can be developed within a concrete countable universe J_2 (the second set in Jensen's constructable hierarchy). The drawback of that work is that both the motivation and the work itself were exclusively based on semantical considerations. Thus, it is unclear how that framework can be turned into a (formal) mathematical theory like, e.g., ZF.

It is important to note that the predicative approach is orthogonal to the debate between the classical and intuitionistic approaches. Thus it may be based classically

on the notion of truth, or, adopting intuitionism, on the concept of provability.[9] We follow this approach too in this work and present two variants for our predicative framework (see Chapters 5 and 6).

[9]Thus Feferman explicitly states that his systems can be based on either of these logics. However, it should be noted here that the predicativist literature tends to presume classical logic.

Chapter 3

Ancestral Logic

In this chapter and the next one we explore Ancestral Logic, AL, which is the logic obtained from FOL by adding to it a transitive closure operator. As noted in the Introduction, we believe that this logic provides a very suitable framework for the formalization of effective mathematics, and is also fundamental in many areas of computer science. In this chapter we mainly focus on the classical point of view of AL, whereas a detailed account of an intuitionistic version of AL is provided in the next chapter.

The chapter is organized as follows: In Section 3.1 the formal definitions of the transitive closure operator and ancestral logic are given. Then, some of the most important model-theoretic properties of ancestral logic are presented, and its expressive power is described. Section 3.2 deals with ancestral logic from a proof-theoretic point of view. Natural Gentzen-style systems which are sound for ancestral logic are presented and their key features are explored. In Section 3.3 we show that in the case of arithmetics the ordinal number of these systems is ε_0, the ordinal of Peano Arithmetic (PA) and Heyting Arithmetic (HA). Finally, in Section 3.4 we provide a completeness theorem for the system for classical AL with respect to a natural Henkin-style semantics.

Sections 3.1–3.3 are mainly based on [22, 23].[1] The results in Section 3.4 have not been published before.

[1]Some of the preliminary results in these sections were already given in the author's M.Sc. thesis [21], so we omit their proofs here.

3.1 The Languages and their Semantics

The essential idea in embedding the concept of the transitive closure operator into a logical framework is that one may treat a first-order formula with two (assigned) free variables as a definition of a binary relation. Below are the corresponding formal definitions of a classical and intuitionistic first-order logics augmented by a transitive closure operator, and their semantics. Following suggestions made in, e.g., [4, 76, 77, 82], we present two types of the transitive closure operator: the reflexive one, and the non-reflexive one.

Let σ be some first-order signature (with or without a distinguished equality symbol), and let $\mathcal{L}$ be the corresponding first-order language.

Definition 3.1.1 (The languages)**.**

- The language $\mathcal{L}_{TC}$ is obtained from $\mathcal{L}$ by the addition of the transitive closure operator (TC), together with the following clause concerning the definition of a formula:
  - for any formula φ in $\mathcal{L}_{TC}$, distinct variables x, y, and terms s, t, $(TC_{x,y}\varphi)(s,t)$ is a formula in $\mathcal{L}_{TC}$. The free occurrences of x and y in φ become bound in this formula.
- The language $\mathcal{L}_{RTC}$ is defined as $\mathcal{L}_{TC}$ with TC replaced by RTC.

The intended meaning of the formula $(TC_{x,y}\varphi)(s,t)$ is the "infinite disjunction":

$$\varphi\{\frac{s}{x},\frac{t}{y}\}\vee\exists w_1(\varphi\{\frac{s}{x},\frac{w_1}{y}\})\wedge\varphi\{\frac{w_1}{x},\frac{t}{y}\})\vee\exists w_1\exists w_2(\varphi\{\frac{s}{x},\frac{w_1}{y}\}\wedge\varphi\{\frac{w_1}{x},\frac{w_2}{y}\}\wedge\varphi\{\frac{w_2}{x},\frac{t}{y}\})\vee\ldots$$

where $w_1, w_2, \ldots$ are all new variables. The intended meaning of $(RTC_{x,y}\varphi)(s,t)$ is $s = t \vee (TC_{x,y}\varphi)(s,t)$.

Definition 3.1.2 (The logics)**.**

- The logic cAL is semantically defined like classical first-order logic, with the following additional clause:

  - The pair $\langle M, v\rangle$ is said to satisfy $(TC_{x,y}\varphi)(s,t)$ if there exist $a_0, ..., a_n \in D$ $(n > 0)$ such that $v(s) = a_0$, $v(t) = a_n$, and $M, v[x := a_i,\ y := a_{i+1}] \models \varphi$ for $0 \leq i \leq n-1$.

- The logics cAL^{ref} is semantically defined like classical first-order logic, with the following additional clause:

  - The pair $\langle M, v\rangle$ is said to satisfy $(RTC_{x,y}\varphi)(s,t)$ if $v(s) = v(t)$, or there exist $a_0, ..., a_n \in D$ $(n > 0)$ such that $v(s) = a_0$, $v(s) = a_n$, and $M, v[x := a_i,\ y := a_{i+1}] \models \varphi$ for $0 \leq i \leq n-1$.

Note 3.1.3. In this chapter we do not provide a precise semantics for the corresponding intuitionistic logics, iAL and iAL^{ref}, but instead refer to the intended meaning of the transitive closure operator. A formal semantics for these logics is explored in the next chapter.

In the presence of equality, the two forms of the transitive closure operator are definable in terms of each other. Without equality, there is a difference between the two forms in the ability to define quantifiers. In the language without equality, the existential quantifier can be defined using the TC operator by: $\exists x\varphi := \left(TC_{u,v}\left(\varphi\left\{\frac{u}{x}\right\} \vee \varphi\left\{\frac{v}{x}\right\}\right)\right)(s,t)$, while it cannot be defined using the RTC operator (see [21]).

Notation. In what follows, when the claim we are stating applies to all four forms of ancestral logic defined above (cAL, cAL^{ref}, iAL and iAL^{ref}) we simply write AL.

Another option that has been investigated ([4, 76]) is to use not only a binary operator TC (RTC), but stronger transitive closure operators, i.e., creating a system where for each $n \in \mathbb{N}$ there is an operator TC^n (RTC^n) which when applied to a $2n$-ary predicate produces a new $2n$-ary predicate. It is easy to see that in a language with equality, if one allows the use of ordered pairs, then all the TC^n (RTC^n) operators can be reduced to TC (RTC).

A simple compactness argument shows that the transitive closure operator is in general not first-order definable. However, it is definable using second-order logic. Thus, ancestral logics are intermediate between first- and second-order logics. As mentioned in the introduction, an important indication that the expressive power of ancestral logic captures a very significant and natural fragment of SOL is provided

by the fact that cAL (and thus also cAL^{ref}) is equivalent in its expressive power to several other logics between FOL and SOL that have been suggested and investigated in the literature.

The natural numbers can be categorically characterized in AL using only equality, 0 and the successor function S (see Section 3.3). However, it was shown in [4] that this language does not suffice for defining addition. It was further shown there that, in contrast, the language of $\{=, 0, S, +\}$ does suffice in AL for defining all recursive predicates and functions. It can also be easily seen (see [102]) that the upward Löwenheim-Skolem theorem fails for AL[2], and that AL is not finitary (i.e., the compactness theorem fails for it). Moreover, the sets of valid formulas of cAL and iAL in the language of $\{=, 0, S, +\}$ are not even arithmetical. Hence, there can be no formal system, for either of these logics, which is sound and complete. Nevertheless, as we shall demonstrate in the next section, there are very natural formal approximations which are sound, and seem to encompass all forms of reasoning for these logics that are used in practice.

3.2 Proof systems for AL

Like in the case of SOL, since there can be no sound and complete system for AL, one should instead look for useful approximations. Such approximations should be:

- natural and effective,
- sound with respect to the intended semantics,
- both sound and complete with respect to some natural generalization of the intended semantics (like Henkin semantics for SOL [102]).

Such Hilbert-style approximations were suggested already in [76, 77, 82]. Those systems are all equivalent, and every rule for the transitive closure operator that have been suggested so far in the literature can be shown to be derivable in them. Nevertheless, the use of Hilbert-type systems is impractical, since they are not suitable

[2]On the other hand, the downward Löwenheim-Skolem theorem holds for AL. This shows that though more expressive than FOL, ancestral logic is still weaker than SOL. For instance, the real numbers and the notion of well-ordering cannot be characterized (up to isomorphism) in AL (while they can be characterized in SOL).

for mechanization. A better proof-theoretical approach would be to explore Gentzen-style systems for AL.

The systems defined below are extensions of Gentzen-style systems for classical first-order logic, $\mathcal{LK}$, and for intuitionistic first-order logic, $\mathcal{LJ}$ (see, e.g., [45, 108]).[3] In what follows the letters Γ, Δ represent finite (possibly empty) multisets of formulas, φ, ψ, ϕ arbitrary formulas, x, y, z, u, v variables, and r, s, t terms.

Definition 3.2.1. Let G be a Gentzen-style system.

- A sequent s is said to be provable from a set of sequents S in G, denoted by $S \vdash_G s$, if there exists a derivation in G of s from S.
- A formula φ is said to be provable from a set of formulas T in G, denoted by $T \vdash_G \varphi$, if there is a derivation in G of $\Rightarrow \varphi$ from the set $\{\Rightarrow \psi \,|\, \psi \in T\}$.

We start with AL^{ref}, for which it is slightly easier to provide an adequate Gentzen-type approximation.

Definition 3.2.2 (*The systems* cAL^{ref}_G and iAL^{ref}_G)**.**

- The system cAL^{ref}_G for $\mathcal{L}_{RTC}$ is defined by adding to $\mathcal{LK}$ the axiom:

$$\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,s) \tag{3.1}$$

and the following inference rules:

$$\frac{\Gamma \Rightarrow \Delta, \varphi\left\{\frac{s}{x}, \frac{t}{y}\right\}}{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)} \tag{3.2}$$

$$\frac{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,r) \quad \Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(r,t)}{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)} \tag{3.3}$$

$$\frac{\Gamma, \psi(x), \varphi(x,y) \Rightarrow \Delta, \psi\left\{\frac{y}{x}\right\}}{\Gamma, \psi\left\{\frac{s}{x}\right\}, \left(RTC_{x,y}\varphi\right)(s,t) \Rightarrow \Delta, \psi\left\{\frac{t}{x}\right\}} \tag{3.4}$$

In all three rules we assume that the terms which are substituted are free for substitution, and that no forbidden capturing occurs. In Rule (3.4) x should not occur free in Γ and Δ, and y should not occur free in Γ, Δ and ψ.

[3] We take here $\mathcal{LK}$ *and* $\mathcal{LJ}$ to include the substitution rule, which was not taken as a logical rule in the original systems.

- The system iAL_G^{ref} for $\mathcal{L}_{RTC}$ is defined by adding to $\mathcal{LJ}$ the same set of axioms and inference rules as to cAL_G^{ref}, with the restriction that $\Delta = \emptyset$ in all of them.

Notation. When what we claim applies to both the classical and the intuitionistic systems, we simply write AL_G^{ref}.

Definition 3.2.3. For languages with equality, the system $AL_G^{=,ref}$ is obtained from AL_G^{ref} by the addition of standard equality axioms (see, e.g., [108]).

Rule (3.4) is a generalized induction principle which states that if t is a φ-descendant of s or equal to it, then if s has some hereditary property which is passed down from one object to another if they are φ-related, then t also has that property. This is actually a generalized form of the induction rule of PA (see Prop. 3.3.2).

The system AL_G^{ref} is adequate for handling the RTC operator, in the sense that it is sound, it give the RTC operator the intended meaning of the reflexive transitive closure operator, and all fundamental rules concerning the RTC operator that have been suggested in the literature (as far as we know) are derivable in it.[4] The Lemma below provides some examples.

Lemma 3.2.4. *The following rules are derivable in* AL_G^{ref}.[5]

$$\frac{\Gamma \Rightarrow \Delta, \varphi\left\{\frac{s}{x}, \frac{r}{y}\right\} \quad \Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(r,t)}{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)} \qquad \frac{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,r) \quad \Gamma \Rightarrow \Delta, \varphi\left\{\frac{r}{x}, \frac{t}{y}\right\}}{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)} \tag{3.5}$$

$$\frac{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)}{\Gamma \Rightarrow \Delta, s = t \vee \exists z\left(\left(RTC_{x,y}\varphi\right)(s,z) \wedge \varphi\left\{\frac{z}{x}, \frac{t}{y}\right\}\right)} \qquad \frac{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)}{\Gamma \Rightarrow \Delta, s = t \vee \exists z\left(\varphi\left\{\frac{s}{x}, \frac{z}{y}\right\} \wedge \left(RTC_{x,y}\varphi\right)(z,t)\right)} \tag{3.6}$$

$$\frac{\Gamma \Rightarrow \Delta, \left(RTC_{x,y}\varphi\right)(s,t)}{\Gamma \Rightarrow \Delta, \left(RTC_{y,x}\varphi\right)(t,s)} \qquad \frac{\left(RTC_{x,y}\varphi\right)(s,t), \Gamma \Rightarrow \Delta}{\left(RTC_{y,x}\varphi\right)(t,s), \Gamma \Rightarrow \Delta} \tag{3.7}$$

[4] Further evidence for the naturalness of AL_G^{ref} is that while it was developed independently of the Hilbert-type systems mentioned above, it is not difficult to show that it is equivalent to them (see [21]).

[5] Rules (3.6), (3.9) and (3.12) contain equality and therefore are derivable in $AL_G^{=,ref}$.

$$\frac{\Gamma \Rightarrow \Delta, (RTC_{x,y}\varphi)(s,t)}{\Gamma \Rightarrow \Delta, \left(RTC_{u,v}\varphi\left\{\frac{u}{x},\frac{v}{y}\right\}\right)(s,t)} \quad \frac{(RTC_{x,y}\varphi)(s,t), \Gamma \Rightarrow \Delta}{\left(RTC_{u,v}\varphi\left\{\frac{u}{x},\frac{v}{y}\right\}\right)(s,t), \Gamma \Rightarrow \Delta} \tag{3.8}$$

$$\frac{\varphi\left\{\frac{s}{x}\right\}, \Gamma \Rightarrow \Delta}{(RTC_{x,y}\varphi)(s,t), \Gamma \Rightarrow s = t, \Delta} \quad \frac{\varphi\left\{\frac{t}{y}\right\}, \Gamma \Rightarrow \Delta}{(RTC_{x,y}\varphi)(s,t), \Gamma \Rightarrow s = t, \Delta} \tag{3.9}$$

$$\frac{\Gamma, \varphi \Rightarrow \Delta, \psi}{\Gamma, (RTC_{x,y}\varphi)(s,t) \Rightarrow \Delta, (RTC_{x,y}\psi)(s,t)} \tag{3.10}$$

$$\frac{(RTC_{x,y}\varphi)(s,t), \Gamma \Rightarrow \Delta}{(RTC_{u,v}(RTC_{x,y}\varphi)(u,v))(s,t), \Gamma \Rightarrow \Delta} \tag{3.11}$$

$$\frac{\Gamma, \varphi(x,y) \Rightarrow \Delta, \phi(x,y) \quad \Gamma, \phi\left\{\frac{u}{x},\frac{v}{y}\right\}, \phi\left\{\frac{v}{x},\frac{w}{y}\right\} \Rightarrow \Delta, \phi\left\{\frac{u}{x},\frac{w}{y}\right\}}{\Gamma, (RTC_{x,y}\varphi)(s,t) \Rightarrow \Delta, s = t, \phi\left\{\frac{s}{x},\frac{t}{y}\right\}} \tag{3.12}$$

Conditions:

- *In all the rules we assume that the terms which are substituted are free for substitution and that no forbidden capturing occurs.*
- *In (3.6) z should not occur free in Γ, Δ and $\varphi\left\{\frac{s}{x},\frac{t}{y}\right\}$.*
- *In (3.8) the conditions are the usual ones concerning the α-rule.*
- *In (3.9) y should not occur free in Γ, Δ or s in the left rule, and x should not occur free in Γ, Δ or t in the right rule.*
- *In (3.10) x, y should not occur free in Γ, Δ.*
- *In (3.11) u, v should not occur free in φ.*
- *In (3.12) x, y should not occur free in Γ and Δ, and u, v, w should not occur free in Γ, Δ, ϕ and ψ.*

Proof. For a detailed proof see [21, 23]. Here we prove (3.12) as an example. For readability we omit the context Γ, Δ from the sequents. Assume that $\varphi(x,y) \Rightarrow \phi(x,y)$ and $\phi\left\{\frac{u}{x},\frac{v}{y}\right\}, \phi\left\{\frac{v}{x},\frac{w}{y}\right\} \Rightarrow \phi\left\{\frac{u}{x},\frac{w}{y}\right\}$ are provable. By substitution we get $\phi(u,x), \phi(x,y) \Rightarrow \phi(u,y)$, from which, using a cut, we get $\phi(u,x), \varphi(x,y) \Rightarrow \phi(u,y)$. Applying Rule (3.4) we get $\phi(s,x), (RTC_{x,y}\varphi)(x,t) \Rightarrow \phi(s,t)$. Using $\varphi(s,x) \Rightarrow \phi(s,x)$ and $s = t \Rightarrow s = t$, applying standard first-order rules we get

a proof of $s = t \vee \exists x \left(\varphi\left(s,x\right) \wedge \left(RC_{x,y}\varphi\right)\left(x,t\right)\right) \Rightarrow s = t \vee \phi\left(s,t\right)$. Now, using Rule (3.6) and a cut we get a proof of $\left(RTC_{x,y}\varphi\right)\left(s,t\right) \Rightarrow s = t \vee \phi\left(s,t\right)$. □

Next we turn to the non-reflexive TC operator. The systems for it below correct and extend the system suggested in [4]. In that system the rules were the exact TC-counterparts of the rules in AL_G^{ref} (but without Axiom (3.1)). However, in [22] it was shown that there are fundamental properties of the TC operator which are unprovable in it. Therefore, we strengthen here that system by replacing its original induction rule by (3.15).[6] In the resulting systems all the TC-counterparts of the rules in Lemma 3.2.4 are provable.

Definition 3.2.5 (*The systems* cAL_G and iAL_G)**.**

- The system cAL_G for $\mathcal{L}_{TC}$ is defined by adding to $\mathcal{LK}$ the following inference rules:

$$\frac{\Gamma \Rightarrow \Delta, \varphi\left\{\frac{s}{x}, \frac{t}{y}\right\}}{\Gamma \Rightarrow \Delta, \left(TC_{x,y}\varphi\right)\left(s,t\right)} \tag{3.13}$$

$$\frac{\Gamma \Rightarrow \Delta, \left(TC_{x,y}\varphi\right)\left(s,r\right) \quad \Gamma \Rightarrow \Delta, \left(TC_{x,y}\varphi\right)\left(r,t\right)}{\Gamma \Rightarrow \Delta, \left(TC_{x,y}\varphi\right)\left(s,t\right)} \tag{3.14}$$

$$\frac{\Gamma, \varphi\left(x,y\right) \Rightarrow \Delta, \phi\left(x,y\right) \quad \Gamma, \phi\left\{\frac{u}{x}, \frac{v}{y}\right\}, \phi\left\{\frac{v}{x}, \frac{w}{y}\right\} \Rightarrow \Delta, \phi\left\{\frac{u}{x}, \frac{w}{y}\right\}}{\Gamma, \left(TC_{x,y}\varphi\right)\left(s,t\right) \Rightarrow \Delta, \phi\left\{\frac{s}{x}, \frac{t}{y}\right\}} \tag{3.15}$$

  In all three rules we assume that the terms which are substituted are free for substitution, and that no forbidden capturing occurs. In Rule (3.15) x, y should not occur free in Γ and Δ, and u, v, w should not occur free in Γ, Δ, ϕ and ψ.

- The system iAL_G for $\mathcal{L}_{TC}$ is defined by adding to $\mathcal{LJ}$ the same set of inference rules as to cAL_G, with the restriction that $\Delta = \emptyset$ in all of them.

Notation. Again, when what we claim applies to both the classical and the intuitionistic systems, we simply write AL_G.

Definition 3.2.6. For languages with equality, the system $AL_G^{=}$ is obtained from AL_G by the addition of standard equality axioms (see, e.g., [108]).

[6] A different extension of the proof system suggested in [4], which is equivalent to the systems presented here, is described in [23].

Note 3.2.7. Rule (3.15) is indeed a generalization of the TC-counterpart of Rule (3.4) (which is, in turn, a generalization of PA's induction rule). This can be easily seen by taking $\phi(x,y)$ to be $\psi(x) \to \psi\left\{\frac{y}{x}\right\}$, for which $\phi\left\{\frac{u}{x},\frac{v}{y}\right\}, \phi\left\{\frac{v}{x},\frac{w}{y}\right\} \Rightarrow \phi\left\{\frac{u}{x},\frac{w}{y}\right\}$ is clearly provable using first-order rules.

Since each of the two forms of the transitive closure operator can be expressed in terms of the other (in the presence of equality), it is interesting to explore the connection between their proof systems.

Definition 3.2.8. Define recursively two interpretations, $'$ from $\mathcal{L}_{RTC}$ to $\mathcal{L}_{TC}$ and $*$ from $\mathcal{L}_{TC}$ to $\mathcal{L}_{RTC}$, as follows:

- $\varphi' = \varphi^* = \varphi$, for φ atomic formula.
- $(\neg\varphi)^* = \neg\varphi^*$ and $(\neg\varphi)' = \neg\varphi'$.
- $(\varphi \circ \psi)^* = \varphi^* \circ \psi^*$ and $(\varphi \circ \psi)' = \varphi' \circ \psi'$, where $\circ \in \{\wedge, \vee, \to\}$.
- $(Qx\varphi)^* = Qx\varphi^*$ and $(Qx\varphi)' = Qx\varphi'$, where $Q \in \{\forall, \exists\}$.
- $\left((TC_{x,y}A)(s,t)\right)^* = \exists z\left(A^*\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge (RTC_{x,y}A^*)(z,t)\right)$.
- $\left((RTC_{x,y}A)(s,t)\right)' = (TC_{x,y}A')(s,t) \vee s = t$.

We use the standard abbreviations: Γ^* for $\{\varphi^* | \varphi \in \Gamma\}$ and Γ' for $\{\varphi' | \varphi \in \Gamma\}$.

The following proposition, which shows that the above interpretations preserve provability, was established in [21, 23]:[7]

Proposition 3.2.9.

1. *If* $\vdash_{AL_G^{=,ref}} \Gamma \Rightarrow \Delta$, *then* $\vdash_{AL_G^{=}} \Gamma' \Rightarrow \Delta'$.
2. *If* $\vdash_{AL_G^{=}} \Gamma \Rightarrow \Delta$, *then* $\vdash_{AL_G^{=,ref}} \Gamma^* \Rightarrow \Delta^*$.

Next we present another strong connection between these translations:

[7] Actually, $AL_G^{=}$ and $AL_G^{=,ref}$ are modifications of the systems for the TC operator and the RTC operator which are given in [23] (denoted there by TC_G and RTC_G, respectively). The systems presented here have a further generalization of the induction rule. However, the same translation given there preserves provability between $AL_G^{=}$ and $AL_G^{=,ref}$ as well.

Proposition 3.2.10. *The following holds:*

1. $\vdash_{AL_G^{=,ref}} (\varphi')^* \Rightarrow \varphi$ *and* $\vdash_{AL_G^{=,ref}} \varphi \Rightarrow (\varphi')^*$.
2. $\vdash_{AL_G^{=}} (\varphi^*)' \Rightarrow \varphi$ *and* $\vdash_{AL_G^{=}} \varphi \Rightarrow (\varphi^*)'$.

Proof. The proofs of both 1. and 2. are carried out by induction on φ. If φ does not contain the TC or RTC operator, then $(\varphi')^*$ and $(\varphi^*)'$ are equal to φ, hence provably equivalent to it.

For 1. assume that $\varphi := (RTC_{x,y}A)(s,t)$. Thus, $(\varphi')^*$ is the formula $\exists z\left((A')^*\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge RTC_{x,y}(A')^*(z,t)\right) \vee s = t$. By the induction hypothesis we have $\vdash_{AL_G^{ref}} (A')^* \Rightarrow A$, thus by (3.10) the sequent $(RTC_{x,y}(A')^*)(s,t) \Rightarrow (RTC_{x,y}A)(s,t)$ is also provable in $AL_G^{=,ref}$. It is easy to check that the sequent $\exists z\left((A')^*\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge RTC_{x,y}(A')^*(z,t)\right) \vee s = t \Rightarrow (RTC_{x,y}(A')^*)(s,t)$ is provable in $AL_G^{=,ref}$ (using (3.5) and (3.1)). A cut on the last two sequents results in a proof of $\exists z\left((A')^*\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge RTC_{x,y}(A')^*(z,t)\right) \vee s = t \Rightarrow (RTC_{x,y}A)(s,t)$. For the converse, denote $\exists z\left((A')^*\left\{\frac{u}{x},\frac{z}{y}\right\} \wedge RTC_{x,y}(A')^*(z,w)\right) \vee s = t$ by ψ (notice that $(\varphi')^*$ is $\psi\left\{\frac{s}{u},\frac{t}{w}\right\}$). It is easy to see that $\psi\left\{\frac{s}{u},\frac{x}{w}\right\}, (A')^* \Rightarrow \psi\left\{\frac{s}{u},\frac{y}{w}\right\}$ is provable in $AL_G^{=,ref}$. Applying Rule (3.4) results in a proof of the sequent $\psi\left\{\frac{s}{u},\frac{s}{w}\right\}, (RTC_{x,y}(A')^*)(s,t) \Rightarrow \psi\left\{\frac{s}{u},\frac{t}{w}\right\}$. The sequent $\Rightarrow \psi\left\{\frac{s}{u},\frac{s}{w}\right\}$ is clearly provable using the equality axiom, thus, a cut entails a proof of the sequent $(RTC_{x,y}(A')^*)(s,t) \Rightarrow (\varphi')^*$. As before, by the induction hypothesis we have that $\vdash_{AL_G^{=,ref}} A \Rightarrow (A')^*$, so by (3.10) the sequent $(RTC_{x,y}A)(s,t) \Rightarrow (RTC_{x,y}(A')^*)(s,t)$ is also provable in $AL_G^{=,ref}$, and by one cut we obtain a proof of $(RTC_{x,y}A)(s,t) \Rightarrow (\varphi')^*$.

For 2. assume that $\varphi := (TC_{x,y}A)(s,t)$. Hence, $(\varphi^*)'$ is the formula $\exists z\left((A^*)'\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge \left(TC_{x,y}(A^*)'(z,t) \vee z = t\right)\right)$. It is easy to check that the sequent $\exists z\left((A^*)'\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge \left(TC_{x,y}(A^*)'(z,t) \vee z = t\right)\right) \Rightarrow \left(TC_{x,y}(A^*)'\right)(s,t)$ is provable in $AL_G^{=}$. By the induction hypothesis we have $\vdash_{AL_G^{=}} (A^*)' \Rightarrow A$, so by the TC-counterpart of (3.10) the sequent $\left(TC_{x,y}(A^*)'\right)(s,t) \Rightarrow (TC_{x,y}A)(s,t)$ is also provable in $AL_G^{=}$. Applying a cut results in a proof of the sequent $\exists z\left((A^*)'\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge \left(TC_{x,y}(A^*)'(z,t) \vee z = t\right)\right) \Rightarrow (TC_{x,y}A)(s,t)$. For the converse, notice that the TC-counterpart of Rule (3.6) entails the provability of $\left(TC_{x,y}(A^*)'\right)(s,t) \Rightarrow (A^*)'\left\{\frac{s}{x},\frac{t}{y}\right\} \vee \exists z\left((A^*)'\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge \left(TC_{x,y}(A^*)'\right)(z,t)\right)$. Clearly, the sequent $(A^*)'\left\{\frac{s}{x},\frac{t}{y}\right\} \Rightarrow \exists z\left((A^*)'\left\{\frac{s}{x},\frac{z}{y}\right\} \wedge z = t\right)$ is provable in $AL_G^{=}$, and again,

using the induction hypothesis on A together with the TC-counterpart of (3.10) we get that $(TC_{x,y}A)(s,t) \Rightarrow \left(TC_{x,y}(A^*)'\right)(s,t)$ is provable in $AL_G^{=}$. By cuts we get $\vdash_{AL_G^{=}} (TC_{x,y}A)(s,t) \Rightarrow \exists z \left((A^*)'\left\{\frac{s}{x}, \frac{z}{y}\right\} \wedge \left(TC_{x,y}(A^*)'(z,t) \vee z = t\right)\right)$. □

Corollary 3.2.11. *$AL_G^{=}$ and $AL_G^{=,ref}$ are equivalent in the following sense:*

1. $\vdash_{AL_G^{=,ref}} \Gamma \Rightarrow \Delta$ *iff* $\vdash_{AL_G^{=}} \Gamma' \Rightarrow \Delta'$.

2. $\vdash_{AL_G^{=}} \Gamma \Rightarrow \Delta$ *iff* $\vdash_{AL_G^{=,ref}} \Gamma^* \Rightarrow \Delta^*$.

Proof. The left-to-right implications are simply Prop. 3.2.9. For the converse, consider $\vdash_{AL_G^{=}} \Gamma' \Rightarrow \Delta'$. By Prop. 3.2.9 we get that $\vdash_{AL_G^{=,ref}} (\Gamma')^* \Rightarrow (\Delta')^*$. Since by Prop. 3.2.10 we have that $\vdash_{AL_G^{=,ref}} (\varphi')^* \Rightarrow \varphi$ and $\vdash_{AL_G^{=,ref}} \varphi \Rightarrow (\varphi')^*$ for any formula φ in $\mathcal{L}_{RTC}$, using cuts we get that $\vdash_{AL_G^{=,ref}} \Gamma \Rightarrow \Delta$. The proof of 2. is similar. □

3.3 Arithmetics in AL

Definition 3.3.1. The system AL_A^{ref} is obtained by augmenting $AL_G^{=,ref}$ with the following axioms for successor and addition, and the axiom characterizing the natural numbers:

$$
\begin{aligned}
s(x) = 0 \Rightarrow \\
s(x) = s(y) \Rightarrow x = y \\
\Rightarrow x + 0 = x \\
\Rightarrow x + s(y) = s(x + y) \\
\Rightarrow (RTC_{w,u}(s(w) = u))(0, x)
\end{aligned}
$$

Proposition 3.3.2. *In AL_A^{ref}, Rule (3.4) entails all instances of the induction rule in PA_G (Gentzen-style system for Peano arithmetics, see e.g. [108]).*

Proof. This can be achieved by taking φ to be $s(x) = y$ in Rule (3.4) (see [4] for further details). □

We next show that for the standard language of PA the system AL^{ref}_A is equivalent to PA_G, in the sense that there is a provability preserving translation algorithm between them.[8] The followings were proven in [21, 23] using Godel's β-function.[9]

Theorem 3.3.3. *There is a β-translation which transforms a formula of $\mathcal{L}_{RTC}$ into a formula in the language of PA, such that the following holds:*[10]

- $\vdash_{AL^{ref}_A} \varphi \Rightarrow \varphi^\beta$ *and* $\vdash_{AL^{ref}_A} \varphi^\beta \Rightarrow \varphi$.
- $\vdash_{AL^{ref}_A} \Gamma \Rightarrow \Delta$ *iff* $\vdash_{PA_G} \Gamma^\beta \Rightarrow \Delta^\beta$.
  In particular, for Γ, Δ in the language of PA, $\vdash_{AL^{ref}_A} \Gamma \Rightarrow \Delta$ iff $\vdash_{PA_G} \Gamma \Rightarrow \Delta$.

A key proof-theoretical method which arises from Gentzen's consistency proof for PA_G (see [46, 108]) is the assignment of ordinals to proof systems. In Gentzen's method, each system is assigned the least ordinal number needed for its constructive consistency proof. This provides a measure for a complexity of a system which is useful for comparing different proof systems. The constructive consistency proof of PA_G entails that the ordinal number of PA_G is at most ε_0, and another theorem of Gentzen [47] shows that it is exactly ε_0.

Corollary 3.3.4. *The ordinal number of the system AL^{ref}_A is ε_0.*

Note 3.3.5. Due to the provability-preserving translation between $AL^{=,ref}_G$ and $AL^{=}_G$ given in Prop. 3.2.11, it can easily be deduced that the ordinal number of AL_A (the system obtained from $AL^{=}_G$ by the addition of the axioms for successor and addition, and the axiom characterizing the natural numbers) is ε_0 as well.

3.4 Henkin-Style Completeness

In this section we introduce a Henkin-style semantics for $\mathcal{L}_{RTC}$ (see, e.g. [54, 102]) and prove the completeness of AL^{ref}_G with respect to it. In what follows, for the sake of readability, we assume that $\mathcal{L}_{RTC}$ is a language based on a signature without function symbols and without equality.[11]

[8] Moreover, if $\mathcal{L}$ is a language that expands the language of PA, and S and T are two systems expanding AL^{ref}_A and PA_G, respectively, to the language $\mathcal{L}$ by the same set of additional first-order axioms, then, using practically the same method we can prove that S and T are equivalent.

[9] The idea was taken from [105].

[10] We use the standard abbreviations: Γ^β for $\{\varphi^\beta | \varphi \in \Gamma\}$.

[11] The completeness result which follows can easily be extended in the standard way to languages that do include them.

First we recall the concepts of frames and Henkin structures. A frame is a relational structure together with some subset of the powerset of its domain (called its set of admissible subsets).

Definition 3.4.1 (Frames)**.** Let σ be a relational signature with constants. A σ-frame M is a triple $\langle D, I, D' \rangle$, where D is a non-empty domain, I is an interpretation function on σ in D, and $D' \subseteq P(D)$. An assignment v in M is defined as in the standard semantics.

A σ-Henkin structure is a frame whose set of admissible subsets satisfies some closure conditions.

Definition 3.4.2 (Henkin structures)**.** A σ-Henkin structure is a σ-frame M that is closed under parametric definability, i.e., for each formula φ of $\mathcal{L}$ and assignment v in M:

$$\{a \in D \mid M, v[x := a] \models \varphi\} \in D'$$

In case $D' = P(D)$, the σ-Henkin structure is called a standard structure.

Notice that in finite structures every subset of the domain is parametrically definable, hence non-standard σ-Henkin structures are necessarily infinite.

Definition 3.4.3. Let $\mathcal{L}_{RTC}$ be the language based on the signature σ. $\mathcal{L}_{RTC}$ formulas are interpreted in σ-frames as in standard structures, except for the following clause:

- $M, v \models (RTC_{x,y}\varphi)(s,t)$ iff for every $A \in D'$, if $v(s) \in A$ and for every $a, b \in D$: $(a \in A \wedge M, v[x := a, y := b] \models \varphi) \rightarrow b \in A$, then $v(t) \in A$.

The next proposition shows that the above definition is equivalent to Def. 3.1.2 when dealing with standard structures.

Proposition 3.4.4. *Let M be a standard structure and v an assignment in M. Then, the followings are equivalent:*

1. *$v(s) = v(t)$ or there exist $a_0, ..., a_n \in D$ $(n > 0)$ such that $v(s) = a_0$, $v(t) = a_n$, and $M, v[x := a_i, y := a_{i+1}] \models \varphi$ for $0 \leq i \leq n-1$.*
2. *for every $A \subseteq D$, if for every $a, b \in D$: $(a \in A \wedge M, v[x := a, y := b] \models \varphi) \rightarrow b \in A$ and $v(s) \in A$, then $v(t) \in A$.*

Proof. Suppose (1). Let $A \subseteq D$ be a set that is closed under φ, and v an assignment such that $v(s) = a_0 \in A$. If $v(s) = v(t)$ we are done, otherwise, by induction on the sequence $a_0, ..., a_n$ we prove that $v(t) = a_n \in A$. For the converse, assume by contradiction that (1) does not hold. Take A to be that set which includes $v(s)$ as well as all $a_n \in D$ such that there exist $a_0, ..., a_{n-1} \in D$ $(n > 0)$ where $v(s) = a_0$, and $M, v[x := a_i, y := a_{i+1}] \models \varphi$ for $0 \leq i \leq n-1$. By assumption, $v(t) \notin A$, which contradictes (2), because A is obviously φ-closed. □

Definition 3.4.5. Let $T \cup \{\varphi\}$ be a set of formulas in a language based on the signature σ. We say that $T \models_H \varphi$ if every σ-Henkin model of T is a model of φ.

It is straightforward to verify the following:

Theorem 3.4.6 (Soundness Theorem)**.** *Let $T \cup \{\varphi\}$ be a set of sentences in $\mathcal{L}_{RTC}$. Then, $T \vdash_{AL^{ref}_G} \varphi$ implies $T \models_H \varphi$.*

The main result of this chapter is Theorem 3.4.7 below, which we shall prove using several lemmas and definitions.

Theorem 3.4.7 (Completeness Theorem)**.** *Let $T \cup \{\varphi\}$ be a set of sentences in $\mathcal{L}_{RTC}$. Then, $T \models_H \varphi$ implies $T \vdash_{AL^{ref}_G} \varphi$.*

We prove the theorem in the standard method by showing that if $T \nvdash_{AL^{ref}_G} \varphi$, then $T \nvDash_H \varphi$. First, we extend the language $\mathcal{L}_{RTC}$ to a language $\mathcal{L}'_{RTC}$ by adding to it countably many new constant symbols, $c_1, c_2, ...$, and countably many new monadic predicates, $P_1, P_2, ...$ It is easy to see that $T \nvdash_{AL^{ref}_G} \varphi$ in the new language as well.

Definition 3.4.8. We say that a set of $\mathcal{L}'_{RTC}$ sentences Γ contains Henkin witnesses if the followings hold:

1. if $\exists x \varphi \in \Gamma$, then $\varphi\left\{\frac{c}{x}\right\} \in \Gamma$ for some constant c.

2. if $\neg (RTC_{x,y}\varphi)(s,t) \in \Gamma$, then $P(s) \wedge \forall x, y (P(x) \wedge \varphi(x,y) \rightarrow P(y)) \wedge \neg P(t) \in \Gamma$ for some monadic predicate P.

3. if φ is a formula of $\mathcal{L}'_{RTC}$ such that . $Fv(\varphi) = \{x\}$, then $\forall x (P(x) \leftrightarrow \varphi) \in \Gamma$ for some monadic predicate P.

Lemma 3.4.9. Let P be a monadic predicate and ψ a formula of $\mathcal{L}'_{RTC}$. Then, $P(s), \forall x,y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)), \neg P(t) \vdash_{AL^{ref}_G} \neg (RTC_{x,y}\psi)(s,t)$.

Proof. Follows immediately from Rule (3.4) (taking $\varphi(x,y) := \psi(x,y)$ and $\psi(x) := P(x)$) using cuts. □

Lemma 3.4.10. *Let T be a set of sentences in $\mathcal{L}'_{RTC}$ such that $T \nvdash_{AL^{ref}_G} \varphi$, and let θ be a sentence of the form $\forall x\,(P(x) \leftrightarrow \psi)$, where P is a fresh monadic predicate (i.e. does not occur in $T \cup \{\varphi, \psi\}$). Then, $T, \theta \nvdash_{AL^{ref}_G} \varphi$.*

Proof. Suppose by contradiction that $T, \forall x\,(P(x) \leftrightarrow \psi) \vdash_{AL^{ref}_G} \varphi$, where P is a fresh monadic predicate. Therefore, there is a proof of φ from $T \cup \{\forall x\,(P(x) \leftrightarrow \psi)\}$ in AL^{ref}_G. First we rename all bound variables in the proof (apart from x in the formula $\forall x\,(P(x) \leftrightarrow \psi)$) with new variables (not occurring in the proof or in $\forall x\,(P(x) \leftrightarrow \psi)$). Now, we replace all the occurrences of formulas of the form $P(t)$ in the proof by $\psi\left\{\frac{t}{x}\right\}$. Then, every occurrence of $\forall x\,(P(x) \leftrightarrow \psi)$ in the proof becomes an occurrence of $\forall x\,(\psi \leftrightarrow \psi)$, which of course is provable in AL^{ref}_G. We now show that the replacement procedure indeed produces a proof of φ from T in AL^{ref}_G. It is straightforward to show that if the replacement is done on an axiom, then the result is still an axiom of AL^{ref}_G. It is also easy to verify that all the inference rules apply equally to the formulas after the replacement. Also notice that since P does not occur in $T \cup \{\varphi\}$, the replacement procedure applied to a formula in $T \cup \{\varphi\}$ results in the same formula. This shows that there is a proof of φ from T in AL^{ref}_G, i.e., $T \vdash_{AL^{ref}_G} \varphi$, which is a contradiction. □

Lemma 3.4.11 (Lindenbaum Lemma)**.** *There exists an extension of T to a set of sentences T' in the language $\mathcal{L}'_{RTC}$ such that:*

1. *T' is a maximal theory in $\mathcal{L}'_{RTC}$ such that $T' \nvdash_{AL^{ref}_G} \varphi$.*
2. *T' contains Henkin witnesses.*

Proof. First, fix two enumerations: one of all sentences of $\mathcal{L}'_{RTC}$: $\psi_1, \psi_2, ...$; and one of all the formulas of $\mathcal{L}'_{RTC}$ with one free variable x: $\theta_1, \theta_2, ...$ We define a sequence of theories $T_0, T_1, ...$ inductively in the following way: $T_0 = T$, and for $i > 0$ T_i is constructed from T_{i-1} as follows:

1. If $i = 2n - 1$ for some $n \in \mathbb{N}$, then:

(a) If $T_{i-1} \cup \{\psi_n\} \vdash_{AL_G^{ref}} \varphi$, then $T_i = T_{i-1}$.

(b) If $T_{i-1} \cup \{\psi_n\} \nvdash_{AL_G^{ref}} \varphi$, then:

i. If ψ_n is not of the form $\exists x\psi$ or $\neg\left(RTC_{x,y}\psi\right)(s,t)$: $T_i = T_{i-1} \cup \{\psi_n\}$.

ii. If $\psi_n = \exists x\psi$, then $T_i = T_{i-1} \cup \left\{\psi_n, \psi\left\{\frac{c_j}{x}\right\}\right\}$, where c_j is a fresh variable not in T_{i-1}.

iii. If $\psi_n = \neg\left(RTC_{x,y}\psi\right)(s,t)$, then:
$T_i = T_{i-1} \cup \{\psi_n, P_j(s) \wedge \forall x, y\left(P_j(x) \wedge \psi(x,y) \rightarrow P_j(y)\right) \wedge \neg P_j(t)\}$, where P_j is a fresh monadic predicate not in T_{i-1}.

2. If $i = 2n$ for some $n \in \mathbb{N}$, then $T_i = T_{i-1} \cup \{\forall x\left(P_j(x) \leftrightarrow \theta_n\right)\}$, where P_j is a fresh monadic predicate not in T_{i-1}.

We show that for every $i \in \mathbb{N}$, $T_i \nvdash_{AL_G^{ref}} \varphi$. First notice that Lemma 3.4.10 entails that if $T_{2n-1} \nvdash_{AL_G^{ref}} \varphi$, then $T_{2n} \nvdash_{AL_G^{ref}} \varphi$. In cases where $i = 2n-1$: Cases (a) and (bi) are trivial, and Case (bii) is provable just as in the standard completeness proof for FOL. We prove here Case (biii). Assume by contradiction that $T_{i-1}, \neg\left(RTC_{x,y}\psi\right)(s,t), P_j(s) \wedge \forall x, y\left(P_j(x) \wedge \psi(x,y) \rightarrow P_j(y)\right) \wedge \neg P_j(t) \vdash_{AL_G^{ref}} \varphi$. Since $P_j(s) \wedge \forall x, y\left(P_j(x) \wedge \psi(x,y) \rightarrow P_j(y)\right) \wedge \neg P_j(t) \vdash_{AL_G^{ref}} \neg\left(RTC_{x,y}\psi\right)(s,t)$ by Lemma 3.4.9, we get that $T_{i-1}, P_j(s) \wedge \forall x, y\left(P_j(x) \wedge \psi(x,y) \rightarrow P_j(y)\right) \wedge \neg P_j(t) \vdash_{AL_G^{ref}} \varphi$. Now, P_j is a monadic predicate which does not appear in $T_{i-1} \cup \{\varphi\}$. Therefore it is straightforward to verify that by replacing all occurrences of formulas of the form $P_j(r)$ in the proof with $\left(RTC_{x,y}\psi\right)(s,r)$ we get a proof in AL_G^{ref} of φ from $T_{i-1} \cup \{\left(RTC_{x,y}\psi\right)(s,s), \forall x, y\left(\left(RTC_{x,y}\psi\right)(s,x) \wedge \psi(x,y) \rightarrow \left(RTC_{x,y}\psi\right)(s,y)\right), \neg\left(RTC_{x,y}\psi\right)(s,t)\}$. Now, $\left(RTC_{x,y}\psi\right)(s,s)$ is an axiom of AL_G^{ref}, and using Rule (3.5) we can deduce that $\vdash_{AL_G^{ref}} \forall x, y\left(\left(RTC_{x,y}\psi\right)(s,x) \wedge \psi(x,y) \rightarrow \left(RTC_{x,y}\psi\right)(s,y)\right)$. Hence, we get that $T_{i-1} \cup \{\neg\left(RTC_{x,y}\psi\right)(s,t)\} \vdash_{AL_G^{ref}} \varphi$, which contradicts the original assumption that $T_{i-1} \cup \{\psi_i\} \nvdash_{AL_G^{ref}} \varphi$. Therefore, $T_i \nvdash_{AL_G^{ref}} \varphi$.
We now define $T' = \bigcup_{i=0}^{\infty} T_i$. The construction of T' entails that it contains Henkin witnesses. □

Definition 3.4.12. Define a σ-frame M by:

- $D = \{c \mid c \text{ is a constant}\}$
- $D' = \{\{c \mid P(c) \in T'\} \mid P \text{ is a monadic predicate}\}$

- $\langle c_1, ..., c_n \rangle \in I(P)$ iff $P(c_1, ..., c_n) \in T'$
- $I(c) = c$

Notice that $D' = \{I(P) \mid P \text{ is a monadic predicate}\}$.
It is standard to verify the following lemma:

Lemma 3.4.13. *Let ψ be a formula in $\mathcal{L}'_{RTC}$. The following holds:*

- $M, v \models \psi$ *iff* $M \models \psi\left\{\frac{v(x_1)}{x_1}, ..., \frac{v(x_n)}{x_n}\right\}$, *where* $Fv(\psi) = \{x_1, ..., x_n\}$.
- $T' \models_H \forall x\psi$ *iff* $T' \models_H \psi\left\{\frac{c}{x}\right\}$ *for every constant c.*

Lemma 3.4.14. *M is a σ-Henkin structure.*

Proof. This is immediate since T' contains Henkin witnesses of the third type in Definition 3.4.8, i.e., a monadic predicate was introduced for each "parametrically" definable subset (using the new constant symbols instead of the parameters). To see this, let ψ be a formula such that $Fv(\psi) = \{x_1, ..., x_n\}$, and let v be an assignment in M. Then, $\{a \in D \mid M, v[x_1 := a] \models \psi\} = \left\{a \in D \mid M, v[x_1 := a] \models \psi\left\{\frac{v(x_2)}{x_2}, ..., \frac{v(x_n)}{x_n}\right\}\right\}$. Now in T' we have a monadic predicate which forms a Henkin witness for $\psi\left\{\frac{v(x_2)}{x_2}, ..., \frac{v(x_n)}{x_n}\right\}$, denote it by $P_k(x_1)$. Thus, $\left\{a \in D \mid M, v[x_1 := a] \models \psi\left\{\frac{v(x_2)}{x_2}, ..., \frac{v(x_n)}{x_n}\right\}\right\} = I(P_k) \in D'$. □

Proposition 3.4.15. *For every sentence θ in $\mathcal{L}'_{RTC}$: $M \models \theta$ iff $\varphi \in T'$.*

Proof. By induction on θ. The base case follows immediately from the definition of M. For the connectives and quantifiers the proof is similar to the standard proof for FOL (using Henkin witnesses for existential formulas). We next prove the case for $\theta = (RTC_{x,y}\psi)(s,t)$.
($\Rightarrow$) : Assume $M \models (RTC_{x,y}\psi)(s,t)$. Hence, for every monadic predicate P, if $I(s) \in I(P)$ and for every $a, b \in D$:$(a \in I(P) \wedge M, v[x := a, y := b] \models \psi) \rightarrow b \in I(P)$, then $I(t) \in I(P)$. Using the induction hypothesis and the base case we get that for any monadic predicate P, if $P(s) \in T'$ and for any two constants a, b, if $P(a) \in T'$ and $\psi(a,b) \in T'$ then $P(b) \in T'$, then $P(t) \in T'$. From this we deduce (using Lemma 3.4.13) that for any monadic predicate P, if

$P(s) \in T'$ and $\forall x, y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)) \in T'$, then $P(t) \in T'$. Assume by contradiction that $(RTC_{x,y}\psi)(s,t) \notin T'$. Then, by the maximality of T', we get that $\neg(RTC_{x,y}\psi)(s,t) \in T'$. Therefore, T' contains a Henkin witness of the type $P(s) \wedge \forall x, y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)) \wedge \neg P(t)$ for some monadic predicate P. From this we get that $P(s) \in T'$, $\forall x, y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)) \in T'$ and $\neg P(t) \in T'$. But this contradicts the consistency of T', since we showed that for any monadic predicate P, if $P(s) \in T'$ and $\forall x, y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)) \in T'$, then $P(t) \in T'$. Therefore, we conclude that $(RTC_{x,y}\psi)(s,t) \in T'$.
($\Leftarrow$) : Assume $M \nvDash (RTC_{x,y}\psi)(s,t)$. So, $M \models \neg(RTC_{x,y}\psi)(s,t)$ and there exists a monadic predicate P such that $I(s) \in I(P)$, $I(t) \notin I(P)$ and for every $a, b \in D$: $(a \in I(P) \wedge M, v\,[x := a, y := b] \models \psi) \rightarrow b \in I(P)$. By the induction hypothesis and the base case we get that there exists a monadic predicate P such that $P(s) \in T'$, $P(t) \notin T'$, and for any two constants a, b, if $P(a) \in T'$ and $\psi(a,b) \in T'$ then $P(b) \in T'$. Therefore, by the maximality of T', $P(s) \in T'$, $\neg P(t) \in T'$ and $\forall x, y\,(P(x) \wedge \psi(x,y) \rightarrow P(y)) \in T'$ (the latter holds since assuming otherwise leads to a contradiction using a Henkin witness for an existential formulas). This entails, by Lemma 3.4.9, $T' \vdash_{AL_G^{ref}} \neg(RTC_{x,y}\psi)(s,t)$. Thus, if $(RTC_{x,y}\psi)(s,t) \in T'$. we get a contradiction to the consistency of T', therefore $(RTC_{x,y}\psi)(s,t) \notin T'$. □

Since $T \subseteq T'$ and $\varphi \notin T'$, we get that $M \models T$ while $M \nvDash \varphi$. Hence, $T \nvDash_H \varphi$, which finally proves Theorem 3.4.7.

Note 3.4.16. Using a similar method to that described in this section it is straightforward to provide a Henkin-style semantics for $\mathcal{L}_{TC}$ and to prove that AL_G is complete with respect to it.

Chapter 4

Constructive (Intuitionistic) Ancestral Logic

In this chapter we present a constructive view of ancestral logic.[1] We introduce a formal system for iAL, iAL_S, which is a refinement of iAL_G and a natural extension of the formal system for intuitionistic FOL given in [31] (which is here denoted by $iFOL_S$). The system $iFOL_S$ can be viewed and used as a dependently typed abstract programming language, particularly suitable for correct-by-construction style of programming [31]. This is because $iFOL_S$ has the key feature that proofs of specifications in its proof system carry their computational content in realizers. Hence, proving a formula in $iFOL_S$ results in a realizer that can be thought of as holding the computational element of a program, and so one can extract programs from proofs that $iFOL_S$ specifications are solvable. Now iAL_S enjoys these features too, but has greater expressive and proof-theoretic power. Accordingly, iAL_S can serve as a much better framework for specifying, developing, and reasoning about programs. The computational power that iAL_S has beyond that of $iFOL_S$ is given by its realizer for the transitive closure operator, which has the property that each of its particular instances corresponds to some recursive program.

The chapter is organized as follows: Section 4.1 reviews the formal system and realizability semantics for $iFOL$, as presented in [31]. In Section 4.2 they are extended to a formal system and a realizability semantics for iAL. We prove that the system for iAL, iAL_S, is strongly sound with respect to this semantics by showing that provable

[1] In this chapter we focus on the non-reflexive TC operator.

formulas are *uniformly realizable*. In Section 4.3 applications of iAL in computer science are explored. In particular, we show that iAL_S subsumes Kleene Algebras with tests [66] and thus serves as a natural programming logic for proving properties of program schemes.

This chapter is mainly based on [24, 25].

4.1 Formal System for $iFOL$

4.1.1 Realizability semantics for $iFOL$

This section reviews the semantics of evidence for *pure*[2] intuitionistic first-order logic ($iFOL$) along the lines of [31], which is simply a compact type theoretic restatement of the BHK semantics.

Let $\mathcal{L}$ be a first-order language consisting of predicates $P_i^{n_i}$ (with arity n_i), together with a designated symbol D. A structure M for $\mathcal{L}$ assigns to D a constructive type, $[D]_M$ (the domain of discourse),[3] and assigns to every formula A over $\mathcal{L}$ a type of objects denoted $[A]_M$, called the *evidence* for A with respect to M.
Convention. In what follows, when there is only one structure involved, the subscript M is sometimes omitted from $[A]_M$.

Below is how evidence is defined for the various kinds of first-order propositional functions. The definition also implicitly provides a formal syntax of first-order formulas.

Definition 4.1.1 ($iFOL$ formulas and their evidence)**.**

- **atomic predicates** $P_i^{n_i}$ are interpreted as functions from $[D]_M^{n_i}$ into $\mathbb{P}$, the type of propositions, and for the atomic proposition $P_i^{n_i}(x_1, ..., x_{n_i})$, the basic

[2] For simplicity we focus on *pure* languages, i.e., equality, constants, and functions are not built-in primitives of the language. However, the systems presented in this chapter can easily be extended to include them.

[3] As a first approximation one may think of types as *constructive sets* [12]. In [2] it was shown how to interpret constructive sets as types in Intuitionistic Type Theory (ITT) [78, 79, 87], and this approach was implemented in MetaPRL for Constructive Type Theory (CTT) [57]. Intuitionists might refer to *species* instead. We do not analyze the structure of the domain further and do not examine the equality relation on the type when dealing with the pure first-order theory, as is standard practice.

evidence must be supplied, say by objects p_i. In the uniform treatment, all of these objects are considered to be equal, and are denoted by the unstructured atomic element $\star$.[4] Thus if an atomic proposition is known by atomic evidence, the evidence is the single element $\star$ of the unit type, $\{\star\}$.[5]

- **conjunction** $[A \wedge B] \ = \ [[A]] \times [B]$, the Cartesian product.
- **existential** $[\exists x.B(x)] \ = \ x : [D]_M \times [B(x)]$, the dependent product.[6]
- **implication** $[A \to B] \ = \ [A] \mapsto [B]$, the function space.[7]
- **universal** $[\forall x.B(x)] \ = \ x : [D]_M \mapsto [[B(x)]]$, the dependent function space.[8]
- **disjunction** $[A \vee B] \ = \ [A] + [B]$, disjoint union.
- **false** $[False] \ = \ \emptyset$, the void type.

Negation is defined by $\neg A := A \to False$.

The following proposition, which is proven classically in [31], shows that the above evidence semantics can be read classically, and it corresponds to Tarski's semantics for FOL and AL.

Proposition 4.1.2. *A formula A is satisfied in a structure M if and only if there is evidence in $[A]_M$.*

4.1.2 The Proof System $iFOL_S$

We next present the proof system $iFOL_S$ for $iFOL$ adopting the presentation style from [31], where the computational content is made explicit using the evidence semantics. The rules of the system are presented in the "top-down style" (also called

[4]Officially this can be done using the set type $\{sq(p_i)|P_i^{n_i}(x_1, ..., x_{n_i})\}$, where $sq(p_i)$ "squashes" the evidence p_i to $\star$ [26].

[5]It might seem that we should introduce atomic evidence terms that might depend on parameters, say $p(x, y)$ as the *atomic evidence* in the atomic proposition $P(x, y)$, but this is unnecessary and uniformity would eliminate any significance to those terms. In CTT and ITT, the evidence for atomic propositions such as equality and ordering is simply an unstructured term such as $\star$.

[6]This type is sometimes denoted by $\Sigma_{x:D}B(x)$.

[7]This function space is interpreted type theoretically and is assumed to consist of *effectively computable deterministic functions.*

[8]This type is sometimes denoted by $\Pi_{x:D}B(x)$.

refinement style), in which the goal comes first and the rule name with parameters generates subgoals. Thus the sequent style trees are grown with the root at the top. This top-down, goal oriented style is common to all of the proof assistants which work in the highly successful tactic mechanism of the Edinburgh LCF proof assistant [49], and it is also compatible with the style for rules and proofs used in the Nuprl proof assistant [30].

The rules are presented in the style of McCarthy's abstract syntax [80]. For each rule a name is provided that is the outer operator of a proof expression with slots to be filled in as the proof is developed. Each sequent in the rules is build so that the left-hand side contains the context, and the right-hand side contains a type and its evidence term. For the right-hand sides the notation "type *by* term" is used (as opposed to "term : type"). The construction rules (often called introduction rules) apply to terms on the right-hand side of the sequent and introduce the canonical proof terms. For each of these construction rules, the constructor needs subterms which build the component pieces of evidence. For each connective and quantifier we also have rules for their occurrence on the left of the sequent. These are the rules for decomposing a connective or a quantifier. They tell us how to use the evidence that was built with the corresponding construction rules, and the formula being decomposed is always named by a label in the list of hypotheses, so there is a variable associated with each rule application. (For a more detailed explanation of the syntax used in the proof rules see [31].)

Definition 4.1.3. The proof system $iFOL_S$ is given in Figure 4.1.

Note 4.1.4. The meta variable d is used to denote objects in the domain of discourse D. In the classical evidence semantics, we assume that D is non-empty by postulating the existence of some d_0.

Note 4.1.5. The system $iFOL_S$ is based on the classical proof system for intuitionistic FOL, with the addition of the realizers. If the realizers are omitted from the rules, we get a natural deduction version of $\mathcal{LJ}$.

[9]This notation shows that $ap(f; sl_a)$ is substituted for v in $g(v)$. In the CTT logic we stipulate in the rule that $v = ap(f; sl_a)\ in\ B$.

[10]In the CTT logic, we use equality to stipulate that $v = ap(f; d)\ in\ B(v)$ just before the hypothesis $v : B(d)$.

Figure 4.1: The proof system $iFOL_S$

And Construction
$H \vdash A \wedge B$ *by* $pair(slot_a; slot_b)$
$H \vdash A$ *by* $slot_a$
$H \vdash B$ *by* $slot_b$

Implication Construction
$H \vdash A \to B$ *by* $\lambda(x.slot_b(x))$ *new* x
$H, x : A, H' \vdash B$ *by* $slot_b(x)$

Hypothesis
$H, d : D, H' \vdash \ d \in D$ *by* $obj(d)$

$H, x : \varphi, H' \vdash \ \varphi$ *by* $hyp(x)$

Or Construction
$H \vdash A \vee B$ *by* $inl(slot_l)$
$H \vdash A$ *by* $slot_l$

$H \vdash A \vee B$ *by* $inr(slot_r)$
$H \vdash B$ *by* $slot_r$

Exists Construction
$H \vdash \exists x.B(x)$ *by* $pair(d; slot_b(d))$
$H \vdash d \in D$ *by* $obj(d)$
$H \vdash B(d)$ *by* $slot_b(d)$

All Construction
$H \vdash \forall x.B(x)$ *by* $\lambda(x.slot_b(x))$
$H, x : D, H' \vdash B(x)$ *by* $slot_b(x)$

And Decomposition
$H, x : A \wedge B, H' \vdash G$ *by* $spread(x; l, r.slot_g(l, r))$ *new* l, r
$H, l : A, r : B, H' \vdash G$ *by* $slot_g(l, r)$

Implication Decomposition
$H, f : A \to B, H' \vdash G$ *by* $apseq\,(f; slot_g; v.sl_g\,[^{ap(f;slot_a)}/_v])$ *new* v[9]
$H \vdash A$ *by* $slot_a$
$H, v : B, H' \vdash G$ *by* $slot_g(v)$

Or Decomposition
$H, y : A \vee B, H' \vdash G$ *by* $decide(y; l.slot_{left}(l); r.slot_{right}(r))$
$H, l : A, H' \vdash G$ *by* $slot_{left}(l)$
$H, r : B, H' \vdash G$ *by* $slot_{right}(r)$

Exists Decomposition
$H, x : \exists y.B(y), H' \vdash G$ *by* $spread(x; d, r.slot_g(d, r))$ *new* d, r
$H, d : D, r : B(d), H' \vdash G$ *by* $slot_g(d, r)$

All Decomposition
$H, f : \forall x.B(x), H' \vdash G$ *by* $apseq(f; d; v.slot_g\,[^{ap(f;d)}/_v])$
$H \vdash d \in D$ *by* $obj(d)$
$H, v : B(d), H' \vdash G$ *by* $slot_g(v)$[10]

False Decomposition
$H, f : False, H' \vdash G$ *by* $any(f)$

Turning to the semantics of the system, in [31] the notion of uniform validity was introduced. A proposition is said to be uniformly valid if it has *polymorphic* evidence that does not depend on the specific evidence which is assigned to atomic propositions in a given structure. Intuitively, the notion of uniform validity is concerned not with the statements that are true in every structure, but with the statements that are true *in the same way* in every structure. Uniformity provides an effective tool for semantics, because one can establish uniform validity by exhibiting a single polymorphic object. The notion of uniformity was then used in [31] to provide a constructive completeness theorem (for its intended semantics) of pure intuitionistic first-order logic.

4.1.3 Constructive (Intuitionistic) Metatheory

Formal semantics of the intuitionistic logics we present is based on extensional constructive type theories such as Intuitionistic Type Theory (ITT) [78, 79, 87] or Constructive Type Theory (CTT) [29, 30]. Critical to constructive type theory is the underlying *computation system*, thus below we survey its key properties. This is essentially the untyped programming language underlying the type theory, and thus underlying $iFOL_S$ and iAL_S (which is introduced in the sequel).

The data and the programs of the computation system are given by closed terms. To explain what we mean by a *closed term* we need to examine the structure of terms further. The precise definition of terms provides the syntax of the programming language. All terms have an outer operator which determines whether a term is *canonical* or *non-canonical*. Table 4.2 below contains the operators that will be used here: the canonical ones, and the corresponding non-canonical ones associated with the canonical. For instance, we use the term $pair(a; b)$ (or more succinctly $< a, b >$) for And construction rule, and for the corresponding decomposition rule we use $spread(x; l, r.t(l, r))$ where the binding variables l, r have a scope that is the subterm $t(l, r)$. The reason to use *spread* instead of the more familiar operators for decomposing a pair p such as $first(p)$ and $second(p)$ (or $p.1$ and $p.2$) is that we need to indicate how the subformulas of a conjunction will be named in the hypothesis list.[11] The canonical terms correspond to the values of the computation system, the data. The non-canonical expressions evaluate to canonical ones under the evaluation

[11] See [31] for a more detailed description of the operators.

Figure 4.2: The Computational Operators

Canonical	**Non-canonical**
pair	*spread*
inl, inr	*decide*
λ	$ap, apseq$
[ ] (list constructor)	*concat*
$\star$	

rules. They correspond in a sense to programs applied to data. This terminology is used by Martin-Löf in relating his type theory to programming languages [78].

Terms having a canonical operator are called *values* and those with non-canonical operators are called *operations*. In building terms we use bound variables, e.g., the notation for functions has the form $\lambda(x.t(x))$, where the subterm x is a *bound variable* with binding operator λ. If the subterm $t(x)$ has an occurrence of the variable x , the expression $t(x)$ is neither an operator, nor a value; it is an *open* term. Open terms have a special status in the computation system because of the ability to substitute terms for variables within open terms. Typically we think of substituting values for variables, but there are reasons that we must also consider substituting variables for variables.

A computation is defined as a sequence of *rewritings* or *reductions* of terms to other terms according to very explicit rules. When all its slots are filled in, each rule form of the proof systems $iFOL_S$ and iAL_S becomes a term in an applied lambda calculus, and there are computation rules that define how to reduce these terms. These rules are given in detail in several papers about Computational Type Theory and Intuitionistic Type Theory, e.g., [30, 78]. The computation systems of $iFOL_S$ and iAL_S have the property that all reductions will terminate in values which are said to be the value of the expression. Canonical terms, such as $\lambda(x.x)$, reduce to themselves. A non-canonical term such as $ap(\lambda(x.x); y)$ reduces in one step to y.

It is important to notice that all of the programs and data of these logics are untyped. They are also said to be *polymorphically* typed because many terms, such as $\lambda(x.x)$, have the type $A \to A$ for any proposition A of the logic.

4.2 Formal System for iAL

Notation. In this chapter, for readability, we use a more succinct presentation of the TC operator than that used in the previous chapter. Thus, the notation $A_{x,y}$ will be used to specify that we treat the formula A as defining a binary relation with respect to variables x and y (x and y distinct variables), and other free variables that may occur in A are taken as parameters. The standard abbreviation $A^+_{x,y}$ is used for $TC_{x,y}A$ (the full notation used in the previous chapter). We also use $A_{x,y}(u,v)$ as an abbreviation of $A\left\{\frac{u}{x},\frac{v}{y}\right\}$. If there is no chance of confusion, the subscript x,y is sometimes omitted.

4.2.1 Realizability Semantics for iAL

To provide evidence for the transitive closure operator, generic and polymorphic constructs are used, in the spirit of using polymorphic functions, pairs, and tags. To know $A^+(x,y)$, a *list* of elements of D is constructed, say $[d,...,d']$[12], and a list of evidence terms $[r,...,r']$ such that r is evidence for $A(x,d)$ and r' is evidence for $A(d',y)$ and the intermediate terms form an *evidence chain.* That is, if d^+ is the list of elements and r^+ is the list of evidence terms, we have that the first element of d^+ is d_1 and first of r^+ is r_1, where $r_1 \in [A(x,d_1)]$, the next element of d^+ is d_2 and the next element of r^+ is r_2, evidence for $A(d_1,d_2)$, and so forth. It is important to notice that the concept of lists is subsumed into the realizers and does not appear in the logic itself.[13]

Definition 4.2.1 (iAL formulas and their evidence)**.** iAL formulas are defined as $iFOL$ formulas with the addition of the following clauses:

- If A is a formula, x,y distinct variables, and u,v variables, then $A^+_{x,y}(u,v)$ is a formula.
- The evidence type for $A^+_{x,y}(u,v)$ consists of lists of the form

$$[\langle d_0,d_1,r_1\rangle, \langle d_1,d_2,r_2\rangle, ..., \langle d_n,d_{n+1},r_{n+1}\rangle]$$

[12]Notice that the notation [] is overloaded and used for lists (and list constructors), as well as for evidence types.

[13]In CTT used in [29], the evidence type for the transitive closure operator can be defined using intersection type, the type $\bigcap_{x:A} B(x)$.

where $n \geq 0$, $d_0, d_1, ..., d_{n+1} : [D]_M$, $d_0 = u$, $d_{n+1} = v$, and $r_i \in [A_{x,y}(d_{i-1}, d_i)]$ for $1 \leq i \leq n+1$.

Notice that the realizers for transitive closure formulas are all polymorphic and thus independent of realizers for particular atomic formulas.

It is also interesting to note the following: according to a standard mathematical definition of the transitive closure operator, $R^+(x, y)$ can be taken as the formula $\exists n \left(N(n) \wedge R^{(n)}(x, y)\right)$, where $R^{(0)} = R$ and $R^{(n+1)} = R^{(n)} \circ R$. This of course is not a legal formula in our language, but this is intuitively what we mean, if we had the natural numbers, N, at our disposal. The realizer for such "formula" will be of the form: $\langle n, \langle isnat(n), \langle x, d_1, ..., d_n, y, \langle r_1, ..., r_{n+1}\rangle\rangle\rangle\rangle$, where $isnat(n)$ realizes $N(n)$. The realizer of the transitive closure correlates nicely to this realizer. A realizer of the form $\langle n, \langle isnat(n), \langle x, d_1, ..., d_n, y, \langle r_1, ..., r_{n+1}\rangle\rangle\rangle\rangle$ can be easily converted into the form $[\langle x, d_1, r_1\rangle, \langle d_1, d_2, r_2\rangle, ..., \langle d_n, y, r_{n+1}\rangle]$ simply by rearranging the data. For the converse, the data can also be rearranged, but some additional data is required: n, which is the length of the list minus 1; and the realizer for it being a natural number, which is available as the length of a list is always a natural number.[14]

4.2.2 The Proof System iAL_S

We present a proof system for iAL which extends $iFOL_S$ by adding construction and decomposition rules for the transitive closure operator. The rules for the transitive closure are the same as in iAL_G (written in the same form as the rules of $iFOL_S$), with the addition of the realizers. We use here the standard canonical operator [] for list constructor, and the non-canonical operator associated with it, *concat*, for concatenating lists.

Definition 4.2.2. The proof system iAL_S is defined by adding to $iFOL_S$ the following rules for the transitive closure operator.

- **TC Base**

$H, x : D, y : D, H' \vdash A^+(x, y)$ *by* $[\langle x, y, slot\rangle]$
$H, x : D, y : D, H' \vdash A(x, y)$ *by* $slot$

[14] By interpreting the naturals as lists on the unit type, the definition of the transitive closure operator by means of the natural numbers is an instance of our definition using lists.

- **TC Trans**

$H, x : D, y : D, H' \vdash A^+(x, y)$ *by* $concat\,(slot_l, slot_r)$
$H, x : D, z : D, H' \vdash A^+(x, z)$ *by* $slot_l$
$H, y : D, z : D, H' \vdash A^+(z, y)$ *by* $slot_r$

- **TC Ind**

$H, x : D, y : D, r^+ : A^+(x, y), H' \vdash B(x, y)$ *by* $tcind\,(r^+; u, v, w, b_1, b_2.tr(u, v, w, b_1, b_2);$
$u, v, r.st(u, v, r))$
$H, u : D, v : D, w : D, b_1 : B(u, v), b_2 : B(v, w), H' \vdash B(u, w)$ *by* $tr(u, v, w, b_1, b_2)$
$H, u : D, v : D, r : A(u, v), H' \vdash B(u, v)$ *by* $st(u, v, r)$
where u, v, w are fresh variables.

Notes:

- Rule TC Base states that the list consisting of the singe triple $\langle x, y, r\rangle$, where r realizes $A(x, y)$, is a realizer for the transitive closure $A^+(x, y)$.

- The crucial point about Rule TC Trans is that it does not nest lists of triples for the same goal. Instead, the lists are "flattened out" as proofs are constructed. This entails that proofs of transitive closure formulas have a distinguished realizer. Furthermore, it provides an adequate mechanism for creating a flat chain of evidence needed for the transitive closure induction rule (TC Ind). We have found this to be the minimal, most natural structure needed for handling a TC-chain.

- The realizer for Rule TC Ind computes on the list r^+ and is recursively defined as follows:
$tcind(r^+; u, v, w, b_1, b_2.tr(u, v, w, b_1, b_2); u, v, r.st(u, v, r))$ computes to:

  if $base(r^+)$ then
  $st(r^+.1_1, r^+.1_2, r^+.1_3)$
  else
  $tr(r^+.1_1, r^+.1_2, r^+.2_2, tcind\,\big(c(r^+); u, v, w, b_1, b_2.tr(u, v, w, b_1, b_2); u, v, r.st(u, v, r)\big)$

The operator $base(r^+)$ is true when r^+ is simply the singleton triple. We use the notation $r^+.u_i$ to denote the ith elements of the triple ($i \in \{1,2,3\}$) in the uth place in the list. The operator $c(r^+)$ returns the list r^+ without its first element.

We next give some examples of fundamental, intuitionistically valid statements concerning the TC operator that are provable in iAL_S. They are counterparts of statements proven in iAL_G. For simplicity of presentation we use the "forgetful" abbreviation $T \vdash A$ (where T is a set of formulas), omitting all the declarations and the realizers.

Proposition 4.2.3. *The following are provable in iAL_S:*

$$A(x,z), A^+(z,y) \vdash A^+(x,y) \qquad A^+(x,z), A(z,y) \vdash A^+(x,y) \tag{4.1}$$

$$A^+(x,y) \vdash A(x,y) \vee \exists z \left(A(x,z) \wedge A^+(z,y)\right) \tag{4.2}$$

$$A^+(x,y) \vdash A(x,y) \vee \exists z \left(A^+(x,z) \wedge A(z,y)\right)$$

$$A^+(x,y) \vdash \exists z A(x,z) \qquad A(x,y) \vdash \exists z A(z,y) \tag{4.3}$$

$$\left(A^+\right)^+(x,y) \vdash A^+(x,y) \tag{4.4}$$

$$\vdash \exists x A \leftrightarrow \left(A\left\{\frac{u}{x}\right\} \vee A\left\{\frac{v}{x}\right\}\right)^+_{u,v}(u,v) \quad (\text{where } u,v \text{ are fresh}) \tag{4.5}$$

Proof. The proofs of the the claims are similar to their proofs in iAL_G. We provide here the proof of (4.5) as an example.[15] Denote by $\varphi(u,v)$ the formula $A(u,\overrightarrow{y}) \vee A(v,\overrightarrow{y})$. The right-to-left implication follows from (4.3) since $\exists z\left(A(u,\overrightarrow{y}) \vee A(z,\overrightarrow{y})\right) \vdash \exists x A(x,\overrightarrow{y})$ can be easily proven in $iFOL_S$, and (4.3) entails that $\varphi^+(u,v) \vdash \exists z \varphi(u,z)$. For the left-to-right implication it suffices to prove $d:D, A(d,\overrightarrow{y}) \vdash \varphi^+(u,v)$. Clearly, in $iFOL_S$, $d:D, A(d,\overrightarrow{y}) \vdash \varphi(d,v)$ is provable, from which we can deduce by TC Base $d:D, A(d,\overrightarrow{y}) \vdash \varphi^+(d,v)$. Since we also have $d:D, A(d,\overrightarrow{y}) \vdash \varphi(u,d)$, by (4.1), we obtain $d:D, A(d,\overrightarrow{y}) \vdash \varphi^+(u,v)$. □

It is important to notice that there is a strong connection between our choice for the realizer of the transitive closure and the standard realizers for $iFOL$. For example, (4.5) entails that the existential quantifier is definable in iAL_S. It is

[15] Notice that (4.5) is the defining formula of the existential quantifier in terms of the transitive closure operator, introduced in Section 3.1.

interesting to see how the realizer for the defining formula correlates to the realizer for an existential formula. For instance, the standard realizer for $\exists x P(x)$ is a pair $\langle d, \star \rangle$, since P is an atomic relation. The realizer for the defining formula, $(P(u) \vee P(v))^{+}(u, v)$, is of the form $[\langle d_0, d_1, r_1 \rangle, \langle d_1, d_2, r_2 \rangle, ..., \langle d_n, d_{n+1}, r_{n+1} \rangle]$ where $d_0 = u$, $d_{n+1} = v$, and each r_i is a realizer for $P(d_i) \vee P(d_{i+1})$. Now, suppose we have a realizer of the form $\langle d, \star \rangle$ of $\exists x P(x)$. The realizer for the defining formula in iAL_S is $[\langle u, d, inr(\star) \rangle]$. For the converse, suppose we have a realizer of the form $[\langle d_0, d_1, r_1 \rangle, \langle d_1, d_2, r_2 \rangle, ..., \langle d_n, d_{n+1}, r_{n+1} \rangle]$ where $d_0 = u$, $d_{n+1} = v$. Then we can create a realizer for $\exists x P(x)$ in the following way: if r_1 is $inl(\star)$ return $\langle u, \star \rangle$, else return $\langle d_1, \star \rangle$.

4.2.3 Soundness for iAL_S

Next we prove that iAL_S is sound by showing that every provable formula is realizable, and even uniformly realizable. We do so by providing a semantics to sequents and then proceed by induction on the structure of the proofs. It is important to note that the realizers are all polymorphic since they do not contain any propositions or types as sub-components, and thus serve to provide evidence for any formulas built from any atomic propositions.

Given a type $[D]_M$ (empty or not) as the domain of discourse, atomic propositional functions from $[D]_M$ to propositions, $\mathbb{P}$, and the type theoretic meaning of the logical operators and the transitive closure operator, an iAL_S sequent can be interpreted over dependent types as follows: a sequent $x_1 : T_1, x_2 : T_2(x_1), ..., x_n : T_n(x_1, ..., x_{n-1}) \vdash G(x_1, ..., x_n)$ defines an effectively computable function from an n-tuple of elements of the dependent product of the types in the hypothesis list to the type of the goal, $G(x_1, ..., x_n)$.

Theorem 4.2.4 (Realizability Theorem)**.** *Every provable formula of iAL_S is uniformly realizable.*

Proof. The proof is carried out by induction on the structure of proofs in iAL_S. The atomic (axiomatic) subgoals are of the form $x_1 : T_1, x_2 : T_2(x_1), ..., x_n : T_n(x_1, ..., x_{n-1}) \vdash T_j(x_1, ..., x_n)$, which is clearly realizable. Also, the proof rules for iAL_S show how a realizer for the goal sequent can straightforwardly be constructed given realizers for the subgoals. □

Since realizability semantics is the definition of constructive truth, this theorem allows us to also say that every provable formula is true in every constructive structure.

Theorem 4.2.5 (Soundness Theorem)**.** *Every provable formula of iAL_S is intuitionistically valid.*

4.3 Applications of iAL

4.3.1 Kleene Algebra

Kleene algebra (KA) [64] arises in many areas of computer science, such as automata theory, the design and analysis of algorithms, dynamic logic, and program semantics. There are many interesting models of KA, yet the theory of relational Kleene algebra (RKA) is of practical interest, particularly for programming language semantics and verification [66, 67].

Definition 4.3.1. A *Kleene algebra* (KA) is a structure $(K, +, \cdot, ^*, 0, 1)$, such that $(K, +, \cdot, 0, 1)$ forms an idempotent semi-ring which satisfies the following axioms:[16]

$$(1)\ \ 1 + xx^* \leq x^* \qquad (2)\ \ 1 + x^*x \leq x^*$$
$$(3)\ \ xp \leq x \to xp^* \leq x \qquad (4)\ \ px \leq x \to p^*x \leq x$$

Elements of K, which are denoted by $p, q, r, x, ...$, are called programs. The upper semi-lattice structure induces a natural partial order on any idempotent semi-ring: $x \leq y \leftrightarrow x + y = y$.

Definition 4.3.2. For an arbitrary set U, the set $P(U \times U)$ of all binary relations on U forms a Kleene algebra $R(U)$ with the interpretations $\cup$ for $+$, composition $\circ$ for $\cdot$, empty relation for 0, identity relation for 1 and reflexive transitive closure for *. A Kleene algebra is *relational* (RKA) if it is a subalgebra of $R(U)$ for some U.

Due to the prominence of relational models in programming language semantics and verification, it is of high interest to characterize them axiomatically or otherwise. We next show that iAL_S with equality ($iAL_S^=$) forms an adequate proof system

[16] We omit $\cdot$, writing xy for $x \cdot y$.

for RKA, as RKA can be embedded in $iAL_S^=$ in such a way that any valid RKA expression is translated into a provable $iAL_S^=$ formula.

Definition 4.3.3. The system $iAL_S^=$, for iAL with equality, is obtained from iAL_S by adding to it the following:

- **Reflexivity Axiom**
  $H \vdash x = x \, by \, Eq$
- **Paramodulation Rule**
  $H, x : D, y : D, r : A, H' \vdash A' \; by \; r$
  $H, x : D, y : D, H' \vdash x = y \; by \; slot$
  where A' is obtained from A by replacing free occurrences of x in A with y, with the standard restrictions.

Note 4.3.4. The axioms for symmetry and transitivity of the equality relation are derivable in $iAL_S^=$. Moreover, it can be easily proven that all of the atomic relations respect equality.

Let $\mathcal{L}$ be a first-order language with equality, consisting of *binary* predicates, $P_1, P_2, ...$ The translation from an RKA expression E into an $iAL_S^=$ formula is defined as follows. We use the notation $A_{rep(x,y)}$ to denote the formula obtained from A by replacing the free occurrences of x in A with y (applying the α-rule if necessary).

Definition 4.3.5. For an RKA expression E define $|E|$ inductively by:

- For atomic p assign a distinct predicate P_i and define: $|p| := P_i(x, y)$
- $|0| := False$
- $|1| := x = y$
- $|E_1 + E_2| := |E_1| \vee |E_2|$
- $|E_1 \cdot E_2| := \exists z \left(|E_1|_{rep(y,z)} \wedge |E_2|_{rep(x,z)} \right)$, where z is a fresh variable.
- $|E^*| := x = y \vee |E|^+$
- $|E_1 = E_2| := (|E_1| \to |E_2|) \wedge (|E_2| \to |E_1|)$

Note that $|p \leq q|$ is the formula: $((|p| \vee |q|) \to |q|) \wedge (|q| \to (|p| \vee |q|))$, which is provably equivalent to $|p| \to |q|$. For convenience, in what follows we shall use this as the translation of $p \leq q$.

Theorem 4.3.6. *For E a valid expression of RKA, $|E|$ is provable in $iAL^{=}_{S}$.*

Proof. The only rule in RKA is the transitivity of equality, which translates to the transitivity of $\leftrightarrow$[17]. This, in turn, is clearly provable in $iFOL_S$. It is easy to see that the translation of all the axioms for the idempotent semiring are provable using the proof rules of $iFOL_S$. It remains to show that the translation of axioms (1)–(4) in Def. 4.3.1 are provable in $iAL^{=}_{S}$.

(1): The translation of the axiom is: $[x = y \vee \exists z.P(x,z) \wedge (z = y \vee P^+(z,y))] \to (x = y \vee P^+(x,y))$. If $x = y$, by Or construction (using inl) we get a proof of $x = y \vee P^+(x,y)$. Otherwise, assume $\exists z\,(P(x,z) \wedge (z = y \vee P^+(z,y)))$. If $z = y$, then by the Paramodulation rule we get $P(x,y)$, from which, by TC Base, we get $P^+(x,y)$. If $P^+(z,y)$, then by (4.1) in Prop. 4.2.3 we get $P^+(x,y)$. Applying Or construction (using inr) results in a proof of $x = y \vee P^+(x,y)$.

(2): Symmetric to the proof of (1).

(3): We need to show that the following rule is derivable in $iAL^{=}_{S}$:
$H, x:D, y:D, H' \vdash \exists z\left(P_1(x,z) \wedge \left(z = y \vee P_2^+(z,y)\right)\right) \to P_1(x,y)$
$H, x:D, y:D, H' \vdash \exists z\,(P_1(x,z) \wedge P_2(z,y)) \to P_1(x,y)$
Assume $\exists z\left(P_1(x,z) \wedge \left(z = y \vee P_2^+(z,y)\right)\right)$. If $z = y$ and $P_1(x,z)$, then $P_1(x,y)$ by the Paramodulation rule. Otherwise, suppose $\exists z\left(P_1(x,z) \wedge P_2^+(z,y)\right)$. Assuming $\exists z\,(xP_1z \wedge zP_2y) \to xP_1y$ we can derive $P_2(z,y) \vdash P_1(x,z) \to P_1(x,y)$ using $iFOL_S$ rules. Since clearly $P_1(x,u) \to P_1(x,v), P_1(x,v) \to P_1(x,w) \vdash P_1(x,u) \to P_1(x,w)$, Rule TC Ind, we obtain $P_2^+(z,y) \vdash P_1(x,z) \to P_1(x,y)$. This entails that $\exists z\left(xP_1z \wedge zP_2^+y\right) \vdash xP_1y$ using $iFOL_S$ rules. Hence, in both cases $P_1(x,y)$ is derivable from the assumptions.

(4): Symmetric to the proof of (3) . □

[17] $A \leftrightarrow B$ is taken as an abbreviation of $(A \to B) \wedge (B \to A)$.

4.3.2 Kleene Algebra with Tests

Many of the applications of KA are enhanced using Kleene algebra with tests (KAT) [66], which is an equational system for program verification that combines KA with Boolean algebra. The presence of tests allows KAT to model basic programming language constructs such as conditionals, while loops, verification conditions, and partial correctness assertions.

Definition 4.3.7. A *Kleene algebra with tests* (KAT) is a KA with an embedded Boolean subalgebra, i.e., a two-sorted structure $(K, B, +, \cdot, ^*, ^-, 0, 1)$ such that:

1. $(K, +, \cdot, ^*, 0, 1)$ is a KA,
2. $(B, +, \cdot, ^-, 0, 1)$ is a Boolean algebra,
3. $B \subseteq K$.

Elements of B, denoted by $b, c, ...$, are called *tests,* and the Boolean complementation operator $^-$ is defined only on them.
Relational Kleene algebra with test ($RKAT$) is RKA with test, where tests are simply subsets of the identity relation on the domain U. The Boolean complementation operator on tests gives the set-theoretic complement in the identity relation.

Let $\mathcal{L}$ be a first-order language with equality, consisting of *binary* predicates, $P_1, P_2, .., B_1, B_2, ...$ We expand the translation from a RKA expression into an $iAL^{=}_{S}$ formula, to a translation of a $RKAT$ expression into an $iAL^{=}_{S}$ formula in the following way.

Definition 4.3.8. Let E be a $RKAT$ expression. the formula $|E|$ is defined inductively as in Def. 4.3.5, with the following additional clause:

- For each atomic test b assign a distinct predicate symbol B_i and define:

$$|b| := B_i(x, y) \wedge x = y$$
$$\left|\bar{b}\right| := \neg B_i(x, y) \wedge x = y$$

The translation of each test may be viewed as a decidable unary predicate. This is because from any formula of the form $B_i(x, y) \wedge x = y$ one can deduce $B_i(x, x)$ using

the Paramodulation rule. Tests in constructive systems are the properties which are decidable, thus the formal system must be extended to include instances of the Law of Excluded Middle for each test (and only for them). This correlates to the fact that in Kleene algebra with tests the Boolean complementation operator is defined only on tests.

Definition 4.3.9. The system $iALT_S^=$ is obtained by adding to $iAL_S^=$ axioms of the form $B_i(x,y) \vee \neg B_i(x,y)$ for each predicate B_i.

Theorem 4.3.10. *For E a valid expression of $RKAT$, $|E|$ is $iALT_S^=$ provable.*

Proof. Clearly all the translations of the axioms of RKA remain provable in $iALT_S^=$. It remains to show that the translated axioms for the Boolean algebra are provable in $iALT_S^=$. The translated axioms of associativity, commutativity and distributivity of $+$ and $\cdot$ are provable as in the case of RKA. We next show that the translations of the remaining axioms are provable.

$b \cdot \bar{b} = 0$: The translation is $\exists z\,(B(x,z) \wedge x = z \wedge \neg B(z,y) \wedge z = y) \leftrightarrow False$. The right-to-left implication is trivial. For the converse direction, notice that the formula $\exists z\,(B(x,z) \wedge x = z \wedge \neg B(z,y) \wedge z = y)$ easily entails $B(x,x) \wedge \neg B(x,x)$, from which $False$ is provable (since $\neg B(x,x)$ is an abbreviation of $B(x,x) \rightarrow False$).

$b + \bar{b} = 1$: The translation is $(B(x,y) \wedge x = y) \vee (\neg B(x,y) \wedge x = y) \leftrightarrow x = y$, which is provably equivalent to $(x = y \wedge (B(x,y) \vee \neg B(x,y))) \rightarrow x = y$. Thus, the left-to-right implication is clearly provable. As $B(x,y) \vee \neg B(x,y)$ is an axiom of the system $iALT_S^=$, the right-to-left implication is also provable.

$b + (b \cdot c) = b$: The translation of the axiom is the formula:
$[(B_1(x,y) \wedge x = y) \vee \exists z.B_1(x,z) \wedge x = z \wedge B_2(z,y) \wedge z = y] \leftrightarrow B_1(x,y) \wedge x = y$.
The right-to-left implication is immediate. For the converse direction, notice that using the formula $\exists z\,(B_1(x,z) \wedge x = z \wedge B_2(z,y) \wedge z = y)$ we can prove $B_1(x,y) \wedge B_2(x,y)$ and $\exists z\,(x = z \wedge z = y)$, from which both $B_1(x,y)$ and $x = y$ are provable.

$b \cdot (b + c) = b$: The translation of the axiom is the formula:
$\exists z\,(B_1(x,z) \wedge x = z \wedge ((B_1(z,y) \wedge z = y) \vee (B_2(z,y) \wedge z = y))) \leftrightarrow B_1(x,y) \wedge x = y$.

The left-to-right implication is proven in a similar way to the proof of the previous axiom. For the converse, using the formula $B_1(x,y) \wedge x = y$ we can deduce $\exists z\,(B_1(x,z) \wedge x = z \wedge B_1(z,y) \wedge z = y)$, and from it the left-hand side is derivable. □

$RKAT$ is especially interesting because it closely models our intuition about programs. For instance, the **if** and **while** program constructs are encoded in $RKAT$ as in propositional Dynamic Logic:

$$\mathbf{if}\, b\, \mathbf{then}\, p\, \mathbf{else}\, q := bp + \bar{b}q$$
$$\mathbf{while}\, b\, \mathbf{do}\, p := (bp)^* \bar{b}$$

By the above translation, the construct $\mathbf{if}\, b\, \mathbf{then}\, p\, \mathbf{else}\, q$ can be expressed by a formula equivalent to $(B(x,x) \wedge P_1(x,y)) \vee (\neg B(x,x) \wedge P_2(x,y))$, and the construct $\mathbf{while}\, b\, \mathbf{do}\, p$ is expressible in $iALT_S^=$ by a formula equivalent to $\left((B(x,x) \wedge P(x,y))^+ \vee x = y\right) \wedge \neg B(y,y)$.

Another connection to programming derives from the relation between $iALT_S^=$ and the theory of flowchart schemes.[18] This theory has a rich history going back to Ianov [60] and Manna [74]. A central question in the theory of flowchart schemes is scheme equivalence. In [74] examples of equivalence proofs done by transformations on the graphs of the schemes are presented. In [66] KAT was used to recast much of the theory of flowchart schemes into an algebraic framework by assigning to each flowchart scheme a KAT expression. Thus, the question of scheme equivalence was replaced by the question of equality between KAT expressions. The translation algorithm given in this section shows that the problem of scheme equivalence amounts to the question of equivalence in $iALT_S^=$ between two formulas.

There are a number of benefits to reasoning about programs in $iALT_S^=$ as opposed to $RKAT$. First, while $RKAT$ can be embedded into $iALT_S^=$, the language of $iALT_S^=$ is far richer than that of $RKAT$. Hence, there are many meaningful statements about programs that cannot be formulated in $RKAT$, but can be captured in $iALT_S^=$ (e.g., "there is a state to which each run gets to (on any input)"). Another key feature of $iALT_S^=$ is that proofs of specifications in $iALT_S^=$ carry their computational content in the realizer. Thus, a proof in $iALT_S^=$ results in a realizer which can be thought of

[18] A flowchart scheme is a vertex-labeled graph that represents an uninterpreted program.

as a program. Even simple assertions, such as $A^+(x, y) \to \exists z A(z, y)$, have interesting realizers that depend on the realizer of the TC Ind rule ($tcind$), and thus correspond to recursive programs. Moreover, since $iALT_S^=$ is an effective, constructive proof system, it is more amenable to implementation.

Chapter 5

The Predicative Framework

In this chapter we pursue the predicative approach described in Section 2.2 and present a framework which is suitable for the formalization of predicative mathematics. This system is based on the approach to predicative set theory suggested in [9], which in turn uses the framework for formalizing set theories developed in [5, 8]. One advantage of this framework is that it is close in spirit and formulation to ZF (and similar systems), since it is purely set-theoretical. Another main advantage of the framework is that its language is type-free, and it includes nothing that is not used also in ordinary mathematical discourse. In particular, it reflects real mathematical practice in making an extensive use of defined abstract set terms of the form $\{x \mid \varphi\}$.[1] A crucial property of the framework is that unlike in the dynamic approach used in current mathematical texts, here the introduction of such set terms does not depend on previously proving corresponding existence theorems. Instead, they are statically defined in a precise, purely syntactic way using a mechanism called safety relations. The use of this mechanism also adds computational content to our system (as will be demonstrated in the sequel).

The formal framework we present has several variants. It can be first-order based, or its language can be a TC-language. Moreover, in each of these options the underlying logic can be classical or intuitionistic.

[1] This is a major advantage over the usual formalizations of set theories, which employ language(s) that are rather poor and remote from those used in everyday mathematics. These formalizations are usually done in languages in which variables are the only terms which are directly provided, thus making them almost useless from a computational point of view.

It should be noted that while our formal systems are predicative, their use is not confined to predicative models. Thus, possible models for the systems include concrete, predicatively accepted universes, but also inconcrete ones. However, the meaning of a term in their languages does not really depend on the choice of model (whether predicatively acceptable or not), because every valid term of the languages has the same interpretation in *all* transitive models of the system which contains the values of its parameters and interprets all the constants in the same way.

The chapter is organized as follows: We start by reviewing the safety relation mechanism and its motivations in Section 5.1. In Section 5.2 we describe the general predicative framework: the family of languages and their corresponding systems. Possible models for the formal systems are explored in Section 5.3. Then, in Section 5.4, the basic languages are extended in a *static* way in order to introduce fundamental set-theoretical concepts. This includes the introduction of classes into the framework, as well as describing the way standard set-theoretical notions (like relations and functions) are dealt with within the framework. In Section 5.5 the natural numbers are incorporated into our framework.

This chapter is mainly based on [10, 11].

5.1 Safety Relations

One of the foundational questions in set theory is which formulas should be excluded from defining sets by an abstract term of the form $\{x \mid \varphi\}$ in order to avoid the paradoxes of naive set theory. More generally, the question is: what formulas can be taken as defining a construction of a set from given objects (including other sets)? Various set theories provide different answers to this question. These answers are usually guided by semantic intuitions (like the limitation of size doctrine [43, 51]), thus rendering them useless for the purpose of mechanization. When interested in a mechanizable or computational system, one has to translate the various semantic principles into *syntactic* (and in our opinion, less ad-hoc) constraints on the logical form of formulas. For this the concept of safety relations introduced in [5, 6, 8] was developed. The concept combines ideas from three seemingly very different sources:

Set theory: Gödel's classical work [48] on the constructible universe L is best known for its use in consistency and independence proofs. However, it is of course of great interest also for the study of the general notion of constructions with sets. Thus, for characterizing the "constructible sets" Gödel identified a set of operations on sets (which we may call "computable"), that can be used for "effectively" constructing new sets from given ones. For example, binary union and intersection are "effective", while the power set operation is not. Gödel has provided a finite list of basic operations, from which all other "effective" (for his purposes) constructions can be obtained through compositions. Another very important idea introduced in [48] is *absoluteness* — a key property (see [69]) of formulas which are used for defining "constructible sets". Roughly, a formula is absolute if its truth value in a transitive class M, for some assignment v of objects from M to its free variables, depends only on v, but not on M (i.e., the truth value is the same in every transitive class M, in which v is legal).

Formal arithmetic: Absoluteness is not a decidable property. Therefore, a certain set Δ_0 of absolute formulas is extensively used in set theory as a syntactically defined approximation. Now a similar set Δ_0 of formulas (also called in [106] "bounded formulas" or "Σ_0-formulas"), which has exactly the same definition (except that $\in$ is replaced by $<$) is used in formal arithmetic in order to characterize the decidable and the semi-decidable (r.e.) relations on the natural numbers. This fact hints at an intimate connection (investigated in [7]) between absoluteness/constructibility and decidability/computability.

Relational database theory: The importance of computations with sets to this area is obvious: to provide an answer to a query in a relational database, a computation should be made in which the input is a finite set of finite sets of tuples (the "tables" of the database), and the output should also be a finite set of tuples. In other words: the computation is done with (finite) sets. Accordingly, for *effective* computations with finite relations some finite set of basic operations has been identified in database theory, and this basic set defines (via composition) what is called there "the relational algebra" [1, 112]. Interestingly, there is a lot of similarity between the list of operations used in the relational algebra and Gödel's list of basic operations mentioned above. However, much more important is again the strong connection (observed in [6, 7, 8]) between

the notion of absoluteness used in set theory, and the notion of domain independence used in database theory, and practically serving as its counterpart of the notion of computability.

A query in a database can be construed as a formula φ in the language of set theory, augmented with constants for the relations in the database. The answer to such a query is the set of all n-tuples that satisfy φ, given the interpretations provided by the database for the extra constants (here n is the number of free variables in φ; if $n = 0$, then the answer to the query is either "yes" or "no"). A domain-independent (d.i.) query is a query the answer to which depends only on the information included in the database, and on the objects which are mentioned in the query. Only such queries are considered meaningful. Moreover: the answer to such queries is always finite and computable. Therefore, practical database query languages (like SQL) are designed so that only d.i. queries can be formulated in them, and each such query language is based on some syntactic criteria that ensure this property. In order to give these criteria a concise logical characterization, and in order to unify the notions of absoluteness and domain-independence, the formula *property* of d.i. was turned in [6, 7] into a *safety relation* $\succ$ between a formula φ and finite subsets of $Fv(\varphi)$. The intuitive meaning of "$\varphi(x_1, ..., x_n, y_1, ..., y_k) \succ \{x_1, ..., x_n\}$" in databases is: "$\varphi(x_1, ..., x_n, d_1, ..., d_k)$ is d.i. for all values $d_1, ..., d_k$". In particular, $\varphi \succ \emptyset$ if φ is absolute in the sense of axiomatic set theory.

In view of the connections noted above between "absolute" and "decidable" and between "domain-independent" and "computable", in the realm of sets we shall intuitively take the meaning of "$\varphi(x_1, ..., x_n, y_1, ..., y_k) \succ \{x_1, ..., x_n\}$" in case $n > 0$ to be: "The collection $\{\langle x_1, ..., x_n\rangle \mid \varphi\}$ is an acceptable set for all acceptable values of $y_1, ..., y_k$, and it can be *constructed* from these values". In case $n = 0$ we take the meaning of "$\varphi(y_1, ..., y_k) \succ \emptyset$" to be: "$\varphi$ describes a *definite* property". In particular, if $\varphi(y_1, ..., y_k) \succ \emptyset$, then the collection $\{\langle y_1, ..., y_k\rangle \in z \mid \varphi\}$ should be an acceptable set for any acceptable value of z.[2] This intuitive meaning will be used in our predicative

[2] In other words, we take "$\varphi \succ \emptyset$" as saying that φ defines a "definite" property in Zermelo's sense (see Introduction), and practically identify "definite" and "absolute".

formalization of ZF Comprehension Axiom in the next section. The differences between the strength of formal systems will intuitively be due to different interpretations of the vague notions of "acceptable" and "can be constructed".

5.2 The Languages and the Systems

Notation. To avoid confusion, we use different parentheses for collections in our formal language and in the meta-language. The parentheses $\{\!|\, |\!\}$ are used in our formal languages, while for a collection in the meta-language we use $\{\,\}$. In the meta-language we use uppercase letters $X, Y, Z, ...$ for collections; Φ, Θ for finite sets of variables; and $x, y, z, ...$ for variables in the formal language.

5.2.1 Languages

Given a set of constants C, we first introduce the language $\mathcal{L}^C_{RST}$, which is a first-order language with equality having a variable binding term operator (vbto, see [32]). The vbto of the language is of the form $\{\!|\, x \mid \varphi \,|\!\}$. Note that in first-order languages with vbto the notion of a term being free for substitution generalizes the usual one, since in such a language a variable can be bound within a term. As usual, the generalization amounts to avoiding the capture of free variables within the scope of a binding operator.

Convention. In what follows, in case $C = \emptyset$ we omit C from our notations for the language and for the systems. For example, we write $\mathcal{L}_{RST}$ instead of $\mathcal{L}^{\emptyset}_{RST}$.

Definition 5.2.1. Let C be a finite set of constants. The language $\mathcal{L}^C_{RST}$ and the associated safety relation $\succ$ are simultaneously defined as follows:

- Terms:
  - Every variable is a term.
  - Every $c \in C$ is a term (taken to be a constant).
  - If x is a variable and φ is a formula such that $\varphi \succ \{x\}$, then $\{\!|\, x \mid \varphi \,|\!\}$ is a term (for which $Fv\left(\{\!|\, x \mid \varphi \,|\!\}\right) = Fv\left(\varphi\right) - \{x\}$).

- Formulas:
  - If t and s are terms, then $t = s$ and $t \in s$ are atomic formulas.
  - If φ and ψ are formulas and x is a variable, then $\neg\varphi, (\varphi \wedge \psi), (\varphi \vee \psi)$, and $\exists x\varphi$ are formulas.
- The safety relation $\succ$ is defined as follows:
  - If φ is an atomic formula, then $\varphi \succ \emptyset$.
  - If t is a term such that $x \notin Fv(t)$, and $\varphi \in \{x \in x, x \in t, x = t, t = x\}$, then $\varphi \succ \{x\}$.
  - If $\varphi \succ \emptyset$, then $\neg\varphi \succ \emptyset$.
  - If $\varphi \succ \Theta$ and $\psi \succ \Theta$, then $\varphi \vee \psi \succ \Theta$.
  - If $\varphi \succ \Theta$, $\psi \succ \Phi$ and either $\Phi \cap Fv(\varphi) = \emptyset$ or $\Theta \cap Fv(\psi) = \emptyset$, then $\varphi \wedge \psi \succ \Theta \cup \Phi$.
  - If $\varphi \succ \Theta$ and $y \in \Theta$, then $\exists y\varphi \succ \Theta - \{y\}$.

Since in our opinion TC-languages provide a better framework for formalizing predicative mathematics, we next introduce an extension of $\mathcal{L}^C_{RST}$ to a TC-language (see Chapter 3).[3]

Definition 5.2.2. The language $\mathcal{L}^C_{RST+TC}$ is obtained from $\mathcal{L}^C_{RST}$ by the addition of the following clauses to its definition:

- If φ is a formula, x, y are distinct variables, and s, t are terms, then $(TC_{x,y}\varphi)(s,t)$ is a formula.
- If $\varphi \succ \Phi$ and $\{x, y\} \cap \Phi \neq \emptyset$, then $(TC_{x,y}\varphi)(x,y) \succ \Phi$.

Notation. In what follows, when what we say applies to both $\mathcal{L}^C_{RST}$ and $\mathcal{L}^C_{RST+TC}$, we simply write $\mathcal{L}^C$.

[3]For convenience we take here the TC operator, but the RTC operator could be taken instead, since $\mathcal{L}^C_{RST}$ includes equality (see Chapter 3).

Note 5.2.3. Though the official languages do not include $\forall$ and $\rightarrow$, we take $\forall x_1 ... \forall x_n (\varphi \rightarrow \psi)$ as an abbreviation for $\neg \exists x_1 ... \exists x_n (\varphi \wedge \neg \psi)$. This can be done if we assume classical logic. If intuitionistic logic is assumed, we must include $\forall$ and $\rightarrow$ in our formal language and add to the definition of the safety relation $\succ$ a clause stating: if $\varphi \succ \{x_1, ..., x_n\}$ and $\psi \succ \emptyset$, then $\forall x_1, ..., x_n (\varphi \rightarrow \psi) \succ \emptyset$.

The next Lemma is straightforward to verify.

Lemma 5.2.4. $\succ$ *has the following properties:*

- *If* $\varphi \succ \Theta$ *and* $\Phi \subseteq \Theta$, *then* $\varphi \succ \Phi$.
- *If* $\varphi \succ \Theta$, $x \in \Theta$, *and* $y \notin Fv(\varphi)$, *then* $\varphi \left\{ \frac{y}{x} \right\} \succ \Theta - \{x\} \cup \{y\}$.
- *If* $\varphi \succ \Theta$ *and* $x \in \Theta$, *then* $\varphi \left\{ \frac{t}{x} \right\} \succ \Theta - \{x\}$.
- *If* $\varphi \succ \Theta$ *and* $x \notin \Theta$, *then* $\varphi \left\{ \frac{t}{x} \right\} \succ \Theta - Fv(t)$.
- *If* $\varphi \succ \{x_1, ..., x_n\}$ *and* $\psi \succ \emptyset$, *then* $\forall x_1, ..., x_n (\varphi \rightarrow \psi) \succ \emptyset$.
- *If* $x \notin Fv(t)$ *and* $\varphi \succ \emptyset$, *then* $\forall x (x \in t \rightarrow \varphi) \succ \emptyset$ *and* $\exists x (x \in t \wedge \varphi) \succ \emptyset$. *Hence,* $\varphi \succ \emptyset$ *for every* Δ_0 *formula in* $\mathcal{L}_{ZF}$.

5.2.2 Logics

We use four different types of logics in our framework. The basic two for $\mathcal{L}^C_{RST}$ are classical first-order logic $cFOL$ (the logic which underlies ZF and related systems), and intuitionistic first-order logic $iFOL$ (the logic which underlies constructive counterparts of ZF, like CZF and IZF). For $\mathcal{L}^C_{RST+TC}$ ancestral logic is used, again in two versions: the classical one cAL, and the intuitionistic one iAL.

5.2.3 Axioms and Systems

Next, the axioms of the formal systems are presented, followed by a discussion that explains the rationale behind them.

Notation 5.2.5. We take the usual definition of $\subseteq$ in terms of $\in$. Note that according to this definition $t \subseteq s \succ \emptyset$. We also denote by $\{\!\{t\}\!\}$ the term $\{\!\{ x \mid x = t \}\!\}$, by $s \cup t$ the term $\{\!\{ x \mid x \in s \vee x \in t \}\!\}$, and by $\emptyset$ the term $\{\!\{ x \mid x \in x \}\!\}$ (see Lemma 5.4.2 in the sequel).

Definition 5.2.6 (*The systems*)**.**

1. The system RST_C^{cFOL} is the classical first-order with vbto system in $\mathcal{L}^C_{RST}$ which is based on the following set of axioms:[4]

   - Extensionality: $\forall z\,(z \in x \leftrightarrow z \in y) \rightarrow x = y$
   - Comprehension Schema: $\forall x\,(x \in \{\!|\, x \mid \varphi \,|\!\} \leftrightarrow \varphi)$
   - Restricted $\in$-induction Schema: $\left(\forall x\left(\forall y\left(y \in x \rightarrow \varphi\left\{\frac{y}{x}\right\}\right) \rightarrow \varphi\right)\right) \rightarrow \forall x\varphi$, where $\varphi \succ \emptyset$.

   If C includes the constant HF, then the following axioms are added:

   - $\emptyset \in HF$
   - $\forall x \forall y\,(x \in HF \wedge y \in HF \rightarrow x \cup \{\!|y|\!\} \in HF)$
   - $\forall y\,(\emptyset \in y \wedge \forall v, w \in y.v \cup \{\!|w|\!\} \in y \rightarrow HF \subseteq y)$

2. The system RST_C^{cAL} is the classical ancestral logic with vbto system in $\mathcal{L}^C_{RST+TC}$ which is based on the same set of axioms as RST_C^{cFOL}.

3. The system RST_C^{iFOL} is the intuitionistic first-order with vbto system in $\mathcal{L}^C_{RST}$ which is based on the same set of axioms as RST_C^{cFOL}, with the additional axiom:

   - Restricted Excluded Middle: $\varphi \vee \neg\varphi$, where $\varphi \succ \emptyset$.

4. The system RST_C^{iAL} is the intuitionistic ancestral logic with vbto system in $\mathcal{L}^C_{RST+TC}$ which is based on the same set of axioms as RST_C^{iFOL}.

Notation. When what we say applies to both the classical and the intuitionistic versions of the corresponding systems we simply write RST_C^{FOL} or RST_C^{AL}. In case it applies to all four systems we write RST_C.

An important feature of RST_C is that its first two axioms directly lead (and are equivalent) to the following *set-theoretical* reduction rules:

[4] In [11] the system RST_C^{cFOL} was denoted by RST^C, and in case $HF \in C$ it was denoted by RST^C_{HF}. In [8] the system RST^{cFOL} was denoted by RST. Also, RST^{cFOL} can be shown to be equivalent to the system obtained from Gandy's basic set theory [44] by adding to it the restricted $\in$-induction schema, and to the system called BST_0 in [99] (see [7]).

- $[(\beta)] \vdash_{RST_C} t \in \{\!|\, x \mid \varphi \,|\!\} \leftrightarrow \varphi\left\{\frac{t}{x}\right\}$ (provided t is free for x in φ).
- $[(\eta)] \vdash_{RST_C} \{\!|\, x \mid x \in t \,|\!\} = t$ (provided $\{\!|\, x \mid x \in t \,|\!\}$ is a legal term, i.e. $x \notin Fv(t)$).

Discussion:

As noted in Section 2.2, to impose the first principle of the predicative approach there should be some restrictions on the Comprehension Axiom. In [7] it was suggested that the predicatively acceptable instances of the axiom are those which determine the collections they define in an absolute way, independently of any "surrounding universe". In other words: in the context of set theory, a formula φ is predicative (with respect to x) if the collection $\{x \mid \varphi(x, y_1, ..., y_n)\}$ is completely and uniquely determined by the identity of the parameters $y_1, ..., y_n$, and the identity of other objects referred to in the formula (all of which should be well-determined beforehand). Note that φ is predicative for $\emptyset$ iff it is absolute in the usual sense of set theory. In order to translate this idea into an exact, *syntactic* definition the safety relation is used. Thus, only those formulas which are safe with respect to $\{x\}$ should be allowed in the Comprehension Axiom Scheme of our systems. In order to show that the safety relation $\succ$ indeed possesses the above property we examine some of its defining clauses. It is not difficult to see that the formula $x \in y$ should be safe w.r.t. $\{x\}$ (but not w.r.t. $\{y\}$), since if the identity of y is predicatively acceptable as a set, then any of its elements must be previously accepted as a set, and $\{x \,|\, x \in y\} = y$. Another example can be given by looking at the clause for negation. The intuitive meaning of $\{x \,|\, \neg\varphi\}$ is the complement (with respect to some universe) of $\{x \,|\, \varphi\}$, which is not in general predicatively accepted. However, if φ is absolute (or "definite"), then its negation will be absolute ("definite") as well. This also explains the additional axiom of Restricted Excluded Middle in RST_C^{iFOL} and RST_C^{iAL}, which asserts the definiteness of absolute formulas. For a more complicated example, assume that $\theta = \varphi \wedge \psi$, where $Fv(\varphi) = \{x, z\}$, $Fv(\psi) = \{x, y, z\}$, $\varphi \succ \{x\}$ and $\psi \succ \{y\}$. Given some absolute set c, by the induction hypothesis the collection $Z(c)$ of all x such that $\varphi(x, c)$ is an absolute set. Again by the induction hypothesis, for every d in this set the collection $W(c, d)$ of all y such that $\psi(d, y, c)$ is an absolute set. Now the collection of all $\langle x, y\rangle$ such that $\theta(x, y, c)$ is the union for $d \in Z(c)$ of the sets $\{d\} \times W(c, d)$. Hence, it is a set containing only previously accepted, absolute collections, and its identity is obviously absolute too. This is exactly what $\theta \succ \{x, y\}$ (which holds in this case

by the clause concerning conjunction in the definition of $\succ$) intuitively means. For a TC-formula, the justification of the safety clause for it can be easily obtained if we look at the intuitive meaning of $(TC_{x,y}\varphi)(x,y)$ as an infinite disjunction (see Section 3.1). The safety clauses for $\exists, \vee, \wedge$ imply that if $\varphi \succ \Phi$ and $x \in \Phi$ (or $y \in \Phi$), then any finite initial of the infinite disjunction is safe w.r.t Φ. From this it easily follows that so is the whole disjunction.

The second principle of the predicative approach requires that $\mathbb{N}$ (the set of natural numbers) constitutes a set. However, in the first-order systems, the collection of hereditary finite sets is the minimal model of RST^{FOL}; hence $\mathbb{N}$ is not definable as a set in RST^{FOL}. To solve this problem, RST^{FOL} is enhanced above by including a special constant HF in C whose intended interpretation is the set of all hereditary finite sets. The axioms for the constant symbol HF ensure (as far as it is possible on the first-order level) that HF is indeed to be interpreted as this collection. These axioms in fact replace the usual infinity axiom of ZF. In contrast, in case ancestral logic is used the natural numbers can be defined using the TC operator. Therefore, in RST^{AL} the additional constant HF and its axioms are not necessary (see Section 5.5 for the introduction of the natural numbers in $RST^{FOL}_{\{HF\}}$ and in RST^{AL}).

It should be clear that the use of ancestral logic is compatible with the predicative approach. The reason is that the second principle of predicativism entails acceptance of principles and ideas implicit in the construction of $\mathbb{N}$. This includes proofs by mathematical induction, as well as the idea of iterating (an operation or a relation) an arbitrary (finite) number of times. Therefore finitary inductive definitions of sets, relations, and functions should be accepted. In particular, the ability to form the transitive closure of a given relation should also be predicatively acceptable.[5]

The inclusion of the Restricted $\in$-induction Scheme is mainly in order to be able to prove predicatively valid statements like $\forall x.x \notin x$ or $\forall x, y.x \in y \rightarrow y \notin x$. Actually, even the addition of the full $\in$-induction does not seem to be in any conflict with the predicative approach. On the contrary, it only imposes further restrictions on the collection of acceptable sets. Nevertheless, to be on the safe side, like in [44], we do not adopt here the full $\in$-induction scheme, but only a very restricted form of it.

[5]It is interesting to note that in case we are interested in pure finitism, and thus do not want to assume any form of infinity, this can be achieved by simply replacing the clause in the safety relation referring to TC-formulas by the following clause: If $\varphi \succ \Phi$ and $\{x, y\} \subseteq \Phi$, then $(TC_{x,y}\varphi)(x,y) \succ \Phi$.

Moreover, we will try to avoid (as much as possible) the use of this axiom, and we shall point out the places where it is used.

Note 5.2.7. The systems RST^{cFOL} and $RST^{cFOL}_{\{HF\}}$ are subsystems of ZF, as all of their axioms are easily provable in ZF. On the other hand, in [8] it was shown that some of the comprehension axioms of ZF (such as the Union axiom) are already derivable in RST^{cFOL}, while the others can be incorporated by adding corresponding clauses to the definition of the safety relation. In particular:

- The full power of restricted comprehension can be achieved by assuming that $\varphi \succ \emptyset$ for any φ (not only atomic ones).[6]
- The power set axiom is equivalent to letting $x \subseteq t \succ \{x\}$ in case $x \notin Fv(t)$.
- The full power of replacement is achieved by letting $\exists y\varphi \wedge \forall y(\varphi \rightarrow \psi) \succ \Theta$ if $\psi \succ \Theta$ and $\Theta \cap Fv(\varphi) = \emptyset$.

The system RST^{iFOL} without the axiom scheme for restricted excluded middle can easily be shown to be is a subsystem of IZF and CZF. It can also be shown (using results from [70]) that RST^{iFOL} is a subsystem of IZF and CZF if one adds to the latter an axiom scheme for excluded middle for bounded formulas (see, e.g., [2]).

Proposition 5.2.8. *Let C be a set of constants which contains HF. The followings are provable in RST_C:*

1. $\forall x(x \in HF \leftrightarrow x = \emptyset \vee \exists u, v \in HF.u \cup \{v\} = x)$.
2. $(\varphi(\emptyset) \wedge \forall x \forall y(\varphi(x) \wedge \varphi(y) \rightarrow \varphi(x \cup \{y\}))) \rightarrow \forall x \in HF.\varphi(x)$, where $\varphi \succ \emptyset$.[7]
3. $\psi(HF) \wedge \forall a(\psi(a) \rightarrow HF \subseteq a)$, where $\psi(a)$ denotes the formula: $\forall x(x \in a \leftrightarrow x = \emptyset \vee \exists u, v \in a.u \cup \{v\} = x)$.

[6] In other words, by assuming that any first-order formula is "definite" in Zermelo's sense (see Introduction).

[7] It is possible to prove the *full* induction principle for HF by using the full $\in$-induction. See Prop. 5.5.6 for an example of how this is done in a similar case.

Proof.

1. The right-to-left implication is provable in RST_C using the first two axioms for HF. For the converse, define $B := \{\!\{x \in HF \mid x = \emptyset \vee \exists u, v \in HF.u \cup \{\!\{v\}\!\} = x\}\!\}$. It is easy to verify that the formula $\emptyset \in B \wedge \forall v, w \in B.v \cup \{\!\{w\}\!\} \in B$ is provable in RST_C. From this, using the third axiom of HF, we get $\forall x \in HF.x \in B$, which by the Restricted Comprehension entails $\forall x \in HF\,(x = \emptyset \vee \exists u, v \in HF.u \cup \{\!\{v\}\!\} = x)$.

2. Suppose $\varphi(\emptyset) \wedge \forall x \forall y\,(\varphi(x) \wedge \varphi(y) \rightarrow \varphi(x \cup \{\!\{y\}\!\}))$. Let $u = \{\!\{z \in HF \mid \varphi(z)\}\!\}$ (This is a legal term, since we assume that $\varphi \succ \emptyset$). From the assumption and the first two axioms for HF it is easy to see that $\emptyset \in u \wedge \forall v, w \in u.v \cup \{\!\{w\}\!\} \in u$. Therefore, by the third axiom for HF we get $\forall x \in HF.x \in u$, which by the Restricted Comprehension entails $\forall x \in HF.\varphi(x)$.

3. By part 1. of the proposition, we have $\psi(HF)$, and by the third axiom for HF we can easily derive $\forall z\,(\psi(z) \rightarrow \forall y \in HF.y \in z)$. □

Note 5.2.9. Note that while the languages $\mathcal{L}^C$ allow the use of set terms, they also provide a mechanizable static check of their validity due to the syntactic safety relation. In order to ensure the decidability of the syntax we did not include in the definition of the safety relation the following natural condition: if $\varphi \succ \Theta$ and ψ is logically equivalent to φ (where $Fv(\varphi) = Fv(\psi)$), then also $\psi \succ \Theta$. This condition would entail that $\{\!\{x \mid \varphi\}\!\}$ is a valid term whenever $\{\!\{x \mid \psi\}\!\}$ is a valid term, and ψ is logically equivalent to φ. What we do however have in RST_C is that if $\vdash_{RST_C} \varphi \leftrightarrow \psi$, then $\vdash_{RST_C} x \in \{\!\{x \mid \varphi\}\!\} \leftrightarrow \psi$, and so $\vdash_{RST_C} \exists z \forall x\,(x \in z \leftrightarrow \psi)$.

5.3 Universes

We next recall the definition of rudimentary functions (see [34, 58]).[8] Rudimentary functions are just the functions obtained by omitting the recursion schema from the standard list of schemata for primitive recursive set functions.

[8]To be precise, the definition we take here is given in The Basis Lemma in [34]. It was shown there that this definition is equivalent to the standard definition of rudimentary functions.

Definition 5.3.1. Every rudimentary function is a composition of the following functions:

- $F_0(x,y) = \{x,y\}$
- $F_1(x,y) = x - y$
- $F_2(x,y) = x \times y$
- $F_3(x,y) = \{\langle u,z,v\rangle \mid z \in x \wedge \langle u,v\rangle \in y\}$
- $F_4(x,y) = \{\langle z,v,u\rangle \mid z \in x \wedge \langle u,v\rangle \in y\}$
- $F_5(x,y) = \{Im(x|_z) \mid z \in y\}$, where $Im(x|_z) = \{w \mid \exists u \in z.\, \langle u,w\rangle \in x\}$
- $F_6(x) = \bigcup_{z\in x} z$
- $F_7(x) = Dom(x) = \{v \mid \exists w.\, \langle v,w\rangle \in x\}$
- $F_8(x) = \{\langle u,v\rangle \mid u \in x \wedge v \in x \wedge u \in v\}$

We now consider the notion of relatively rudimentary functions.

Definition 5.3.2. Let C be a set of constants.

1. A function is called C-rudimentary if it can be generated by composition of the functions $F_0, ..., F_8$ in Definition 5.3.1, and the following constant functions:
   - $F_9^c(x) = c$, for $c \in C$.
2. A function is said to be C-rudimentary relative to $\mathcal{C}\ell$ if it can be generated from the functions $F_0, ..., F_8$ in Definition 5.3.1 and the functions F_9^c for any $c \in C$, by composition or by the following operation scheme:
   - $\mathcal{C}\ell(x,H) = \bigcup_{m\in\mathbb{N}} H^m(x)$, where H is a C-rudimentary function relative to $\mathcal{C}\ell$, and $H^0(x) = x$ (i.e., $\mathcal{C}\ell(x,H)$ denotes the closure of x under the function H).

Notation. In case $C = \emptyset$ we write rudimentary relative to $\mathcal{C}\ell$, instead of $\emptyset$-rudimentary relative to $\mathcal{C}\ell$.

Definition 5.3.3. A *universe* (of sets) is a transitive collection of sets such that:

- In the first-order case: it is closed under rudimentary functions.
- In the AL case: it is closed under rudimentary functions relative to $\mathcal{C}\ell$.

Terminology. In what follows, we do not distinguish between the universe W and the structure for $\mathcal{L}^C$ with domain W and an interpretation function I that assigns the obvious interpretations to the symbols $\in$ and $=$, and assigns to any $c \in C$ an element in W. In case $HF \in C$, $I[HF] = \mathcal{HF}$ (the set of hereditary finite sets).

Note 5.3.4. For simplicity of presentation, in what follows we assume the platonic universe V of ZF (whatever this universe is). The assumption of such an all-encompassing collection V, which includes all potential "sets" and contains all "universes", but is itself a universe too (meaning that classical logic holds within it), is in fact doubtful from a predicativist point of view. However, from Theorem 5.3.12 below it should be clear that we can actually do without this assumption, since that theorem entails "universe independence".

Notation. If $\vec{x}, \overrightarrow{a}$ are two vectors of the same length, we abbreviate $v[\vec{x} := \overrightarrow{a}]$ for $v[x_1 := a_1, ..., y_n := x_n]$. We denote by $[x_1 := a_1, ..., x_n := a_n]$ any assignment which assigns to each x_i the element a_i.[9]

Definition 5.3.5. Let W be a universe and v an assignment in W. For any term t and formula φ of $\mathcal{L}^C$, define recursively a collection $\|t\|_v^W$ and a truth value $\|\varphi\|_v^W \in \{\mathbf{t}, \mathbf{f}\}$, respectively, by:

- $\|t\|_v^W = v(x)$ for x a variable.
- $\|c\|_v^W = I(c)$ for $c \in C$.
- $\|\{x \mid \varphi\}\|_v^W = \left\{a \in W \mid \|\varphi\|_{v[x:=a]}^W = \mathbf{t}\right\}$
- $\|t = s\|_v^W = \mathbf{t}$ iff $\|t\|_v^W = \|s\|_v^W$; $\|t \in s\|_v^W = \mathbf{t}$ iff $\|t\|_v^W \in \|s\|_v^W$
- $\|\neg\varphi\|_v^W = \mathbf{t}$ iff $\|\varphi\|_v^W = \mathbf{f}$
- $\|\varphi \wedge \psi\|_v^W = \mathbf{t}$ iff $\|\varphi\|_v^W = \mathbf{t} \wedge \|\psi\|_v^W = \mathbf{t}$

[9] As long as we apply $[x_1 := a_1, ..., x_n := a_n]$ to expressions whose set of free variables is contained in $\{x_1, ..., x_n\}$ the exact assignment does not matter of course.

- $\|\varphi \vee \psi\|_v^W = \mathbf{t}$ iff $\|\varphi\|_v^W = \mathbf{t} \vee \|\psi\|_v^W = \mathbf{t}$
- $\|\exists x\varphi\|_v^W = \mathbf{t}$ iff $\exists a \left(a \in W \wedge \|\varphi\|_{v[x:=a]}^W = \mathbf{t}\right)$
- $\|(TC_{x,y}\varphi)(s,t)\|_v^W = \mathbf{t}$ iff
  $\exists a_1, ..., a_n \in W \left(\|s\|_v^W = a_1 \wedge \|t\|_v^W = a_n \wedge \bigwedge_{i\in\{0,1,...,n-1\}} \|\varphi\|_{v[x:=a_i,y:=a_{i+1}]}^W = \mathbf{t}\right)$

Note 5.3.6. From Theorem 5.3.8 below it follows that $\|t\|_v^W$ is an element of W (and it denotes the value in W that the term t gets under v), and $\|\varphi\|_v^W$ denotes the truth value of the formula φ under W and v.

Notation. In case *exp* is a closed term or a closed formula, we denote by $\|exp\|^W$ the value of *exp* in W, and at times we omit the superscript W and simply write $\|exp\|$.

The following theorem is a generalization of a theorem proven in [9].

Theorem 5.3.7. *Let C be a set of constants.*

1. *If F is an n-ary C-rudimentary function (relative to $\mathcal{C}\ell$), then there exists a formula φ_F of $\mathcal{L}_{RST}^C$ ($\mathcal{L}_{RST+TC}^C$) such that:*
   - $Fv(\varphi_F) \subseteq \{y, x_1, ..., x_n\}$
   - $\varphi_F \succ \{y\}$
   - $F(x_1, ..., x_n) = \{y \mid \varphi_F\}$
2. *If φ is a formula of $\mathcal{L}_{RST}^C$ ($\mathcal{L}_{RST+TC}^C$) such that:*
   - $Fv(\varphi) \subseteq \{y_1, ..., y_k, x_1, ..., x_n\}$
   - $\varphi \succ \{y_1, ..., y_k\}$

   then there exists a C-rudimentary function (relative to $\mathcal{C}\ell$) F_φ such that:

   $$F_\varphi(x_1, ..., x_n) = \{\langle y_1, ..., y_k\rangle \mid \varphi\}$$

3. *If t is a term of $\mathcal{L}_{RST}^C$ ($\mathcal{L}_{RST+TC}^C$) such that $Fv(t) \subseteq \{x_1, ..., x_n\}$, then there exists a C-rudimentary function (relative to $\mathcal{C}\ell$) F_t such that $F_t(x_1, ..., x_n) = t$ for every $x_1, ..., x_n$.*

Proof. The corresponding theorem in [9] establishes the connection between $\mathcal{L}_{RST}$ and rudimentary functions. Thus, the only modification required here is the treatment of the new functions and operations in 1., and the treatment of the constants in C and the transitive closure in 2. and 3. (which are then incorporated in the original proof that was carried out by induction). For 1., it is easy to verify that $\varphi_{F_9^c} := y = c$. For $\mathcal{C}\ell(x, H)$, take $\varphi_{\mathcal{C}\ell(x,H)}$ to be the formula $(TC_{x,y}\varphi_H)(x, y)$, where φ_H exists due to the induction hypothesis. The fact that $\varphi_H \succ \{y\}$ entails that $\varphi_{\mathcal{C}\ell(x,H)} \succ \{y\}$. For 2. and 3. (which are proven by simultaneous induction on the structure of terms and formulas), the case for constants is immediate from the definition of C-rudimentary functions. For the case of a TC-formula, suppose $\psi = (TC_{x,y}\varphi)(x, y)$ such that (w.l.o.g.) $\psi \succ \{y, y_2, ..., y_k\}$. This implies that also $\varphi \succ \{y, y_2, ..., y_k\}$, and by the induction hypothesis there is a C-rudimentary function relative to $\mathcal{C}\ell$, F_φ, such that $F_\varphi(x, x_2, ..., x_n) = \{\langle y, y_2, ..., y_k\rangle \mid \varphi\}$. Now, let $H(x, x_2, ..., x_n) = \bigcup_{\langle y,y_2,...,y_k\rangle \in x} F_\varphi(y, x_2, ..., x_n)$, which is also C-rudimentary relative to $\mathcal{C}\ell$. Then, define: $F_\psi := \mathcal{C}\ell(F_\varphi(x, x_2, ..., x_n), H)$. □

Theorem 5.3.8. *Let W be a universe, and let v be an assignment in W.*

1. *For t a term of $\mathcal{L}^C$, $\|t\|_v^W \in W$.*

2. *For φ a formula of $\mathcal{L}^C$:*

   (a) *If $\varphi \succ \{y_1, ..., y_n\}$ and $n > 0$, then:*

   $$\left\{\langle a_1, ..., a_n\rangle \in W^n \mid \|\varphi\|^W_{v\left[\vec{y}:=\vec{a}\right]} = \mathbf{t}\right\} \in W$$

   (b) *If $\varphi \succ \emptyset$ and $\{y_1, ..., y_n\} \subseteq Fv(\varphi)$, then for any $X \in W$:*

   $$\left\{\langle a_1, ..., a_n\rangle \in X^n \mid \|\varphi\|^W_{v\left[\vec{y}:=\vec{a}\right]} = \mathbf{t}\right\} \in W$$

Proof. The proof is straightforward using Theorem 5.3.7. Claims (1) and (2a) are immediate. For (2b) let φ be a formula s.t. $\varphi \succ \emptyset$ and $\{y_1, ..., y_n\} \subseteq Fv(\varphi)$. Using Theorem 5.3.7 we get that φ defines a C-rudimentary predicate (relative to $\mathcal{C}\ell$), P_φ

(i.e. one whose characteristic function is rudimentary). Define:

$$H\left(x_1,...,x_k,y_1,...,y_n\right)=\begin{cases}\{\langle y_1,...,y_n\rangle\} & \text{if } P_\varphi\left(x_1,...,x_k,y_1,...,y_n\right)\\ \emptyset & \text{otherwise}\end{cases}$$

H is a C-rudimentary function (relative to $\mathcal{C}\ell$) (see Lemma 1.1 in [34]). Now, define:

$$F\left(z,x_1,...,x_k\right)=z^n\cap\left\{\langle y_1,...,y_n\rangle\,|P_\varphi\left(x_1,...,x_k,y_1,...,y_n\right)\right\}$$

F is also C-rudimentary (relative to $\mathcal{C}\ell$) since

$$F\left(z,x_1,...,x_k\right)=\bigcup_{\langle y_1,...,y_n\rangle\in z^n}H\left(x_1,...,x_k,y_1,...,y_n\right)$$

Now, the fact that W is a universe entails that for every assignment v in W and every $X\in W$, $F\left(X,v\left(x_1\right),....,v\left(x_k\right)\right)\in W$. In other words, $\left\{\langle a_1,...,a_n\rangle\in X^n\mid\|\varphi\|^W_{v\left[\vec{y}:=\vec{a}\right]}=\mathbf{t}\right\}\in W$. □

Proposition 5.3.9. *Let W be a universe. Then, W is a model of RST_C.*

Proof. The Extensionality axiom is clearly satisfied in any universe. Theorem 5.3.8 entails that the interpretation of any term is an element of the universe, from this immediately follows that the other axioms are satisfied in any universe. In case $HF\in C$, it is straightforward to verify that the interpretation of HF as $\mathcal{HF}$ satisfies the three axioms for HF; and in case AL is used, it is also easy to see that any universe satisfies all the rules of the TC operator. □

Note 5.3.10. The converse of the last proposition is also true, i.e., any transitive collection of sets which is a model of RST_C is a universe (see [7]).

The proof of the following lemma is straightforward.

Lemma 5.3.11 (Substitution Theorem)**.** *Let $t,s_1,...,s_n$ be terms and φ a formula of $\mathcal{L}^C$. If v is an assignment in W, then:*

- $\left\|t\left\{\frac{s_i}{x_i},...,\frac{s_n}{x_n}\right\}\right\|^W_v=\|t\|^W_{v\left[x_1:=\|s_1\|^W_v,...,x_n:=\|s_n\|^W_v\right]}$
- $\left\|\varphi\left\{\frac{s_i}{x_i},...,\frac{s_n}{x_n}\right\}\right\|^W_v=\|\varphi\|^W_{v\left[x_1:=\|s_1\|^W_v,...,x_n:=\|s_n\|^W_v\right]}$

Next we show that the meaning of terms in $\mathcal{L}^C$ is actually independent of W. The following theorem is a more precise and more general formulation of Theorem 4 in [9].

Theorem 5.3.12. *Let W_1, W_2 be two universes which agree on the interpretations of all $c \in C$.*

1. *If v_1, v_2 are assignments in W_1 and W_2, respectively, that agree on the values of all the free variables in a term t, then $\|t\|_{v_1}^{W_1} = \|t\|_{v_2}^{W_2}$.*

2. *If v_1, v_2 are assignments in W_1 and W_2, respectively, that agree on the values of all the free variables in a formula φ, then $\|\varphi\|_{v_1}^{W_1} = \|\varphi\|_{v_2}^{W_2}$.*

Proof. The proof is carried out by simultaneous induction on t and φ. □

Note 5.3.13. Theorem 5.3.12 shows that indeed every term of $\mathcal{L}^C$ has the same interpretation in all transitive models of RST_C (i.e., universes) which contain the values of its parameters and interpret the constants in C in the same way. Thus, the identity of the set denoted by a term t is independent of the exact extension of the assumed surrounding universe of sets. Note again that a universe is here any transitive collection of sets which is closed under certain operations. For instance, a platonist can take W to be V, the cumulative universe of ZF. One can also take W to be V_κ for any κ such that V_κ is a universe. However, W can also be taken as a much smaller, and very concrete set. Thus in Section 6.1 we work within the universe J_2 (the second set in Jensen's constructible hierarchy). This is in fact the minimal first-order universe which includes an infinite set. However, bigger (but still concrete and effectively constructible) universes, like J_ω or J_{ω^ω}, might also be considered. The latter is the minimal AL universe (see Section 6.2).

5.4 Basic Set Theoretical Notions

5.4.1 Standard Set Notations

In $\mathcal{L}^C$ we can introduce as abbreviations many standard mathematical notations and prove their basic properties in RST_C. The lemma below specifies some examples.

Note 5.4.1. As mentioned, we are using different parenthesis in the formal languages and in the metalanguage ($\{\!|\ |\!\}$ and $\{\ \}$, respectively). To be extremely precise we should have also used different notations for each of the abbreviations mentioned below, as well as for standard symbols such as $+, \cdot$ which appear in the sequel. (In fact, we should have used different notations for $\in$ and $=$ as well). However, for readability we shall not do so, and trust the reader to deduce the correct use from the context.

Lemma 5.4.2. *The following notations are available in* RST^{FOL}*:*

- $\emptyset := \{\!|\, x \mid x \in x \,|\!\}$.
- $\{\!|t_1, ..., t_n|\!\} := \{\!|\, x \mid x = t_1 \vee ... \vee x = t_n \,|\!\}$, *where* x *is fresh.*
- $\langle s, t\rangle := \{\!|\{\!|s|\!\}, \{\!|s, t|\!\}|\!\}$. $\langle t_1, ..., t_n\rangle := \langle \langle t_1, ..., t_{n-1}\rangle, t_n\rangle$.
- $\{\!|\, x \in t \mid \varphi \,|\!\} := \{\!|\, x \mid x \in t \wedge \varphi \,|\!\}$, *provided* $\varphi \succ \emptyset$ *and* $x \notin Fv(t)$.
- $\{\!|\, t \mid x \in s \,|\!\} := \{\!|\, y \mid \exists x. x \in s \wedge y = t \,|\!\}$, *where* y *is fresh and* $x \notin Fv(s)$.
- $s \times t := \{\!|\, x \mid \exists a \exists b. a \in s \wedge b \in t \wedge x = \langle a, b\rangle \,|\!\}$, *where* x, a, b *are fresh.*
- $s \cup t := \{\!|\, x \mid x \in s \vee x \in t \,|\!\}$, *where* x *is fresh.*
- $s \cap t := \{\!|\, x \mid x \in s \wedge x \in t \,|\!\}$, *where* x *is fresh.*
- $\cup t := \{\!|\, x \mid \exists y \in t. x \in y \,|\!\}$, *where* x, y *are fresh.*
- $\cap t := \{\!|\, x \mid x \in \cup t \wedge \forall y \in t. x \in y \,|\!\}$, *where* x, y *are fresh.*
- $\iota x.\varphi := \bigcup \{\!|\, x \mid \varphi \,|\!\}$, *provided* $\varphi \succ \{x\}$.[10]
- $Dom(t) := \{\!|\, x \mid \exists z \exists v \exists y. z \in t \wedge v \in z \wedge y \in v \wedge x \in v \wedge z = \langle x, y\rangle \,|\!\}$, *where* $z, v, x,$ *and* y *are fresh.*
- $Im(t) := \{\!|\, y \mid \exists z \exists v \exists x. z \in t \wedge v \in z \wedge y \in v \wedge x \in v \wedge z = \langle x, y\rangle \,|\!\}$, *where* $z, v, x,$ *and* y *are fresh.*

In RST^{AL} *the following notations are also available:*

[10] Due to the Extensionality Axiom, if $\varphi \succ \{x\}$, then the term above for $\iota x.\varphi$ denotes $\emptyset$ if there is no set which satisfies φ, and it denotes the union of all the sets which satisfy φ otherwise. In particular: this term has the property that if there is exactly one set which satisfies φ, then $\iota x.\varphi$ denotes this unique set since $\cup\{a\} = a$. Note that the definition of $\iota x.\varphi$ taken here is simpler than the definition used in [9], which was $\cap\{\!|\, x \mid \varphi \,|\!\}$ (where some caution was taken so that the term is always well defined).

- $TH(x) := x \cup \{\!| y \mid (TC_{x,y} y \in x)(x,y) |\!\}$, *the transitive hull of* x.[11]
- $\mathbb{N} := \{\!| x \mid x = \emptyset \vee \exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x,y) |\!\}$.[12]
- $\mathcal{HF} = \{\!| x \mid \exists y \exists z.x \in y \wedge z = \{\!|\emptyset|\!\} \wedge (TC_{z,y} \exists u \in z \exists v \in z.y = z \cup \{\!| u \cup \{\!| v |\!\}|\!\})(z,y) |\!\}$.

It is routine to verify that all these terms are indeed well defined, and that their basic properties are provable in RST^{FOL} or in RST^{AL}. Note that to prove that $\neg\exists x.x \in \emptyset$ we use the Restricted $\in$-induction Scheme. To give another example, $\forall x(x \in \{\!| x \in t \mid \varphi |\!\} \leftrightarrow x \in t \wedge \varphi)$ is a trivial consequence of the Comprehension Axiom and the definition of $\{\!| x \in t \mid \varphi |\!\}$ as $\{\!| x \mid x \in t \wedge \varphi |\!\}$. Also, by Definition 5.3.5, Theorem 5.3.8, and the fact that $x \notin Fv(t)$, we get that $\|\{\!| x \in t \mid \varphi |\!\}\|_v^W = \left\{a \in W \mid \|x \in t \wedge \varphi\|_{v[x:=a]}^W = \mathbf{t}\right\} = \left\{a \in \|t\|_v^W \mid \|\varphi\|_{v[x:=a]}^W = \mathbf{t}\right\}$.

The fact that $\langle s,t\rangle$ is a term in our language implies only that if $z \notin Fv(t) \cup Fv(s)$, then $z = \langle s,t\rangle \succ \{z\}$ and $\langle s,t\rangle = z \succ \{z\}$. However, in [8] another formula was constructed that states that t is equal to the ordered pair $\langle r,s\rangle$:

$$t \check{=} \langle r,s\rangle := \exists u \exists v (P(t,u,v) \wedge P(u,r,r) \wedge P(v,r,s))$$

where $P(t,x,y) = x \in t \wedge y \in t \wedge \forall w(w \in t \rightarrow w = x \vee w = y)$ and w is a fresh variable. Denote by $\langle r,s\rangle \check{\in} t$ the formula: $\exists u \in t(u \check{=} \langle r,s\rangle)$, where u is a fresh variable which does not occur in t, r, or s. The following is then proved in [8]:

Lemma 5.4.3.

1. $t \check{=} \langle x,s\rangle \succ \{x\}$, $t \check{=} \langle s,x\rangle \succ \{x\}$ *and* $t \check{=} \langle x,y\rangle \succ \{x,y\}$ *for* $x,y \notin Fv(t)$.
2. $\langle x,s\rangle \check{\in} t \succ \{x\}$, $\langle s,x\rangle \check{\in} t \succ \{x\}$ *and* $\langle x,y\rangle \check{\in} t \succ \{x,y\}$ *for* $x,y \notin Fv(t)$.
3. $\vdash_{RST_C} r = \langle s,t\rangle \leftrightarrow r \check{=} \langle s,t\rangle$.

Note that the first and second coordinate of a pair can be extracted using ι:

$$\pi_1(z) := \iota x.\exists y.z \check{=} \langle x,y\rangle \ , \ \pi_2(z) := \iota y.\exists x.z \check{=} \langle x,y\rangle$$

[11] That is, the smallest transitive set (with respect to inclusion) which contains x.
[12] See Section 5.5 for more details.

5.4.2 Classes

In standard mathematical practice, one always works with extensions of the base language by definitions of new relations and functions. We wish to allow such extensions in our framework too.[13] We start with the problem of introducing new predicate symbols (leaving the addition of new function symbols to Subsection 5.4.3). Since n-ary predicates can be reduced in the framework of set theory to unary predicates, we first focus on the introduction of new unary predicates. In standard practice an extension of this sort is carried out by introducing a new unary predicate symbol P and either treating $P(t)$ as an abbreviation for $\varphi(t)$ for some formula φ, or (what is more practical) adding $\forall x\,(P(x) \leftrightarrow \varphi)$ as an axiom to the (current version of the base) theory, obtaining by this a conservative theory in the extended language . However, in the set theoretical framework it is possible and frequently more convenient to uniformly use class terms, rather than introduce a new predicate symbol each time (the origins of this approach date back to [92]). Thus, instead of writing "$P(t)$" one uses an appropriate class term S and writes "$t \in S$". Whatever approach is chosen, the definition of a safety relation demands that any new atomic formula which is generated in an extension of the language should be safe with respect to $\emptyset$. Therefore, class terms should be restricted in our framework so that "$t \in S$" is safe with respect to $\emptyset$. Accordingly, we extend our language by incorporating class terms which are objects of the form $\{\!|x \,\hat{|}\, \varphi|\!\}$, where $\varphi \succ \emptyset$. (A closed class term is a class term $\{\!|x \,\hat{|}\, \varphi|\!\}$ where $Fv(\varphi) \subseteq \{x\}$.) The extension of the language is constructed in the standard way, as described, e.g., in [72]:

- $t \in \{\!|x \,\hat{|}\, \varphi|\!\}$ (where t is free for x in φ) is equivalent to (and may be taken as an abbreviation for) $\varphi\left\{\frac{t}{x}\right\}$.

Note that the new atomic formulas of the form "$t \in \{\!|x \,\hat{|}\, \varphi|\!\}$" indeed represent formulas which are safe with respect to $\emptyset$ (by Lemma 5.2.4). Further standard abbreviations (see again [72]) are:

- $t \subseteq \{\!|x \,\hat{|}\, \varphi|\!\}$ is an abbreviation for $\forall z \left(z \in t \rightarrow z \in \{\!|x \,\hat{|}\, \varphi|\!\}\right)$.
- $t = \{\!|x \,\hat{|}\, \varphi|\!\}$ and $\{\!|x \,\hat{|}\, \varphi|\!\} = t$ stand for $\forall z \left(z \in t \leftrightarrow z \in \{\!|x \,\hat{|}\, \varphi|\!\}\right)$.

[13] Actually, we have already used such an extension in the introduction of $\subseteq$.

- $\{x \mid \varphi\} = \{y \mid \psi\}$ is an abbreviation for $\forall z \left(z \in \{x \mid \varphi\} \leftrightarrow z \in \{y \mid \psi\} \right)$.
- $\{x \mid \varphi\} \in t$ is an abbreviation for $\exists z. z = \{x \mid \varphi\} \wedge z \in t$.
- $\{x \mid \varphi\} \in \{y \mid \psi\}$ is an abbreviation for $\exists z. z = \{x \mid \varphi\} \wedge z \in \{y \mid \psi\}$.

Note that these formulas are merely abbreviations for formulas which are *not* necessarily atomic (even though, $t \subseteq \{x \mid \varphi\}$ also happens to be safe with respect to $\emptyset$).

It should be emphasized that a class term is not a valid set term in the language, only a definable predicate. This means that everything we can say using the new notation can be formulated already in $\mathcal{L}^C$; i.e., adding the new notation does not enhance the expressive power of $\mathcal{L}^C$, it only increases the ease of using it.

A further conservative extension of the language we shall use incorporates free class variables, $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$, and free function variables, $\boldsymbol{F}, \boldsymbol{G}$, into $\mathcal{L}^C$ (as in free-variable second-order logic, see [102]). These variables stand for arbitrary class or function terms (the latter will be defined in the sequel — see Def. 5.4.12), and they may only appear as *free* variables, *never to be quantified*.[14] We allow occurrences of such variables inside a formula in a class term or a function term. One may think of a formula with such variables as a schema, where the variables play the role of "place holders" and whose substitution instances are the official formulas of the language. In fact, their immediate instances are not formulas of $\mathcal{L}^C$, but formulas of the extended language with class terms, but as noted above, there is no difficulty in interpreting such formulas in $\mathcal{L}^C$ (for an example of such translation see Example 6.1.18 on page 100). These kinds of schemes are similar to the schemes in the formalization of the Comprehension Schema and the Restricted $\in$-induction Schema, where φ is a "place holder" to be substituted by any valid formula of the language. In effect, a formula $\psi(\boldsymbol{X})$ with free class variable $\boldsymbol{X}$ can be intuitively interpreted as "for any *given* class X, $\psi(X)$ holds". Thus, a free-variable formulation has the flavor of a universal formula.[15] Therefore, this addition enables us to make statements about *all* potential classes and *all* potential functions.

[14] Admitting the quantification over such variables would lead to stronger languages and systems.

[15] It is worth noting that in such a language there isn't, in general, a formula which states the existence of a class or a function.

For a class term $\{\!|x \,\hat{|}\, \varphi|\!\}$ we define $\left\|\{\!|x \,\hat{|}\, \varphi|\!\}\right\|_v^W = \left\{a \in W \mid \|\varphi\|^W_{v[x:=a]} = \mathbf{t}\right\}$. We say that the class term defines the latter collection. Note that this collection may not be an element of W. If t is a closed set term of $\mathcal{L}^C$ such that $\|t\|^W = X$ we also say that t *defines* X (and so X *is definable by* t).

Definition 5.4.4. Let X be a collection of elements in W.

- X is a $\succ$*-set* if there is a closed term that defines it.
- X is a $\succ$*-class* if there is a closed class term which defines it.

Note that, by Theorem 5.3.8, X is a $\succ$*-set* iff $X \in W$.

Notation. If X is a $\succ$-set, $\widetilde{X}$ denotes some closed term which defines it. If X is a $\succ$-class, $\bar{X}$ denotes some class term which defines it. (The exact choices are irrelevant).

Proposition 5.4.5. *The following holds:*

1. *Every $\succ$-set is a $\succ$-class.*
2. *The intersection of a $\succ$-class with a $\succ$-set is a $\succ$-set.*
3. *Every $\succ$-class that is contained in a $\succ$-set is a $\succ$-set.*

Proof.

1. If X is a $\succ$-set, the formula $x \in \widetilde{X}$ is safe with respect to $\{x\}$, thus by Lemma 5.2.4 we get that $x \in \widetilde{X} \succ \emptyset$. This implies that $\{\!|x \,\hat{|}\, x \in \widetilde{X}|\!\}$ is a class term which defines X, hence X is a $\succ$-class.
2. Let X be a $\succ$-class and Y be a $\succ$-set. Then, $X \cap Y$ can be defined by the term $\{\!|z \mid z \in \bar{X} \wedge z \in \widetilde{Y}|\!\}$. Since $z \in \bar{X} \succ \emptyset$ and $z \in \widetilde{Y} \succ \{z\}$, we get that $z \in \bar{X} \wedge z \in \widetilde{Y} \succ \{z\}$, hence $X \cap Y$ is a $\succ$-set.
3. Follows immediately from 2., since if $X \subseteq Y$, then $X = X \cap Y$. □

Proposition 5.4.6. *The following holds:*

- *Let Y be a $\succ$-set. If $\varphi \succ \emptyset$ and $Fv(\varphi) \subseteq \{x\}$, then $\{x \in Y \mid \varphi\}$ is a $\succ$-set.*
- *If $\varphi \succ \{x_1, ..., x_n\}$, then $\{\langle x_1, ..., x_n\rangle \mid \varphi\}$ is a $\succ$-set.*

Proof.

- $\{x \in Y \mid \varphi\}$ is definable by $\{\!|\, x \mid x \in \widetilde{Y} \wedge \varphi \,|\!\}$.
- $\{\langle x_1, ..., x_n\rangle \mid \varphi\}$ is definable by $\{\!|\, z \mid \exists x_1 ... \exists x_n \left(\varphi \wedge z = \langle x_1, ..., x_n\rangle\right) |\!\}$, where z is a fresh variable. □

Proposition 5.4.7. *For every n-ary C-rudimentary function (relative to $\mathcal{C}\ell$) there is a term t of $\mathcal{L}^C_{RST}$ ($\mathcal{L}^C_{RST+TC}$) with $Fv(t) \subseteq \{x_1, ...x_n\}$ that defines the function in the following sense: for any $\succ$-sets $X_1, ..., X_n$, it returns the $\succ$-set $\|t\|^W_{[x_1:=X_1,...,x_n:=X_n]}$.*

Proof. From Lemma 5.2.4 it follows that if $X_1, ..., X_n$ are $\succ$-sets and φ is a formula such that $Fv(\varphi) \subseteq \{y, v_1, ...v_n\}$ and $\varphi \succ \{y\}$, then $\left\{y \mid \varphi\left\{\frac{\widetilde{X_1}}{v_1}, ..., \frac{\widetilde{X_n}}{v_n}\right\}\right\}$ is a $\succ$-set. Therefore the proposition easily follows from Theorem 5.3.7. □

Proposition 5.4.8. *If X, Y are $\succ$-classes, so are $X \cup Y$, $X \cap Y$, $X \times Y$, $W - X$.*

Proof.

- $\overline{X \cup Y} = \{\!|\, x \,\hat{|}\, x \in \bar{X} \vee x \in \bar{Y} \,|\!\}$, and $x \in \bar{X} \vee x \in \bar{Y} \succ \emptyset$.
- $\overline{X \cap Y} = \{\!|\, x \,\hat{|}\, x \in \bar{X} \wedge x \in \bar{Y} \,|\!\}$, and $x \in \bar{X} \wedge x \in \bar{Y} \succ \emptyset$.
- $\overline{X \times Y} = \{\!|\, x \,\hat{|}\, \exists a \exists b \left(a \in \bar{X} \wedge b \in \bar{Y} \wedge x \doteq \langle a, b\rangle\right) |\!\}$. $a \in \bar{X} \wedge b \in \bar{Y} \succ \emptyset$ and by Prop. 5.4.3 $x \doteq \langle a, b\rangle \succ \{a, b\}$, thus $\exists a \exists b \left(a \in \bar{X} \wedge b \in \bar{Y} \wedge x \doteq \langle a, b\rangle\right) \succ \emptyset$.
- $\overline{W - X} = \{\!|\, x \,\hat{|}\, x \notin \bar{X} \,|\!\}$, and since $x \in \bar{X} \succ \emptyset$ also $x \notin \bar{X} \succ \emptyset$. □

Proposition 5.4.9. *The following holds:*

- *If X is a $\succ$-class, then $z \subseteq \bar{X} \succ \emptyset$.*
- *If X is a $\succ$-class, then so is $P_W(X) = \{z \in W \mid z \subseteq X\}$.*

Proof.

- Since $a \in \bar{X} \succ \emptyset$, by Lemma 5.2.4, $\forall a \left(a \in z \to a \in \bar{X}\right) \succ \emptyset$.
- Follows from the first claim of the Proposition, since $P_W(X)$ is definable by $\{\!|\, z \,\hat{|}\, z \subseteq \bar{X} \,|\!\}$. □

For a class term s we denote by 2^s the class term $\{\!|z \,\hat{|}\, z \subseteq s|\!\}$. Note that for any assignment v in W and class term s, $\|2^s\|_v^W$ is equal to $P_W\left(\|s\|_v^W\right)$, i.e., the intersection of the power set of $\|s\|_v^W$ and W. In general, for every other universe the interpretation of 2^s will be the collection of all subsets of $\|s\|_v^W$ in that universe. This demonstrates the main difference between set terms and class terms. The interpretation of set terms is absolute, whereas the interpretation of class terms might not be (though membership in the interpretation of a class term is absolute).

5.4.3 Relations and Functions

Definition 5.4.10. A $\succ$-*relation from* a $\succ$-*class* X *to* a $\succ$-*class* Y is a $\succ$-class A such that $A \subseteq X \times Y$. A $\succ$-relation is called *small* if it is a $\succ$-set.

Terminology. In what follows, claiming that a certain object is *available in* RST_C *as a* $\succ$-*relation* means that it is definable as a $\succ$-relation in $\mathcal{L}^C$, and that its basic properties are provable in RST_C.[16]

Proposition 5.4.11. *Let* X, Y *be* $\succ$-*classes and* R *a* $\succ$-*relation from* X *to* Y.

1. R *is small iff* $Dom\,(R)$ *and* $Im\,(R)$ *are* $\succ$-*sets.*
2. $R^{-1} = \{\langle y, x\rangle \,|\, \langle x, y\rangle \in R\}$ *is available in* RST_C *as a* $\succ$-*relation from* Y *to* X. *If* R *is small, then so is* R^{-1}.
3. *If* $Z \subseteq X$ *is a* $\succ$-*class and* $U \subseteq Y$ *is a* $\succ$-*class, then* $R \cap (Z \times U)$ *is available in* RST_C *as a* $\succ$-*relation from* Z *to* U .

Proof.

1. ($\Rightarrow$) If R is a $\succ$-set, then $\exists y.\, \langle x, y\rangle \,\check{\in} \widetilde{R} \succ \{x\}$ and $\exists x.\, \langle x, y\rangle \,\check{\in} \widetilde{R} \succ \{y\}$. Thus, $Dom\,(R)$ is defined by $\{\!|\, x \mid \exists y.\, \langle x, y\rangle \,\check{\in} \widetilde{R} |\!\}$ and $Im\,(R)$ by $\{\!|\, y \mid \exists x.\, \langle x, y\rangle \,\check{\in} \widetilde{R} |\!\}$. Note that we can use $\check{\in}$ since the $\succ$-relation is small.
   ($\Leftarrow$) If $Dom\,(R)$ and $Im\,(R)$ are $\succ$-sets, then $Dom\,(R) \times Im\,(R)$ is a $\succ$-set, as $\times$ is a rudimentary function. Since R is a $\succ$-class such that $R \subseteq Dom\,(R) \times Im\,(R)$, Prop. 5.4.5 entails that R is a $\succ$-set.

[16] The "basic properties" of a certain object is of course a fuzzy notion. However, it is not difficult to identify its meaning in each particular case, as will be demonstrated in several examples below.

2. Since R is a $\succ$-class, we can define $\overline{R^{-1}} = \{\!|z \,\hat{|}\, \exists x \exists y \left(\langle x,y\rangle \in \bar{R} \wedge z \doteq \check{} \langle y,x\rangle\right)|\!\}$. $\exists x \exists y \left(\langle x,y\rangle \in \bar{R} \wedge z \doteq \check{} \langle y,x\rangle\right) \succ \emptyset$, as $z \doteq \check{} \langle y,x\rangle \succ \{x,y\}$ and $\langle x,y\rangle \in \bar{R} \succ \emptyset$. It is standard to prove in RST_C related basic properties, such as $\langle x,y\rangle \in R \leftrightarrow \langle y,x\rangle \in R^{-1}$ and $(R^{-1})^{-1} = R$. If R is a $\succ$-set, R^{-1} can be defined by $\{\!|z \mid \exists x \exists y \left(\langle x,y\rangle \in \widetilde{R} \wedge z = \langle y,x\rangle\right)|\!\}$, hence R^{-1} is a $\succ$-set.

3. Surely $R \cap (Z \times U) \subseteq Z \times U$. By Prop. 5.4.8, since R, Z, U are $\succ$-classes, we have that $R \cap (Z \times U)$ is a $\succ$-class. □

Next we extend our framework by the introduction of new function symbols. This poses a new difficulty. While new relation symbols are introduced in a static way, new function symbols are usually introduced *dynamically*: a new function symbol is made available after appropriate existence and uniqueness theorems have been proven (see, e.g., [103] for a precise detailed description of the process). However, one of the main guiding principles of our framework is that its languages should be treated exclusively in a *static* way, independent of any set of axioms. Thus function symbols, too, are introduced only as abbreviations for definable operations on sets.

Definition 5.4.12. For a $\succ$-class X and a term t of $\mathcal{L}^C$, $\lambda x \in \bar{X}.t$ is a *function term* which is an abbreviation for $\{\!|z \,\hat{|}\, \exists x \exists y \left(z \doteq \check{} \langle x,y\rangle \wedge x \in \bar{X} \wedge y = t\right)|\!\}$.

Definition 5.4.13.

- A $\succ$-class F is called a *$\succ$-function on a $\succ$-class* X if there is a term t of $\mathcal{L}^C$ such that $Fv(t) \subseteq \{x\}$, and $F = \left\|\lambda x \in \bar{X}.t\right\|$. t is said to be a term which *represents* F.

- A $\succ$-function on $\succ$-class X is called *small* if it is a $\succ$-set.

- A $\succ$-class is called a $\succ$-function if it is a $\succ$-function on some $\succ$-class.

From this definition it follows that the standard functionality condition is always satisfied in a *$\succ$-function*.

Terminology. In what follows, claiming that an object is *available in RST_C as a $\succ$-function* means that it is definable as a $\succ$-function in $\mathcal{L}^C$, and that its basic properties are provable in RST_C.

Proposition 5.4.14. *A $\succ$-set is a function according to the standard mathematical definition (a relation that satisfies the functionality condition) iff it is a small $\succ$-function.*

Proof. Let A be a $\succ$-set which is a relation that satisfies the functionality condition. Since A is a $\succ$-set, there is a closed term $\widetilde{A}$ that defines it. A is a $\succ$-function on the $\succ$-set $Dom(A)$ since the term $t = \iota y.\langle x, y\rangle \check{\in} \widetilde{A}$ represents it (see Lemma 5.4.2). The term t is legal and it represents A since $\langle x, y\rangle \check{\in} \widetilde{A} \succ \{y\}$ and A satisfies the functionality condition. The converse is trivial, since for every small $\succ$-function there is a term representing it, and thus the functionality condition clearly holds by the equality axioms of FOL. □

Notation. Let $F = \left\|\lambda x \in \bar{X}.t\right\|$ be a $\succ$-function. We employ standard β-reduction for λ terms. That is, if s is a term free for x in t, we write $F(s)$ for $t\left\{\frac{s}{x}\right\}$. Hence $F(s) = y$ is an abbreviation for $t\left\{\frac{s}{x}\right\} = y$, and so if $y \notin Fv[t] \cup Fv[s] \setminus \{x\}$, then $F(s) = y \succ \{y\}$.

Proposition 5.4.15 (The predicative class form of the axiom of replacement)**.** *Let F be a $\succ$-function on a $\succ$-class X. Then for every $\succ$-set $A \subseteq X$, the image of A under F (i.e., $F[A] = \{f(a) \mid a \in A\}$) is a $\succ$-set.*

Proof. Let F be a $\succ$-function on a $\succ$-class X, and let A be a $\succ$-subset of X. Then, $\{\!|\, y \mid \exists a \in \widetilde{A}.F(a) = y \,|\!\}$ is a closed term which defines $F[A]$. □

The next Lemma provides a straightforward generalization of Definition 5.4.13 to functions of several variables. We omit its straightforward proof.

Lemma 5.4.16. *If $X_1, ..., X_n$ are $\succ$-classes and t is a term such that . $Fv(t) \subseteq \{x_1, ..., x_n\}$, then $F = \left\|\lambda x_1 \in \bar{X_1}, ..., x_n \in \bar{X_n}.t\right\|$ is available in RST_C as a $\succ$-function on $X_1 \times ... \times X_n$. (where $\lambda x_1 \in \bar{X_1}, ..., x_n \in \bar{X_n}.t$ is an abbreviation for $\{\!|\, \langle \langle x_1, ..., x_n\rangle, t\rangle \mid \langle x_1, ..., x_n\rangle \in \bar{X_1} \times ... \times \bar{X_n} \,|\!\}$).*

Corollary 5.4.17. *Every C-rudimentary function (relative to $\mathcal{C}\ell$) is available in RST_C^{FOL} (RST_C^{AL}) as a $\succ$-function.*

Proof. Follows immediately from Lemma 5.4.16 and Prop. 5.4.7. □

Proposition 5.4.18. *Let F be a $\succ$-function on a $\succ$-class X.*

1. *F is small iff X is a $\succ$-set.*
2. *If Y_0 is a $\succ$-class, then $F^{-1}[Y_0] = \{a \in X \mid F(a) \in Y_0\}$ is a $\succ$-class. If F is small, then $F^{-1}[Y_0]$ is a $\succ$-set.*
3. *If $X_0 \subseteq X$ is a $\succ$-class, then $F \restriction_{X_0}$ is available in RST_C as a $\succ$-function on X_0.*
4. *If G is a $\succ$-function on a $\succ$-class Y and $Im(F) \subseteq Y$, then $G \circ F$ is available in RST_C as a $\succ$-function on X.*
5. *If G is a $\succ$-function on a $\succ$-class Y and F and G agree on $X \cap Y$, then $G \cup F$ is available in RST_C as a $\succ$-function on $X \cup Y$.*
6. *If Z is a $\succ$-class, then the identity map on Z and any constant function on Z are available in RST_C as $\succ$-functions.*

Proof.

1. ($\Rightarrow$) F is a $\succ$-set, thus $Dom(F) = X$ is also a $\succ$-set, since Dom is a rudimentary function.
   ($\Leftarrow$) Let t be a term that represents F. If X is a $\succ$-set, then $F = \left\| \lambda x \in \widetilde{X}.t \right\|$ which is a $\succ$-set.

2. $\overline{F^{-1}[Y_0]} = \left\{ a \,\hat{|}\, a \in \bar{X} \wedge F(a) \in \bar{Y_0} \right\}$. Since $a \in \bar{X} \wedge F(a) \in \bar{Y_0} \succ \emptyset$, we get that $F^{-1}[Y_0]$ is a $\succ$-class. If F is small, then by 1. we have that X is a $\succ$-set. The fact that $F^{-1}[Y_0] \subseteq X$ implies that $F^{-1}[Y_0]$ is a $\succ$-set.

3. If t is a term that represents F, the same term t represents $F \restriction_{X_0}$.

4. Denote by t_F, t_G terms that represent the $\succ$-functions F, G, respectively. Thus, $G \circ F = \left\| \lambda x \in \overline{X}.t_G \left\{ \frac{t_F(x)}{x} \right\} \right\|$. It is easy to see that standard properties, such as the associativity of $\circ$, are provable in RST_C.

5. Denote by t_F, t_G terms that represent the $\succ$-functions F, G, respectively. Thus, $F \cup G = \left\| \lambda x \in \bar{X} \cup \bar{Y}.\imath y. \left(\left(x \in \bar{X} \wedge y = t_F \right) \vee \left(x \in \bar{Y} - \bar{X} \wedge y = t_G \right) \right) \right\|$. Since each of the adjuncts is safe with respect to $\{y\}$, we get that the term is valid. It is

easy to verify that in RST_C basic properties, such as $\forall x \in \bar{X}.\overline{G \cup F}(x) = \bar{F}(x)$, are provable.

6. $id_Z = \left\|\lambda z \in \bar{Z}.z\right\|$, and for any $A \in J_2$, $const_A = \left\|\lambda z \in \bar{Z}.\tilde{A}\right\|$. Proving properties such as $\forall x, y \in \bar{Z}.const_A(x) = const_A(y)$ in RST_C is straightforward. □

5.5 The Natural Numbers

We follow the standard construction of the natural numbers:

$$0 := \emptyset,$$
$$n+1 := S(n),$$

where $S(n) = n \cup \{n\}$. Obviously, each $n \in \mathbb{N}$ is a $\succ$-set, and $\mathbb{N}$ (the set of natural numbers) is contained in $\mathcal{HF}$.

5.5.1 The Natural Numbers in RST_C^{FOL}

In this section we assume $HF \in C$.

Proposition 5.5.1. *$\mathbb{N} = \{0, 1, 2, ...\}$ is a $\succ$-set.*

Proof. Denote by $Ord(x)$ the formula $Trans(x) \wedge Linear(x)$, where:

$$Linear(x) := \forall z \forall y (z \in x \wedge y \in x \rightarrow (z \in y \vee y \in z \vee z = y))$$
$$Trans(n) := \forall z \forall y (y \in x \wedge z \in y \rightarrow z \in x)$$

It is straightforward to verify that $Ord(n) \succ \emptyset$. Hence, $\mathbb{N}$ is definable by the term $\{\!|\, n \in HF \mid Ord(n) \,|\!\}$. □

Lemma 5.5.2. *The followings are provable in RST_C^{FOL}:*

1. $\forall a.Ord(a) \rightarrow \forall z \in a.Ord(z)$
2. $\forall a, b.(Ord(a) \wedge Ord(b)) \rightarrow (a \in b \vee b \in a \vee a = b)$
3. $\forall x \in HF.x = \emptyset \vee \exists z \in x \neg Ord(z) \vee \exists z.max(x) = z$, *where* $max(x) = z$ denotes the formula $z \in x \wedge \forall w \in x.w \in z \vee w = z$.

Proof. The proofs of 1. and 2. are standard. For 3., let $\varphi(x)$ be the formula $x = \emptyset \vee \exists z \in x \neg Ord(z) \vee \exists z.max(x) = z$. Then $\varphi \succ \emptyset$, and so we can prove $\forall x \in HF.\varphi$ by induction on HF (Prop. 5.2.8(2)). Clearly we have $\varphi(\emptyset)$. Assume $\varphi(x) \wedge \varphi(y)$. We prove $\varphi(x \cup \{y\})$. If $x = \emptyset$, then $max(x \cup \{y\}) = y$, thus $\varphi(x \cup \{y\})$ obtains. If $\exists z \in x \neg Ord(z)$, then $\exists z \in x \cup \{y\} . \neg Ord(z)$, and again $\varphi(x \cup \{y\})$ holds. Otherwise, we have that $x \neq \emptyset \wedge \forall z \in x Ord(z) \wedge \exists z.max(x) = z$. If y is not an ordinal, then $\exists z \in x \cup \{y\} . \neg Ord(z)$, otherwise, denote by z_0 the maximum of x. Then we have $Ord(y) \wedge Ord(z_0)$ and by claim (2) of this Lemma we get $y \in z_0 \vee z_0 \in y \vee y = z_0$. If $y \in z_0$ or $y = z_0$ then $max(x \cup \{y\}) = z_0$, otherwise the transitivity of y implies that $max(x \cup \{y\}) = y$.[17] □

Proposition 5.5.3. $\vdash_{RST_C^{FOL}} \forall n \in \widetilde{\mathbb{N}}.n = 0 \vee \exists k \in n.n = S(k)$.

Proof. Let n be an element in $\mathbb{N}$. Then $Ord(n)$ and $n \in HF$. By Lemma 5.5.2(1), we get $\forall z \in n.Ord(z)$. Hence Lemma 5.5.2(2) implies that $n = \emptyset \vee \exists z.max(n) = z$. If $n = \emptyset$ we are done. Otherwise, denote by z_0 the maximum of n. We prove that $n = S(z_0)$. If $x \in n$, then $x \in z_0$ (i.e., $x \in z_0 \vee x = z_0$) by the maximality of z_0. For the converse, assume $x \in z_0$. If $x = z_0$ then clearly $x \in n$. If $x \in z_0$, then by the transitivity of n we again conclude that $x \in n$. □

The next proposition shows that the induction rule of Peano's arithmetic is available in RST_C^{FOL} for $\varphi \succ \emptyset$.

Proposition 5.5.4. $\vdash_{RST_C^{FOL}} (\varphi(0) \wedge \forall x(\varphi \to \varphi(S(x)))) \to \forall x \in \widetilde{\mathbb{N}}.\varphi$, *where* $\varphi \succ \emptyset$

Proof. Let $\varphi \succ \emptyset$, and assume $\varphi(0) \wedge \forall x(\varphi \to \varphi(S(x)))$. Denote by $\psi(x)$ the formula $Ord(x) \to \varphi(x)$. Since $\varphi(x) \succ \emptyset$, using Lemma 5.2.8(2) we deduce $(\psi(\emptyset) \wedge \forall x \forall y(\psi(x) \wedge \psi(y) \to \psi(x \cup \{y\}))) \to \forall x \in HF.\psi(x)$. Clearly $\psi(\emptyset)$ is provable in RST_C^{FOL}, since we have $\varphi(0)$. Now, assume $\psi(x) \wedge \psi(y)$. We show that we can prove $\psi(x \cup \{y\})$. Suppose $Ord(x \cup \{y\})$. Thus, it must be the case that both $Ord(x)$ and $Ord(y)$ (since otherwise we get $\neg Ord(x \cup \{y\})$ using Lemma 5.5.2). This, together with the assumption $\psi(x) \wedge \psi(y)$, entails $\varphi(x) \wedge \varphi(y)$. Now, by Lemma 5.5.2(2) we get that $x \cup \{y\} \in x \vee x \in x \cup \{y\} \vee x \cup \{y\} = x$. In case $x \cup \{y\} = x$, we get that $\varphi(x \cup \{y\})$, since we have $\varphi(x)$. If $x \cup \{y\} \in x$, then by $Trans(x)$ we

[17] Note that the proofs of 1. and 2. can be carried out in RST^{FOL}.

get that $y \in x$ and thus again $x \cup \{y\} = x$. In case $x \in x \cup \{y\}$, then $x = y$[18] and thus $x \cup \{y\} = x \cup \{x\}$. By the initial assumption we have $\varphi(x \cup \{x\})$ which entails $\varphi(x \cup \{y\})$. □

Next we give a direct construction of addition and multiplication on $\mathbb{N}$ as small $\succ$-functions. This formalization enables proving the basic properties of these $\succ$-functions in RST_C^{FOL}.

Proposition 5.5.5. *The following holds:*

1. *The standard ordering $\leq$ on $\mathbb{N}$ is available in RST_C^{FOL} as a small $\succ$-relation.*
2. *The standard addition and multiplication of natural numbers are available in RST_C^{FOL} as small $\succ$-functions.*

Proof.

1. The standard ordering $<$ on $\mathbb{N}$ coincides with $\in$. Thus $\leq$ is definable by the term $\{ \langle m, n\rangle \check{\in} \widetilde{\mathbb{N} \times \mathbb{N}} \mid m = n \vee m \in n \}$. Since $\mathbb{N}$ is a $\succ$-set, so is $\mathbb{N} \times \mathbb{N}$, and $m = n \vee m \in n \succ \emptyset$, thus by Proposition 5.4.6 we get that $\leq$ is a $\succ$-set. It is now straightforward to prove in RST_C^{FOL} the basic properties of $\leq$, like it being a linear order or the existence of a successor for each element in $\mathbb{N}$.

2. Let $Func(f)$ denote the formula $\forall a, b, c\,(\langle a, b\rangle \check{\in} f \wedge \langle a, c\rangle \check{\in} f \to b = c)$, and $add(z, u, n, f)$ denote the formula:
   $(z = 0 \wedge u = n) \vee \exists z_1, u_1 \in \widetilde{\mathbb{N}}\,(\langle z_1, u_1\rangle \check{\in} f \wedge z = S(z_1) \wedge u = S(u_1))$.
   Define:

$$\psi_{add}(n, k, f) :=$$
$$Func(f) \wedge \forall x \left(x \in f \leftrightarrow \exists z, u \in \widetilde{\mathbb{N}}\,(z \leq k \wedge x = \langle z, u\rangle \wedge add(z, u, n, f)) \right)$$

   Intuitively, the formula $\psi_{add}(n, k.f)$ states that f stands for the collection $\{\langle 0, n\rangle, \langle 1, n+1\rangle, \langle 2, n+2\rangle, ..., \langle k, n+k\rangle\}$. It is standard to check that $\psi_{add}(n, f) \succ \emptyset$. Now, define addition (using Lemma 5.4.16) by the term

$$+ := \lambda n \in \widetilde{\mathbb{N}}, k \in \widetilde{\mathbb{N}}.\iota m.\exists f \in HF\,(\psi_{add}(n, k, f) \wedge \langle k, m\rangle \check{\in} f)\,.$$

[18] Here we use the fact that $\forall x.x \notin x$ which is provable in RST_C due to the Restricted $\in$-induction.

$n+k$ is a valid term since $f \in HF \succ \{f\}$ and $\langle k,m\rangle \check{\in} f \wedge \psi_{add}(n,f) \succ \{m\}$. Addition is a small $\succ$-function, as $\mathbb{N}\times\mathbb{N}$ is a $\succ$-set.
Multiplication is defined using the same method by replacing $add\,(z,u,n,f)$ with $mult\,(z,u,n,f)$, which is defined by:
$(z=0\wedge u=0)\vee\exists z_1,u_1\in\widetilde{\mathbb{N}}\,(\langle z_1,u_1\rangle\check{\in}f\wedge z=S\,(z_1)\wedge u=n+u_1)$. Since $+$ is a small $\succ$-function, we get that $\psi_{mult}(n,f)\succ\emptyset$. Thus multiplication is defined (using Lemma 5.4.16) by the term

$$\cdot := \lambda n\in\widetilde{\mathbb{N}},k\in\widetilde{\mathbb{N}}.\iota m.\exists f\in HF\,(\psi_{mult}\,(n,k,f)\wedge\langle k,m\rangle\check{\in}f)\,.$$

It is not difficult to prove in RST_C (by induction, using Prop. 5.5.4) the standard properties of addition and multiplication, such as commutativity and associativity, as well as the connection between our definitions for addition and multiplication and their standard recursive definitions, i.e.,

$$\begin{aligned} \forall x.x+0=x \quad &\text{and} \quad \forall x,y.x+S\,(y)=S\,(x+y)\\ \forall x.x\cdot 0=0 \quad &\text{and} \quad \forall x,y.x\cdot S\,(y)=x+(x\cdot y)\end{aligned}$$

□

Basic properties of the natural numbers which can be formulated in the language of first-order Peano arithmetics are provable in RST_C^{FOL} using the restricted induction principle given in Prop. 5.5.4. This is due to the fact that in their translation to $\mathcal{L}^C_{RST}$, all the quantifications are bounded in $\mathbb{N}$, and thus they are safe with respect to $\emptyset$. However, if one wishes to get the power of full induction over $\mathbb{N}$ (i.e. for *any* formula φ), this can be accomplished by adding to RST_C^{FOL} the full $\in$-induction scheme (i.e., for any formula φ) as is shown in the next proposition.

Proposition 5.5.6. *Let $RST_C^{FOL}+(\in)$ be the system obtained by omitting the restriction of $\varphi\succ\emptyset$ from the Restricted $\in$-induction Scheme. Then,*

$$\vdash_{RST_C^{FOL}+(\in)}(\varphi\,(0)\wedge\forall x\,(\varphi\to\varphi\,(S\,(x))))\to\forall x\in\widetilde{\mathbb{N}}.\varphi$$

Proof. Assume $\varphi\,(0)\wedge\forall x\,(\varphi\to\varphi\,(S\,(x)))$. Let ψ be the formula $x\in\widetilde{\mathbb{N}}\to\varphi$. Suppose $\forall x\in a.\psi\,(x)$. We shall prove $\psi\,(a)$. Assume $a\in\widetilde{\mathbb{N}}$. We have $a=0\vee\exists k\in a.a=S(k)$ by Prop. 5.5.3. In case $a=0$ we get $\varphi\,(0)$, and thus $\psi\,(a)$. Otherwise, there is $k\in a$

such that $a = S(k)$. By the second assumption we have that $\psi(k)$, and by the first assumption this entails $\psi(S(x))$. Thus, we have proven $\forall a((\forall x \in a.\psi(x)) \rightarrow \psi(a))$, and by the full $\in$-induction scheme we get $\forall a.\psi(a)$, i.e., $\forall a \in \widetilde{\mathbb{N}}.\varphi(a)$. $\square$

5.5.2 The Natural Numbers in RST^{AL}

Using the transitive closure operator, the natural numbers can be defined as a $\succ$-set, even without the constant HF and its defining axioms. This can be done by the defining formula given in Lemma 5.4.2:

$$\mathbb{N} := \{\!| x \mid x = \emptyset \vee \exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x, y) |\!\}$$

In [9] it was proven that all types of finitary inductive definitions are available in RST^{AL}. From this it follows that the collection $\mathcal{HF}$ is available in RST^{AL} as a $\succ$-set, since it is definable as the least X such that $\emptyset \in X$, and $a \cup \{b\} \in X$ whenever $a, b \in X$. Denote by t_{HF} the set term of $\mathcal{L}_{RST+TC}$ which defines $\mathcal{HF}$ given in Lemma 5.4.2. It is easy to verify that all of the axioms of HF are provable in RST^{AL} if we replace HF by t_{HF}. We now show that the definition of the natural numbers in AL is equivalent to the one given in the previous section.

Proposition 5.5.7.

$$\vdash_{RST^{AL}} \{\!| x \mid x = \emptyset \vee \exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x, y) |\!\} = \{\!| n \in t_{HF} \mid Ord(n) |\!\}$$

Proof. First notice that a formula of the form $\exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x, y)$ is provably equivalent in RST^{AL} to $(TC_{x,y}(x = S(y)))(x, \emptyset)$ (using standard first-order rules). Accordingly, assume $x = \emptyset \vee (TC_{x,y}(x = S(y)))(x, \emptyset)$. Denote by $\psi(x)$ the formula $x \in t_{HF} \wedge Ord(x)$. First we show that $\psi(x)$ implies $\psi(S(x))$. By the t_{HF}-counterpart of the second axiom of HF, we get that $x \in t_{HF}$ entails $S(x) \in t_{HF}$. To show $Linear(S(x))$, let $u, v \in x \cup \{x\}$. If $u = v = x$ we are done. If $u, v \in x$, then by linearity of x we are done. Otherwise (w.l.o.g.) $u \in x = v$ and again we are done. To show $Trans(S(x))$, let $a \in x \cup \{x\}$ and $b \in a$. In case $a \in x$ we get that $b \in x$ by the transitivity of x, and thus $b \in x \cup \{x\}$. Otherwise, $b \in a = x \subseteq x \cup \{x\}$. Hence,

we have shown that $\psi(x)$ implies $\psi(S(x))$. Now, $\psi(\emptyset)$ clearly holds. Then, using TC-induction rule (Rule (3.15)[19]) we get that $(TC_{x,y}(x = S(y)))(x, \emptyset)$ implies $\psi(x)$.

For the converse, denote by $\varphi(x)$ the formula $x = \emptyset \vee \exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x, y)$. We prove this direction using the t_{HF}-counterpart of the restricted induction given in Prop. 5.5.4 (recall that $\varphi(x) \succ \emptyset$). Clearly $\varphi(\emptyset)$ is provable. Now, assume $\varphi(x)$ and we show $\varphi(S(x))$. First notice that by Rule (3.13) we have $(TC_{x,y}(x = S(y)))(S(x), x)$. In case $x = \emptyset$, clearly $\exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(S(\emptyset), y)$ is provable. In case $\exists y.y = \emptyset \wedge (TC_{x,y}(x = S(y)))(x, y)$, using Rule 3.14 and standard first-order rules, we get $\varphi(S(x))$. Thus, in any case we have proven $\varphi(S(x))$. Applying Prop. 5.5.4 we get that $x \in t_{HF} \wedge Ord(x)$ implies $\varphi(x)$. □

From the above proposition it follows that all the results of the last section can easily be derived in RST^{AL} using the TC definition of $\mathbb{N}$. However, the use of AL makes their proofs more direct and natural. Below we provide as an example the definition of addition as a small $\succ$-function in RST^{AL}.

Example 5.5.8. Addition on $\mathbb{N}$ can be defined by the term:

$$\begin{aligned} + : \; = \; & \lambda n \in \widetilde{\mathbb{N}}, k \in \widetilde{\mathbb{N}}.\imath m.\,(k = 0 \wedge n = m) \vee \\ & \exists x, y.x = \langle \emptyset, n \rangle \wedge y \check{=} \langle k, m \rangle \wedge (TC_{x,y}(y = \langle S(\pi_1(x)), S(\pi_2(x)) \rangle))(x, y) \end{aligned}$$

This above term is valid since $(TC_{x,y}(y = \langle S(\pi_1(x)), S(\pi_2(x)) \rangle))(x, y) \succ_{AL} \{y\}$, $x = \langle \emptyset, n \rangle \succ_{AL} \{x\}$, and $y \check{=} \langle k, m \rangle \succ_{AL} \{m\}$.

Apparently, the *full* power of mathematical induction is not available in RST_C^{FOL} (even in case $HF \in C$). In contrast, the fact that AL is the underling logic makes the full induction on $\mathbb{N}$ available already in RST^{AL} (see Prop. 3.3.2).

[19] See Note 3.2.7 for the precise instance of the rule applied here.

Chapter 6

Formalizing Analysis in the Minimal Frameworks

This chapter is devoted to the minimal systems among those described in the previous chapter which meet the predicative principles. The minimal such first-order system is $RST^{FOL}_{\{HF\}}$.[1] This system, in turn, has a minimal model: the universe J_2 (in Jensen's constructible hierarchy [62]). J_2 indeed seems to be the smallest universe that suffices for predicative mathematics as conceived by Weyl, Poincaré, and Feferman.[2] Beyond the predicative philosophy behind it, working in such a minimal framework has several advantages. J_2 has the property that every element of it is definable by some closed term (and we have reduction rules for such terms, like in the λ-calculus). This allows for a concrete, computationally-oriented interpretation of the theory. What is more, it makes the framework appropriate for mechanical manipulations and for interactive theorem proving. However, the restriction to this minimal, concrete framework also has of course its price. Not all standard mathematical structures are elements of J_2 (the real line is a case in point). Hence, we have to treat such objects in a different manner, i.e., as proper classes.

[1] Though RST^{FOL} is a subsystem of $RST^{FOL}_{\{HF\}}$, $RST^{FOL}_{\{HF\}}$ is the minimal system that allows the introduction of the natural numbers as a complete set.

[2] The thesis that J_2 is sufficient for core mathematics was already put forward in [114]. However, both the motivation and the work itself were purely semantical there, and not connected to any (formal) axiomatic system. Further comparison to this work is discussed in Note 6.1.5.

We believe that this minimal predicative framework indeed suffices for scientifically applicable mathematics. We substantiate this claim by developing in it fundamental portions of classical analysis in a predicatively acceptable way. Note that the formalization presented in this chapter is based on classical logic. However, similar formalization of constructive mathematics (see, e.g., [15, 83]) can be carried out in the framework using intuitionistic logic.

Now, the minimal AL system is RST^{AL}, and its minimal model is J_{ω^ω}. We show that this system subsumes RST^{FOL}_{HF}, and offers a more congenial environment for practicing standard mathematics since all standard mathematical structures can be treated in it as sets.

The chapter is organized as follows: In Section 6.1 we investigate the minimal first-order framework. First, its key properties are presented in Subsection 6.1.1. Then, in Subsection 6.1.2, we turn to classical real analysis and demonstrate how it can be developed in this minimal predicative framework, although the reals form a proper class in it. This includes the introduction of the real line (Subsubsection 6.1.2.1), formulating and proving basic topological properties of it (Subsubsection 6.1.2.2), and formalization of fundamental properties of continuous real functions (Subsubsection 6.1.2.3). In Subsection 6.1.3 we explore extensions of the minimal first-order framework using constants. Then, in Section 6.2, we study the minimal AL framework, and compare its power to that of the minimal first-order framework.

This chapter is mainly based on [10] (with some extensions and modifications).

6.1 The Minimal First-Order Framework

In this section we focus on the framework which is the minimal among the classical first-order frameworks described in the previous chapter: the system $RST^{cFOL}_{\{HF\}}$.

Notation. For readability, in what follows we write RST^{cFOL}_{HF} instead of $RST^{cFOL}_{\{HF\}}$ and $\mathcal{L}^{HF}_{RST}$ instead of $\mathcal{L}^{\{HF\}}_{RST}$.

6.1.1 The Minimal Model

In addition to choosing the minimal formal language and system, we also restrict ourselves in this section to the minimal model of RST^{cFOL}_{HF}.

Definition 6.1.1. Jensen's constructible hierarchy [62] is defined as follows:.

$$\begin{aligned} J_0 &= \emptyset \\ J_{\alpha+1} &= Rud\,(J_\alpha) \\ J_\lambda &= \bigcup_{\alpha<\lambda} J_\alpha \quad \text{if } \lambda \text{ is a limit ordinal} \end{aligned}$$

where $Rud\,(x)$ denotes the smallest set y such that $x \subseteq y$, $x \in y$, and y is closed under application of all rudimentary functions.

Proposition 6.1.2. *Let J_1 and J_2 be the first two universes in Jensen's constructible hierarchy. J_2 with the interpretation of HF as J_1 is a model of RST^{cFOL}_{HF}.*

Proof. J_2 is clearly a universe. Since $J_1 = \mathcal{HF}$, the claim follows from Prop. 5.3.9. □

Theorem 6.1.3. *$X \in J_2$ iff there is a closed term t of $\mathcal{L}^{HF}_{RST}$ such that $\|t\|^{J_2} = X$.*

Proof. Theorem 5.3.8 entails the right-to-left implication. The converse is proven by induction. Clearly, $\|\{\!|\, x \mid x \in x \,|\!\}\|^{J_2} = \emptyset$ and $\|HF\|^{J_2} = J_1$. Now, suppose that for $A, B \in J_2$ there are closed terms t_A and t_B such that $\|t_A\|^{J_2} = A$ and $\|t_B\|^{J_2} = B$. We show that there are closed terms for any of the results of applications of $F_0, ..., F_8$ to A and B.

- $F_0\,(A, B) = \|\{\!|t_A, t_B|\!\}\|^{J_2}$
- $F_1\,(A, B) = \|t_A - t_B\|^{J_2}$
- $F_2\,(A, B) = \|t_A \times t_B\|^{J_2}$
- $F_3\,(A, B) = \|\{\!|\, x \mid \exists z \in t_A \exists u, v.\, \langle u, v\rangle \check{\in} t_B \wedge x = \langle u, z, v\rangle \,|\!\}\|^{J_2}$
- $F_4\,(A, B) = \|\{\!|\, x \mid \exists z \in t_A \exists u, v.\, \langle u, v\rangle \check{\in} t_B \wedge x = \langle z, v, u\rangle \,|\!\}\|^{J_2}$
- $F_5\,(A, B) = \|\{\!|\, Im\,(\{\!|\, w \mid w \in t_A \wedge \pi_1\,(w) \in z \,|\!\}) \mid z \in t_B \,|\!\}\|^{J_2}$

- $F_6(A) = \|\{\!|\, x \mid \exists u \in t_A . x \in u \,|\!\}\|^{J_2}$
- $F_7(A) = \|Dom(t_A)\|^{J_2}$
- $F_8(A) = \|\{\!|\, x \mid \exists u \in t_A \exists v \in t_a . u \in v \wedge x \check{=} \langle u, v \rangle \,|\!\}\|^{J_2}$ □

Corollary 6.1.4. *Let X be a collection of elements in J_2. X is a $\succ$-set iff there is a closed term of $\mathcal{L}^{HF}_{RST}$ that defines it.*

Proof. Follows immediately from the definition of $\succ$-set and Theorem 6.1.3. □

It follows from this that proper $\succ$-classes are subsets of J_2 which are not elements of J_2.

6.1.2 Real Analysis

Before we start we note that it is not difficult to formalize the definitions, propositions, and proofs of this section in the formal system RST^{cFOL}_{HF}. These translations are usually straightforward, but rather tedious. Therefore we shall usually omit them, with the exception of a few outlined examples.

6.1.2.1 The Construction of the Real Line

The standard construction of $\mathbb{Z}$, the set of integers, as the set of ordered pairs $(\mathbb{N} \times \{0\}) \cup (\{0\} \times \mathbb{N})$ can be straightforwardly carried out in RST^{cFOL}_{HF}, as can the usual construction of $\mathbb{Q}$, the set of rationals, in terms of ordered pairs of relatively prime integers. There is also no difficulty in defining the standard orderings on $\mathbb{Z}$ and $\mathbb{Q}$ as small $\succ$-relations, as well as the standard functions of addition and multiplication as small $\succ$-functions. It is easy to see that all the main properties of addition and multiplication are provable in RST^{cFOL}_{HF}, as the standard proofs by induction can be carried out within it. Furthermore, all the basic properties of $\mathbb{Z}$ and $\mathbb{Q}$ (such as $\mathbb{Q}$ being a dense unbounded field) are straightforwardly proven in RST^{cFOL}_{HF}.

Now we turn to the standard construction of the real line using Dedekind cuts. Since it is well known that the real line and its open segments (such as $(0, 1)$) are not absolute, they cannot be $\succ$-sets, only proper $\succ$-classes. Thus the collection of real numbers in RST^{cFOL}_{HF} will not be definable by a term but merely by a *definable predicate* (see Subsection 5.4.2).

Note 6.1.5. The idea of treating the collection of reals as a proper class is due to [114] (see Footnote 2 on page 91). However, the fact that our formalization is carried out within a formal system (as opposed to the work in [114]) has the benefit that our framework is significantly simpler, even from the semantical point of view. For instance, a semantic counterpart of our notion of a $\succ$-class has been used in [114], and is called there an ι-class. It is defined there as a definable subset of J_2 whose intersection with any ι-set (i.e., an element of J_2) is an ι-set. The last condition in this definition seems somewhat ad hoc. More importantly, it is not clear how this condition can be checked in general, and what kind of set theory is needed to establish the claim that certain collections are ι-classes. The definition we use here of a $\succ$-class is, in contrast, motivated by and based on purely syntactical considerations, and it is a simplification of the notion of ι-class. Note that, by Prop. 5.4.5(2), every $\succ$-class is an ι-class. It remains to be determined whether the converse holds as well.

Now define

$$\psi(u) := \forall x, y \in \widetilde{\mathbb{Q}}\,(x \in u \wedge y < x \to y \in u)$$
$$\varphi(u) := \neg\exists x \in u \forall y \in u.x \leq y$$

where $x \leq y$ stands for $x < y \vee x = y$. The formula $\psi(u)$ states that u contains every rational number less than any rational number it contains, and $\varphi(u)$ states that u has no greatest element.

Definition 6.1.6 (The Reals)**.** $\mathbb{R}$ is $\left\| \{\!| u \in \overline{P_{J_2}(\mathbb{Q}) \setminus \{\emptyset, \mathbb{Q}\}}\,\hat{|}\,\psi(u) \wedge \varphi(u) |\!\} \right\|$

Note that the term defining $\mathbb{R}$ in the above definition is a valid class term as $P_{J_2}(\mathbb{Q}) \setminus \{\emptyset, \mathbb{Q}\}$ is a $\succ$class, and it is easy to verify that $\varphi, \psi \succ \emptyset$.

It is important to note that the $\succ$-class $\mathbb{R}$ is not the "real" real-line (if such a thing really exists). However, it does contain all *computable* real numbers (such as $\sqrt{2}$, π) . For example, to see that π is a member of $\mathbb{R}$ it suffices to know that π is the interpretation of the term: $\{\!| r \in \mathbb{Q} \mid \exists n \in \mathbb{N}.r < 4 \cdot \sum_{k=0}^{n} \left(\frac{1}{4k+1} - \frac{1}{4k+3}\right) |\!\}$ (a variant of the Leibniz series).

Notation 6.1.7. We use the following standard notations: $\mathbb{Q}^+ = \{q \in \mathbb{Q} \mid 0 < q\}$ and $\mathbb{R}^+ = \{r \in \mathbb{R} \mid 0 < r\}$.[3] We also use the standard notations for intervals, i.e., for real numbers a, b: $(a, b) = \{r \in \mathbb{R} \mid a < r < b\}$ and $[a, b] = \{r \in \mathbb{R} \mid a \leq r \leq b\}$.

Proposition 6.1.8. *The following holds:*

1. *The standard ordering $<_{\mathbb{R}}$ on $\mathbb{R}$ is available in RST_{HF}^{cFOL} as a $\succ$-relation.*

2. *The standard addition and multiplication of reals are available in RST_{HF}^{cFOL} as $\succ$-functions.*

Proof.

1. The relation $<_{\mathbb{R}}$ on $\mathbb{R}$ coincides with $\subset$, thus we can define the relation $<_{\mathbb{R}}$ by $\{\!|\, \langle x, y\rangle \in \overline{\mathbb{R} \times \mathbb{R}} \,\hat{|}\, x \subset y \,|\!\}$. We have that $x \subset y \succ \emptyset$, hence $<_{\mathbb{R}}$ is a $\succ$-class. It is straightforward to prove in RST_{HF}^{cFOL} basic properties concerning $<$, such as it being a total order on $\mathbb{R}$, the density of the rationals in $\mathbb{R}$, the Archimedean Principle, etc.

2. The $\succ$-function $+_{\mathbb{R}}$ can be represented (using Lemma 5.4.16) by the term

$$+_{\mathbb{R}} = \lambda x \in \bar{\mathbb{R}}, y \in \bar{\mathbb{R}}.\{\!|\, z \mid \exists u \in x \exists v \in y.z = u + v \,|\!\}$$

since $\exists u \in x \exists v \in y.z = u + v \succ \{z\}$.
To define multiplication, let F_1 be the $\succ$-function:

$$F_1 = \left\| \lambda a \in \bar{\mathbb{R}^+}, b \in \bar{\mathbb{R}^+}.\{\!|\, z \mid \exists u \in a \exists v \in b\,(0 \leq u \wedge 0 \leq v \wedge z = u \cdot v)\,|\!\} \cup \{\!|\, x \in \widetilde{\mathbb{Q}} \mid x < 0 \,|\!\} \right\|$$

Next, define the $\succ$-function $-$ on $\mathbb{R}$ by

$$- = \left\| \lambda x \in \bar{\mathbb{R}}.\{z \,\hat{|}\, \exists u \in \widetilde{\mathbb{Q}\backslash x} \exists a \in \widetilde{\mathbb{Q}}.z + b = a\} \right\|.$$

Then, for $0 \leq a \wedge b < 0$ define $F_2(\langle a, b\rangle) := -F_1(\langle a, -b\rangle)$, for $a < 0 \wedge 0 \leq b$ define $F_3(\langle a, b\rangle) := -F_1(\langle -a, b\rangle)$, and for $a < 0 \wedge b < 0$ define $F_4(\langle a, b\rangle) :=$

[3] Notice that $\mathbb{Q}^+$ is a $\succ$-set and $\mathbb{R}^+$ is a $\succ$-class.

$F_1(\langle -a, -b\rangle)$. Now the $\succ$-function $\cdot_{\mathbb{R}}$ on $\mathbb{R}\times\mathbb{R}$ can be defined by

$$\cdot_{\mathbb{R}} := F_1 \cup F_2 \cup F_3 \cup F_4.$$

Proving in RST^{cFOL}_{HF} basic properties regarding these $\succ$-functions, such as $\mathbb{R}$ being an ordered field, is again straightforward. □

6.1.2.2 The Topology of the Reals

First we show that, as opposed to Weyl's original approach to predicative analysis [116], the least upper bound principle is provable in RST^{cFOL}_{HF} for $\succ$-subsets of $\mathbb{R}$.

Theorem 6.1.9. *It is provable in RST^{cFOL}_{HF} that every nonempty $\succ$-subset of $\mathbb{R}$ that is bounded above has a least upper bound in $\mathbb{R}$. Furthermore, the map (l.u.b) that takes each nonempty $\succ$-subset of $\mathbb{R}$ that is bounded above to its least upper bound is available in RST^{cFOL}_{HF} as a $\succ$-function.*

Proof. Let X be a nonempty $\succ$-subset of $\mathbb{R}$ that is bounded above. $\cup X$ is a $\succ$-set (since $\cup$ is a rudimentary function), and since X is bounded above, standard arguments show that $\cup X$ is a Dedekind cut and thus belongs to $\mathbb{R}$. Since the order $\succ$-relation $\leq$ coincides with the inclusion relation, it follows that $\cup X$ is a least upper bound for X. Moreover, the function that maps each X to $\cup X$ is a rudimentary function (from $P_{J_2}(\mathbb{R})$ to $P_{J_2}(\mathbb{Q})$), and hence it is a $\succ$-function. Denote it by F. The desired function *l.u.b* is $F \restriction_{F^{-1}[\mathbb{R}]}$ (recall that $\vdash_{RST^{cFOL}_{HF}} \mathbb{R} \subseteq P_{J_2}(\mathbb{Q})$), which by Proposition 5.4.18 is a $\succ$-function. □

It should be emphasized that Theorem 6.1.9 only states that $\succ$-*subsets* of $\mathbb{R}$ have the least upper bound property. Thus, it is insufficient for the development of most of standard mathematics in RST^{cFOL}_{HF}. The reason is that in RST^{cFOL}_{HF} even the most basic substructures of $\mathbb{R}$, like the intervals, are not $\succ$-sets, but proper $\succ$-classes. Hence we need a stronger version of the theorem which ensures that the least upper bound property holds for standard $\succ$-subclasses of $\mathbb{R}$. Such an extension is given in Theorem 6.1.21 in the sequel, but to state it we need some additional definitions and propositions.

First we consider $\succ$-classes $U \subseteq \mathbb{R}$ which are open. These $\succ$-classes are generally not $\succ$-sets (unless they are empty), since they contain an interval of positive length,

which is a proper $\succ$-class and thus cannot be contained in a $\succ$-set (see Prop. 5.4.5(3)). Clearly, there is no such thing as a $\succ$-set of $\succ$-classes, since a proper $\succ$-class can never be an element of another $\succ$-set or $\succ$-class. However, the use of coding (following [104] and [114][4]) allows us, for example, to replace the meaningless statement "the union of a $\succ$-set of $\succ$-classes is a $\succ$-class" with the statement "given a $\succ$-set of codes for $\succ$-classes, the union of the corresponding $\succ$-classes is a $\succ$-class".

The coding technique we shall use is based on the idea behind the standard notation used in mathematics for a "family of sets", $(A_i)_{i\in I}$, where I is a set and A_i is a set for each $i \in I$. In RST^{cFOL}_{HF} if A_i is a $\succ$-class for $i \in I$ and I is a $\succ$-set we cannot construct the collection of all such A_i's. Thus, we shall treat the $\succ$-set I as a code for the "family of classes" $(A_i)_{i\in I}$. In fact, we mainly use the union of such a family, i.e., $\bigcup_{i\in I} A_i$.

Definition 6.1.10. For every $p \in \mathbb{R}$ and $q \in \mathbb{R}^+$, the open ball $B_q(p)$ is the $\succ$-class $\{r \in \mathbb{R} \mid |r - p| < q\}$.[5]

Next we define an open $\succ$-class as a union of a $\succ$-set of balls with rational centers and rational radii.

Definition 6.1.11. Let $U \subseteq \mathbb{R}$ be a $\succ$-class.

- A $\succ$-set $u \subseteq \mathbb{Q} \times \mathbb{Q}^+$ is called a *code* for U if

$$U = \bigcup_{\langle p,q\rangle \in u} B_q(p) = \{r \in \mathbb{R} \mid \exists p, q\,(\langle p, q\rangle \in u \wedge |r - p| < q)\}$$

- $U \subseteq \mathbb{R}$ is called *open* if it has a code.

In what follows, the formalizations in RST^{cFOL}_{HF} are carried out in the following way. In order to quantify over open $\succ$-classes we use their codes and write $Qu \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^+}$ ($Q \in \{\forall, \exists\}$). If we wish to refer to the open $\succ$-class U whose code is u, we use the following:

$$decode(u) := \{\!|r \in \bar{\mathbb{R}} \,\hat|\, \exists p, q\,(\langle p, q\rangle \check{\in} u \wedge |r - p| < q)|\!\}$$

[4] In [114] such codings are called "proxies".

[5] Note that an open ball in $\mathbb{R}$ is an open, finite interval (see Def. 6.1.23).

To state that a class variable $\boldsymbol{U}$ denotes an open $\succ$-class we use:

$$Open\,(\boldsymbol{U}) := \exists u \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^+}.\boldsymbol{U} = decode\,(u)$$

Proposition 6.1.12. *The followings are provable in* RST^{cFOL}_{HF}*:*

1. *If* $u \subseteq \mathbb{R} \times \mathbb{R}^+$ *is a* $\succ$*-set, then* $U = \{r \in \mathbb{R} \mid \exists p, q\,(\langle p, q\rangle \in u \wedge |r - p| < q)\}$ *is an open* $\succ$*-class.*

2. *The open ball* $B_q\,(p) = \{r \in \mathbb{R} \mid |r - p| < q\}$ *is an open* $\succ$*-class for any* $p \in \mathbb{R}$ *and* $q \in \mathbb{R}^+$*.*

Proof.

1. Define $w = \left\| \{\!|\ \langle p, q\rangle \in \widetilde{\mathbb{Q} \times \mathbb{Q}^+} \mid \exists r, s\,(\langle r, s\rangle \check{\in} u \wedge q + |r - p| \leq s)\ |\!\} \right\|$. Since $\mathbb{Q} \times \mathbb{Q}^+$ is a $\succ$-set and $\exists r, s\,(\langle r, s\rangle \check{\in} u \wedge q + |r - p| \leq s) \succ \emptyset$, w is a $\succ$-set that is a code for an open $\succ$-class. It can easily be proven in RST^{cFOL}_{HF} that w codes exactly the $\succ$-class U.

2. Take $u = \{\langle p, q\rangle\}$ (which is clearly a $\succ$-set) as the code of $B_q\,(p)$ (by 1.). □

Proposition 6.1.13. *The followings are provable in* RST^{cFOL}_{HF}*:*

1. *The union of any* $\succ$*-set of open* $\succ$*-classes is an open* $\succ$*-class. i.e, given a* $\succ$*-set of codes of open* $\succ$*-classes, the union of the corresponding open* $\succ$*-classes is an open* $\succ$*-class.*

2. *An intersection of two open* $\succ$*-classes is an open* $\succ$*-class.*

Proof.

1. Let X be a $\succ$-set of codes of open $\succ$-classes. Thus, $\cup X$ is a code for the union of the corresponding open $\succ$-classes.

2. If U and V are open $\succ$-classes, a code for their intersection is obtained by intersecting every ball in a code for U with every ball in a code for V. □

Example 6.1.14. As an example of the use of the coding technique, we demonstrate the formalization of Prop. 6.1.13(1) in $\mathcal{L}^{HF}_{RST}$:

$$\forall z \left(\left(\forall x \in z.x \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^+} \right) \rightarrow \right.$$
$$\left. \exists w \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^+}.decode\,(w) = \{ r \mid \exists x \in z.r \in decode\,(x) \} \right)$$

Definition 6.1.15. A $\succ$-class $X \subseteq \mathbb{R}$ is called *closed* if $\mathbb{R} - X$ is open.

Lemma 6.1.16. *It is provable in RST^{cFOL}_{HF} that if $U \subseteq \mathbb{R}$ is an open $\succ$-class, then for every $x \in U$ there is an open ball about x which is contained in U.*

Proof. If $x \in U$, then there is some $\langle p, q \rangle$ in the code of U such that $x \in B_q\,(p)$. Take $\varepsilon = |p - x|$. It is straightforward to see that $B_\varepsilon\,(x) \subseteq B_q\,(p) \subseteq U$. □

The proof of the next lemma is trivial.

Lemma 6.1.17. *Let $X \subseteq \mathbb{R}$ be a $\succ$-class and $A \subseteq X$ be a $\succ$-set. The followings are equivalent in RST^{cFOL}_{HF}:*

1. *Every open ball about a point in X intersects A.*

2. *Every open $\succ$-class that intersects X also intersects A.*

Example 6.1.18. To provide an example of a full formalization which uses class variables, the formalization of the lemma above is:

$$\varphi := \boldsymbol{X} \subseteq \bar{\mathbb{R}} \;\rightarrow\; \forall a \subseteq \boldsymbol{X} \left(\forall x \in \boldsymbol{X} \forall \varepsilon \in \bar{\mathbb{R}^+}\,(B_\varepsilon\,(x) \cap a \neq \emptyset) \leftrightarrow \right.$$
$$\left. \forall u \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^+}\,(decode\,(u) \cap \boldsymbol{X} \neq \emptyset \rightarrow decode\,(u) \cap a \neq \emptyset) \right)$$

We next demonstrate how one can get a formula in the basic $\mathcal{L}^{HF}_{RST}$ (i.e., with no class terms or variables) by replacing each appearance of a class term or variable with the formula it stands for. First, we explain in detail the translation of $x \in \bar{\mathbb{R}}$ to $\mathcal{L}^{HF}_{RST}$. One iteration of the translation gives us $x \in \overline{P_{J_2}\,(\mathbb{Q}) \setminus \{\emptyset, \mathbb{Q}\}} \wedge \varphi\,(x) \wedge \psi\,(x)$, where φ, ψ are as in Def. 6.1.6. A second iteration results in $x \subseteq \tilde{\mathbb{Q}} \wedge x \neq \tilde{\mathbb{Q}} \wedge x \neq \emptyset \wedge \varphi\,(x) \wedge \psi\,(x)$ which is in $\mathcal{L}^{HF}_{RST}$. Denote this formula by $R\,(x)$. Now, for the full translation of φ,

we first substitute $\{\!\{x \mid \psi\}\!\}$ for $\boldsymbol{X}$, where $\psi \succ \emptyset$. Then we proceed with the translation to finally obtain the following formula (scheme) of $\mathcal{L}^{HF}_{RST}$, for $\psi \succ \emptyset$:

$$\forall b\,(\psi\,(b) \to R\,(b)) \to \forall a\,((\forall z.z \in a \to \psi\,(z)) \to$$
$$[\forall x\,(\psi\,(x) \to \forall \varepsilon\,((R\,(\varepsilon) \wedge 0 < \varepsilon) \to \exists w.\,|w - x| < \varepsilon \wedge w \in a)) \leftrightarrow$$
$$\forall u \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^{+}}\,((\exists w.R\,(w) \wedge \exists p, q\,(\langle p, q\rangle \check{\in} u \wedge |w - p| < q) \wedge \psi\,(w)) \to$$
$$(\exists w.R\,(w) \wedge \exists p, q\,(\langle p, q\rangle \check{\in} u \wedge |w - p| < q) \wedge w \in a)])$$

Note 6.1.19. When we say that a theorem about a $\succ$-class or a $\succ$-function is provable in RST^{cFOL}_{HF} (like in the previous lemma), we mean that it can be formalized and proved as a scheme. This is to say that the proof can be carried out in RST^{cFOL}_{HF} using a uniform scheme. The one exception is theorems concerning open $\succ$-classes, which due to the coding machinery can be fully formalized and proved in RST^{cFOL}_{HF}.

Definition 6.1.20. Let $X \subseteq \mathbb{R}$ be a $\succ$-class, and $A \subseteq X$ a $\succ$-set. A is called *dense in* X if one of the two equivalent conditions of Lemma 6.1.17 holds. X is called *separable* if it contains a dense $\succ$-subset.

Now we can turn to the proof that the least upper bound property holds not only for nonempty $\succ$-subsets of $\mathbb{R}$, but to any nonempty separable $\succ$-subclass of $\mathbb{R}$.

Theorem 6.1.21. *It is provable in* RST^{cFOL}_{HF} *that every nonempty separable* $\succ$*-subclass of* $\mathbb{R}$ *that is bounded above has a least upper bound in* $\mathbb{R}$.

Proof. Let X be a nonempty separable $\succ$-subclass of $\mathbb{R}$ that is bounded above. Thus, there is a $\succ$-subset $A \subseteq X$ that is dense in X. Now, by Theorem 6.1.9, A has a least upper bound, denote it by m. Suppose m is not an upper bound of X, i.e., there exists $x \in X - A$ such that $x > m$. Take $\varepsilon_m = |x - m|$. Then, there is no $a \in A$ such that $a \in (x - \varepsilon_m, x + \varepsilon_m)$, which contradicts the fact that A is dense in X. The fact that m is the least upper bound of X is immediate. □

Example 6.1.22. To demonstrate the formalization in $\mathcal{L}^{HF}_{RST}$ of the last theorem, denote by $\mathit{separ}\,(\boldsymbol{U})$ the formula $\exists d.d \subseteq \boldsymbol{U} \wedge \forall x \in \boldsymbol{U} \forall \varepsilon \in \overline{\mathbb{R}^{+}} B_{\varepsilon}\,(x) \cap \boldsymbol{U} \neq \emptyset$, and by $\mathit{bound}_{\boldsymbol{U}}\,(w)$ the formula $\forall x \in \boldsymbol{U}.x \leq w$. Now, the full formalization is:

$$\big(\boldsymbol{U} \subseteq \mathbb{R} \wedge \boldsymbol{U} \neq \emptyset \wedge \mathit{separ}\,(\boldsymbol{U}) \wedge \exists w \in \bar{\mathbb{R}}.\mathit{bound}_{\boldsymbol{U}}\,(w)\big) \to$$
$$\exists v \in \bar{\mathbb{R}}\,\big(\mathit{bound}_{\boldsymbol{U}}\,(v) \wedge \forall w \in \bar{\mathbb{R}}\,(\mathit{bound}_{\boldsymbol{U}}\,(w) \to v \leq w)\big)$$

Definition 6.1.23. A $\succ$-class $X \subseteq \mathbb{R}$ is called an *interval* if for any $a, b \in X$ such that $a < b$: if $c \in \mathbb{R} \wedge a < c < b$, then $c \in X$.

Proposition 6.1.24. *It is provable in RST^{cFOL}_{HF} that any non-degenerate interval is separable.*

Proof. Let X be a non-degenerate interval. Take A to be $X \cap \mathbb{Q}$. By Prop. 5.4.5(2), A is a $\succ$-set. A standard argument shows that in every open ball about a point in X there is a rational number, and thus it intersects A. □

Corollary 6.1.25. *It is provable in RST^{cFOL}_{HF} that any non-degenerate interval that is bounded above has a least upper bound.*

Proposition 6.1.26. *It is provable in RST^{cFOL}_{HF} that an open $\succ$-subclass of a separable $\succ$-class is separable.*

Proof. Let X be an open $\succ$-subclass of the separable $\succ$-class S, and let D be the dense $\succ$-subset in S. Let B be any open ball with center $x \in X$. By Lemma 6.1.16, there is a ball B' about x such that $B' \subseteq B \cap X$. Since D is dense in S, $B' \cap D \neq \emptyset$. Hence, $B \cap D \cap X \neq \emptyset$, and so $D \cap X$ is dense in X. □

Definition 6.1.27. A $\succ$-class $X \subseteq \mathbb{R}$ is called *connected* if there are no open $\succ$-classes U and V such that $X \subseteq U \cup V$, $U \cap V \neq \emptyset$, $X \cap U \neq \emptyset$, and $X \cap V \neq \emptyset$.

Example 6.1.28. The formalization of the above definition can be given by:

$$\begin{aligned} &\mathit{connected}\,(\boldsymbol{X}) := \neg\exists u, v \subseteq \widetilde{\mathbb{Q} \times \mathbb{Q}^{+}}\,(\boldsymbol{X} \subseteq \mathit{decode}\,(u) \cup \mathit{decode}\,(v) \wedge \\ &\mathit{decode}\,(u) \cap \mathit{decode}\,(v) \neq \emptyset \wedge \boldsymbol{X} \cap \mathit{decode}\,(u) \neq \emptyset \wedge \boldsymbol{X} \cap \mathit{decode}\,(v) \neq \emptyset) \end{aligned}$$

Proposition 6.1.29. *Let $X \subseteq \mathbb{R}$ be a $\succ$-class. It is provable in RST^{cFOL}_{HF} that X is connected if and only if it is an interval.*

Proof. Assume that there are open $\succ$-classes U and V such that $X \subseteq U \cup V$, $U \cap V \neq \emptyset$, $X \cap U \neq \emptyset$, and $X \cap V \neq \emptyset$ (recall that the formalization of the existence of open $\succ$-classes is done using their codes). Choose $u \in U$ and $v \in V$ and assume that $u < v$. Let $U_0 = U \cap \{z \in \mathbb{R} \mid z < v\}$ and $V_0 = V \cap \{z \in \mathbb{R} \mid z > u\}$. Prop. 6.1.13 and Prop. 6.1.26 entail that U_0 and V_0 are open, separable $\succ$-subclass of $\mathbb{R}$. Standard arguments show that they are non-empty and bounded above. Thus, by Theorem

6.1.21, U_0 and V_0 have least upper bounds. Following the standard proof found in ordinary textbooks we can deduce that the least upper bounds are elements in $[u, v]$, but not elements of U_0 or of V_0, which is a contradiction, since $[u, v] \subseteq U_0 \cup V_0$. The classical proof of the converse direction can easily be carried out in RST^{cFOL}_{HF}. □

6.1.2.3 Real Functions

Definition 6.1.30. Let X be a $\succ$-class. A $\succ$-*sequence* in X is a $\succ$-function on $\mathbb{N}$ whose image is contained in X.

Note 6.1.31. Note that any $\succ$-sequence is a small $\succ$-function

Lemma 6.1.32. *It is provable in RST^{cFOL}_{HF} that every Cauchy $\succ$-sequence in $\mathbb{R}$ converges to a limit in $\mathbb{R}$. The map (lim) that takes Cauchy $\succ$-sequences in $\mathbb{R}$ to their limits is available in RST^{cFOL}_{HF} as a $\succ$-function.*

Proof. Let a be a Cauchy $\succ$-sequence, and let a_k abbreviate $a\,(k)$. For each $n \in \mathbb{N}$ define $v_n := \bigcap_{k \geq n} a_k$. The $l.u.b$ of $\lambda n.v_n$ is equal to the limit of $\lambda n.a_n$ (see [59]). Thus, $lim\,\lambda n.a_n := \bigcup\{\!|\, v_n \mid n \in \widetilde{\mathbb{N}} \,|\!\}$. □

Proposition 6.1.33. *It is provable in RST^{cFOL}_{HF} that if $X \subseteq \mathbb{R}$ is closed, then every Cauchy $\succ$-sequence in X converges to a limit in X.*

Proof. Let a be a Cauchy $\succ$-sequence in X, and let a_k abbreviate $a\,(k)$. By Lemma 6.1.32, $lim\,\lambda n.a_n$ is an element in $\mathbb{R}$, denote it by l. Assume by contradiction that $l \in \mathbb{R} - X$. Since X is closed, $\mathbb{R} - X$ is open, and thus there exists $\varepsilon > 0$ such that $B_\varepsilon\,(l) \subseteq \mathbb{R} - X$. This means that in $B_\varepsilon\,(l)$ there is no element from X. In particular, for every a_k, $a_k \notin B_\varepsilon\,(l)$ (i.e., $|a_k - l| \geq \varepsilon$), which contradicts the fact that $lim\,\lambda n.a_n = l$. □

Next we want to study sequences of functions on X whose images are contained in Y, where X, Y are given $\succ$-classes. However, we cannot apply Definition 6.1.30 as is, because a $\succ$-function which is not small cannot be a value of a $\succ$-function (and in particular not a value of a $\succ$-sequence). Instead, we use the standard procedure of Uncurrying.

Definition 6.1.34. Let X and Y be $\succ$-classes. A *$\succ$-sequence of $\succ$-functions* on X whose image is contained in Y is a $\succ$-function on $\mathbb{N} \times X$ whose image is contained in Y. (Intuitively, this $\succ$-function, call it F, denotes the sequence $f(0), f(1), f(2), ...$ where $f(n) = \lambda x \in X.F(n,x)$).

Proposition 6.1.35. *Let $X \subseteq \mathbb{R}$ be a $\succ$-class. Any point-wise limit of a $\succ$-sequence of $\succ$-functions on X whose image is contained in $\mathbb{R}$ is available in RST^{cFOL}_{HF} as a $\succ$-function.*

Proof. Let F be a $\succ$-sequence of $\succ$-functions on X whose image is contained in $\mathbb{R}$. Suppose that for each $a \in X$ the $\succ$-sequence $\lambda n.F(n,a)$ is converging, and so it is Cauchy. Define: $G_a := \{\!\{ \langle n, \bar{F}(n,a) \rangle \mid n \in \widetilde{\mathbb{N}} \}\!\}$. Then, $\left\| \lambda a \in \bar{X}.limG_a \right\|$ is the desired $\succ$-function. □

Next we turn to continuous real $\succ$-functions. One possibility of doing so, adopted e.g., in [116] and [104], is to introduce codes for continuous real $\succ$-functions (similar to the use of codes for open $\succ$-classes). This is of course possible as such $\succ$-functions are determined by their values on the $\succ$-set $\mathbb{Q}$. However, we prefer to present here another approach, which allows for almost direct translations of proofs in standard analysis textbook into our system. This is done by using free function variables. Accordingly, the theorems which follows are schemes (see Note 6.1.19).

Note 6.1.36. Implicitly, Chapter 5 can also be read and understood as done in the manner described above. Therefore, in what follows we freely use results from it.

Definition 6.1.37. Let $X \subseteq \mathbb{R}$ be a $\succ$-class and let F be a $\succ$-function on X whose image is contained in $\mathbb{R}$. F is called a *continuous real $\succ$-function* if:

$$\forall a \in X \forall \varepsilon \in \mathbb{R}^+ \exists \delta \in \mathbb{R}^+ \forall x \in X \left(|x - a| < \delta \rightarrow |F(x) - F(a)| < \varepsilon \right)$$

Proposition 6.1.38. *Let $X \subseteq \mathbb{R}$ be a $\succ$-class and let F be a $\succ$-function on X whose image is contained in $\mathbb{R}$. It is provable in RST^{cFOL}_{HF} that if for every open $\succ$-class $B \subseteq \mathbb{R}$, there is an open $\succ$-class A such that $F^{-1}[B] = A \cap X$, then F is continuous.*

Proof. Let $a \in X$ and $\varepsilon > 0$. Denote by V the open ball $B_\varepsilon(F(a))$. Since V is an open $\succ$-class, it follows from the assumption that there is an open $\succ$-class A such that $F^{-1}[V] = A \cap X$ (notice that $F^{-1}[V]$ is a $\succ$-class by Prop. 5.4.18(2)). Also,

as $F(a) \in V$ we have that $a \in F^{-1}[V]$ and thus $a \in A$. Since A is open, there exists δ_a such that $B_{\delta_a}(a) \subseteq A$. Take $\delta = \delta_a$. For any $x \in X$, if $|x - a| < \delta_a$, then $x \in B_{\delta_a}(a) \subseteq A$. Hence $x \in A \cap X = F^{-1}[V]$, and therefore $F(x) \in V = B_\varepsilon(F(a))$, i.e., $|F(x) - F(a)| < \varepsilon$. □

Lemma 6.1.39. *The followings are provable in* RST_{HF}^{cFOL}*:*

1. *The composition, sum, and product of two continuous real* $\succ$*-functions is a continuous real* $\succ$*-function.*

2. *The limit of a* $\succ$*-sequence of continuous real* $\succ$*-functions is a continuous real* $\succ$*-function.*

Proof. The standard proofs of these claims can be easily carried out in RST_{HF}^{cFOL}. Note that they require the triangle inequality which is provable in RST_{HF}^{cFOL}. □

Next we prove, as examples, the Intermediate Value Theorem and the Extreme Value Theorem, which are two key properties of continuous real functions.

Theorem 6.1.40 (Intermediate Value Theorem)**.** *Let* F *be a continuous real* $\succ$*-function on an interval* $[a, b]$*, and suppose* $F(a) < F(b)$*. It is provable in* RST_{HF}^{cFOL} *that for any* $d \in \mathbb{R}$ *such that* $F(a) < d < F(b)$*, there is* $c \in [a, b]$ *such that* $F(c) = d$*.*

Proof. Let $d \in \mathbb{R}$ and assume that $F(a) < d < F(b)$. Define Q_d to be $\left\| \{\!\!\{ x \in \tilde{\mathbb{Q}} \mid x \in \overline{[a,b]} \wedge F(x) \leq d \}\!\!\} \right\|$. Q_d is clearly bounded, for instance by b. Since $F(a) < d$, standard arguments that use the continuity of F and the denseness of $\mathbb{Q}$ in $\mathbb{R}$ show that there is a rational $a \leq q$ such that $F(q) \leq d$. Thus, Q_d is non-empty. Therefore, by Thm. 6.1.9, it has a least upper bound, denote it by c. Since Q_d is non-empty and b is an upper bound for it, $c \in [a, b]$. We prove that $F(c) = d$. Assume by contradiction that $F(c) < d$. Pick $\varepsilon = d - F(c)$. By the continuity of F we know that there exists $\delta > 0$ such that for any $x \in [a, b]$, if $|x - c| < \delta$, then $|F(x) - F(c)| < \varepsilon = d - F(c)$. This yields the existence of a rational $q \in (c, c + \delta)$ (again, by the denseness of $\mathbb{Q}$ in $\mathbb{R}$) such that $F(q) < d$, which is a contradiction. Now, assume by contradiction that $F(c) > d$, and pick $\varepsilon = F(c) - d$. In this case we get that there exists $\delta > 0$ such that for any $x \in [a, b]$, if $|x - c| < \delta$, then $F(x) > d$. But then $c - \delta$ is also an upper bound for Q_d, which is again a contradiction. Hence, $F(c) = d$. □

Theorem 6.1.41 (Extreme Value Theorem)**.** *Let F be a continuous real $\succ$-function on a non-degenerate interval $[a,b]$. It is provable in RST^{cFOL}_{HF} that F attains its maximum and minimum.*

Proof. Let Q be the $\succ$-set $[a,b] \cap \mathbb{Q}$. $F[Q]$ is a $\succ$-set by Prop. 5.4.15, and it is non-empty by the denseness of $\mathbb{Q}$ in $\mathbb{R}$. We first show that $F[Q]$ is bounded. Assume by contradiction that it is not bounded, and define for every $n \in \mathbb{N}$ $C_n = \left\| \{ x \in \tilde{Q} \mid F(x) > n \} \right\|$. By the assumption C_n is a non-empty, bounded $\succ$-set. Therefore, by Thm. 6.1.9, each C_n has a least upper bound, denote it by c_n. It is easy to see that $c_n \in [a,b]$ for each $n \in \mathbb{N}$. Now, define the $\succ$-sequence $\lambda n \in \mathbb{N}.c_n$ (this is indeed a $\succ$-sequence since Thm. 6.1.9 also entails that the map $l.u.b$ is available in RST^{cFOL}_{HF} as a $\succ$-function). Standard arguments show that since $[a,b]$ is closed and bounded, there is a subsequence of $\lambda n \in \mathbb{N}.c_n$, $\lambda k \in \mathbb{N}.c_{n_k}$, which converges to a limit, denote it by m. By Prop. 6.1.33 we have that $m \in [a,b]$. Now, since F is continuous, by standard arguments we get that $\lambda k \in \mathbb{N}.F(c_{n_k})$ converges to $F(m)$. But, for each $k \in \mathbb{N}$: $F(c_{n_k}) > n_k \geq k$, which contradicts the convergence of the sequence. Hence, $F[Q]$ is bounded, and again by Thm. 6.1.9 we get that it has a least upper bound, denote it by d. Assume by contradiction that there exists $u \in [a,b]$ such that $F(u) > d$. Picking $\varepsilon = F(u) - d$, the continuity of F entails that there exists δ such that for every $x \in B_\delta(u)$, $F(x) \geq d$. But the denseness of $\mathbb{Q}$ entails that there is a rational number $q \in B_\delta(u)$, and thus $F(q) \geq d$, which is a contradiction. It remains to show that there exists $x \in [a,b]$ such that $F(x) = d$. This can be proven using arguments similar to the ones used in the proof of Thm. 6.1.40 for the $\succ$-set Q_d. The proof that F attains its minimum is similar. □

The next step is to introduce in RST^{cFOL}_{HF} the concepts of differentiation, integration, power series, etc, and develop their theories. It should now be clear that there is no difficulty in doing so. Since a thorough exposition obviously could not fit in one chapter, we omit the details here. However, we shall use some relevant facts regarding these concepts in what follows.

We end this section by showing that all elementary functions that are relevant to J_2 are available in RST^{cFOL}_{HF} in the sense that they are formalizable as $\succ$-functions and their basic properties are provable in RST^{cFOL}_{HF}. Of course, not all constant functions on the "real" real line are available in J_2, even though for every y in $\mathbb{R}$, $\lambda x \in \mathbb{R}.y$ is

available in RST^{cFOL}_{HF} as a $\succ$-function. The reason is that $\lambda x \in \mathbb{R}.y$ does not exists in J_2 for *every* "real" number y (for the simple fact that not every "real" real number is available in RST^{cFOL}_{HF}). Thus we next define what is an "J_2-elementary function" (see, for example, [95] for a standard definition of "elementary function").

Definition 6.1.42. The collection of J_2-elementary functions is the minimal collection that is closed under addition, substraction, multiplication, division, and composition, and includes the following:

- J_2-constant functions: $\lambda x \in \mathbb{R}.c$ where c is a real number in J_2.
- Exponential: $\lambda x \in \mathbb{R}.e^x$.
- Natural logarithm: $\lambda x \in \mathbb{R}^+.\ln x$.
- Trigonometric functions: $\lambda x \in \mathbb{R}.\sin x$.
- Inverse trigonometric functions: $\lambda x \in [-1,1]\,.\arcsin x$.

Proposition 6.1.43. *All J_2-polynomials (i.e., with coefficients in J_2) on $\mathbb{R}$ are available in RST^{cFOL}_{HF} as $\succ$-functions, and it is provable in RST^{cFOL}_{HF} that they are continuous.*

Proof. J_2-constant functions and the identity function are available in RST^{cFOL}_{HF} by Prop. 5.4.18, and the proofs of their continuity is immediate. Composition of $\succ$-functions is also available in RST^{cFOL}_{HF}. All J_2-polynomials on $\mathbb{R}$ are therefore available in RST^{cFOL}_{HF}, since $+$ and $\cdot$ are $\succ$-functions, and they are continuous by Lemma 6.1.39. □

Proposition 6.1.44. *The exponential and trigonometric functions are available in RST^{cFOL}_{HF}, and it is provable in RST^{cFOL}_{HF} that they are continuous.*

Proof. Since the exponential and the trigonometric functions all have power series, their definability as $\succ$-functions follows from Prop. 6.1.35. It is straightforward to verify that the basic properties of these $\succ$-functions are provable in RST^{cFOL}_{HF}. Examples of such properties are: the monotonicity of the exponential, the power rules of the exponential, trigonometric identities like $\sin(\alpha+\beta) = \sin\alpha\cos\beta + \sin\beta\cos\alpha$,

the fact that sin has a period of 2π (where π is its first positive root), etc.[6] The continuity of these functions follows from Lemma 6.1.39 and Prop. 6.1.35. □

Lemma 6.1.45. *Let F be a continuous, monotone real $\succ$-function on a real interval $[a,b]$, and suppose $F(a) < F(b)$. It is provable in RST_{HF}^{cFOL} that*

$$\forall y \in \bar{\mathbb{R}} \left(\exists x \in \overline{[a,b]}.\bar{F}(x) = y \leftrightarrow y \in \overline{[F(a), F(b)]} \right)$$

Proof. The left-to-right implication is immediate from the monotonicity of F. The right-to-left implication follows from Thm. 6.1.40. □

Proposition 6.1.46. *Let F be a continuous, strictly monotone real $\succ$-function on a real interval. Then it is provable in RST_{HF}^{cFOL} that the inverse function F^{-1} is available in RST_{HF}^{cFOL} as a $\succ$-function, and its continuity is provable in RST_{HF}^{cFOL}.*

Proof. We here prove the claim for continuous, strictly monotone real $\succ$-function on a finite closed interval $[a,b]$. The extension from finite closed intervals to arbitrary interval is standard. Suppose F is increasing. The proof is similar to the proof of Thm. 6.1.40. For any $y \in [F(a), F(b)]$ define the $\succ$-set $Q_y := \left\| \{ q \in \widetilde{\mathbb{Q}} \mid q \in \overline{[a,b]} \wedge F(q) \leq y \} \right\|$. It is easy to see that Q_y is non-empty and bounded, thus, by Thm. 6.1.9, Q_y has a least upper bound. Now, $\left\| \lambda y \in \overline{[F(a), F(b)]}.l.u.b\widetilde{Q_y} \right\|$ is the desired inverse $\succ$-function. It is not difficult to prove the basic properties of the inverse function in RST_{HF}^{cFOL}. We demonstrate the proof that $F^{-1} \circ F = id_{[a,b]}$. For this we need to show that for any $x \in [a,b]$, $l.u.bQ_{F(x)} = x$. By the monotonicity of F, x is clearly an upper bound for $Q_{F(x)}$. Assume by contradiction that there is a real number $w < x$ which is an upper bound of $Q_{F(x)}$. Thus, in the interval (w,x) there is a rational number q such that $F(q) \leq F(x)$ (by monotonicity). But then, $q \in Q_{F(x)}$ and $w < q$, which is a contradiction. □

Proposition 6.1.47. *All J_2-elementary functions are available in RST_{HF}^{cFOL}.*

Proof. Props. 6.1.43 and 6.1.44 show that J_2-polynomials on $\mathbb{R}$, the exponential, and the trigonometric functions are available in RST_{HF}^{cFOL}. Prop. 6.1.46 then enables the availability in RST_{HF}^{cFOL} of the inverse trigonometric functions, and of the natural logarithm as the inverse of the exponential. □

[6] We can prove the standard properties of the exponent and the trigonometric functions as listed, e.g., in [3], using the notion of differentiation.

It is not difficult to see that many standard discontinuous functions are also available in RST^{cFOL}_{HF}, as the next proposition shows.

Proposition 6.1.48. *Any piece-wise defined function with finitely many pieces such that its restriction to any of the pieces is a J_2-elementary function, is available in RST^{cFOL}_{HF}.*

Proof. If the function has finitely many pieces and each of the pieces is a J_2-elementary function, then it can be constructed in RST^{cFOL}_{HF} using Prop. 5.4.18(5). □

6.1.3 Going Beyond the Minimal Framework

In this section we explore possible extensions of the minimal first-order framework RST^{cFOL}_{HF} using constants. In particular, we add to the minimal framework constants and appropriate closure axioms concerning them, in a way that incorporates the real line as a $\succ$-set.

Terminology. In what follows we take C to be a set of constants that includes both HF and U.

Definition 6.1.49. The system RST^{cFOL}_{C} is extended by adding to its definition (Def. 5.2.6) the following clause:

- In case $U \in C$ the following axioms are added:
  - $HF \in U$.
  - $\forall x \forall y\,(x \in U \wedge y \in x \rightarrow y \in U)$.
  - $\forall y_1, ..., y_n \in U . \{\!|\, x \mid \varphi \,|\!\} \in U$, provided $\varphi \succ \{x\}$, $Fv\,(\varphi) = \{y_1, ..., y_n\}$, and U does not occur in φ.[7]

Notes:

- The new constant symbol U is to be interpreted as an element of the universe W that includes $\mathcal{HF}$ and is a universe (and so it is closed under rudimentary operations). This imposes some constraints on W, which now must

[7] To be more precise, for a concrete C we should specify which other constants may occur in φ or not. HF is always allowed. As for the other constants in C, this will depend on their intended interpretation. In this section we may assume that $C = \{HF, U\}$.

contain as an element both $\mathcal{HF}$ and some universe. The minimal such W is J_3, in which we can take the interpretation of U to be J_2. From a definitional/computational/constructive point of view, a better choice for W might be J_{ω^ω} (which is the minimal model of RST^{cAL}, see Section 6.2). In J_{ω^ω} one can use J_ω as a sufficiently extensive interpretation of U.

- While for HF we have a unique interpretation, the interpretation of U is deliberately left open to allow stronger extensions of the system. The development of applicable mathematics which is outlined below is independent of the interpretation of U.

The real line is available as a $\succ$-*set* in RST_C^{cFOL} since it can be defined by the set term $\{\!|\, u \in U \mid D(u) \,|\!\}$, where $D(u)$ is the formula stating that u is a Dedekind cut (see Subsection 6.1.2). Next we show that any recursive function is available as a $\succ$-function in RST_C^{cFOL}.

Proposition 6.1.50. *Let F be a $\succ$-function which for ordered pairs in $\|U\|^W$ returns an element of $\|U\|^W$, and let A be an $\succ$-set in U. Then the function H_A^F with domain $\mathbb{N}$ which is defined by*

$$H_A^F(0) = A,$$
$$H_A^F(S(n)) = F\left(A, H_A^F(n)\right),$$

is available as a $\succ$-function in RST_C^{cFOL}.

Proof. By a well-known theorem (see, e.g., Theorem 3.6 in [58]), the proof of which can easily be reproduced in RST_C^{cFOL}, it suffices to show that given a unary function F with the same property, there is a function H_A^F with domain $\mathbb{N}$ such that

$$H_A^F(0) = A,$$
$$H_A^F(S(n)) = F\left(H_A^F(n)\right).$$

Let $Seq_{fin}(f)$ stand for the formula $Func(f) \wedge \exists m \in \mathbb{N}.Dom(f) = S(m)$, and take FIN_A^F to be:

$$\{\!|\, f \in U \mid Seq_{fin}(f) \wedge \left\langle 0, \tilde{A} \right\rangle \in f \wedge \forall n < S(m)\, \forall z\, (\langle n, z\rangle \,\tilde{\in} f \rightarrow \langle S(n), F(z)\rangle \in f) \,|\!\}$$

Next, define H_A^F to be $\bigcup FIN_A^F$. □

It is important to notice that although some of the mathematical structures used in Section 6.1.2 are now $\succ$-sets (as opposed to $\succ$-classes in RST_{HF}^{cFOL}), we still need to work with $\succ$-classes in RST_C^{cFOL}. For example, the collection of all real $\succ$-functions is not available as a $\succ$-set in RST_C^{cFOL}. However, we can treat this collection in RST_C^{cFOL} as a proper $\succ$-class using the fact that the property of being a real $\succ$-function is definable in our language by a formula which is safe with respect to $\emptyset$. Another approach is to continue the method of introducing new constants for bigger universes, from which we are going to take our real functions. This can still be done in our framework if we take W to be J_{ω^ω}, the interpretation of U as J_ω, and handling n-order constructs as elements of J_{ω^n}, after introducing the necessary constant symbols. In practice, scientifically applicable mathematics uses at most 4-order constructs, so we shall not need more than a finite number of constants.

6.2 The Minimal AL Framework

In the case of AL the minimal classical system in RST^{cAL}. In [9] it was shown that its minimal model is J_{ω^ω}. This universe (like J_2 in the case of $\mathcal{L}_{RST}^{HF}$) has the important property that each element in it is definable by some closed term of $\mathcal{L}_{RST+TC}$. This allows for a natural interpretation of cumulative type theory, in which J_ω, J_{ω^2}, J_{ω^3},... are taken as the major types (as they are all definable in RST^{cAL}). These are the types for which, in the case of RST_{HF}^{cFOL}, we had to enrich the system with new constant symbols and corresponding axioms. Using RST^{cAL} we get them "for free".

Another element that was not available in RST_{HF}^{cFOL} whereas in RST^{cAL} we get it "for free" is the treatment of the real line as a $\succ$-set. Thus, the reals can be taken in RST^{cAL} to be Dedekind cuts which are available in J_ω, i.e., $\mathbb{R}$ can be defined as $\left\| \{\!| u \in \widetilde{J_\omega} \mid D(u) |\!\} \right\|^{J_{\omega^\omega}}$, where $D(u)$ is the formula stating that u is a Dedekind cut. Since $D(u) \succ \emptyset$, we get that $\mathbb{R}$ is available as a $\succ$-set in RST^{cAL}. As before, $\mathbb{R}$ may not be the "real" real line, but it contains much more than what was available in RST_{HF}^{cFOL}, which as we have shown in the previous section, seems to be sufficient for the development of mainstream mathematics. Moreover, as opposed to the situation in RST_{HF}^{cFOL}, in RST^{cAL} $\mathbb{R}$ itself is an ordinary object of 'type' J_{ω^2}.

It follows that in RST^{cAL} the development of analysis can be done in a similar way to the way it has been carried out in RST^{cFOL}_{HF}, with the (important) exception that every standard mathematical structure is a $\succ$-set in RST^{cAL}.[8] This implies that, in contrast to the work in RST^{cFOL}_{HF}, in RST^{cAL} one does not have to work with $\succ$-classes and codings. Moreover, since in RST^{cAL} there is no need to use class terms and class variables, all claims are now fully provable in RST^{cAL} as theorems in its language, not merely as schemes, as was the case in RST^{cFOL}_{HF}. For example, the counterpart of Lemma 6.1.17 can be formulated in RST^{cAL} in the following way:

$$\forall x \subseteq \tilde{\mathbb{R}} \forall a \subseteq x \left(\forall y \in x \forall \varepsilon \in \widetilde{\mathbb{R}^+} \left(B_\varepsilon (y) \cap a \neq \emptyset \right) \leftrightarrow \right.$$
$$\left. \forall u \left(open (u) \rightarrow (u \cap x \neq \emptyset \rightarrow u \cap a \neq \emptyset) \right) \right)$$

where $open\,(u)$ stands for $\forall x \in u \exists \varepsilon \in \widetilde{\mathbb{R}^+}.B_\varepsilon (x) \subseteq u$ (note that now the open ball $B_\varepsilon (x)$ is a $\succ$-set). Thus, the formalization of analysis in RST^{cAL} seems somewhat more natural and compatible with ordinary mathematical practice.

As mentioned, in [9] it was shown that in RST^{cAL} it possible to provide inductive definitions of relations and functions which are sets, and in certain cases even to define global relations. However, this is far from capturing the potential of predicative set theory. Thus, for example, although ω^n is definable in RST^{cAL} for each $n \in \mathbb{N}$, and there is an effective procedure to derive a definition of ω^{n+1} from a a definition of ω^n, the set $\{\omega^n | n \in \mathbb{N}\}$ and the function $\lambda n \in \mathbb{N}.\omega^n$ are not definable in $\mathcal{L}_{RST+TC}$, even though their identity is clearly absolute and predicatively acceptable. One possible way to remedy this is by extending the definability power of RST^{cAL}. This can be done by invoking the method of adding new constant symbols together with appropriate closure axioms which was used on the first-order level (both for the introduction of the natural numbers in Section 5.5.1, as well as for the extension of the minimal framework in Subsection 6.1.3). In a similar manner, it is not difficult to show that by adding to $\mathcal{L}_{RST+TC}$ a constant denoting J_{ω^ω} with appropriate closure axioms, we get a system in which it is easy to construct closed terms for $\lambda n \in \mathbb{N}.\omega^n$ and for ω^ω, and prove their main properties.

[8] This is in contrast to the first-order case, in which even in the extended system described in Subsection 6.1.3, some standard mathematical structures were still only available as proper $\succ$-classes.

Note 6.2.1. Obviously the above extension process can be repeated using transfinite recursion, creating a transfinite progression of languages and theories. To do so, we need first of all to precisely define the process of passing from a theory T_α to $T_{\alpha+1}$, and of constructing T_α for limit α. Moreover, like in the systems for predicative analysis of Feferman and Schütte (see [37, 100]), the progression should be autonomous, in the sense that only ordinals justified in previous systems may be used. Now instead of using indirect systems of (numerical) notations for ordinals, it would be much more natural to use terms of our systems which provably denote in them von Neumann's ordinals. We conjecture that every ordinal less than Γ_0, the Feferman-Schütte ordinal for predicativity ([37, 38, 101]), should be obtainable in this way.

Chapter 7

Summary and Further Work

This thesis aimed at a formalization of applicable mathematics on the basis of a user-friendly formal set theoretical framework which is amenable for mechanization and reflects standard mathematical practice. We have identified what we believe are the minimal ontological commitments the framework should employ. In term of the logic, we have shown that ancestral logic offers a suitable base logic for the formalization of mathematics, and created natural, effective proof systems for it. In the classical case we have developed a Hankin-style semantics and proved that the system developed is complete with respect to it. For the constructive version of the system we have developed a realizability semantics and proved that all provable formulas are uniformly realizable. We have also explored some applications of the constructive system for computer science by relating it to the concepts of Kleene algebras and program schemes.We have then identified weak sets of set theoretical axioms which are suitable for the predicative approach to mathematics, and based variants of our formal system on them. A key property of these basic predicative theories is that they are definitional, which allows for a very concrete, computationally-oriented interpretation. We then demonstrated the usefulness of the framework for the formalization of mathematics by developing in it large portions of classical analysis. We showed that while the the minimal first-order variant of the framework is already sufficient for the formalization of most of classical analysis, using the minimal framework which is based on ancestral logic allows for a more natural development of classical analysis.

The field of MKM is still at relatively early stages of its development, with new lines of research, potential applications, promising extensions, and theoretical problems emerging all the time. We believe that both the formal framework for formalizing mathematics used in this work as well as the use of ancestral logic have demonstrated their usefulness and naturalness in this thesis. Evidently, there are various open questions and possible promising extensions. The main directions for further research include the following:

Further Developing AL

Cut elimination theorem for AL_G: The consistency of PA_G was proven by providing a constructive method for transforming any proof of the empty sequent into a cut-free proof (i.e., a proof in which there is no application of the cut rule of $\mathcal{LK}$). A crucial step in the proof is the elimination of all appearances of PA_G's induction rule from the end-piece of the proof.[1] In AL_G the generalization for the induction rule of PA_G renders this method inapplicable. This is because Gentzen's elimination of the induction rule uses special features of the natural numbers that do not exist in AL_G. In the general case, the formula φ in the induction rules of the systems for AL is an arbitrary formula. Thus, unlike in PA_G, we do not have a "built in" measure for the φ-distance between two arbitrary closed terms s and t. The φ-path from s to t is not known apriori, and it does not have to be unique. Therefore, an interesting task is to try to prove some restricted form of cut-elimination theorem for AL_G. One option is to search for a suitable definition of the term "subformula" under which some form of analytical cut elimination can be proven. It is clear that the usual definition of a subformula should be revised.[2] Thus the induction rule of AL satisfies the subformula property only if we take a formula to be a subformula of every substitution instance of it.

[1] The end-piece of a proof consists of all the sequents of the proof encountered if we ascend each path starting from the end-sequent and stop when we arrive to an operational inference rule. Thus the lower sequent of this inference rule belongs to the end-piece, but its upper sequents do not.

[2] Exactly as the straightforward notion of subformula used in propositional languages is changed on the first-order level, where for example a formula of the form $\psi\{\frac{t}{x}\}$ is considered to be a subformula of $\forall x\psi$, even though it might be much longer than the latter.

Useful Fragments of AL: Determine and explore fragments of AL that are more convenient to work with (e.g., they admit full cut-elimination), but are still sufficient for at least some concrete applications. An example of such a fragment may be the one which corresponds to the use of the *deterministic* transitive closure operator (see, e.g., [61]). Another option worth investigating is to restrict the induction rule by allowing only φ's of the form $y = t$, where $Fv(t) = \{x\}$. Implicitly, this is e.g., the fragment of AL used in PA_G.

Extending AL_G: Natural as the systems for AL are, they are of course not complete. Take for example cAL_G. It is not difficult to express its consistency in the language $\{=, 0, S, +\}$ as a logically valid sentence Con_{cAL_G} of cAL. By Gödel theorem on consistency proofs, Con_{cAL_G} is not a theorem of cAL_G. It would be interesting to find what valid principles of cAL (not available in cAL_G) can be used to derive it, and whether those principles are valid for iAL as well.

The model theory of AL: In [102] it is noted that Craig interpolation theorem and Beth definability theorem fail for logics in which the notion of finiteness can be expressed. Thus, an interesting task is to find appropriate AL counterparts (whenever such exist) to central model-theoretic properties of FOL or $iFOL$ such as these.

Uniform completeness for iAL_S: As noted, in [31] a new concept of uniform validity was introduced, and the system $iFOL_S$ was proven to be complete with respect to it. An important research direction is to determine whether the proof system iAL_S presented in Chapter 4 is uniformly complete with respect to realizability semantics (and try to complete it in case it is not).

Applications of iAL_S: In Chapter 4 we established the connection between iAL_S and Kleene Algebras with tests. It is interesting to investigate the connection between iAL_S and new useful variations of KAT, such as: $KAT + B!$ [50] and Kleene Algebra with Equations [68] (which are mostly motivated by concrete applications for verication), and Kleene algebras in software defined networks [42]. iAL_S can also be used to develop efficient deployed algorithms from the natural constructive proofs of theorems about data structures expressible in it. It is also interesting to explore specific direct use of such proofs, e.g., in building distributed protocols and making them attack-tolerant.

Further Developing the Predicative Framework

Formalization of mathematics: In Chapter 6 we demonstrated how large portions of classical analysis can be formalized within even the minimal system of our predicative framework. Of course, a lot of further work of formalizing larger portions of mathematics within our framework is still required in order to fully substantiate the thesis that predicative mathematics is indeed sufficient for the development of applicable mathematics. This includes first of all much more analysis, but also topology and algebra. An important criterion for the adequacy of our system for the task of formalizing mathematics is the extent to which things will be done in a natural way.

Incorporating the Axiom of Choice: There are theorems of standard analysis that require of course some form of the Axiom of Choice. Since in this work we focused on the minimal possible framework for the development of classical analysis, we have avoided any use of this axiom. However, in order to fully recast analysis as presented in standard textbooks, the addition of some form of the axiom of choice to the system is required. This can be achieved by extending the set of terms, using Hilbert's ε function symbol, together with its usual characterizing axiom: $\exists x\varphi \to \varphi\left\{\frac{\varepsilon x\varphi}{x}\right\}$ (which is equivalent to the axiom of global choice).

Going beyond predicativity: For philosophical as well as practical reasons this work focused on a predicative setting for applicable mathematics. However, as noted, the framework employed is not confined to the predicative approach and can easily be extended. Thus, an interesting research task is to practice reverse mathematics (see, e.g., [104]) to determine what set theoretical assumptions are essential for various fragments of mathematics.

The dynamic approach: In this work we aimed at keeping all parts of the framework static. Another interesting possibility is to replace the static approach to terms with a dynamic approach, in which both being a legal term and equality of terms are major judgments. The goal is that a user would be able to introduce any term she/he finds natural and useful. For this we might like $\{x|\varphi\}$ to be a valid term whenever $\{x|\psi\}$ is a valid term, and φ is logically equivalent to ψ (according to the formal logical system which underlies the set theory

used). Note that in such a dynamic framework all parts of a theory (terms, formulas, safety relation, logical principles, and non-logical axioms) are defined by simultaneous recursion.

In addition, a major future research task is to implement and test the formal systems developed in this work, and then use it for concrete applications. A particularly important application we envisage is the development of user-friendly proof assistants.

Bibliography

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of databases*, volume 8. Addison-Wesley Reading, 1995.

[2] P. Aczel. The type theoretic interpretation of constructive set theory. In A. Macintyre, L. Pacholski, and J. Paris, editors, *Logic Colloquium '77*, volume 96 of *Studies in Logic and the Foundations of Mathematics*, pages 55 – 66. Elsevier, 1978.

[3] L. V. Ahlfors. *Complex analysis: An introduction to the theory of analytic functions of one complex variable*. McGraw-Hill, 1966.

[4] A. Avron. Transitive closure and the mechanization of mathematics. In F. D. Kamareddine, editor, *Thirty Five Years of Automating Mathematics*, volume 28 of *Applied Logic Series*, pages 149–171. Springer, Netherlands, 2003.

[5] A. Avron. Formalizing set theory as it is actually used. In *Mathematical Knowledge Management*, pages 32–43. Springer, 2004.

[6] A. Avron. Safety signatures for first-order languages and their applications. *Hendricks, et al.(eds.) First-Order Logic Revisited*, pages 37–58, 2004.

[7] A. Avron. Constructibility and decidability versus domain independence and absoluteness. *Theoretical Computer Science*, 394(3):144–158, 2008.

[8] A. Avron. A framework for formalizing set theories based on the use of static set terms. In *Pillars of Computer Science*, pages 87–106. Springer, 2008.

[9] A. Avron. A new approach to predicative set theory. *Ways of Proof Theory*, pages 31–63, 2010.

[10] A. Avron and L. Cohen. A minimal framework for applicable mathematics. *Submitted*.

[11] A. Avron and L. Cohen. Formalizing scientifically applicable mathematics in a definitional framework. *Journal of Formalized Reasoning*, 2015.

[12] B. Barras. Sets in coq, coq in sets. *Journal of Fromalized Reasoning*, 3(1):29–48, 2010.

[13] M. J. Beeson. *Foundations of constructive mathematics: Metamathematical studies*, volume 6. Springer Science & Business Media, 2012.

[14] Y. Bertot and P. Castéran. *Interactive theorem proving and program development: Coq'Art: the calculus of inductive constructions*. springer, 2004.

[15] E. Bishop. *Foundations of constructive analysis*, volume 60. McGraw-Hill New York, 1967.

[16] A. Bove, P. Dybjer, and U. Norell. A brief overview of Agda–a functional language with dependent types. In S. Berghofer, T. Nipkow, C. Urban, and M. Wenzel, editors, *LNCS 5674, Theorem Proving in Higher Order Logics*, pages 73–78. Springer, 2009.

[17] R. Boyer. The QED manifesto. *Automated Deduction–CADE*, 12:238–251, 1994.

[18] A. Chlipala. An introduction to programming and proving with dependent types in Coq. *Journal of Formalized Reasoning (JFR)*, 3(2):1–93, 2010.

[19] A. Chlipala. A Verified Compiler for an Impure Functional Language. In *POPL*, pages 93–106, 2010.

[20] A. Chlipala. *Certified Programming with Dependent Types*. MIT Press, Cambridge, MA, 2013.

[21] L. Cohen. Ancestral logic and equivalent systems. Master's thesis, Tel-Aviv University, Israel, 2010.

[22] L. Cohen and A. Avron. Ancestral logic: A proof theoretical study. In U. Kohlenbach et al., editor, *Logic, Language, Information, and Computation*,

volume 8652 of *Lecture Notes in Computer Science*, pages 137–151. Springer, 2014.

[23] L. Cohen and A. Avron. The middle ground–ancestral logic. *Synthese*, pages 1–23, 2015.

[24] L. Cohen and R. L. Constable. Intuitionistic ancestral logic. *Journal of Logic and Computation*, 2015.

[25] L. Cohen and R. L. Constable. Intuitionistic ancestral logic as a dependently typed abstract programming language. In *Logic, Language, Information, and Computation*, pages 14–26. Springer, 2015.

[26] R. L. Constable. Constructive mathematics as a programming logic I: Some principles of theory. *North-Holland Mathematics Studies*, 102:21–37, 1985.

[27] R. L. Constable. Evidence semantics and refinement rules for first-order logics: Minimal, intuitionistic, and classical. Technical report, 2012.

[28] R. L. Constable. Virtual evidence: A constructive semantics for classical logics. *CoRR*, abs/1409.0266, 2014.

[29] R. L. Constable, S. F. Allen, M. Bickford, R. Eaton, C. Kreitz, L. Lorigo, and E. Moran. Innovations in computational type theory using nuprl. *Journal of Applied Logic*, 4(4):428–469, 2006.

[30] R. L. Constable, S. F. Allen, M. Bromley, R. Cleaveland, J. F. Cremer, R. W. Harper, D. J. Howe, T. B. Knoblock, N. P. Mendler, P. Panangaden, J. T. Sasaki, and S. F. Smith. *Implementing mathematics with the Nuprl proof development system*. Prentice Hall, 1986.

[31] R. L. Constable and M. Bickford. Intuitionistic completeness of first-order logic. *Annals of Pure and Applied Logic*, 1(165):164–198, 2014.

[32] J. Corcoran, W. Hatcher, and J. Herring. Variable binding term operators. *Mathematical Logic Quarterly*, 18(12):177–182, 1972.

[33] N. G. de Bruijn. *A survey of the project AUTOMATH*. Academic Press, 1980.

[34] K. J. Devlin. *Constructibility*. Springer, 1984.

[35] J. Diller and A. S. Troelstra. *Realizability and intuitionistic logic*. Springer, 1984.

[36] HD. Ebbinghaus and J. Flum. *Finite Model Theory*, volume 2. Springer, 1995.

[37] S. Feferman. Systems of predicative analysis, I. *Journal of Symbolic logic*, pages 1–30, 1964.

[38] S. Feferman. Systems of predicative analysis, II: Representations of ordinals. *Journal of Symbolic Logic*, pages 193–220, 1968.

[39] S. Feferman. A more perspicuous formal system for predicativity. *Konstruktionen versus Positionen*, 1:68–93, 1978.

[40] S. Feferman. Weyl vindicated: Das kontinuum 70 years later. *Termi e prospettive della logica e della filosofia della scienza contemporanea*, 1:59–93, 1988.

[41] S. Feferman. Finitary inductively presented logics. *Studies in Logic and the Foundations of Mathematics*, 127:191–220, 1989.

[42] N. Foster, D. Kozen, M. Milano, A. Silva, and L. Thompson. A coalgebraic decision procedure for netkat. In *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 343–355. ACM, 2015.

[43] A. A. Fraenkel, Y. Bar-Hillel, and A. Levy. *Foundations of set theory*. Elsevier, 1973.

[44] R. O. Gandy. Set-theoretic functions for elementary syntax. In *Proceedings of Symposia in Pure Mathematics*, volume 13, pages 103–126, 1974.

[45] G. Gentzen. Untersuchungen über das logische schließen. I. *Mathematische Zeitschrift*, 39(1):176–210, 1935.

[46] G. Gentzen. Die gegenwärtige lage in der mathematischen grundlagenforschung. neue fassung des widerspruchsfreiheitsbeweises für die reine zahlentheorie. *Bull. Amer. Math. Soc. 45*, 1939.

[47] G. Gentzen. Beweisbarkeit und unbeweisbarkeit von anfangsfällen der transfiniten induktion in der reinen zahlentheorie. *Mathematische Annalen*, 119(1):140–161, 1943.

[48] K. Gödel. The consistency of the continuum hypothesis. Princeton University Press, 1940.

[49] M. Gordon, R. Milner, and C. Wadsworth. *Edinburgh LCF: a mechanized logic of computation*, volume 78. Springer, 1979.

[50] N. B. B. Grathwohl, D. Kozen, and K Mamouras. Kat+ B! In *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, page 44. ACM, 2014.

[51] M. Hallett. *Cantorian set theory and limitation of size*. Clarendon Press Oxford, 1984.

[52] J. Y. Halpern, R. Harper, N. Immerman, P. G. Kolaitis, M. Y. Vardi, and V. Vianu. On the unusual effectiveness of logic in computer science. *Bulletin of Symbolic Logic*, 7(02):213–236, 2001.

[53] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the ACM (JACM)*, 40(1):143–184, 1993.

[54] L. Henkin. Completeness in the theory of types. *Journal of Symbolic Logic*, 15(2):81–91, 1950.

[55] A. Heyting. Mathematische grundlagenforschung intuitionismus, beweistheorie. 1934.

[56] A. Heyting. L.E.J. brouwer collected works, volume I, Philosophy and foundations of mathematics. *Bulletin of the American Mathematical Society 83*, 1977.

[57] J. Hickey. *The MetaPRL logical programming environment*. PhD thesis, Cornell University, 2001.

[58] K. Hrbacek and T. Jech. *Introduction to Set Theory, Revised and Expanded*, volume 220. CRC Press, 1999.

[59] A. E. Hurd and P. A. Loeb. *An introduction to nonstandard real analysis*, volume 118. Academic Press, 1985.

[60] Y. I. Ianov. The logical schemes of algorithms. *Problems of Cybernetics*, 1:82–140, 1960.

[61] N. Immerman. Languages that capture complexity classes. *SIAM Journal on Computing*, 16(4):760–778, 1987.

[62] R. B. Jensen. The fine structure of the constructible hierarchy. *Annals of Mathematical Logic*, 4(3):229–308, 1972.

[63] S. C. Kleene. On the interpretation of intuitionistic number theory. *Journal of Symbolic Logic*, 10(4):pp. 109–124, 1945.

[64] S. C. Kleene. *Representation of Events in Nerve Nets and Finite Automata.* Memorandum. Rand Corporation, 1951.

[65] A. Kolmogoroff. Zur deutung der intuitionistischen logik. *Mathematische Zeitschrift*, 35(1):58–65, 1932.

[66] D. Kozen and A. Angus. Kleene algebra with tests and program schematology. Technical report, 2001.

[67] D. Kozen and A. Barth. Equational verification of cache blocking in lu decomposition using Kleene algebra with tests. Technical report, 2002.

[68] D. Kozen and K. Mamouras. Kleene algebra with equations. In *Automata, Languages, and Programming*, pages 280–292. Springer, 2014.

[69] K. Kunen. *Set theory: An introduction to independence proofs.* North-Holland, 1980.

[70] S. Lev and A. Avron. A syntactic characterization of domain independence with parameters. unpublished manuscript.

[71] H. J. Levesque, R. Reiter, Y. Lesperance, F. Lin, and R. B. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1):59–83, 1997.

[72] A. Levy. *Basic Set Theory. Perspectives in Mathematical Logic*. Spring, 1979.

[73] L. Libkin. *Elements of finite model theory*. Springer Science & Business Media, 2013.

[74] Z. Manna. *Mathematical theory of computation*. McGraw-Hill, Inc., 1974.

[75] Z. Manna and R. Waldinger. *The logical basis for computer programming*, volume 1. Addison-Wesley Reading, 1985.

[76] R. M. Martin. A homogeneous system for formal logic. *Journal of Symbolic Logic*, 8(1):pp. 1–23, 1943.

[77] R. M. Martin. A note on nominalism and recursive functions. *Journal of Symbolic Logic*, 14(1):pp. 27–31, 1949.

[78] P. Martin-Löf. Constructive mathematics and computer programming. *Studies in Logic and the Foundations of Mathematics*, 104:153–175, 1982.

[79] P. Martin-Löf and G. Sambin. *Intuitionistic type theory*. Studies in proof theory. Bibliopolis, 1984.

[80] J. McCarthy. A formal description of a subset of algol. Technical report, DTIC Document, 1964.

[81] N. Megill. *Metamath: A Computer Language for Pure Mathematics*. Elsevier Science, 1997.

[82] J. Myhill. A derivation of number theory from ancestral theory. *Journal of Symbolic Logic*, 17(3):pp. 192–197, 1952.

[83] J. Myhill. Constructive set theory. *Journal of Symbolic Logic*, 40(03):347–382, 1975.

[84] R. P. Nederpelt, J. H. Geuvers, and R.C. De Vrijer. *Selected papers on Automath*. Elsevier, 1994.

[85] E. Nelson. *Predicative Arithmetic*. Princeton University Press, 1986.

[86] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL: a proof assistant for higher-order logic*, volume 2283 of *LNCS*. Springer, 2002.

[87] B. Nordström, K. Petersson, and J. M. Smith. *Programming in Martin-Löf's type theory: an introduction*. International Series of Monographs on Computer Science. Clarendon Press, 1990.

[88] H. Poincaré. Les mathématiques et la logique. *Revue de Métaphysique Et de Morale*, 14(3):294–317, 1906.

[89] H. Poincaré. Dernières pensées, Flammarion, 1913.

[90] H. Putnam. What is mathematical truth? *Historia Mathematica*, 2(4):529–533, 1975.

[91] W. V. Quine. Carnap and logical truth. *Synthese*, 12(4):350–374, 1960.

[92] W. V. Quine. *Set theory and its logic*, volume 9. Harvard University Press, 1969.

[93] W. V. Quine. *Philosophy of Logic*. Harvard University Press, 1986.

[94] F. Ramsey. The foundations of mathematics. *Proceedings of the London Mathematical Society*, 2(1):338–384, 1926.

[95] R. H. Risch. Algebraic properties of the elementary functions of analysis. *American Journal of Mathematics*, 101(4):743–759, 1979.

[96] P. Rudnicki. An overview of the mizar project. In *Proceedings of the 1992 Workshop on Types for Proofs and Programs*, pages 311–330, 1992.

[97] B. Russell. Les paradoxes de la logique. *Revue de métaphysique et de morale*, 14(5):627–650, 1906.

[98] B. Russell. Mathematical logic as based on the theory of types. *American journal of mathematics*, 30(3):222–262, 1908.

[99] V. Y. Sazonov. On bounded set theory. In *Logic and Scientific Methods*, pages 85–103. Springer, 1997.

[100] K. Schütte. *Predicative well-orderings*. North-Holland Publ., 1965.

[101] K. Schütte. *Proof theory*. Springer Verlag, 1977.

[102] S. Shapiro. *Foundations without foundationalism: A case for second-order logic*. Oxford University Press, 1991.

[103] J. R. Shoenfield. *Mathematical logic*. Addison-Wesley Reading, 1967.

[104] S. G. Simpson. *Subsystems of second order arithmetic*, volume 1 of *Perspectives in Logic*. Cambridge University Press, 2009.

[105] P. Smith. Ancestral arithmetic and Isaacson's thesis. *Analysis*, 68(297):1–10, 2008.

[106] R. M. Smullyan. *Gödels incompleteness theorems*. Oxford University Press, 1992.

[107] M. H. Sørensen and P. Urzyczyn. *Lectures on the Curry-Howard isomorphism*, volume 149. Elsevier, 2006.

[108] G. Takeuti. *Proof theory*. Courier Dover Publications, 1987.

[109] A. S. Troelstra. Aspects of constructive mathematics. *Studies in Logic and the Foundations of Mathematics*, 90:973–1052, 1977.

[110] A. S. Troelstra and D. Van Dalen. *Constructivism in Mathematics: An Introduction*. Number 1 in Constructivism in Mathematics. North-Holland, 1988.

[111] A. Trybulec and H. A. Blair. Computer aided reasoning. In *Logics of Programs*, pages 406–412. Springer, 1985.

[112] J. D. Ullman. *Principles of Database and Knowledge-base Systems, Vol. I*. Computer Science Press, Inc., New York, NY, USA, 1988.

[113] W. P. Van Stigt. *Brouwer's intuitionism*. North-Holland Amsterdam, 1990.

[114] N. Weaver. Analysis in J_2. *arXiv preprint math/0509245*, unpublished manuscript, 2005.

[115] N. Weaver. Mathematical conceptualism. *arXiv preprint math/0509246*, unpublished manuscript, 2005.

[116] H. Weyl. *Das Kontinuum: Kritische Untersuchungen über die Grundlagen der Analysis*. W. de Gruyter, 1932.

[117] A. Whitehead and B. Russell. *Principia mathematica*, volume I, II and III. Cambridge University Press, 1910–1913.

[118] F. Wiedijk. The QED manifesto revisited. *Studies in Logic, Grammar and Rhetoric*, 10(23):121–133, 2007.

[119] S. Wohrle and W. Thomas. Model checking synchronized products of infinite transition systems. In *Logic in Computer Science, 2004. Proceedings of the 19th Annual IEEE Symposium on*, pages 2–11. IEEE, 2004.

[120] E. Zermelo. Untersuchungen über die grundlagen der mengenlehre. I. *Mathematische Annalen*, 65(2):261–281, 1908.

[121] J. Zucker. Formalization of classical mathematics in automath. *Studies in Logic and the Foundations of Mathematics*, 133:127–139, 1994.