Homotopy Type-Theoretic Interpretations of Constructive Set Theories

Cesare Gallozzi

Submitted in accordance with the requirements
for the degree of Doctor of Philosophy

The University of Leeds
School of Mathematics

July 2018

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

To all my teachers.

Acknowledgements

I wish to thank all those who made this thesis possible. I thank all my teachers and in particular my supervisor Nicola Gambino for his tireless generosity and all the help and guidance he offered during the course of the PhD.

I thank the University of Leeds and the School of Mathematics for their financial support.

I thank all those who contributed to improve this thesis by answering my questions or making comments on the material: my co-supervisor Michael Rathjen, and also Peter Hancock, John Truss, Stan Wainer, Martin Hofmann, Helmut Schwichtenberg, Michael Toppel, Anton Freund, Andrew Swan, Jakob Vidmar, Nicolai Kraus and Fredrik Nordvall Forsberg.

I thank my parents and my grandmother for all their care and for encouraging my interests in science and mathematics since my early childhood. I also thank all my friends.

Abstract

This thesis deals primarily with type-theoretic interpretations of constructive set theories using notions and ideas from homotopy type theory.

We introduce a family of interpretations $[\![\cdot]\!]_{k,h}$ for $2 \leq k \leq \infty$ and $1 \leq h \leq \infty$ of the set theory BCS into the type theory H, in which sets and formulas are interpreted respectively as types of homotopy level k and h. Depending on the values of the parameters k and h we are able to interpret different theories, like Aczel's CZF and Myhill's CST. We relate the family $[\![\cdot]\!]_{k,h}$ to the other interpretations of CST into homotopy type theory already studied in the literature in [Uni13] and [Gyl16a].

We characterise a class of sentences valid in the interpretations $[\![\cdot]\!]_{k,\infty}$ in terms of the $\Pi\Sigma$ axiom of choice, generalising the characterisation of [RT06] for Aczel's interpretation.

We also define a proposition-as-hproposition interpretation in the context of logic-enriched type theories. The formulas valid in this interpretation are then characterised in terms of the axiom of unique choice.

We extend the analysis of Aczel's interpretation provided in [GA06] to the interpretations of CST into homotopy type theory, providing a comparative analysis. This is done formulating in the logic-enriched type theory the key principles used in the proofs of the two interpretations.

We also investigate the notion of feasible ordinal formalised in the context of a linear type theory equipped with a type of resources. This type theory was originally introduced by Hofmann in [Hof03]. We disprove Hofmann's conjecture on the definable ordinals, by showing that for any given $k \in \mathbb{N}$ the ordinal ω^k is definable.

Contents

Introduction

Background

The topic of this thesis belongs to the general area of the study of the foundations of mathematics, and focuses on how different foundational frameworks are related. The expression 'foundations of mathematics' suggests often a reductionist approach in which some concepts are seen as fundamental and all other mathematical concepts ought to be defined in their terms.

Here, however, we consider foundations in the sense of an approach and language for the development of mathematics. From a pragmatic viewpoint, different languages and concepts are suited to study and be applied to different areas of mathematics.

The three main options for developing the foundations of mathematics are set theory, category theory and type theory, with a lot of diversity within each one. Other foundational theories are systems of arithmetic used in reverse mathematics [Sim09] and untyped formal systems, for example Feferman's system of explicit mathematics [Fef75].

When using the language and techniques of set theory, category theory or type theory, the mathematics developed there has distinct flavours, different ideas come into play and different questions arise naturally. In other words, the mathematics we do and the way we do it depend on the foundational framework. One can argue that any single perspective has a blind spot, from which comes the importance of having available different foundational frameworks, and to understand their relationships.

Despite the diversity of different foundational theories nearly all of what is used by the working mathematician can be developed in these different frameworks and to a large extent also in subsystems of second order arithmetic. This underlines the interconnectedness of these quite different foundational frameworks. In particular there are various ways to relate set theory, category theory and type theory with each other. See [Fef88] for the foundational and conceptual implications of interpreting a theory into another one. See also [MP02] and [Awo11] for more details on relating set theory, category theory and type theory.

In this thesis we interpret constructive set theories into type theories taking inspiration from concepts and ideas from homotopy type theory. We are also interested in understanding how these different interpretations relate to each other.

Constructive set theories and type theories

Here we give some context and an introduction to the theories we consider in this thesis.

Set theories. The set theories we consider are Myhill's constructive set theory CST [Myh75] and Aczel's Constructive Zermelo-Fraenkel CZF [Acz78]. We also consider for convenience the theory that we call Basic Constructive Set Theory, BCS for short, which comprises of the axioms shared by both CST and CZF. See appendix A for their axioms. The only difference between CST and CZF is that the Replacement and Exponentiation axioms of CST are replaced in CZF by the Strong Collection and Subset Collection axioms, which are stronger (see [AR10, Section 5.1]).

The theories CST and CZF are constructive at two levels: in the sense that the underlying logic is intuitionistic rather than classical, and also because they are predicative theories. At the first level, one can see them intuitively as the result of the process that starts from ZFC and either removes or reformulates the axioms that are incompatible with intuitionistic logic. The axiom of choice is removed and the Foundation axiom is reformulated as the $\in$-induction axiom, since they both imply the law of excluded middle.

Moreover, CST and CZF are predicative, meaning that sets are conceived as being constructed from already constructed sets. Definitions of a set that use a collection to which the set belongs are prevented by further restrictions on the axioms. For example the usual definition of closure in a topological space is impredicative: 'given a subset of a space $S \subseteq X$, the closure $\bar{S}$ is the smallest closed set containing S'. Instead, in a predicative theory expressions like 'the smallest set such that' appear only as a shorthand for an explicit construction that generates the set from ones that are already constructed.

The axioms that need restriction in order to guarantee predicativity are the Power Set and the Separation axioms. The former is weakened to either the Exponentiation axiom in CST or the Subset Collection axiom in CZF. The latter is weakened to the Bounded Separation axiom, which is like the usual Separation axiom but applies only to bounded formulas. See appendix A for the details.

Type theories. Type theories comprise types and terms. Types are defined together with their terms by deduction rules that introduce terms and govern how

terms interact. The rules of type theory allow one to derive a judgement, say $\mathcal{J}$, in a type-theoretic context, say Γ, which is written as $\Gamma \vdash \mathcal{J}$. Judgements are expressions of the form $A : \mathsf{type}$ or $A = B : \mathsf{type}$ for types or $a : A$ or $a = b : A$ for terms. A context is a finite list of declarations of variables $\Gamma = \{\ x_1\ :\ A_1\ ,\ \ldots x_n :\ A_n\ \}$ that appear in the body of the judgement.
The types of a type theory can either be base types, e.g. the type of natural numbers N (see table A7), or be formed from already constructed types via a type-constructor, e.g. given types A and B we can form their disjoint union $A + B$ (see table A10).

The type theories we consider are all variants or extensions of Martin-Löf type theory (see [ML84] for a readable introduction to the extensional theory). Martin-Löf type theory is a dependent type theory, i.e. types can depend on terms. This means that in context $x : A$ one may derive the judgement $B(x) : \mathsf{type}$, which is formally expressed as $x : A \vdash B(x) : \mathsf{type}$, and one may construct a dependent term $x : A \vdash b(x) : B(x)$. The dependency of types on terms allows one to express a family of types internally as $x : A \vdash B(x) : \mathsf{type}$ and to introduce type constructors that take as input a family and return a type. The main examples are Π-types, Σ-types and W-types that construct the dependent function type, the dependent product type and the type of well-founded trees, respectively. See appendix B for the rules.

The rules of Martin-Löf type theory are a decoration with proof terms of the rules of natural deduction, which specify for each formula A its proofs $a : A$. The correspondence between type theory, set theory and proof theory can be depicted as follows:

type theory	set theory	logic
type A	set A	formula A
term $a : A$	element $a \in A$	proof of A
$(\Pi x : A)B(x)$	dependent function space	universal quantification
$(\Sigma x : A)B(x)$	disjoint union	existential quantification
type N	set $\mathbb{N}$	formula $(\exists x \in \mathbb{N})\top$

This correspondence also extends to category theory, in a locally cartesian closed category, and to computation in the context of functional programming languages.

One key feature of type theory, highlighted in the table above is the correspondence with logical systems, which is called *propositions-as-types* or *Curry-Howard correspondence* (see [SU06] for an introduction to the subject). This allows one to treat formulas and types uniformly, unlike what happens in set theory where there are two layers of the theory: the underlying logic and the set-theoretic axioms.

However, in chapter 4 we consider logic-enriched type theories, which are type theories with primitive judgements expressing formulas $\phi : \mathsf{prop}$ and $\phi_1, \ldots, \phi_n \Rightarrow \phi$ within the type theory. In this respect logic-enriched type theories gain more flexibility while preserving the good proof theoretic properties of type theories.

One of the most notable features of type theories are normalisation theorems which are proved analysing the structure of terms and how they interact, and provide syntactic proofs of consistency without any use of models, therefore providing a strong justification for such theories. See [ML75, Theorem 3.3] for a normalisation theorem for Martin-Löf type theory.

Another important aspect of Martin-Löf type theories are identity types. Among the primitive judgements of a type theory there are equality judgements, for example $A = B : \mathsf{type}$ and $a = b : A$. These are called *definitional* or *judgemental equalities*, and they are proved by syntactic manipulation of terms. One may want to introduce alongside them a *propositional equality*, i.e. a type constructor following the propositions-as-types correspondence that represents the equality proposition within type theory. In other words, a constructor that given a type $A : \mathsf{type}$ and two terms $a, b : A$ returns a new type $\mathsf{Id}_A(a, b)$. Its terms are understood as proofs that the terms $a, b : A$ are equal.

There are two approaches to identity types, both due to Martin-Löf. One is to consider *extensional* identity types, in which the type $\mathsf{Id}_A(a, b)$ reflects internally as a type the judgemental equality $a = b$, as a consequence all terms of Id_A are equal (see [ML84]). On the other hand, one can introduce *intensional* identity types in which the type Id_A is defined as the least reflexive relation on A (see table A12 for the rules). In this way one does not have anymore that all terms of Id_A are propositionally nor judgementally equal, which was proved in [HS94] and [HS98]. Intensional identity types turn out to be remarkably more complex giving a higher categorical structure on every type (see [Lum09]). In this thesis we deal mostly with type theories with intensional identity types.

Homotopy type theories. Homotopy type theory is a type theory that deepens this connection with abstract homotopy theory arising from intensional identity types. It extends Martin-Löf type theory with two new main features: the univalence axiom and higher inductive types, which are inspired by homotopical notions and constructions. Homotopy type theory is closely connected to Voevodsky's idea of univalent foundations, which is an approach to the foundations of mathematics based on homotopy types and the univalence axiom. The standard reference for homotopy type theory and univalent foundations is [Uni13].
The univalence axiom describes the identity type of the universe between two small

types $\mathsf{Id}_{\mathsf{U}}(A, B)$ as the type of equivalences between A and B (see chapter 1 for details). As a consequence of univalence and the elimination rule for Id-types, all the properties expressible in type theory through the propositions-as-types correspondence become invariant under equivalence of types.

The univalence axiom presents a different approach for the investigation of the meaning of equality. Type theory, differently from set theory, with its distinction between judgement and proposition, invites a distinction between judgemental and propositional equality, which in turn allows for a deeper analysis of equality. With intensional identity types one can keep track of different ways to prove the equality between two terms. And in homotopy type theory in particular, the notion of equality is made in correspondence with the notion of equivalence, which is interpreted in homotopy theory as homotopy equivalence between homotopy types.
Thus, taking homotopy type theory as a foundation of mathematics means taking homotopy types as the intended model for the primitive objects of the theory and homotopy equivalence as the natural notion of equality. Only logical properties that are invariant under homotopy equivalence can be expressed internally through the propositions-as-types correspondence.

Then one can define the notion of set in homotopy type theory as a discrete homotopy type, which is called a homotopy sets (or hset for short). Another important notion is the one of hproposition (or hprop for short), which is a type such that all its terms are related by an identity, so classically a hprop is either empty or contractible. Hprops and hsets are special cases of a general definition of homotopy level of a type, which is due to Voevodsky [Voe15]. We recall the definitions of hsets and hprops in chapter 1. In chapter 1 and chapter 4 we use hprops to model formulas of set theory. [1]

The other distinguishing feature of homotopy type theory are higher inductive types. In type theory a new type is defined by specifying inductively its terms and how they behave (see for example the natural numbers in table A7). But the idea that a type comes together with its identity type suggests the definition of new kinds of inductive types in which terms of the type are defined together with terms of its identity type and of its iterated identity types. For example this approach allows one to define spheres in type theory [Uni13, Section 6.4]. A higher inductive type that we use often is truncation, which for each natural number h is an operation that given a type A returns a new type $\|A\|_h$ such that all towers of iterated identity types of length h or more are trivial (see table 1.1).

It is worth noting that a drawback of the homotopy type theories present up to now is that the univalence axiom and higher inductive types in their current

[1]Note that when counting the homotopy level of a type we diverge from the usual convention and start to count from 0 for contractible types, 1 for hprops, 2 for hsets, and so on.

formulation break the proof of the normalisation theorem. The recent proof of canonicity for cubical type theory [Hub16] is a step towards solving this problem.

Linear type theories. We also consider a different topic, namely the application of type theory to the study of feasible ordinals.

Our starting point is [Hof03], in which Hofmann designed a system called *Linear Functional Programming Language*, LFPL for short, designed to capture polytime non-size increasing computation. See Section 6.1 for the rules of the system. This is achieved by introducing a resource type $\diamond$ which provides tokens that have to be used to type size increasing functions. With regard to the structural rules, weakening is allowed but not contraction. This ensures that resources cannot be duplicated but can be discarded. Hofmann in [Hof03, Corollary 5.5.1], and Aehlig and Schwichtenberg in [AS02, Corollary 4.10], prove that the closed terms of type $\mathsf{N} \multimap \mathsf{N}$ are non-size increasing polytime functions. In Section 6.1 we introduce a type of ordinal notations on top of LFPL and study lower and upper bounds for the definable ordinals.

This is an example application of linear logic as a logic of knowability and feasibility. There are many applications of linear logic to the field of implicit computational complexity in which linear logic is used to model and control the use of resources in order to provide logical characterisations of complexity classes. See for example bounded linear logic [GSS92] and soft linear logic [Laf04].
One way to see linear logic is indeed as a refinement of intuitionistic logic that takes into account resources and their relevance for the notions of proof and knowledge.

The conceptual relevance of this kind of investigation into feasible mathematics in general, and feasible ordinals in particular, comes from the observation that truth is not an absolute concept, independent from the knower of that truth. As human beings our resources, both in terms of computational time and space are finite, hence there is an epistemological boundary to the length of proofs and arguments that we can check. So one can envision a development of feasible mathematics alongside the one of constructive mathematics, [BS13].

In this thesis we equate feasible computation with polytime computation, as it is commonly done. We do not investigate its adequacy and limitations.

The state of the literature

The first interpretation of constructive set theory into type theory dates back to [Acz78]. The idea is to interpret sets as trees where nodes represent sets and edges the membership relation so that the root represents the given set. This intuition is formalised by interpreting the universe of sets as the W-type of all small types over the universe, i.e. $\mathsf{V} := (Wx : \mathsf{U})\mathsf{T}(x)$.

The notion of equivalence between trees that makes them extensional is a bisimulation relation, so that trees are identified up to permutation and repetition of their branches. Then to every set-theoretic formula ϕ one associates a type $[\![\phi]\!]$ using the propositions-as-types correspondence, and for every axiom of CZF one constructs a term inhabiting its interpretation.

More work has been done since then in order to understand this interpretation. For our purposes [RT06] and [GA06] are relevant, since we generalise some of their results. Rathjen and Tupailo in [RT06] give a characterisation of Aczel's interpretation in terms of the choice principle ΠΣ-AC (see Axiom 2.1.5). In Theorem 2.2.2 we state their result. Aczel and Gambino in [AG00, Theorem 2] characterise the propositions-as-types interpretation $[\![\cdot]\!]_{PT}$ from the logic-enriched type theory LE, defined in Section 4.1, into Martin-Löf type theory. Then in [GA06] they construct a generalised type-theoretic interpretation of CZF using logic-enriched type theories.

We address the same questions for a number of new interpretations of constructive set theories into type theories: for the proof-theoretic characterisation by Rathjen and Tupailo in chapter 2, and for the analysis of interpretations using logic-enriched type theories in chapter 4 and chapter 5.

Since the introduction of homotopy theory there has been work to interpret set theories into homotopy type theory. The paper [RS15] studies the category of hsets in homotopy type theory. In the HoTT book [Uni13, Section 10.5] there is also an interpretation of set theory directly into type theory, along the lines of Aczel's interpretation. Set-theoretic formulas are interpreted as hpropositions by applying the operation of propositional truncation to type formers that do not respect hprops. This interpretation uses a higher inductive type V_{H} defined for the purpose of interpreting set theory, its identity type interprets set-theoretic equality. See Section 3.3 for the details. This interpretation validates the axioms of CST, see [Uni13, Theorem 10.5.8]. Then adding the axiom of choice formulated in type theory for the logic of hpropositions they interpret full ZFC, [Uni13, Theorem 10.5.11].

Another interpretation of set theory into type theory appeared in the paper by Gylterud [Gyl16b]. Univalence is used there to construct a model of a theory of multisets[2] into type theory. In the other paper [Gyl16a], it is shown how to construct a model of CST into homotopy type theory from the multisets model. Sets are interpreted as terms of a type V_{G} which is a subtype of the one used by Aczel. We recall its definition in Section 3.2. Gylterud's interpretation takes place in the theory that in this thesis is called HoTT (see Section 1.1 for the definition). It uses univalence, propositional truncations and set quotients.

[2]Multisets are sets in which elements are counted with multiplicites.

In [Gyl16a, Proposition 8:5 and Remark 8:6] the equivalences between the types $\mathsf{V}_{\mathsf{G}} \simeq \mathsf{V}_{\infty}/_{\approx_1}$ and the types $\mathsf{V}_{\infty}/_{\approx_1} \simeq \mathsf{V}_{\mathsf{H}}$ are stated, providing a connection with the HoTT book interpretation. In stating the equivalence we used the notation used in this thesis.

The situation however does not look completely clear, some features of homotopy type theory come into play, most prominently hprops being used to interpret formulas of set theory. In the HoTT book the higher inductive type V_{H} is used but there is no need for univalence nor set-quotients as we observe in Remark 3.3.1. On the other hand in Gylterud's work univalence and set quotients are used. Also it is not clear what role hsets play in Gylterud's and the HoTT book interpretations. Both the types of sets V_{G} and V_{H} are hsets themselves, but the types that interpret sets are arbitrary types and are not hsets in general.

Outline of the thesis and main results

Considering the situation described so far one may be interested in investigating the following questions.

(1) Is it possible to construct an interpretation in which sets are interpreted as hsets and formulas as hprops?
(2) What is the role played by hsets and more in general by homotopy levels in the interpretations?
(3) How do the HoTT book and Gylterud's interpretations relate to the new interpretations we introduce and to Aczel's original interpretation?

This thesis provides some answers to these questions. In chapter 1 we investigate the first question. Our aim is to construct interpretations that use the notions of hset and hprop from homotopy type theory, but that are as close as possible to Aczel's original interpretation, so that it is easier to understand what is essential.

We also want to understand what are the properties of hsets and hprops that make the interpretations work, and check if the same result can be obtained with differnet homotopy levels. For these reasons we develop a hierarchy of interpretations $[\![\cdot]\!]_{k,h}$, for $1 \leq h \leq \infty$ and $2 \leq k \leq \infty$, of set theory into the type theory H (see Section 1.2). The idea is to interpret sets as trees with indices for the branches coming only from types of fixed homotopy level k, and to interpret formulas as types of homotopy level h by using truncations. Aczel's original interpretation coincides with the interpretation $[\![\cdot]\!]_{\infty,\infty}$. We show in Theorem 1.3.14 that depending on the values of the parameters k and h we can interpret certain set theories in certain type theories, starting to address the second question of the list.

Chapter 2 is devoted to looking more deeply at the role played by homotopy levels. Our starting point is the characterisation of Aczel's interpretation by

Rathjen and Tupailo in [RT06]. We extend their result to the family of interpretations $[\![\cdot]\!]_{k,\infty}$, for $2 \leq k \leq \infty$, in which sets are interpreted as k-types but formulas are not truncated. We show that all the interpretations $[\![\cdot]\!]_{k,\infty}$ for $2 \leq k \leq \infty$, validate the same CC sentences, which are defined in Definition 2.2.1. Morally speaking, CZF cannot distinguish between the interpretations $[\![\cdot]\!]_{k,\infty}$.

Chapter 3 investigates the relation between on the one hand the interpretations in the HoTT book and by Gylterud and on the other hand the family $[\![\cdot]\!]_{k,h}$ that we have introduced. Already in [Gyl16a, Proposition 8:5 and Remark 8:6] it is stated that the types V_{G}, V_{H} and $\mathsf{V}_{\infty}/_{\approx_1}$ are equivalent as types. We show that there is a logical equivalence between the $[\![\cdot]\!]_{k,1}$ interpretations, Gylterud's interpretation and the HoTT book interpretation. This equivalence is induced by an equivalence between the setoids $(\mathsf{V}_{\mathsf{G}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{G}}})$, $(\mathsf{V}_{\mathsf{H}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{H}}})$ and $(\mathsf{V}_{\infty}, \approx_1)$. In chapter 3 we recall the relevant details of Gylterud's work and fill the details of the proofs, then we use the logical equivalence between interpretations to validate the Replacement and Exponentiation axioms in our interpretations $[\![\cdot]\!]_{k,1}$.

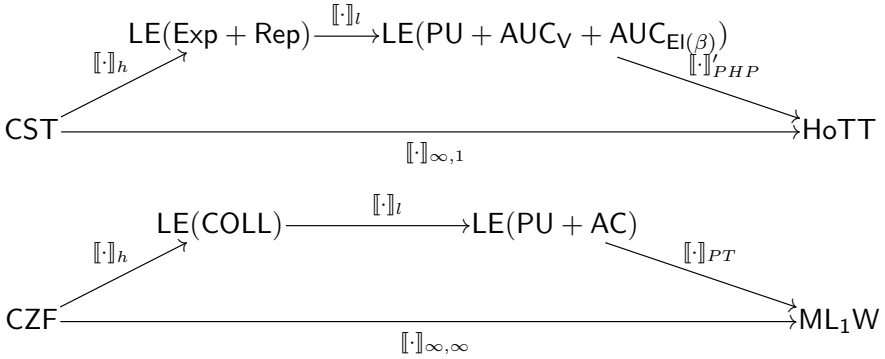
In chapter 4, we investigate the interpretation of formulas as hprops, which is key to all the equivalent interpretations of CST into HoTT. For this purpose, we introduce the tool of logic-enriched type theories. We introduce an appropriate logic-enriched type theory LEH in Section 4.1 and define abstractly in that context a propositions-as-hprops interpretation into the type theory H using truncations. This interpretation mirrors the structure of the interpretations $[\![\cdot]\!]_{k,1}$. Then we proceed to adapt to our situation the characterisation of [AG00, Theorem 2]. We prove in Theorem 4.3.4 the characterisation of the formulas valid in the propositions-as-hprop interpretation by using the axiom of unique choice formulated in the logic-enriched type theory LEH, see table 4.5.

The rest of chapter 4 and chapter 5 are devoted to understanding the similarities and differences between the equivalent interpretations of CST into HoTT on one hand and Aczel's interpretation on the other hand. We follow the example of [GA06], where the generalised type-theoretic interpretation of CZF is introduced and logic-enriched type theories are used to factor the interpretation into intermediate steps. In this way different aspects of the interpretation are isolated and captured by principles expressed within the logic-enriched type theory.

In Section 4.4 we introduce a second propositions-as-hprops interpretation from the logic-enriched type theory LE to HoTT. The inductive definition of the interpretation is the same as the other propositions-as-hprops interpretation defined in Section 4.2, but the domain and codomain theories are different. We also formulate the two choice principles $\mathsf{AUC}_{\mathsf{V}}$ in table 4.7 and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ in table 4.8, which are then used in chapter 5 for the analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$. The principles

$\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ are validated in the propositions-as-hprops interpretation into HoTT in Theorem 4.4.1 and Theorem 4.4.2, and are therefore justified.

Chapter 5 addresses the third question of the list. It provides a comparative analysis of Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$ on one hand, and of the interpretation $[\![\cdot]\!]_{\infty,1}$ on the other hand. The interpretation $[\![\cdot]\!]_{\infty,1}$ is the best suited for this kind of analysis among the equivalent interpretations of CST into HoTT. The first step of the analysis is the factorisation through the propositions-as-hprops interpretations of $\mathsf{LE(PU + AUC_V + AUC_{El(\beta)})}$ into HoTT of Theorem 5.1.6. The propositions-as-types principle PU, defined in table 4.6, and two forms of the Axiom of Unique Choice $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ allow us to validate respectively the Bounded Separation, Replacement and Exponentiation axioms of CST.
Then following the example of [GA06], we factor further the interpretation of CZF into $\mathsf{LE(PU+AUC_V+AUC_{El(\beta)})}$ via another logic-enriched type theory $\mathsf{LE(Rep+Exp)}$ in which there are Exponentiation and Replacement rules that mirror the set-theoretic axioms in the logic-enriched type theory. The rule Exp is defined in table 5.4 and Rep in table 5.3 . Under the assumption of PU the Replacement and Exponentiation Rules follow from $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$, respectively.
The main result of this chapter is Theorem 5.3.5, which can be depicted in the following commutative diagrams:



The two factorisations of $[\![\cdot]\!]_{\infty,\infty}$ and $[\![\cdot]\!]_{\infty,1}$ follow the same structure very closely in the first two steps. These are the hybrid interpretation $[\![\cdot]\!]_h$ and an interpretation of one logic-enriched type theory into the other $[\![\cdot]\!]_l$. The differences are isolated in the last step of the interpretation where the propositions-as-types $[\![\cdot]\!]_{PT}$ and the propositions-as-hprops $[\![\cdot]\!]'_{PHP}$ interpretations take place.
Thus, we obtain a precise parallel analysis of all the known interpretations of CST into homotopy type theory on one hand and of CZF into Martin-Löf type theory on the other hand.

Chapter 6 deals with the application of linear type theory to the study of feasible ordinals. We introduce a type of ordinal notations $\mathcal{O}$ in table 6.9 in the context of Hofmann's LFPL. We investigate what ordinals can be typed in the system in Section 6.3. The main result is Theorem 6.3.7, in which we disprove Hofmann's conjecture on the definable ordinals of LFPL by proving that all ordinals less than ω^ω can be defined in $\mathsf{LFPL} + \mathcal{O}$. The rest of chapter 6 consists in a discussion on possible approaches for proving upper bounds and what are the issues that arise in doing so.

Summarising the discussion above, the main results of this thesis are:

- the new interpretations of Theorem 1.3.14;
- the proof-theoretic characterisation of Theorem 2.5.1 and Corollary 2.5.2;
- the characterisation of the propositions-as-hprops interpretation of Theorem 4.3.4;
- the factorisation of the interpretation $[\![\cdot]\!]_{\infty,1}$ of Theorem 5.3.5;
- Theorem 6.3.7, disproving Hofmann's conjecture.

CHAPTER 1

Homotopy type-theoretic interpretations

In this chapter we construct a hierarchy of interpretations $[\![\cdot]\!]_{k,h}$, for $1 \leq h \leq \infty$ and $2 \leq k \leq \infty$, of set theory in type theory. This hierarchy arises naturally when we investigate the relationship between sets and formulas of set theory and the homotopic notions of a hset and a hproposition (see Definition 1.1.3).

For our construction we take inspiration from Aczel's interpretation of CZF into Martin-Löf type theory [Acz78] and from the more recent interpretations of Myhill's constructive set theory CST in homotopy type theory developed in the HoTT book [Uni13] and by Gylterud [Gyl16a]. A starting point is the observation that although the latter two interpretations interpret formulas as hprops they do not interpret sets as hsets. Then a natural question is whether the existing interpretations can be adapted to interpret sets as hsets. As we see in Proposition 1.3.7, the answer turns out to be positive. A second question can be posed of how much of the interpretations of constructive set theories depends on homotopy levels. To answer this question we introduce the family of interpretations $[\![\cdot]\!]_{k,h}$ for $2 \leq k \leq \infty$ and $1 \leq h \leq \infty$ (see Section 1.2). We follows Aczel's original interpretation as closely as possible, but also interpret sets as k-types[1] and formulas as h-types. This is achieved for sets by restricting to k-types in the formation of the type of well-founded trees V_k (defined in table 1.2) which represents the universe of sets, and for formulas by using the truncation operation $\| \cdot \|_h$.

In our interpretations $[\![\cdot]\!]_{k,h}$ the identity type $\mathsf{Id}_{\mathsf{V}_k}$ plays a subtle, indirect role. Indeed, it is not itself used to interpret formulas nor sets, but it is used in the definitions of k-types and of truncations. Identity types are used as a technical tool in the proof that the $\Pi\Sigma$ axiom of choice is valid in $[\![\cdot]\!]_{k,\infty}$ (see Corollary 2.4.10).

Moreover, observe that V_k has the same homotopy level of the universe U, hence if univalence is assumed then V_k is not a hset.

A notable interpretation is $[\![\cdot]\!]_{2,1}$ which interprets sets as hsets and formulas as hprops. Note that the interpretation $[\![\cdot]\!]_{\infty,\infty}$ where no restrictions are imposed on the homotopy levels coincides with Aczel's interpretation. In chapter 3 we apply some results by Gylterud to our context to compare our interpretation $[\![\cdot]\!]_{k,h}$ with

[1] More precisely, we interpret sets as well-founded trees in which branches are indexed by k-types.

the ones by Gylterud and the HoTT book. In chapter 5 we compare them with Aczel's interpretation.

The main result of this chapter is Theorem 1.3.14 which summarises the results on the interpretations in which different constructive set theories are interpreted in type theories depending on the values of the parameters k, h. We leave open the question if stronger set theories can be interpreted by $[\![\cdot]\!]_{k,h}$. In particular whether additional axioms for CZF like choice principles or the Regular Extension axiom are valid. In chapter 2 we prove that the $\Pi\Sigma$ axiom of choice is valid in the interpretations $[\![\cdot]\!]_{k,\infty}$ for $2 \leq k \leq \infty$, see Corollary 2.4.10.

Outline. In Section 1.1 we introduce the type theories H and HoTT that are used in the chapter as well as in the rest of the thesis. Then in Section 1.2 we define the family of interpretations $[\![\cdot]\!]_{k,h}$, and in Section 1.3 we validate the axioms of the constructive set theories CZF and CST via the interpretations $[\![\cdot]\!]_{k,h}$. Finally, Section 1.4 present an alternative interpretation of equality that gives rise to variant interpretations $[\![\cdot]\!]'_{k,h}$.

1.1. Type-theoretic preliminaries

All the type theories we consider in this thesis are extensions of Martin-Löf type theory. We always assume the propositional η-rule for Π-types (see appendix B.1).

The type theory H. Here we introduce the type theory H, which extends Martin-Löf type theory. It is given by the following:

- $\mathsf{ML}_1^{\mathsf{Id}}\mathsf{W}$ Martin-Löf type theory with one universe, intensional identity types and W types;
- the axiom of function extensionality FE (see Axiom 1.1.2)
- $\| \cdot \|_n$ truncations for each homotopy level $n \in \mathbb{N}$ (see table 1.1);

We refer the reader to appendix B for the complete list of rules.

First, we recall some notions that are used to state the function extensionality axiom.

Definition 1.1.1. Given a function $f : A \to B$ the type $\mathsf{isEquiv}(f)$ is defined asking the existence of a left and a right inverse for f following the propositions-as-types correspondence:

$$(\Sigma g : B \to A)(\Pi x : B)\mathsf{Id}_B(fg(x), x) \times (\Sigma h : B \to A)(\Pi x : A)\mathsf{Id}_A(hf(x), x).$$

The function f is an *equivalence* if and only if the type $\mathsf{isEquiv}(f)$ is inhabited, in which case we write $A \simeq B$.

The type $\mathsf{Equiv}(A, B)$ is defined as a pair consisting of a function and a proof that the function is an equivalence:

$$\mathsf{Equiv}(A, B) := (\Sigma f : A \to B)\mathsf{isEquiv}(f).$$

The above definition of $\mathsf{isEquiv}$ may look unnecessarily complicated since we are requiring two inverses g and h. The extra complexity is needed to ensure that for any two terms $p, q : \mathsf{isEquiv}(f)$ we have $\mathsf{Id}_{\mathsf{isEquiv}(f)}(p, q)$, which would not be necessarily true for the obvious definition of equivalence. For more details on the notions of equivalence in homotopy type theory see [Uni13, chapter 4].

Now we are ready to introduce the function extensionality axiom which can be summarised informally as saying that two functions are equal if and only if they are point-wise equal. Note that we can define the map:

$$\mathsf{happly} : \mathsf{Id}_{(\Pi x:A)B(x)}(f, g) \to (\Pi x : A)\mathsf{Id}_{B(x)}(f(x), g(x)) \tag{1.1}$$

by using the elimination rule for $\mathsf{Id}_{(\Pi x:A)B(x)}$. Indeed, suppose we have the reflexivity path $\mathsf{refl}_f : \mathsf{Id}_{(\Pi x:A)B(x)}(f, f)$. Then we can define the term $\lambda x.\mathsf{refl}_{f(x)}$ inhabiting the codomain $(\Pi x : A)\mathsf{Id}_{B(x)}(f(x), f(x))$. So if two functions are equal in the sense of the identity types then they are also point-wise equal.

Axiom 1.1.2. The *function extensionality axiom* (FE) requires the canonical map happly of (1.1) to be an equivalence, or in other words it requires the existence of a term:

$$\mathsf{funext} : \mathsf{isEquiv}(\mathsf{happly})$$

Next we recall the notion of homotopy level of a type.

Definition 1.1.3. For any type A and every natural number $n \in \mathbb{N}$ we define inductively the types $\mathsf{is}\text{-}n\text{-}\mathsf{type}(A)$ by letting:

$$\begin{cases} \mathsf{is}\text{-}0\text{-}\mathsf{type}(A) := (\Sigma a : A)(\Pi x : A)\mathsf{Id}_A(a, x) \\ \mathsf{is}\text{-}(n+1)\text{-}\mathsf{type}(A) := (\Pi x, y : A)\mathsf{is}\text{-}n\text{-}\mathsf{type}(\mathsf{Id}_A(x, y)) \end{cases}$$

A type A is called an n-type if and only if $\mathsf{is}\text{-}n\text{-}\mathsf{type}(A)$ is inhabited. We call a 0-type *contractible*, a 1-type a *hproposition*, and finally a 2-type a *hset*.

We also define the types $\mathsf{isHProp}(A)$ as $\mathsf{is}\text{-}1\text{-}\mathsf{type}(A)$ and the type $\mathsf{isHSet}(A)$ as $\mathsf{is}\text{-}2\text{-}\mathsf{type}(A)$.

Note that our notation for homotopy levels follows Voevodsky and differs from the one in [Uni13]. An n-type in our notation is an $(n-2)$-type in their notation. In [Uni13] one starts counting from -2 to fit the terminology of n-types already

established in homotopy theory where a set (i.e. discrete homotopy type) is called a 0-type.

Now we recall the *truncation* type constructor that given a type A produces an n-type $\|A\|_n$. The 0-truncation is easily defined as:

$$\|A\|_0 := \mathsf{1}$$

Before we state the rules for general truncations, let us first introduce the following notation for iterated identity types:

$$\begin{aligned} \mathsf{Id}^0_A(x,y) \quad &\text{is defined as} \quad A \\ \mathsf{Id}^{n+1}_A(x,y) \quad &\text{is defined as} \quad \mathsf{Id}_{\mathsf{Id}^n_A}(x,y) \end{aligned}$$

The rules defining truncations add terms to $\|A\|_n$ from terms of A, and freely add terms in its iterated identity types, so that $\|A\|_n$ is an n-type by construction. The last two rules express a recursion principle for truncations. Namely, given $a : A$ and a family of n-types $B(x)$ for $x : \|A\|_n$, in order to construct a term in $B(a)$ it is enough to do so assuming one has a term in A.

$$\frac{A : \mathsf{type}}{\|A\|_n : \mathsf{type}} \qquad \frac{a : \mathsf{U}}{\|a\|_n : \mathsf{U}} \qquad \frac{}{\mathsf{T}(\|a\|_n) = \|\mathsf{T}(a)\|_n}$$

$$\frac{a : A}{|a|_n : \|A\|_n} \qquad \frac{x, y : A}{\mathsf{i}(x,y) : \mathsf{Id}_{\|A\|_n}(x,y)} \qquad \dots \qquad \frac{p, q : \mathsf{Id}^{n-1}_{\|A\|_n}}{\mathsf{i}(p,q) : \mathsf{Id}^n_{\|A\|_n}(p,q)}$$

$$\frac{a : \|A\|_n \qquad x : \|A\|_n \vdash e(x) : \mathsf{is}\text{-}n\text{-}\mathsf{type}(B(x)) \qquad y : A \vdash b(y) : B(|y|_n)}{c(a,e,b) : B(a)}$$

$$\frac{a : \|A\|_n \qquad x : \|A\|_n \vdash e(x) : \mathsf{is}\text{-}n\text{-}\mathsf{type}(B(x)) \qquad y : A \vdash b(y) : B(|y|_n)}{c(|a|_n,e,b) = b(a) : B(|a|_n)}$$

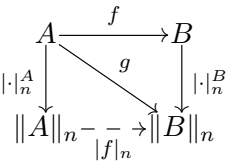
TABLE 1.1. Rules for truncations

We simply write $\|\cdot\|$ to refer to propositional truncation $\|\cdot\|_1$.

Functions between types induce functions between their truncations as detailed in the following lemma.

Lemma 1.1.4. *Every function $f : A \to B$ induces a function on the truncations $|f|_n : \|A\|_n \to \|B\|_n$ such that $|f|_n(|a|_n) = |f(a)|_n$ for all $a : A$.*

Proof. Given $f : A \to B$ consider the function $g := \lambda a.|f(a)|_n^B : A \to \|B\|_n$, where $|\cdot|_n^B : B \to \|B\|_n$ is the projection from B to its truncation. Since the type $\|B\|_n$ is an n-type, applying the elimination rule for $\|A\|_n$ gives the desired function $|f|_n$.



□

Remark 1.1.5. Note that the operators Π and $\|\cdot\|_n$ do not commute in general. [Uni13, Section 3.8, on the axiom of choice].

Now let us recall some basic properties of intensional identity types.

Lemma 1.1.6. *In $\mathsf{ML_1W}$, consider a family of types $B(x)$ for $x : A$. If there is a term $p : \mathsf{Id}_A(x, y)$, then we have a function called* transport *$p_* : B(x) \to B(y)$, which is an equivalence of types.*

Proof. See [Uni13, Lemma 2.3.1]. □

Lemma 1.1.7. *Given a dependent function $f : (\Pi x : A)B(x)$, it induces a function on the identity types:*

$$\mathsf{apd_f} : (\Pi p : \mathsf{Id}_A(x, y))\mathsf{Id}_{B(y)}(p_* f(x), f(y)).$$

In case of non-dependent function the statement simplifies to:

$$\mathsf{apd_f} : \mathsf{Id}_A(x, y) \to \mathsf{Id}_B(f(x), f(y)).$$

Proof. See [Uni13, Lemma 2.3.4]. □

The following lemma gives closure properties of homotopy levels.

Theorem 1.1.8. *In H we can prove the following properties of homotopy levels:*

(i) *the types $\mathsf{0}$ and $\mathsf{1}$ are hprops;*
(ii) *the types $\mathsf{2}$ and N are hsets;*
(iii) *given any type A and a family of n-types $B(x)$ for $x : A$, then $(\Pi x : A)B(x)$ is an n-type;*
(iv) *given an n-type A and a family of n-types $B(x)$ for $x : A$, then the Σ-type $(\Sigma x : A)B(x)$ is an n-type;*

(v) *given two n-types A and B, then $A \times B$ is an n-type;*
(vi) *given two $(n+2)$-types A and B, then $A + B$ is an $(n+2)$-type;*
(vii) *given an $(n+1)$-type A, and any family of types $B(x)$ for $x : A$, then $(Wx : A)B(x)$ is an $(n+1)$-type;*
(viii) *a k-type is also a $(k+1)$-type.*

Proof. $(i)-(ii)$ The cases of $\mathsf{0}, \mathsf{1}$ and $\mathsf{2}$ are straightforward. For the type of natural numbers N the claim follows from the characterisation of the identity type Id_{N}:

$$\mathsf{Id}_{\mathsf{N}}(0,0) \simeq \mathsf{1}$$
$$\mathsf{Id}_{\mathsf{N}}(\mathsf{s}(n),0) \simeq \mathsf{0}$$
$$\mathsf{Id}_{\mathsf{N}}(0,\mathsf{s}(m)) \simeq \mathsf{0}$$
$$\mathsf{Id}_{\mathsf{N}}(\mathsf{s}(n),\mathsf{s}(m)) \simeq \mathsf{Id}_{\mathsf{N}}(n,m)$$

See [Uni13, Section 2.13] for the proof.

(iii) Follows from function extensionality.

(iv) Follows from the fact that the identity type $\mathsf{Id}_{(\Sigma x:A)B(x)}(z,w)$ is equivalent to:

$$(\Sigma p : \mathsf{Id}_A(z.1, w.1))\mathsf{Id}_{B(w.2)}(p_*(z.2), w.2),$$

where p_* is transport, introduced in Lemma 1.1.6. See [Uni13, Theorem 2.7.2] for the proof.

$(v)-(vi)$ Follow from (iv) by recalling that $\times$ is a non-dependent Σ and that $A + B$ is equivalent to the Σ-type of A and B over $\mathsf{2}$.

(vii) Follows from $(iii), (iv)$ and the fact that the identity type

$$\mathsf{Id}_{(Wx:A)B(x)}(\mathsf{sup}(a,t), \mathsf{sup}(a',t')),$$

is equivalent to the type:

$$(\Sigma p : \mathsf{Id}_A(a,a'))(\Pi b : B(a))\mathsf{Id}_{(Wx:A)B(x)}(t(b), t'(p_*b)).$$

$(viii)$ It is a straightforward induction on k. □

Remark 1.1.9. The following examples show the need for the restrictions on the homotopy levels that appear in Theorem 1.1.8.

(a) There are two hpropositions A and B, such that the type $A + B$ is *not* a hprop. Indeed, take $A = B = \mathsf{1}$ which is a hprop, but $\mathsf{1} + \mathsf{1} \simeq \mathsf{2}$ which is a hset but not a hprop.
(b) There are contractible types A and $B(x)$ for $x : A$ such that $(Wx : A)B(x)$ is not contractible. Indeed, take $A = \mathsf{1}$ and $B(x) = \mathsf{1}$, then $(Wx : \mathsf{1})\mathsf{1} \simeq \mathsf{0}$ which is not contractible.

(c) There is a type A and a family of hprops over it $B(x)$ for $x : A$, such that the type $(\Sigma x : A)B(x)$ is *not* a hprop. Indeed, take $A = \mathsf{N}$ and $B(x) = \mathsf{1}$, then $(\Sigma x : \mathsf{N})\mathsf{1} \simeq \mathsf{N}$ which is a hset and not a hprop.

The theory H can be seen as a 'type theory for homotopy levels', since it has a well-behaved notion of homotopy levels thanks to the function extensionality axiom, and truncation operations to turn any type into a type of a given homotopy level.

An easy but useful fact is the following lemma that we recall from [Uni13], it is called the *principle of unique choice*.

Recall that we use $\|A\|$ as a shorthand for the propositional truncation $\|A\|_1$.

Lemma 1.1.10 (Principle of Unique Choice)**.** *In the type theory* H*, given a family of types* $x : A \vdash P(x) : \mathsf{type}$ *such that:*

(i) *for each* $x : A$ *the type* $P(x)$ *is a hproposition;*
(ii) *for each* $x : A$ *we have* $\|P(x)\|$*.*

Then we have $(\Pi x : A)P(x)$*.*

Proof. Immediate from the observation that if $P(x)$ is a hproposition, then there is an equivalence $P(x) \simeq \|P(x)\|$. □

The type theory HoTT. Here we define the type theory HoTT that is used to strengthen H for some of the interpretations $[\![\cdot]\!]_{k,h}$ of constructive set theories. It consists of the following:

- the type theory H;
- the univalence axiom UA (see Axiom 1.1.11);
- A/R set-quotients for families of hprops $x, y : A \vdash R(x, y) : \mathsf{type}$ (see page 20 in this section).

See the appendix B for the complete list of rules. This theory is the background theory for chapter 3.

Firstly, we recall the univalence axiom, which has been formulated by Voevodsky (see [Voe15]). Note that we have a canonical map:

$$\mathsf{idtoeqv} : \mathsf{Id}_{\mathsf{U}}(a, b) \to \mathsf{Equiv}(a, b).$$

Indeed, by the elimination rule of the identity type we can assume $\mathsf{refl}_a : \mathsf{Id}_{\mathsf{U}}(a, a)$, which we can map to the identity function 1_a, which is an equivalence.

Axiom 1.1.11. The *univalence axiom* asks that the canonical map $\mathsf{idtoeqv}$ is an equivalence. In other words that we have a term:

$$\mathsf{ua} : \mathsf{isEquiv}(\mathsf{idtoeqv}).$$

Adding the univalence axiom to Martin-Löf type theory allows to reconstruct in type theory a number of homotopical concepts, for example some calculations of homotopy groups of spheres. See [Uni13, part II] for more details.

Also recall that the univalence axiom plus the propositional η-rule imply the function extensionality axiom (see [Uni13, Section 4.9]).

The last ingredient of the type theory HoTT are *set-quotients*. Recall from [Uni13, Section 6.10] that we can introduce a new type constructor that given a type A, and a family of types $x : A, y : A \vdash R(x, y) : \mathsf{type}$ such that $\mathsf{isHprop}(R(x, y))$ for all $x, y : A$, forms the type A/R given by:

(i) a function $q : A \to A/R$;
(ii) for each $x, y : A$ such that $R(x, y)$ we have a term in the identity type $\mathsf{Id}_{A/R}(q(x), q(y))$;
(iii) for all $x, y : A/R$ and $r, s : \mathsf{Id}_{A/R}(x, y)$ we have a term in $\mathsf{Id}_{\mathsf{Id}_{A/R}}(r, s)$.

It is possible to write down explicit rules for set-quotients that follow the pattern of introduction, elimination and computation common to the other higher inductive types. In particular the elimination term $\mathsf{R}_{A/R} : (\Pi x : A/R)B(x)$ satisfies a definitional computation rule when applied to points $x : A/R$, and a propositional computation rule when applied to paths in $\mathsf{Id}_{A/R}(x, y)$.

We omit these details since we use the elimination and computation rules only via the universal property detailed in Lemma 1.1.13.

We now recall a lemma that is useful in proving some properties of the quotient A/R, including its universal property.

Lemma 1.1.12. *The function q is surjective, i.e.* $(\Pi x : A/R)\left\|(\Sigma \alpha : A)\mathsf{Id}_{A/R}(x, q(\alpha))\right\|$.

Proof. See [Uni13, Lemma 6.10.2]. □

The set-quotient satisfies the following universal property.

Lemma 1.1.13. *For any hset B, precomposing with q gives an equivalence:*

$$(A/R \to B) \simeq (\Sigma\, f : A \to B)(\Pi x, y : A)[\, R(x, y) \to \mathsf{Id}_B(f(x), f(y))\,]$$

Proof. See [Uni13, Lemma 6.10.3]. □

1.2. The homotopy type-theoretic interpretations

Here we define a hierarchy of interpretations $[\![\cdot]\!]_{k,h}$ of the language of set theory into the type theory H.

Firstly, we define the types V_k that interpret the universe of sets, so that sets are interpreted as terms of V_k. Aczel's interpretation of CZF into Martin-Löf type theory uses the W-type of all small types over the universe as a type of sets, i.e. $(Wx : \mathsf{U})\mathsf{T}(x)$. Here we define in a similar way types V_k for $2 \leq k \leq \infty$ by using all small types of fixed homotopy level k.

Definition 1.2.1.

$$k\text{-}\mathsf{Types}_{\mathsf{U}} := (\Sigma x : \mathsf{U})\mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(x))$$
$$\infty\text{-}\mathsf{Types}_{\mathsf{U}} := \mathsf{U}$$

Then for any $2 \leq k \leq \infty$ we form the W-type over k-$\mathsf{Types}_{\mathsf{U}}$:

$$\mathsf{V}_k := (Wy : k\text{-}\mathsf{Types}_{\mathsf{U}})\mathsf{T}(y.1)$$

The explicit rules for these types V_k are special cases of the rules for W-types and Σ-types for the specific type family $x : k\text{-}\mathsf{Types}_{\mathsf{U}} \vdash \mathsf{T}(x) : \mathsf{type}$. The elimination and the computation rules for this type simply express induction on V_k.

$$\frac{a : k\text{-}\mathsf{Types}_{\mathsf{U}} \qquad b : \mathsf{T}(a.1) \to \mathsf{V}_k}{\mathsf{sup}(a,b) : \mathsf{V}_k}$$

$$\frac{\begin{array}{c} c : \mathsf{V}_k \\ x : k\text{-}\mathsf{Types}_{\mathsf{U}}, y : \mathsf{T}(x.1) \to \mathsf{V}_k, z : (\Pi v : \mathsf{T}(x.1))C(y(v)) \vdash d : C(\mathsf{sup}(x,y)) \end{array}}{\mathsf{R}(c, d(x,y,z)) : C(c)}$$

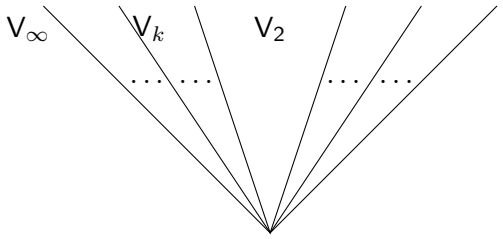
$$\frac{\begin{array}{c} a : k\text{-}\mathsf{Types}_{\mathsf{U}} \qquad f : \mathsf{T}(a.1) \to \mathsf{V}_k \\ x : k\text{-}\mathsf{Types}_{\mathsf{U}}, y : \mathsf{T}(x.1) \to \mathsf{V}_k, z : (\Pi v : \mathsf{T}(x.1))C(y(v)) \vdash d : C(\mathsf{sup}(x,y)) \end{array}}{\mathsf{R}(\mathsf{sup}(a,f), d(x,y,z)) = d(a, f, \lambda v.\mathsf{R}(f(v), d(x,y,z))) : C(c)}$$

TABLE 1.2. Rules for the types of sets V_k

The terms of the type V_k are well-founded trees $\mathsf{sup}((a,p),f)$ such that $\mathsf{T}(a)$ is a k-type. The idea is that only types with a certain level of homotopical information can be used to form its terms, so that the resulting V_k has fewer terms for smaller values of k.

For example, V_2 does not have terms constructed from the fundamental groupoid of a type, defined as $\Pi_1(X, x, y) := \|\mathsf{Id}_X(x, y)\|_3$.

Since any k-type is also a $(k+1)$-type thanks to Theorem 1.1.8, there are canonical embedding maps of V_k into $\mathsf{V}_{k'}$ for each $k \leq k'$. Hence, we can represent these V_k pictorically as:



Here V_2 is the type constructed from homotopy sets, and V_∞ is Aczel's type, in which terms are constructed from arbitrary small types.

Lemma 1.2.2. *There is a function* $\mathsf{V}_k \to (\Sigma t : k\text{-}\mathsf{Types}_\mathsf{U})(\mathsf{T}(t.1) \to \mathsf{V}_k)$ *assigning to every* $\alpha : \mathsf{V}_k$ *the following terms: a small type* $\mathsf{el}(\alpha) : \mathsf{U}$*, a term witnessing its homotopy level* $\mathsf{p}_\alpha : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{el}(\alpha))$ *and the function* $\alpha_\_ : \mathsf{T}(\mathsf{el}(\alpha)) \to \mathsf{V}_k$*.*

They are such that given $\mathsf{sup}((a, p), f)$ *with* $(a, p) : k\text{-}\mathsf{Types}_\mathsf{U}$ *and* $f : \mathsf{T}(a) \to \mathsf{V}_k$*, we have* $\mathsf{el}(\mathsf{sup}((a, p), f)) = a$*, and* $\mathsf{p}_{\mathsf{sup}((\mathsf{a},\mathsf{p}),\mathsf{f})} = p$*, and* $\mathsf{sup}((a, p), f)_x = f(x)$*.*

Proof. By recursion on V_k, for $\alpha = \mathsf{sup}((a, p), f)$ we can define $\mathsf{el}(\mathsf{sup}((a, p), f)) = a$, and $\mathsf{p}_{\mathsf{sup}((\mathsf{a},\mathsf{p}),\mathsf{f})} = p$ and $\mathsf{sup}(a, f)_x = f(x)$. □

For a given $\alpha : \mathsf{V}_k$ we introduce the notation $\mathsf{El}(\alpha) := \mathsf{T}(\mathsf{el}(\alpha))$.

In order to make the interpretation of variables simpler we consider the following extension of the language of set theory. Let $\mathcal{L}$ be the language of set theory and $\mathcal{L}_{\mathsf{V}_k}$ the language obtained by adding to $\mathcal{L}$ a constant for each term $\alpha : \mathsf{V}_k$.

Now we define the interpretation of set-theoretic equality as a bisimulation relation given by Π and the h-th truncation of Σ. The reason for this definition is that we want to follow the structure of Aczel's interpretation as closely as possible, but we also want to interpret formulas as types of homotopy level h, hence the need of truncations.

Given two sets $\alpha, \beta : \mathsf{V}_k$ we define the type $\alpha \approx_h \beta$ as follows:

$$(\Pi x : \mathsf{El}(\alpha))\left\|(\Sigma y : \mathsf{El}(\beta))(\alpha_x \approx_h \beta_y)\right\|_h \times (\Pi y : \mathsf{El}(\beta))\left\|(\Sigma x : \mathsf{El}(\alpha))(\alpha_x \approx_h \beta_y)\right\|_h$$

Note that by convention we consider $\|\cdot\|_\infty$ to be the identity operator.

Finally, we give the interpretation of the other set-theoretic formulas. Formulas are interpreted as types following the propositions-as-types correspondence, but any

type which may not be a h-type is truncated with $\| \cdot \|_h$.

$$\begin{aligned}
[\![\alpha \doteq \beta]\!]_{k,h} &:= (\alpha \approx_h \beta) \\
[\![\bot]\!]_{k,h} &:= \mathsf{0} \\
[\![\phi \Rightarrow \psi]\!]_{k,h} &:= [\![\phi]\!]_{k,h} \to [\![\psi]\!]_{k,h} \\
[\![\phi \wedge \psi]\!]_{k,h} &:= [\![\phi]\!]_{k,h} \times [\![\psi]\!]_{k,h} \\
[\![\phi \vee \psi]\!]_{k,h} &:= \left\| [\![\phi]\!]_{k,h} + [\![\psi]\!]_{k,h} \right\|_h \\
[\![(\forall x \in \alpha)\phi(x)]\!]_{k,h} &:= (\Pi x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h} \\
[\![(\exists x \in \alpha)\phi(x)]\!]_{k,h} &:= \left\| (\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h} \right\|_h \\
[\![\forall x\, \phi(x)]\!]_{k,h} &:= (\Pi \alpha : \mathsf{V}_k)[\![\phi(\alpha)]\!]_{k,h} \\
[\![\exists x\, \phi(x)]\!]_{k,h} &:= \left\| (\Sigma \alpha : \mathsf{V}_k)[\![\phi(\alpha)]\!]_{k,h} \right\|_h \\
[\![\alpha \in \beta]\!]_{k,h} &:= \left\| (\Sigma y : \mathsf{El}(\beta))(\alpha \approx_h \beta_y) \right\|_h
\end{aligned}$$

Note that the interpretations of all bounded formulas are k-types, due to the closure properties of k-types. All formulas, including unbounded ones, are interpreted as h-types thanks to the truncation operations.

Definition 1.2.3. We say that a set-theoretic formula $\phi(x_1, \ldots, x_n)$ is *valid* in the type-theoretic interpretation if and only if the type of the interpretation of its universal closure $[\![\forall x_1, \ldots, x_n\, \phi(x_1, \ldots, x_n)]\!]_{k,h}$ is inhabited.

Remark 1.2.4. Note that for certain values of k and h truncations are not necessary:

(i) for $h = \infty$, no truncation is performed;
(ii) for $h > 1$, the truncation for the $+$-type is unnecessary because $(n+1)$-types are closed under $+$, (see Theorem 1.1.8);
(iii) for $h \geq k$, the truncations in the definitions of $[\![(\exists x \in \alpha)\phi(x)]\!]_{k,h}$ and $[\![\alpha \in \beta]\!]_{k,h}$ are unnecessary. Indeed, in these cases $\mathsf{El}(\beta)$ is a k-type, hence a h-type, and h-types are closed under Σ (see Theorem 1.1.8).

1.3. Interpreting constructive set theories

In this section we investigate which axioms have a valid $[\![\cdot]\!]_{k,h}$ interpretation into the type theories H and HoTT. Depending on the values of the parameters k, h we are able to validate different axioms of constructive set theories.

Our proof follows the structure of Aczel's proof of the interpretation $[\![\cdot]\!]_{\infty,\infty}$ from [Acz78]. The idea of Aczel's interpretation is that by interpreting sets as well-founded trees in V_∞ one can use recursion on V_∞ to prove the $\in$-Induction axiom. Extensionality follows from the interpretation of set-theoretic equality as the bisimulation relation $\approx_\infty$. The validity of Pairing, Union, Bounded Separation and Infinity follows from the type-theoretic rules of $+$, Σ and the types $\mathsf{0}$, $\mathsf{1}$, $\mathsf{2}$ and N. The Collection axioms are validated using the type-theoretic axiom of choice, in particular for the Subset Collection axiom function types are used.

In our context we construct sets using k-types and interpret formulas using h-types. Therefore we need to keep track of the homotopy levels involved in the constructions and rework Aczel's proof accordingly.

Note that in our notation the base types and type constructors $\mathsf{0}$, $\mathsf{1}$, $\mathsf{2}$, N, Π, $\to$, Σ, $\times$, $+$, W, Id_A are represented in the universe using the corresponding terms and term constructors: $0, 1, 2, \mathsf{n}, \pi, \mathsf{exp}, \sigma, \mathsf{times}, \mathsf{plus}, \mathsf{w}, \mathsf{i}_a$. For more details on the type-theoretic rules see appendix B

Some lemmas on equality and formulas. We start with some lemmas that take care of basic properties of equality and set-theoretic formulas.

Lemma 1.3.1. *The interpretation of set-theoretical equality is an equivalence relation, i.e. for all $\alpha, \beta, \gamma : \mathsf{V}_k$ the following formulas are valid:*

(i) $\alpha \doteq \alpha$;
(ii) $\alpha \doteq \beta \Rightarrow \beta \doteq \alpha$;
(iii) $\alpha \doteq \beta \wedge \beta \doteq \gamma \Rightarrow \alpha \doteq \gamma$.

Proof. (i) We apply the elimination rule of V_k to the family of types $C(\alpha) := [\![\alpha \doteq \alpha]\!]_{k,h}$ so that we can assume $\alpha = \mathsf{sup}((a, v), f)$, with $a : \mathsf{U}$ and $v : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(a))$.
We want to find a term in the type:

$$(\Pi x : \mathsf{T}(a))\Big\|(\Sigma y : \mathsf{T}(a))(f(x) \approx_h f(y))\Big\|_h \times (\Pi y : \mathsf{T}(a))\Big\|(\Sigma x : \mathsf{T}(a))(f(x) \approx_h f(y))\Big\|_h.$$

By inductive hypothesis we have $d(x) : (f(x) \approx_h f(x))$, then the desired term is $(\lambda x.|(x, d(x))|_h, \lambda x.|(x, d(x))|_h)$.

(ii) Similarly, we first reduce to the case $\alpha = \mathsf{sup}((a, v), f)$ and $\beta = \mathsf{sup}((b, w), g)$ by recursion on V_k. Then, given $p : (\mathsf{sup}((a, v), f) \approx_h \mathsf{sup}((b, w), g))$, we want to construct a term in the type $(\mathsf{sup}((b, w), g) \approx_h \mathsf{sup}((a, v), f))$.
By inductive hypothesis we have a term

$$F : (f(x) \approx_h g(y)) \to (g(y) \approx_h f(x)),$$

so we can consider the term $p.1(x) : \left\|(\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y))\right\|_h$. Now observe that we have a function $G := \lambda t.(t.1, F(t.2))$ that works for the untruncated types:

$$G : (\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y)) \to (\Sigma y : \mathsf{T}(b))(g(y) \approx_h f(x)),$$

which induces a function $|G|$ with truncated domain and codomain by Lemma 1.1.4. Therefore we get a term:

$$\lambda x.|G|(p.1(x)) : (\Pi x : \mathsf{T}(a))\left\|(\Sigma y : \mathsf{T}(b))(g(y) \approx_h f(x))\right\|_h.$$

We argue similarly for the second component $p.2$, and pairing the terms obtained from the two components we obtain the thesis.

(iii) As before we proceed by recursion on V_k, so we consider terms $\alpha = \mathsf{sup}((a, v), f)$, and $\beta = \mathsf{sup}((b, w), g)$ and $\gamma = \mathsf{sup}((c, u), h)$.
Given $x : \mathsf{T}(a)$, let us consider the following types:

$$A(x) := \left\|(\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y))\right\|_h,$$
$$B := (\Pi y : \mathsf{T}(b))\left\|(\Sigma z : \mathsf{T}(c))(g(y) \approx_h h(z))\right\|_h,$$
$$C(x) := \left\|(\Sigma z : \mathsf{T}(c))(f(x) \approx_h h(z))\right\|_h.$$

By reasoning separately on the two conjuncts of the interpretation of equality our goal reduces to showing that assuming given a term in:

$$(\Pi x : \mathsf{T}(a))A(x) \times B,$$

we can construct a dependent function inhabiting the type:

$$E := (\Pi x : \mathsf{T}(a))C(x).$$

So given $x : \mathsf{T}(a)$, it suffices to find a function of type:

$$A(x) \times B \to C(x).$$

To find a term in $A(x) \times B \to C(x)$ is equivalent to find a term inhabiting $A(x) \to (B \to C(x))$, which is in turn equivalent to find one in:

$$(\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y)) \to (B \to C(x)), \tag{1.1}$$

by the elimination rule of truncation applied to $A(x)$.
To do so notice that by inductive hypothesis we have a function:

$$F : (f(x) \approx_h g(y)) \times (g(y) \approx_h h(z)) \to (f(x) \approx_h h(z)),$$

which we can use together with $p : (\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y))$ to construct the function $G(p) := \lambda t.(t.1, F(p.2, t.2))$ inhabiting the following type:

$$(\Sigma z : \mathsf{T}(c))(g(y) \approx_h h(z)) \to (\Sigma z : \mathsf{T}(c))(f(x) \approx_h h(z)).$$

The function $G(p)$ induces a function $|G(p)|$ with truncated domain and codomain by Lemma 1.1.4. Therefore

$$\lambda p.\lambda f.|G(p)|(f(p.1)) : (\Sigma y : \mathsf{T}(b))(f(x) \approx_h g(y)) \to (B \to C(x)),$$

gives us the required term inhabiting the type in (1.1).

□

Lemma 1.3.2. *If $\phi_1, \dots \phi_n \vdash \phi$ is derivable in intuitionistic logic and $\phi_1, \dots \phi_n$ are valid in the interpretation then so is ϕ.*

Proof. Straightforward. The only points that need care are the rules for $\vee$ and $\exists$ which follow from the definition of the interpretation $[\![\cdot]\!]_{k,h}$ and the rules for truncation. Observe that given the requirement that we eliminate into a h-type in the elimination rule for $\|\cdot\|_h$ (see table1.1), it is important that in the definition of $[\![\cdot]\!]_{k,h}$ we ensure that all formulas are h-types, including unbounded ones. □

Definition 1.3.3. A family of types $B(x)$ over V_k *satisfies the Leibniz rule with respect to* $\approx_h$ if and only if the following type is inhabited:

$$(\Pi x, y : \mathsf{V}_k)((x \approx_h y) \times B(x) \to B(y)).$$

Lemma 1.3.4. *For every formula $\phi(x_1, \dots, x_n)$ of the language $\mathcal{L}_{\mathsf{V}_k}$, the family $[\![\phi(x_1, \dots, x_n)]\!]_{k,h}$ satisfies in every variable the Leibniz rule with respect to $\approx_h$.*

Proof. The proof proceeds by induction on the structure of the formula. First of all consider the case of atomic formulas: $(\alpha \approx_h \beta)$ satisfies the Leibniz rule because of transitivity that we proved in Lemma 1.3.1.

Next we consider the formula $\beta \in \gamma$, we want to prove that it satisfies the Leibniz rule in the first variable, i.e. construct a function of type:

$$(\alpha \approx_h \beta) \times \left\| (\Sigma x : \mathsf{El}(\gamma))(\gamma_x \approx_h \beta) \right\|_h \to \left\| (\Sigma x : \mathsf{El}(\gamma))(\gamma_x \approx_h \alpha) \right\|_h.$$

So suppose we have a term $t : (\alpha \approx_h \beta)$, and recall from Lemma 1.3.1 that we have a term Tr witnessing transitivity, which we use to construct the function $F(t) := \lambda p.(p.1, \mathsf{Tr}(p.2, t))$ inhabiting the following type, where Σ-types are not truncated:

$$(\Sigma x : \mathsf{El}(\gamma))(\gamma_x \approx_h \beta) \to (\Sigma x : \mathsf{El}(\gamma))(\gamma_x \approx_h \alpha).$$

The function F induces a function $|F(t)| : [\![\beta \in \gamma]\!]_{k,h} \to [\![\alpha \in \gamma]\!]_{k,h}$. We conclude by lambda-abstraction on t.

Regarding the Leibniz rule in the other variable, we want to construct a function inhabiting the type:

$$(\gamma \approx_h \delta) \times [\![\alpha \in \gamma]\!]_{k,h} \to [\![\alpha \in \delta]\!]_{k,h},$$

which we can rewrite as follows by swapping the types in the domain, currying, and recalling the definition of $[\![\alpha \in \gamma]\!]_{k,h}$:

$$\left\| (\Sigma x : \mathsf{El}(\gamma))(\alpha \approx_h \gamma_x) \right\|_h \to ((\gamma \approx_h \delta) \to [\![\alpha \in \delta]\!]_{k,h}).$$

Since the codomain $((\gamma \approx_h \delta) \to [\![\alpha \in \delta]\!]_{k,h})$ is a h-type, by the elimination rule of truncation it is enough to construct an inhabitant of:

$$(\Sigma x : \mathsf{El}(\gamma))(\alpha \approx_h \gamma_x) \to ((\gamma \approx_h \delta) \to [\![\alpha \in \delta]\!]_{k,h}).$$

Assuming only the first conjunct of the definition of the equality $(\gamma \approx_h \delta)$ is enough, so at this point our goal is to construct an inhabitant of the type:

(1.2)

$$(\Sigma x : \mathsf{El}(\gamma))(\alpha \approx_h \gamma_x) \to ((\Pi x : \mathsf{El}(\gamma)) \left\| (\Sigma y : \mathsf{El}(\delta))(\gamma_x \approx_h \delta_y) \right\|_h \to [\![\alpha \in \delta]\!]_{k,h}).$$

Its construction is similar to the one for transitivity in Lemma 1.3.1. Suppose to have a term in the domain $p : (\Sigma x : \mathsf{El}(\gamma))(\alpha \approx_h \gamma_x)$, we use it to construct the function $G(p)$ defined as $\lambda t : (t.1, \mathsf{Tr}(p.2, t.2))$, where Tr is the term witnessing transitivity of equality. The function $G(p)$ inhabits the type:

$$G(p) : (\Sigma y : \mathsf{El}(\delta))(\gamma_x \approx_h \delta_y) \to (\Sigma y : \mathsf{El}(\delta))(\delta_y \approx_h \alpha).$$

By Lemma 1.1.4, $G(p)$ induces a function $|G(p)|$ with truncated domain and codomain, and finally: $\lambda p.\lambda f.|G(p)|(f(p.1))$ inhabits the type in (1.2).

The inductive steps for the non-atomic formulas are straightforward. □

Lemma 1.3.5. *The structural defining axioms for the bounded quantifiers are valid, i.e. the types*

$$[\![(\forall x \in y)\phi(x) \Leftrightarrow \forall x(x \in y \Rightarrow \phi(x))]\!]_{k,h},$$

and

$$[\![(\exists x \in y)\phi(x) \Leftrightarrow \exists x(x \in y \wedge \phi(x))]\!]_{k,h},$$

are inhabited.

Proof.

We have that $[\![\phi(x)]\!]_{k,h}$ satisfies the Leibniz rule in x by Lemma 1.3.4. Let

$$\mathsf{Leib} : (\Pi x, y : \mathsf{V}_k)((x \approx_h y) \times [\![\phi(x)]\!]_{k,h} \to [\![\phi(y)]\!]_{k,h})$$

be the term constructed in that proof.

(∃) For the existential quantifier, firstly we want to prove that there is a type-theoretic function with domain:

$$\Big\|(\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}\Big\|_h,$$

and codomain:

$$\Big\|(\Sigma z : \mathsf{V}_k)\Big(\Big\|(\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y)\Big\|_h \times [\![\phi(z)]\!]_{k,h}\Big)\Big\|_h.$$

Thanks to Lemma 1.1.4 it is enough to construct a function with untruncated domain and codomain.

Given $p : (\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}$ the term that we want to construct is a pair where the first component has to be in V_k, so we choose $\alpha_{p.1}$ to inhabit that type. The second component of the pair has to inhabit the cartesian product:

$$\Big\|(\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y)\Big\|_h \times [\![\phi(z)]\!]_{k,h}.$$

For the first component of the cartesian product we use the term $|(p.1, \mathsf{r}(\alpha_{p.1}))|_h$, where $\mathsf{r}(\alpha_{p.1})$ is the term witnessing the reflexivity of $(\alpha_{p.1} \approx_h \alpha_{p.1})$ constructed in Lemma 1.3.1. Finally, for the second component of the cartesian product we choose $\mathsf{Leib}(\mathsf{r}(\alpha_{p.1}), p.2)$. It is straightforward to check that these terms are well-typed and give the desired function.

Now we want to prove the converse, i.e. that there is a type-theoretic function with domain:

$$\Big\|(\Sigma z : \mathsf{V}_k)\Big(\Big\|(\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y)\Big\|_h \times [\![\phi(z)]\!]_{k,h}\Big)\Big\|_h,$$

and codomain:

$$\Big\|(\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}\Big\|_h.$$

Since the codomain is a h-type, it is enough to show that there is a function G with untruncated domain. In order to construct G we assume given $t : [\![\phi(z)]\!]_{k,h}$ and consider the function $F(t)$ defined as $\lambda p.(p.1, \mathsf{ext}(p.2, t))$, which inhabits the type:

$$F(t) : (\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y) \to (\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}.$$

The function $F(t)$ induces a function $|F(t)|_h$ with truncated domain and codomain. Then $G := \lambda s.|F(s.2.2)|_h(s.2.1)$ is the desired term.

(∀) For the universal quantifier we first want to prove that there is a type-theoretic function with domain:

$$(\Pi x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h},$$

and codomain:

$$(\Pi z : \mathsf{V}_k)\Big(\Big\|(\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y)\Big\|_h \to [\![\phi(z)]\!]_{k,h}\Big).$$

So given $f : (\Pi x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}$ and $z : \mathsf{V}_k$ we want to construct a term $F(z,f) : \Big\|(\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y)\Big\|_h \to [\![\phi(z)]\!]_{k,h}$ and conclude by λ-abstraction on f and z. Since $[\![\phi(z)]\!]_{k,h}$ is a h-type, the function $F(z,f)$ is induced by the term:

$$\lambda p.\mathsf{Leib}(p.2, f(p.1)) : (\Sigma y : \mathsf{El}(\alpha))(z \approx_h \alpha_y) \to [\![\phi(z)]\!]_{k,h}.$$

The other direction is straightforward.

□

Note that the proofs of Lemma 1.3.1, Lemma 1.3.4 and Lemma 1.3.5 simplify considerably for $h \geq k$ since in that case truncations of $\vee$ and bounded $\exists$ are unnecessary. Even more for $h = \infty$, since no truncation takes place and the proofs of the corresponding lemmas of [Acz78] carry over to $[\![\cdot]\!]_{k,\infty}$.

Now let us gather one last preliminary remark before moving to the validity of set-theoretic axioms.

Remark 1.3.6. The formulas stating that any set is a subset of itself $\alpha \subseteq \alpha$, is valid in the interpretation $[\![\cdot]\!]_{k,h}$. We use this fact in Proposition 1.3.7.

Indeed, for every set α there is a term $\alpha^* : [\![(\forall x \in \alpha)(x \in \alpha)]\!]_{k,h}$. It is enough to define $\alpha^* := \lambda x.|(x, \mathsf{r}(\alpha_x))|_h$, where $\mathsf{r}(\beta) : (\beta \approx_h \beta)$ witnesses reflexivity.

Validity of the set-theoretic axioms. We can now prove the validity of some of the axioms of set theory in the interpretations $[\![\cdot]\!]_{k,h}$. Recall from appendix A that BCS (Basic Constructive Set theory) is the constructive set theory given by following axioms: Extensionality, Set-Induction, Pairing, Union, Bounded Separation and Infinity.

Proposition 1.3.7. *The axioms of* BCS *are valid in the interpretation* $[\![\cdot]\!]_{k,h}$.

Proof.

(i) For Extensionality and Set Induction no changes are required with respect to the proof of the original interpretation [Acz78]. However, note that for the proof of Extensionality we use Lemma 1.3.1, Lemma 1.3.4 and Lemma 1.3.5, which needed to be reproved for the interpretation $[\![\cdot]\!]_{k,h}$ so the actual work needed to establish Extensionality for $[\![\cdot]\!]_{k,h}$ lies in the proof of these three lemmas.

(ii) Pairing: given sets $\alpha, \beta : \mathsf{V}_k$, we know from Theorem 1.1.8 that 2 is a hset, hence it is a k-type for all $k \geq 2$. Let $p : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(2)$ be the term witnessing this fact. We now define $g : 2 \to \mathsf{V}_k$ using the elimination rule for 2 as $g(1) := \alpha$ and $g(2) := \beta$. Then we define the pair as $\gamma := \mathsf{sup}((n_2, p), g)$.

Then $|(\gamma, (\gamma^*(1), \gamma^*(2)))|_h : [\![\exists z(\alpha \in z \wedge \beta \in z)]\!]_{k,h}$.

(iii) Union: given the set $\alpha : \mathsf{V}_k$ let us consider the type $(\Sigma x : \mathsf{El}(\alpha))\mathsf{El}(\alpha_x)$ which is a small type since it is definitionally equal to $\mathsf{T}(\sigma)$ where $\sigma := \sigma(\mathsf{el}(\alpha), \mathsf{el}(\alpha_x))$. We know by hypothesis that $\mathsf{El}(\alpha)$ and all $\mathsf{El}(\alpha_x)$ are k-types. By Theorem 1.1.8 there is a term $p : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(\sigma))$.

Consider the function $g : (\Sigma x : \mathsf{El}(\alpha))\mathsf{El}(\alpha_x) \to \mathsf{V}_k$ given by the elimination rule for Σ as $g((x, y)) := (\alpha_x)_y$ for $x : \mathsf{El}(\alpha)$ and $y : \mathsf{El}(\alpha_x)$. Define the union set as $\gamma := \mathsf{sup}((\sigma, p), g)$.

Then the term $|(\gamma, \lambda y.\lambda x.\gamma^*((x, y)))|_h$ which inhabits the type:

$$[\![\exists z(\forall x \in \alpha)(\forall y \in x)(y \in z)]\!]_{k,h},$$

shows that Union is valid.

(iv) Bounded Separation: given a bounded formula $\phi(x) \in \mathcal{L}_{\mathsf{V}_k}$ with at most x free, we want to show that the instance of the Bounded Separation axiom for ϕ

$$\exists\gamma[(\forall x \in \gamma)(x \in \alpha \wedge \phi(x)) \wedge (\forall x \in \alpha)(\phi(x) \Rightarrow x \in \gamma)],$$

is valid.

Given a set $\alpha : \mathsf{V}_k$ we form the type $A := (\Sigma x : \mathsf{El}(\alpha))[\![\phi(\alpha_x)]\!]_{k,h}$. Given our definition of $[\![\cdot]\!]_{k,h}$ and the fact that $\phi(x)$ is bounded, we have that A is a small k-type, even when $h \geq k$. Namely, $A = \mathsf{T}(\sigma)$ for some $\sigma : \mathsf{U}$, and we have a term $p : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(\sigma))$ witnessing the fact that $\mathsf{T}(\sigma)$ is a k-type.

Next we define $g : A \to \mathsf{V}_k$ using the elimination rule for Σ, as follows: $g((x, v)) := \alpha_x$, where $x : \mathsf{El}(\alpha)$ and $v : [\![\phi(\alpha_x)]\!]_{k,h}$. The set obtained by separation is $\gamma := \mathsf{sup}((\sigma, p), g)$.

Then we have the following terms:

$$t_1 := \lambda t.(\alpha^*(t), t.2) : [\![(\forall x \in \gamma)(x \in \alpha \wedge \phi(x))]\!]_{k,h},$$

and

$$t_2 := \lambda t.\lambda v.\gamma^*((t, v)) : [\![(\forall x \in \alpha)(\phi(x) \Rightarrow x \in \gamma)]\!]_{k,h}.$$

Hence $|(\gamma, (t_1, t_2))|_h$ is a term witnessing the validity of the instance of the Bounded Separation axiom for ϕ.

(v) Infinity is also straightforward but longer, the key point is to use the fact that $\mathsf{0}, \mathsf{1}, \mathsf{2}$ and N are hsets, and hence k-types for all $2 \leq k \leq \infty$, and to apply the constructor $|\cdot|_h$ appropriately during the construction of the terms.

First of all we construct internally in V_k the sets corresponding to the empty-set, to the successor $\alpha \cup \{\alpha\}$, and to the set of natural numbers. We know by Theorem 1.1.8 that $\mathsf{0}$, $\mathsf{1}$ and N are small k-types so that there are terms $p_0 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{0})$, $p_1 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{1})$ and $p_2 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{N})$ witnessing these facts. Since they are small types we also have $\mathsf{0} = \mathsf{T}(\mathsf{n_0})$, $\mathsf{1} = \mathsf{T}(\mathsf{n_1})$ and $\mathsf{N} = \mathsf{T}(\mathsf{n})$. Now let $f_0 : \mathsf{0} \to \mathsf{V}_k$ be the function given by the elimination rule for $\mathsf{0}$, and define $\emptyset := \mathsf{sup}((\mathsf{n_0}, p_0), f_0)$.

Then given a set $\alpha : \mathsf{V}_k$ consider the small type $A := \mathsf{El}(\alpha) + \mathsf{1}$ which is definitionally equal to $\mathsf{T}(\mathsf{plus}(\mathsf{el}(\alpha), n_1))$. Since by hypothesis $\mathsf{El}(\alpha)$ is a k-type, then by Theorem 1.1.8 A is also a k-type and we have a term $p_3 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(A)$ witnessing this fact. Then we define the function $h(\alpha) : \mathsf{El}(\alpha) + \mathsf{1} \to \mathsf{V}_k$ using the elimination rule for $+$ as $h(\alpha)(\mathsf{i}(x)) := \alpha_x$ and $h(\alpha)(\mathsf{j}(1)) := \alpha$. Finally we define:

$$S(\alpha) := \mathsf{sup}((\mathsf{plus}(\mathsf{el}(\alpha), \mathsf{n_1}), p_3), h(\alpha)).$$

For the set of natural numbers we consider $\Delta : \mathsf{N} \to \mathsf{V}_k$ defined using the elimination rule for N as $\Delta(0) := \emptyset$ and $\Delta(\mathsf{s}(n)) := S(\Delta(n))$. Then we define the term $\omega := \mathsf{sup}((\mathsf{n}, p_2), \Delta)$ in V_k that interprets the set of natural numbers.

Now recall that $Zero(\alpha)$ is the formula $(\forall x \in \alpha)\bot$, and $Succ(\alpha, \beta)$ is the following formula:

$$(\forall x \in \alpha)(x \in \beta) \wedge (\alpha \in \beta) \wedge (\forall x \in \beta)(x \in \alpha \vee x \doteq \alpha).$$

Firstly, we want to show that the following formulas are valid: $Zero(\emptyset)$ and $Succ(\alpha, S(\alpha))$. Observe that the canonical function $g_0 : \mathsf{0} \to \mathsf{0}$ provides a witness for $[\![Zero(\emptyset)]\!]_{k,h}$, then from g_0 we can construct the term $|(0, g_0)|_h : [\![(\exists x \in \omega)Zero(x)]\!]_{k,h}$.
For the other formula, we have a term inhabiting the following type:

$$g_1(\alpha) := \lambda x.S(\alpha)^*(\mathsf{i}(x)) : [\![(\forall x \in \alpha)(x \in S(\alpha))]\!]_{k,h}.$$

Moreover, given $x : \mathsf{El}(\alpha)$ we define $g_2(\alpha)(\mathsf{i}(x)) := |\mathsf{i}(\alpha^*(x))|_h$, and $g_2(\alpha)(\mathsf{j}(1)) := |\mathsf{j}(\mathsf{r_0}(\alpha))|_h$, where $\mathsf{r_0}(\alpha)$ witnesses the reflexivity of equality. So that we have:

$$g_2(\alpha) : [\![(\forall x \in S(\alpha))(x \in \alpha \vee x \doteq \alpha)]\!]_{k,h}.$$

Then it follows that:

$$g(\alpha) := (g_1(\alpha), S(\alpha)^*(\mathsf{j}(1)), g_2(\alpha)) : [\![Succ(\alpha, S(\alpha))]\!]_{k,h}.$$

At this point the only things left to check in order to validate the Axiom of Infinity are the validity of the formulas:

$$(\forall x \in \omega)(Zero(x) \vee (\exists y \in x)Succ(y, x)),$$

and

$$(\forall y \in \omega)(\exists x \in \omega)Succ(y, x).$$

The first is witnessed by the term defined by

$$f(0) := |\mathsf{i}(g_0)|_h,$$

and

$$f(\mathsf{s}(n)) := |\mathsf{j}(|(\mathsf{j}(1), g(\Delta(n)))|_h)|_h.$$

Whereas the second is witnessed by the term $h := \lambda y.|(\mathsf{s}(y), g(\Delta(y)))|_h$.

□

When dealing with the Collection axioms of CZF we use the notation $\forall\exists\frac{x\in\alpha}{y\in\beta}\phi(x, y)$ to abbreviate the formula:

$$(\forall x \in \alpha)(\exists y \in \beta)\phi(x, y) \wedge (\forall y \in \beta)(\exists x \in \alpha)\phi(x, y).$$

Before moving to the validity of the Collection axioms of CZF we state this preliminary remark that is used in the proofs of Lemma 1.3.9 and Lemma 1.3.10.

Remark 1.3.8. In the proofs of validity of the Collection axioms we use the following construction from [Acz78].

Consider a formula $\phi(x, y) \in \mathcal{L}_{\mathsf{V}_k}$, with at most x and y free, and consider terms $\alpha, \beta : \mathsf{V}_k$ such that $\mathsf{el}(\alpha) = \mathsf{el}(\beta)$. If there is a term $f(x) : [\![\phi(\alpha_x, \beta_x)]\!]_{k,h}$ for $x : \mathsf{El}(\alpha)$, then we construct the term $K(f) := (\lambda x.(x, f(x)), \lambda x.(x, f(x)))$.

The term $K(f)$ witnesses the validity of the formula $\forall\exists\frac{x\in\alpha}{y\in\beta}\phi(x, y)$.

Note that we crucially use that $\mathsf{el}(\alpha) = \mathsf{el}(\beta)$.

Lemma 1.3.9. *Strong Collection is valid in the interpretations* $[\![\cdot]\!]_{k,\infty}$*, for the values* $2 \leq k \leq \infty$.

Proof. Recall the formulation of the Strong Collection axiom:

$$(\forall x \in \alpha)\exists y\ \phi(x, y) \Rightarrow \exists z(\forall\exists\frac{x \in \alpha}{y \in z}\phi(x, y)).$$

The proof is analogous to the one of the original interpretation [Acz78]. In order to show its validity we assume the existence a term witnessing the premiss:

$$a : [\![(\forall x \in \alpha)\exists y \phi(x,y)]\!]_{k,\infty}.$$

Since we want to prove the existence of the set z in the conclusion, we consider the function $b := \lambda x.(a(x).1) : \mathsf{El}(\alpha) \to \mathsf{V}_k$, and then define $\beta := \mathsf{sup}((\mathsf{el}(\alpha), p_\alpha), b)$ that provides a witness for the existential quantifier, where $p_\alpha : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{El}(\alpha))$.

Observe that $c := \lambda x.(a(x).2) : (\Pi x : \mathsf{El}(\alpha))[\![\phi(\alpha_x, \beta)]\!]_{k,\infty}$. Now the preliminary Remark 1.3.8 applies and we have a term $K(c) : [\![\forall\exists \frac{x\in\alpha}{y\in\beta}\phi(x,y)]\!]_{k,\infty}$ that together with the term β witnesses the validity of the conclusion.

□

Lemma 1.3.10. *The Subset Collection axiom is valid in the interpretations* $[\![\cdot]\!]_{k,h}$ *for the following values of the parameters:* $2 \leq k \leq h \leq \infty$.

Proof. Recall the formulation of the Subset Collection axiom from appendix A:

$$\exists\gamma\forall u[(\forall x \in \alpha)(\exists y \in \beta)\phi(x,y,u) \Rightarrow (\exists z \in \gamma)(\forall\exists \frac{x \in \alpha}{y \in z}\phi(x,y,u))].$$

Given sets $\alpha, \beta : \mathsf{V}_k$ we consider the type $A := \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$. It is a small type since $A = \mathsf{T}(\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)))$, where exp represents function types internally in U. Moreover, A is a k-type by Theorem 1.1.8, so that we have a term $p : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(A)$ witnessing this fact.

Now consider the function $G : A \to \mathsf{V}_k$ defined as:

$$G(z) := \mathsf{sup}((\mathsf{el}(\alpha), p_\alpha), \beta_{\_} \circ z),$$

for $z : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$, and where $\beta_{\_}$ is the function introduced in Lemma 1.2.2. Here $p_\alpha : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{El}(\alpha))$ and $\beta_{\_} : \mathsf{El}(\beta) \to \mathsf{V}_k$. Then we define the set:

$$\gamma := \mathsf{sup}((\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)), p), G),$$

which is a witness for the existential quantifier.

Next, suppose we have a term in the interpretation of the premiss of the implication:

$$a : [\![(\forall x \in \alpha)(\exists y \in \beta)\phi(x,y,u)]\!]_{k,h},$$

since $h \geq k$, the Σ-type interpreting the existential quantifier is already a h-type, so that the truncation is unnecessary. Thus we have:

$$a : (\Pi x : \mathsf{El}(\alpha))(\Sigma y : \mathsf{El}(\beta))[\![\phi(\alpha_x, \beta_y, u)]\!]_{k,h},$$

from the term $a(x)$ we can project to the two components of the Σ-type, to get the term $b := \lambda x.(a(x).1) : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$ and the term:

$$c := \lambda x.a(x) : (\Pi x : \mathsf{El}(\alpha))[\![\phi(\alpha_x, \beta_{b(x)}, u)]\!]_{k,h}.$$

Then the preliminary Remark 1.3.8 applies to c and we get a term in the conclusion of the implication:

$$(b, K(c)) : [\![(\exists z \in \gamma)(\forall\exists \frac{x \in \alpha}{y \in z} \phi(x, y, u))]\!]_{k,h}.$$

□

Observe that the same proofs of Proposition 1.3.7, Lemma 1.3.9 and Lemma 1.3.10 that involve V_k and the family k-Types_U, use only some of the properties of those types. It can be easily generalised to any class of types that is definable internally in type theory, contains N, 0,1, 2 and is closed under $+$, Σ and Π.

We are not able to show that Strong Collection is not provable in the interpretations $[\![\cdot]\!]_{k,h}$ for $h < \infty$ and Subset Collection for $1 \leq h < k$. But we can indicate that the proof we used fails.

Remark 1.3.11. The standard way to construct terms validating the Strong Collection axiom fails in the interpretations $[\![\cdot]\!]_{k,h}$ for $h < \infty$, and for the Subset Collection axiom it fails for $1 \leq h < k$.

(i) Recall from Lemma 1.3.9 that in trying to validate this axiom we start with a term in the premiss $a : [\![(\forall x \in \alpha)\exists y \phi(x, y)]\!]_{k,h}$ and consider the function $b := \lambda x.(a(x).1) : \mathsf{El}(\alpha) \to V$, and then define $\beta := \mathsf{sup}((\mathsf{el}(\alpha), p), b)$ that provides a witness for the existential quantifier in the conclusion. However, when we work with the interpretation $[\![\cdot]\!]_{k,h}$ with $h < \infty$ we cannot define b by taking the first projection of $a(x)$ because of the presence of truncations:

$$a(x) : \left\| (\Sigma y : \mathsf{V}_k)[\![\phi(x, y)]\!]_{k,h} \right\|_h,$$

and V_k is not a h-type in general;

(ii) For Subset Collection the situation is similar. Recall that in the proof of Lemma 1.3.10 we start with a term in the premiss of the implication $a : [\![(\forall x \in \alpha)(\exists y \in \beta)\phi(x, y, u)]\!]_{k,h}$, which we use to construct the term $b := \lambda x.(a(x).1)$ by using the first projection of a Σ-type. However, when using $[\![\cdot]\!]_{k,h}$ for $1 \leq h < k$ we cannot take the projection $a(x).1$ since Σ-types are truncated.

Remark 1.3.12. Similarly, we can indicate how the standard proofs fail for the Replacement axiom for $h < \infty$, and for the Exponentiation axiom for $1 \leq h < k$.

(i) Replacement: the problem with this axiom is similar to the one of Strong Collection. The extra hypothesis of the uniqueness of y in the premiss fails to give enough information to extract the witness for the existential quantifier in the conclusion. Indeed, recall that set-theoretic equality is interpreted as the bisimulation relation with truncations (Section 1.2);

(ii) The Exponentiation axiom:

$$\forall \alpha, \beta \, \exists \gamma [\forall f (\alpha \xrightarrow{f} \beta) \Rightarrow f \in \gamma],$$

in CZF it follows from the Subset Collection axiom, but it can easily be validated directly in $[\![\cdot]\!]_{k,h}$ for $2 \leq k \leq h \leq \infty$. Here we recall how to do so. The type-theoretic axiom of choice is used in the proof, but it cannot be applied in the interpretation $[\![\cdot]\!]_{k,h}$ for $1 \leq h < k$ due to the presence of truncations.

Given $\alpha, \beta : \mathsf{V}_k$ we construct γ as follows: consider the function type $\mathsf{El}(\alpha) \to \mathsf{El}(\beta)$ which is definitionally equal to $\mathsf{T}(\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)))$. Then let us define the function:

$$\gamma_{\_} := \lambda z.\mathsf{sup}((\mathsf{el}(\alpha), p), \lambda x.\langle \alpha_x, \beta_{z(x)} \rangle),$$

inhabiting the type:

$$(\mathsf{El}(\alpha) \to \mathsf{El}(\beta)) \to \mathsf{V}_k.$$

Then $\gamma := \mathsf{sup}((\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)), p), \gamma_{\_})$. In order to validate the axiom we are given $f : \mathsf{V}_k$ and a term witnessing the premiss of the implication. Our aim is to validate:

$$[\![f \in \gamma]\!]_{k,h} = (\Sigma t : \mathsf{El}(\gamma))(f \approx_h \gamma_t).$$

The function $t : \mathsf{El}(\gamma)$, where $\mathsf{El}(\gamma) = \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$, is provided by the type-theoretic axiom of choice applied to the part of the premiss:

$$[\![(\forall x \in \alpha)(\exists! y \in \beta)(\langle \alpha, \beta \rangle \in f)]\!]_{k,h}.$$

We can apply the type-theoretic axiom of choice here because there are no truncations for $k \leq h \leq \infty$, whereas we cannot apply it in presence of truncations for $1 \leq h < k$. Then the only thing left to prove is that $(\gamma_t \approx_h f)$ is inhabited, which is straightforward.

However, in contrast to Remark 1.3.12, for $h = 1$ it is possible to remedy via a different proof so that both Replacement and Exponentiation are valid, as we shall

see in chapter 3.

Lemma 1.3.13. *The Replacement and Exponentiation axioms are valid in the interpretations* $[\![\cdot]\!]_{k,1}$ *of* CST *into the type theory* HoTT*, for* $2 \leq k \leq \infty$*.*

Proof. See Corollary 3.5.3 of chapter 3. □

We can summarise the results obtained in this section by saying that we have hierarchies of interpretations indexed by homotopy levels as detailed in the following theorem.

Theorem 1.3.14. *For* $2 \leq k \leq \infty$ *we have the following validity results for* $[\![\cdot]\!]_{k,h}$*:*

(i) *for* $h = 1$, $\quad [\![\cdot]\!]_{k,1} : \mathsf{CST} \longrightarrow \mathsf{HoTT}$*;*
(ii) *for* $1 < h < k$, $\quad [\![\cdot]\!]_{k,h} : \mathsf{BCS} \longrightarrow \mathsf{H}$*;*
(iii) *for* $k \leq h < \infty$, $\quad [\![\cdot]\!]_{k,h} : \mathsf{BCS} + \mathsf{SubColl} \longrightarrow \mathsf{H}$*;*
(iv) *for* $h = \infty$, $\quad [\![\cdot]\!]_{k,\infty} : \mathsf{CZF} \longrightarrow \mathsf{H}$.

Proof.
(i) for $h = 1$, use Lemma 1.3.13 and Proposition 1.3.7;
(ii) for $1 < h < k$, use Proposition 1.3.7;
(iii) for $k \leq h < \infty$, use Lemma 1.3.10 and Proposition 1.3.7;
(iv) for $h = \infty$, use Proposition 1.3.7, Lemma 1.3.10, Lemma 1.3.9.

□

To understand informally this result, note that we consider $2 \leq k \leq \infty$ fixed. Then, for any $1 \leq h \leq \infty$, we have that all the axioms of BCS are valid under $[\![\cdot]\!]_{k,h}$. When $h = 1$, we are able to validate also Replacement and Exponentiation, see chapter 3 for more details. When $1 < h < k$, we cannot say anything due to the presence of truncations. When $k \leq h < \infty$, some truncations disappear (see Remark 1.2.4) and we are able to validate Subset Collection. Finally, when $h = \infty$, no truncations are necessary and we can use the type-theoretic axiom of choice to validate all the axioms of CZF.

1.4. An alternative interpretation of equality

Here we explore a natural variant of the interpretations $[\![\cdot]\!]_{k,h}$ where set-theoretic equality is interpreted differently but formulas are still interpreted as in $[\![\cdot]\!]_{k,h}$ (see Section 1.2 for the definition). We denote the variant interpretation as $[\![\cdot]\!]'_{k,h}$. The interpretation $[\![\cdot]\!]'_{k,h}$ is defined so that instead of truncating the Σ-types inside the definition of bisimulation (as defined in Section 1.2), we first form the whole bisimulation relation and then truncate it:

$$\Big\| (\Pi x : \mathsf{El}(\alpha))(\Sigma y : \mathsf{El}(\beta))(\alpha_x \approx'_h \beta_y) \times (\Pi y : \mathsf{El}(\beta))(\Sigma x : \mathsf{El}(\alpha))(\alpha_x \approx'_h \beta_y) \Big\|_h .$$

The rest of the inductive definition of $[\![\cdot]\!]'_{k,h}$ is the same as the one of $[\![\cdot]\!]_{k,h}$.

We give some details on the differences between the two interpretations of equality. In short, in $[\![\cdot]\!]'_{k,h}$ the proof of the Extensionality axiom fails, on the other hand the proofs of the preliminary Lemma 1.3.1 on equality is simplified.

In order to make some of the next proofs more readable we introduce the following notation $(\alpha \doteq \beta)'$, defined as the following type:

$$(\Pi x : \mathsf{El}(\alpha))(\Sigma y : \mathsf{El}(\beta))(\alpha_x \approx'_h \beta_y) \times (\Pi y : \mathsf{El}(\beta))(\Sigma x : \mathsf{El}(\alpha))(\alpha_x \approx'_h \beta_y),$$

so that $(\alpha \approx'_h \beta) = \|(\alpha \doteq \beta)'\|_h$.

Lemma 1.4.1. *For $\alpha, \beta : \mathsf{V}_k$, the family of types $(\alpha \approx'_h \beta)$ is an equivalence relation.*

Proof. The type $(\alpha \approx'_h \alpha)$ is inhabited since it is the truncation of an inhabited type.

The type $(\alpha \approx'_h \beta) \to (\beta \approx'_h \alpha)$ is inhabited by Lemma 1.1.4 since it is of the form $\|(\alpha \doteq \beta)'\|_h \to \|(\beta \doteq \alpha)'\|_h$ and $(\alpha \doteq \beta)' \to (\beta \doteq \alpha)'$ is inhabited.

For transitivity we want to construct a term inhabiting the type:

$$\|(\alpha \doteq \beta)'\|_h \to (\|(\beta \doteq \gamma)'\|_h \to \|(\alpha \doteq \gamma)'\|_h).$$

Since the codomain $(\|(\beta \doteq \gamma)'\|_h \to \|(\alpha \doteq \gamma)'\|_h)$ is a h-type we can remove the truncation from the domain and equivalently construct a term inhabiting the type $(\alpha \doteq \beta)' \to (\|(\beta \doteq \gamma)'\|_h \to \|(\alpha \doteq \gamma)'\|_h)$. By Lemma 1.1.4, it is enough to find a term inhabiting $(\alpha \doteq \beta)' \to ((\beta \doteq \gamma)' \to (\alpha \doteq \gamma)')$ which is straightforward. □

Lemma 1.4.2. *The interpretation $[\![\cdot]\!]'_{k,h}$ of every formula satisfies the Leibniz rule with respect to $\approx'_h$, i.e. given a formula $\phi(x)$ we have:*

$$(\Pi x, y : \mathsf{V}_k)((x \approx'_h y) \times [\![\phi(x)]\!]'_{k,h} \to [\![\phi(y)]\!]'_{k,h}).$$

Proof. By induction on the structure of the formula. The only case we consider is the one of atomic formulas of the form $\alpha \in \beta$ since the others are straightforward.

We want to construct terms inhabiting the type:

$$(\alpha \approx'_h \beta) \times [\![\beta \in \gamma]\!]'_{k,h} \to [\![\alpha \in \gamma]\!]'_{k,h}.$$

The proof for this case is as the one of Lemma 1.3.4, but using transitivity of $(\alpha \approx'_h \beta)$.

The Leibniz rule in the other variable is:

$$(\gamma \approx'_h \delta) \times [\![\alpha \in \gamma]\!]'_{k,h} \to [\![\alpha \in \delta]\!]'_{k,h},$$

which by definition of the interpretation is:

$$\|(\gamma \doteq \delta)'\|_h \to \Big(\Big\|(\Sigma x : \mathsf{El}(\gamma))\|(\gamma_x \doteq \alpha)'\|_h\Big\|_h \to \Big\|(\Sigma y : \mathsf{El}(\delta))\|(\delta_y \doteq \alpha)'\|_h\Big\|_h\Big).$$

By the universal property of truncation and Lemma 1.1.4, it is enough to construct a term inhabiting the following type:

$$(\gamma \doteq \delta)' \to \Big((\Sigma x : \mathsf{El}(\gamma))\|(\gamma_x \doteq \alpha)'\|_h \to (\Sigma y : \mathsf{El}(\delta))\|(\delta_y \doteq \alpha)'\|_h\Big).$$

Recall that the first conjunct of the definition of $(\gamma \doteq \delta)'$ gives a term:

$$F : (\Pi x : \mathsf{El}(\gamma))(\Sigma y : \mathsf{El}(\delta))(\gamma_x \approx'_h \delta_y),$$

applying it to $x : \mathsf{El}(\gamma)$ in the premiss of the implication gives $F(x) : \mathsf{El}(\delta)$ which provides the first component of $(\Sigma y : \mathsf{El}(\delta))\|(\delta_y \doteq \alpha)'\|_h$. The second component is provided by transitivity by the two equalities $(\gamma_x \approx'_h \delta_y)$ and $(\gamma_x \approx'_h \alpha)$. □

Lemma 1.4.3. *The structural axioms for defining the quantifiers are valid, i.e. the types:*

$$[\![(\forall x \in y)\phi(x) \Leftrightarrow \forall x(x \in y \Rightarrow \phi(x))]\!]_{k,h},$$

and

$$[\![(\exists x \in y)\phi(x) \Leftrightarrow \exists x(x \in y \wedge \phi(x))]\!]_{k,h},$$

are inhabited.

Proof. The proof is analogous to the one of Lemma 1.3.5. □

Remark 1.4.4.

(i) In the interpretations $[\![\cdot]\!]'_{k,h}$ the standard proof of the validity of Extensionality fails. We leave open the question to prove that it is not valid in $[\![\cdot]\!]'_{k,h}$. Recall that thanks to Lemma 1.4.3 we have the validity of the following set-theoretic formula:

$$(\alpha \subseteq \beta \wedge \beta \subseteq \alpha) \Leftrightarrow \forall x\,(x \in \alpha \Leftrightarrow x \in \beta).$$

Therefore, in order to validate the Extensionality axiom it is enough to show that $(\alpha \doteq \beta) \Leftrightarrow (\alpha \subseteq \beta \wedge \beta \subseteq \alpha)$ is valid. It is straightforward to validate the implication that from the premiss $\|(\alpha \doteq \beta)'\|_h$ gives the conclusion:

$$(\Pi x : \mathsf{El}(\alpha))\Big\|(\Sigma y : \mathsf{El}(\beta))(\alpha_x \approx'_h \beta_y)'\Big\|_h \times (\Pi y : \mathsf{El}(\beta))\Big\|(\Sigma x : \mathsf{El}(\alpha))(\alpha_x \approx'_h \beta_y)'\Big\|_h.$$

But given how the truncations are positioned in the types above, we do not see any way to validate the converse implication;

(ii) the standard proofs of the validity of the Strong Collection, Subset Collection, Replacement and Exponentiation axioms presented in Proposition 1.3.7 have the same probles as for $[\![\cdot]\!]_{k,h}$, that we have detailed in Remark 1.3.11 and Remark 1.3.12.

Theorem 1.4.5. *The following axioms of set theory are valid in the interpretation* $[\![\cdot]\!]'_{k,h}$*: Set-Induction, Pairing, Union, Bounded Separation and Infinity.*

Proof. The proof is analogous to the one of Proposition 1.3.7. □

CHAPTER 2

A characterisation of the interpretations $[\![\cdot]\!]_{k,\infty}$

This chapter is devoted to a more in depth analysis of the family of interpretations $[\![\cdot]\!]_{k,\infty}$ for $2 \leq k \leq \infty$. Recall that in these interpretations sets are interpreted as k-types whereas formulas are interpreted without introducing truncations, simply following the propositions-as-types correspondence.

Since the family of interpretations $[\![\cdot]\!]_{k,\infty}$ is very close in its definition to Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$, comparing these families of interpretations can shed light on the role of homotopy levels in the interpretation of set theory.

As we show in Theorem 2.5.1 and Corollary 2.5.2, the interpretations $[\![\cdot]\!]_{k,\infty}$ and $[\![\cdot]\!]_{\infty,\infty}$ contain the same relevant set-theoretic information, in other words we may say that CZF is not able to distinguish between the different interpretations $[\![\cdot]\!]_{k,\infty}$.

To do this, we follow the characterisation of Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$ by Rathjen and Tupailo in [RT06]. Their starting point is the observation that some additional axioms are valid in type theory, for example certain forms of the axiom of choice like the Dependent Choice axiom. A natural question is to determine what is the strongest set theory interpreted by $[\![\cdot]\!]_{\infty,\infty}$ in Martin-Löf type theory. As it turns out in their work this amounts to find the strongest choice axiom valid in the interpretation $[\![\cdot]\!]_{\infty,\infty}$.

They introduce the notion of CC formulas (see Definition 2.2.1), which is a technical condition that they use in their proof. As they show in [RT06, Section 5.1] it is a natural notion of formula, and most formulas used in mathematical practice satisfy that condition. Then they characterise the class of CC sentences valid in the interpretation $[\![\cdot]\!]_{\infty,\infty}$ as the ones provable in CZF from the $\Pi\Sigma$ axiom of choice (see Axiom 2.1.5 for its formulation).

Here, we apply their machinery to our context, showing that all the interpretations $[\![\cdot]\!]_{k,\infty}$ that model CZF via the types of sets V_k validate the same CC sentences. So we can describe the situation as having a hierachy of models $\mathsf{V}_2, \ldots, \mathsf{V}_\mathsf{k}, \ldots \mathsf{V}_\infty$ that satisfy the same CC sentences.

Outline. Section 2.1 introduces some set-theoretic preliminaries in CZF, and Section 2.2 gives a sketch of the proof of the characterisation theorem of [RT06].

Then in order to prove one direction of the characterisation of $[\![\cdot]\!]_{k,\infty}$ we apply the result of Rathjen and Tupailo to our context. For that we just need to factor the interpretation $[\![\cdot]\!]_{\infty,\infty}$ through $[\![\cdot]\!]_{k,\infty}$, which is done in Section 2.3. Then in Section 2.4 we prove the other direction of the characterisation by adapting Aczel's proof of the validity of the $\mathsf{\Pi\Sigma}$-AC in the interpretations $[\![\cdot]\!]_{k,\infty}$. Section 2.5 states the theorem of the characterisation and draws some conclusions. Finally, Section 2.6 explores a side question related to the $\mathsf{\Pi\Sigma I}$ axiom of choice which is an equivalent formulation of the $\mathsf{\Pi\Sigma}$ axiom of choice.

2.1. Set-theoretic preliminaries

Let us recall from [Acz82] some set-theoretic preliminaries within CZF that allow us to formulate the $\Pi\Sigma$ axiom of choice.

Definition 2.1.1. In CZF, given a family of sets B indexed by a set A, we define:

- $\Pi(A, B) := \{f \in C^A \mid (\forall x \in A)\, f(x) \in B(x)\}$, where $C := \bigcup_{x\in A} B(x)$;
- $\Sigma(A, B) := \bigsqcup_{x\in A} B(x)$ is the disjoint union of the family B.

Definition 2.1.2. In CZF we say that a class X is $\Pi\Sigma$-*closed* if and only if:

- $\omega \in X$;
- $\Pi(A, B) \in X$ for all families B of sets in X indexed by a set $A \in X$;
- $\Sigma(A, B) \in X$ for all families B of sets in X indexed by a set $A \in X$.

Theorem 2.1.3 (Aczel)**.** *In* CZF *there is a smallest* $\Pi\Sigma$*-closed class, called the class of* $\Pi\Sigma$-generated sets*.*

Proof. See [Acz82, Section 4.2]. □

Definition 2.1.4. In CZF a set A is called a *base* if and only if for every relation R from A to B such that for all $x \in A$ there is a $y \in B$, there is a function $f : A \to B$ such that for all $a \in A$ we have $f(a) \in R(a)$.

The $\Pi\Sigma$ axiom of choice informally states that one can apply the axiom of choice to a class of basic sets. The notion of basic set that is required is that of being inductively generated from ω by using the set-theoretic operations of Π and Σ. More succinctly:

Axiom 2.1.5 ($\mathsf{\Pi\Sigma}$-AC)**.** Every $\Pi\Sigma$-generated set is a base.

2.2. Proof-theoretic preliminaries

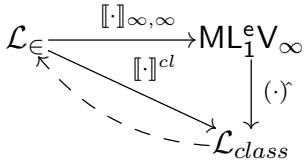
In order to characterise the formulas of set theory validated in the interpretations $[\![\cdot]\!]_{k,\infty}$ of CZF in V_k we use the proof-theoretic machinery developed in [RT06], which was originally devised in order to characterise the formulas validated in Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$. The characterisation is limited to a class of formulas satisfying a technical condition that is used to make the realizability techniques work.

Definition 2.2.1. A formula of set theory is called a *CC formula* if and only if whenever an unbounded quantifier appears in an implication, it can only appear in the conclusion of the implication.

The notion of *CC* formula may seem somewhat arbitrary, but [RT06, Section 5.1] provides arguments to convince the reader that this is a natural and rich notion.

This section presents a sketch of the proof of the main theorem of [RT06], which we represent with the diagram:

(2.1)



In the diagram, $\mathcal{L}_\in$ is the language of set theory and $\mathcal{L}_{class}$ represents informally sets and classes of set theory. The arrow $[\![\cdot]\!]^{cl} : \mathcal{L}_\in \to \mathcal{L}_{class}$ is a formulas-as-classes interpretation of $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC}$ into itself.

Here, V_∞ is Aczel's type $(Wx : \mathsf{U})\mathsf{T}(x)$, meaning that the type theory $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ does not have general rules for W-types but only for the single W-type V_∞. The superscript 'e' in $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ indicates that the theory is endowed with extensional identity types. In the sequel we use the superscript 'i' to indicate that the theory has intensional identity types. The rules for extensional identity types can be found in [ML84].

There are three key points to the proof that can be depicted as in the diagram above. They are performed in the meta-theory $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC}$.

(i) one constructs a formulas-as-classes interpretation of $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC}$ into itself using the machinery of set-recursion theory. See [RT06, Section 4]. This interpretation is represented in the diagram as the arrow $\mathcal{L}_\in \xrightarrow{[\![\cdot]\!]^{cl}} \mathcal{L}_{class}$;

(ii) one constructs a realizability interpretation $(\cdot)^\curlywedge$ of the type theory $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ into $\mathcal{L}_{class}$ such that CZF proves that if $(t : [\![\phi]\!]_{\infty,\infty})^\curlywedge$ then $\exists u(u \in [\![\phi]\!]^{cl})$. See [RT06, Section 6];

(iii) finally, by constructing an inner model $H(Y^*)$ of $\mathsf{CZF}{+}\Pi\Sigma\text{-}\mathsf{AC}$ and proving an absoluteness result of CC sentences for $H(Y^*)$ one obtains that for such a sentence ϕ we have $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC} \vdash (\exists i \in [\![\phi]\!]^{cl}) \Rightarrow \phi$. See [RT06, Section 5]. This step is represented in the diagram by the dashed arrow.

Combining these three steps together one gets the following result.

Theorem 2.2.2 (Rathjen and Tupailo)**.** *Let ϕ be a CC sentence, then we have that $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC} \vdash \phi$ if and only if $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash t : [\![\phi]\!]_{\infty,\infty}$ for some term t.*

Proof. If $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC} \vdash \phi$ then $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash \mathsf{t} : [\![\phi]\!]_{\infty,\infty}$ for some term t because $\Pi\Sigma\text{-}\mathsf{AC}$ is valid in the type-theoretic interpretation of CZF. See [Acz82, Theorem 6.8].

Conversely, given $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash t : [\![\phi]\!]_{\infty,\infty}$, we first apply the interpretation $(\cdot)^\curlywedge$ obtaining a formula of CZF say $(t : [\![\phi]\!]_{\infty,\infty})^\curlywedge$. Then by point (ii) of 2.1 we have that $\exists u(u \in [\![\phi]\!]^{cl})$. Finally, we apply point (iii) of 2.1 and obtain $\mathsf{CZF}{+}\Pi\Sigma\text{-}\mathsf{AC} \vdash \phi$. □

The underlying idea for the formulas-as-classes interpretation $[\![\cdot]\!]^{cl}$ is to interpret formulas as classes mirroring the structure of the interpretation $[\![\cdot]\!]_{\infty,\infty}$ of CZF in the type theory $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$.

However, there is a problem when we come to define the class interpreting an implication $\phi \Rightarrow \psi$ with ϕ unbounded, or the class interpreting an unbounded universal quantification. In the bounded cases we can consider the class of functions with appropriate domain and codomain, but when the domain is a proper class we cannot use functions. An alternative would be to use class functions instead but we then have the issue of collecting these class functions into a single class.

As a workaround a set-recursion machinery is developed in [Rat06] (which is then used in [RT06]) providing meaning to expressions of the form $\{e\}(x) \simeq y$ with x, y sets, and providing indices for well-behaved class functions (the $E_{\mathcal{P}}$-recursive functions defined in [RT06, Definition 4.5]).

The interpretation of an unbounded universal quantifier is the class of indices of the class functions with prescribed domain and codomain. This set-recursion machinery is used also to construct the interpretation $(\cdot)^\curlywedge$ of $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ into CZF.

Definition 2.2.3. Given a class A such that $B(x)$ is a class for every $x \in A$, we define:

$$(\tilde{\Pi}x \in A)B(x) := \{s \,|\, \forall x \in A\, \{s\}(x) \in B(x)\}.$$

Then $\tilde{\to}$ is the non-dependent version of $\tilde{\Pi}$, where the family of classes $B(x)$ is constant.

Similarly to the construction of $[\![\cdot]\!]^{cl}$, the interpretation $(\cdot)^\frown$ uses functions to interpret terms of $(\Pi x : A)B(x)$ when A is small, and uses indices of $E_{\mathcal{P}}$-recursive functions for the case of a large A.

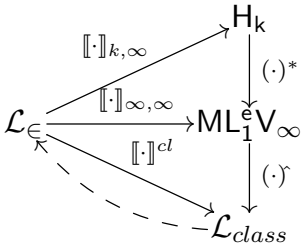
2.3. Relating the interpretations $[\![\cdot]\!]_{\infty,\infty}$ and $[\![\cdot]\!]_{k,\infty}$

In this section we prove one implication of the characterisation of Theorem 2.5.1. Our aim is to prove that every CC sentence ϕ valid in the interpretation $[\![\cdot]\!]_{k,\infty}$ is provable in $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC}$.

First of all, observe that the interpretations $[\![\cdot]\!]_{k,\infty}$ do not make use of truncations since formulas are interpreted using the propositions-as-types correspondence. Moreover, for the whole family of interpretations $[\![\cdot]\!]_{k,h}$, not all W-types are needed, and for fixed values of k and h only the rules for the type V_k are actually used. Hence the interpretations $[\![\cdot]\!]_{k,\infty}$ can be carried out in the type theory $\mathsf{H_k} := \mathsf{ML}_1^{\mathsf{i}}\mathsf{V_k} + \mathsf{FE}$.

In chapter 1 we have chosen to include all W-types since it is convenient for the characterisation of Theorem 4.3.4 of chapter 4.

In order to characterise the image of the interpretation $[\![\cdot]\!]_{k,\infty}$ of CZF in the type theory $\mathsf{H_k}$, as in Theorem 2.2.2, we aim to extend the diagram in 2.1 as follows:



This amounts to establish the following steps:

(i) construct a sound interpretation $(\cdot)^*$ of the type theory H_k into $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$;

(ii) prove that $(\cdot)^*$ maps the interpretation $[\![\cdot]\!]_{k,\infty}$ to $[\![\cdot]\!]_{\infty,\infty}$ (in the precise sense of Lemma 2.3.3).

We could have made a different choice defining directly a new interpretation of H_k into $\mathcal{L}_{class}$ along the lines of $(\cdot)^\frown$ and reproving the relevant lemmas. Our choice to interpret H_k into $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ minimises the amount of details to check, and it turns out to be straightforward given the results of [RT06].

The next lemma is essentially folklore. Since we were not able to find a reference we give a sketch of the proof.

Lemma 2.3.1. *In presence of extensional identity types, function extensionality is equivalent to the (definitional) η-rule:*

$$\vdash f = \lambda x.f(x) : (\Pi x : A)B(x).$$

Proof. Assume function extensionality, we want to prove the η-rule. By the computation rule for Π-types we can derive the judgement $x : A \vdash f(x) = (\lambda y.f(y))x$. Then the thesis follows by using the introduction rule for Id^e, function extensionality, and the elimination rule for Id^e.

Conversely, assume the η-rule and the premiss of function extensionality, i.e. the existence of a term $t : (\Pi x : A)\mathsf{Id}_{B(x)}(f(x), g(x))$. Applying in sequence the elimination rule for Id^e, a λ-abstraction on both sides and the η-rule we derive the judgement $x : A \vdash f = g : (\Pi x : A)B(x)$. We conclude applying the introduction rule for Id^e.

□

Now we construct an interpretation $(\cdot)^*$ of H_k into $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ that maps terms to terms, types to types, contexts to contexts and judgements to judgements, with the only exception of terms of the (Tarski) universe of H_k that are mapped to terms of the (Russell) universe of $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ which are types themselves. For the sake of clarity in this construction we refer to types and term constructors of $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ using the superscript e and the ones of H_k using the superscript i.

Recall that in our notation the base types and type constructors $\mathsf{0}$, $\mathsf{1}$, $\mathsf{2}$, N, Π, $\to$, Σ, $\times$, $+$, W, Id_A are represented in the universe using the corresponding terms and term constructors: $0, 1, 2, \mathsf{n}, \pi, \mathsf{exp}, \sigma, \mathsf{times}, \mathsf{plus}, \mathsf{w}, \mathsf{i}_a$.

The construction of this interpretation is straightforward, the extra complication of Tarski vs Russell style universes comes merely from our choice to use the former, whereas [RT06] uses the latter.

In order to take care of the difference between the Tarski and Russell universes we construct $(\cdot)^*$ as follows:

- $(\mathsf{U}^i)^* := \mathsf{U}^e$;
- for $t : \mathsf{U}^i$ we define $t^* := \mathsf{T}(t)^*$;
- for a variable $x : \mathsf{U}^i$ we define $\mathsf{T}(x)^* := A$, a type variable $A : \mathsf{type}$;
- $\mathsf{T}(\mathsf{n})^* := \mathsf{N}^e$;
- $\mathsf{T}(0)^* := \mathsf{0}^e$;
- $\mathsf{T}(1)^* := \mathsf{1}^e$;
- $\mathsf{T}(2)^* := \mathsf{2}^e$;
- $\mathsf{T}(\mathsf{plus}(a, b))^* := \mathsf{T}(a)^* +^e \mathsf{T}(b)^*$;

- $\mathsf{T}(\pi(a,b))^* := (\Pi^e x : \mathsf{T}(a)^*)\mathsf{T}(b(x))^*$;
- $\mathsf{T}(\sigma(a,b))^* := (\Sigma^e x : \mathsf{T}(a)^*)\mathsf{T}(b(x))^*$;
- $\mathsf{T}(\mathsf{i_a}(x,y))^* := \mathsf{Id}^e_{\mathsf{T}(a)^*}(x^*, y^*)$.

Regarding the identity type, they only non-trivial point is to interpret the elimination term J of intensional identity types. Formally:

- $(\mathsf{Id}^i)^* := \mathsf{Id}^e$;
- $(\mathsf{refl}^i)^* := \mathsf{refl}^e$;
- $\mathsf{J}(a,b,i,d)^* := \mathsf{d}^*(a^*)$, where $a, b : A$ and $i : \mathsf{Id}^i_A(a,b)$ and for $x : A$ we have $\mathsf{d}(x) : C(x,x,\mathsf{refl}_x)$ as in the premiss of the elimination rule for Id^i.

A reference for the fact that the term J is definable in extensional type theory is [NPS90, Section 8.2].

The interpretation of the types of iterative sets V_∞ and V_k is straightforward: since the rules defining V_k require more structure in the premiss than the ones of V_∞ we just need to forget that structure.

- $\mathsf{V}^*_\mathsf{k} := \mathsf{V}_\infty$;
- $\mathsf{sup}^i(a,f)^* := \mathsf{sup}^e(a.1^*, f^*)$, where we have $a : (\Sigma x : \mathsf{U})\mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(x))$ and $f : \mathsf{T}(a.1) \to \mathsf{V}_k$;
- $\mathsf{R}^i(\alpha,d)^* := \mathsf{R}^e(\alpha^*, d^*)$, where $\alpha : \mathsf{V}_k$ and $\mathsf{d}(x,y,z) : C(\mathsf{sup}^i(x,y))$ which depends on $x : k\text{-}\mathsf{Types}_U$ and $y : \mathsf{T}(x.1) \to \mathsf{V}_k$ and $z : (\Pi^i v : \mathsf{T}(x.1))C(v(y))$.

Finally, since in $\mathsf{ML}^\mathsf{e}_1\mathsf{V}_\infty$ function extensionality is equivalent to the definitional η-rule which is one of the rules of the system (see the previous Lemma 2.3.1), we require that the interpretation maps the term witnessing function extensionality in one theory to the corresponding term in the other theory:

- $(\mathsf{funext})^* := \mathsf{funext}^e$.

Then the interpretation $(\cdot)^*$ is extended homomorphically to all other types, terms, contexts and judgements.

Lemma 2.3.2. *The interpretation $(\cdot)^*$ of H_k into $\mathsf{ML}^\mathsf{e}_1\mathsf{V}_\infty$ is sound, i.e. if $\mathsf{H}_k \vdash \mathcal{J}$ for some judgement $\mathcal{J}$, then $\mathsf{ML}^\mathsf{e}_1\mathsf{V}_\infty \vdash \mathcal{J}^*$.*

Proof. This proof is a straightforward induction on the derivation of the judgement. If $\mathcal{J}$ is derived using one of the rules for the universe U^i, either introduction or computation, then $\mathcal{J}^*$ by construction is derived respectively by an introduction rule for U^e or it is a judgemental equality of the form $A = A : \mathsf{U}^e$.

For the identity type the only thing to check is the case of the computation rule as the others are immediate by construction. We apply the interpretation $(\cdot)^*$ to both sides of the equality in the computation rule $\mathsf{J}(a,a,\mathsf{refl}_a,\mathsf{d}) = \mathsf{d}(a)$. By

definition $\mathsf{J}(a, a, \mathsf{refl}_a, \mathsf{d})^* = \mathsf{d}^*(a^*)$, on the other hand $\mathsf{d}(a)^* = \mathsf{d}^*(a^*)$. Therefore the computation rule for Id^i is sound.

The interpretation of the types of iterative sets V_∞ and V_k is straightforward since the rules for V_k differ from those of V_∞ only for the requirement of the homotopy level of $\mathsf{T}(x)$ for $x : \mathsf{U}^i$. Therefore by applying $(\cdot)^*$, the premisses of the rules for V_∞ can be obtained projecting the terms $x : \mathsf{is}\text{-}k\text{-}\mathsf{Type}$ onto U^i.

All other rules and the axiom of function extensionality are immediate. □

Lemma 2.3.3. *For all formulas ϕ of CZF and all $\alpha_1, \ldots, \alpha_N : \mathsf{V}_k$, if we have that $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash (t : [\![\phi(\underline{\alpha})]\!]_{k,\infty})^*$ is derivable for some term t, then $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash t^* : [\![\phi(\underline{\alpha}^*)]\!]_{\infty,\infty}$.*

Proof. Recalling the definition of the interpretations $[\![\cdot]\!]_{\infty,\infty}$ and $[\![\cdot]\!]_{k,\infty}$ we notice that the inductive clauses of the definitions are the same, the only difference being in the type V_k that appears in the interpretation of unbounded quantifiers and equality instead of V_∞. The conclusion follows immediately since by construction the translation $(\cdot)^*$ maps V_k to V_∞ and terms of V_k to terms of V_∞ while respecting all the other type-theoretic constructors that appear in the definitions of $[\![\cdot]\!]_{\infty,\infty}$ and $[\![\cdot]\!]_{k,\infty}$. □

We can put together the results of this section with the ones by Rathjen and Tupailo [RT06].

Theorem 2.3.4. *Let ϕ be a CC sentence of set theory. If $\mathsf{H}_k \vdash t : [\![\phi]\!]_{k,\infty}$ for some term t of H_k, then $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC} \vdash \phi$.*

Proof. By Lemma 2.3.2 $\mathsf{H}_k \vdash t : [\![\phi]\!]_{k,\infty}$ implies $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash (t : [\![\phi]\!]_{k,\infty})^*$. Thus by Lemma 2.3.3, we have $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash t^* : [\![\phi]\!]_{\infty,\infty}$, and we conclude using the implication from right to left of Theorem 2.2.2. □

2.4. $\Pi\Sigma$-AC is valid in the interpretations $[\![\cdot]\!]_{k,\infty}$

In this section our goal is to show that $\Pi\Sigma\text{-}\mathsf{AC}$ holds in our type theory H_k, which provides the other implication of the characterisation of Theorem 2.5.1. The proof that $\Pi\Sigma\text{-}\mathsf{AC}$ holds in $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ is due to Aczel (see [Acz82, Theorem 6.8]). When we adapt a proof from $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$ to H_k we have to check that:

(i) all constructions done in V_∞ with arbitrary small types can be performed in V_k with small k-types;

(ii) all proofs that involve extensional identity types can be performed with intensional identity types.

For the rest of this section we work in H_k and prove a series of lemmas following closely Aczel's article [Acz82] that lead to the proof that ΠΣ-AC holds in V_k in Corollary 2.4.11.

The idea of the proof is to introduce the notion of *strong base* which is a strengthening of the notion of base (see Definition 2.4.6). The next step is to show that the class of strong bases is ΠΣ-closed, which implies that the class of ΠΣ-generated sets is contained in the class of strong bases.

Definition 2.4.1. Given terms in the type of sets $\alpha, \beta : \mathsf{V}_k$ such that $\mathsf{el}(\alpha) = \mathsf{el}(\beta)$. We can define type-theoretically the set $S(\alpha, \beta) : \mathsf{V}_k$ of pairs of elements of α and β, as follows:

$$\mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \lambda x.\langle \alpha_x, \beta_x \rangle),$$

where $\langle \cdot, \cdot \rangle$ is the set-theoretic pair, and $\mathsf{p}_\alpha : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{el}(\alpha))$ witnesses that α is a k-type.

One can then prove that $S(\alpha, \beta)$ is a relation with domain α and codomain β. See [Acz82, Lemma 5.1(i)] for the details.

Injectively presented sets.

Definition 2.4.2. A set $\alpha : \mathsf{V}_k$ is called *injectively presented* if and only if for all $x_1, x_2 : \mathsf{El}(\alpha)$ we have:

$$(\alpha_{x_1} \approx_\infty \alpha_{x_2}) \to \mathsf{Id}_{\mathsf{El}(\alpha)}(x_1, x_2).$$

Here $\approx_\infty$ is the bisimulation relation defined in Section 1.2.

The next lemma establishes some properties of $S(\alpha, \beta)$ under the assumption that α is injectively presented.

Here we use $A \leftrightarrow B$ as a shorthand for $(A \to B) \times (B \to A)$, where A and B are types.

Lemma 2.4.3. *Let $\alpha : \mathsf{V}_k$ be injectively presented.*

(i) *Given $\delta : \mathsf{V}_k$ and $\beta\_ : \mathsf{El}(\alpha) \to \mathsf{V}_k$, consider the set $\beta := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \beta\_)$. Then for all $x : \mathsf{El}(\alpha)$ we have the following characterisation for elements of $S(\alpha, \beta)$:*

$$[\![\langle \alpha_x, \delta \rangle \in S(\alpha, \beta)]\!]_{k,\infty} \leftrightarrow (\delta \approx_\infty \beta_x).$$

(ii) *the term* $\gamma : \mathsf{V}_k$ *is a set-theoretic function with domain* α *if and only if there exists a function* $\beta\_ : \mathsf{El}(\alpha) \to \mathsf{V}_k$ *such that* $(\gamma \approx_\infty S(\alpha, \beta))$*, where* $\beta := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \beta\_)$*;*

(iii) *given* $(\beta_1)\_, (\beta_2)\_ : \mathsf{El}(\alpha) \to \mathsf{V}_k$ *define the sets* $\beta_1 := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), (\beta_1)\_)$ *and* $\beta_2 := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), (\beta_2)\_)$*, then we have the following characterisation of the interpretation of equality between the relations* $S(\alpha, \beta_1)$ *and* $S(\alpha, \beta_2)$

$$(S(\alpha, \beta_1) \approx_\infty S(\alpha, \beta_2)) \leftrightarrow (\Pi x : \mathsf{El}(\alpha))((\beta_1)_x \approx_\infty (\beta_2)_x).$$

Proof. For (i), given $\alpha : \mathsf{V}_k$ injectively presented and β as above, we have the following chain of equivalences:

$$\begin{aligned}
&[\![\langle \alpha_x, \delta\rangle \in S(\alpha, \beta)]\!]_{k,\infty} \leftrightarrow && \\
&(\Sigma y : \mathsf{El}(\alpha))(\langle \alpha_x, \delta\rangle \approx_\infty \langle \alpha_y, \beta_y\rangle) \leftrightarrow && \text{(by definition of } S(\alpha, \beta)), \\
&(\Sigma y : \mathsf{El}(\alpha))(\alpha_x \approx_\infty \alpha_y) \times (\delta \approx_\infty \beta_y) \leftrightarrow && \text{(since } \langle \cdot, \cdot \rangle \text{ is the set-theoretic pair)}, \\
&(\Sigma y : \mathsf{El}(\alpha))\mathsf{Id}_{\mathsf{El}(\alpha)}(x, y) \times (\delta \approx_\infty \beta_y) && \text{(since } \alpha \text{ is injectively presented)}.
\end{aligned}$$

Now recall from Lemma 1.1.7 that a function $f : A \to B$, induces a function on the identity types $\mathsf{Id}_A(a, b) \to \mathsf{Id}_B(f(a), f(b))$. Therefore we have $\mathsf{Id}_{\mathsf{V}_k}(\beta_x, \beta_y)$ which in turn implies $(\beta_x \approx_\infty \beta_y)$, which gives $(\delta \approx_\infty \beta_x)$ by transitivity.

For (ii) and (iii) the adaptation of the proof of [Acz82, Lemma 5.3] is straightforward. □

Theorem 2.4.4. *Every injectively presented set is a base.*

Proof. The adaptation of this proof is straightforward, for the details see [Acz82, Theorem 5.4]. The proof uses Lemma 2.4.3. □

Lemma 2.4.5. *The set of natural numbers* $\omega : \mathsf{V}_k$ *is injectively presented.*

Proof. The aim is to prove that given $n_1, n_2 : \mathsf{N}$ we have:

$$(\omega_{n_1} \approx_\infty \omega_{n_2}) \to \mathsf{Id}_{\mathsf{N}}(n_1, n_2).$$

The proof proceeds by a straightforward double induction on $n_1, n_2 : \mathsf{N}$, and it uses the characterisation of Id_{N} that we used in the proof of Theorem 1.1.8. See [Acz82, Lemma 5.5] for the details. □

Strong bases. One drawback of the notion of injectively presented set is that it is not compatible with respect to the type-theoretic equivalence relation $\approx_\infty$, meaning that there could be two sets $\alpha, \beta : \mathsf{V}_k$ with $(\alpha \approx_\infty \beta)$, and α injectively presented, but not β. Therefore [Acz82, Definition 6.1] introduces the following

strengthening of the notion of base.

Definition 2.4.6. We say that $\alpha : \mathsf{V}_k$ is a *strong base* if and only if there exists a $\beta : \mathsf{V}_k$ injectively presented such that $(\alpha \approx_\infty \beta)$ holds.

We now define operations Π and Σ in V_k corresponding to the respective set-theoretic operations introduced in Definition 2.1.1.

Given sets $\alpha, \beta : \mathsf{V}_k$, with $\mathsf{el}(\alpha) = \mathsf{el}(\beta)$ we define $C := (\Sigma x : \mathsf{El}(\alpha))\mathsf{El}(\beta_x)$, which is a small k-type $C = \mathsf{T}(c)$. Thus by Theorem 1.1.8 we have a term witnessing the homotopy level $p_1 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(c))$. Then, for $z : C$, let us consider the function $F(z) := (\beta_{z.1})_{z.2}$, so that we can define:

$$\Sigma(\alpha, \beta) := \mathsf{sup}((c, p_1), \lambda z.\langle \alpha_{z.1}, F(z)\rangle).$$

Similarly, we define the Π-type $D := (\Pi x : \mathsf{El}(\alpha))\mathsf{El}(\beta_x)$ which is also a small k-type $D = \mathsf{T}(d)$. Thus we have a proof $p_2 : \mathsf{is}\text{-}k\text{-}\mathsf{Type}(\mathsf{T}(d))$. Let $z : D$, and consider the function $G : \mathsf{T}(d) \to \mathsf{V}_k$ given by $G(z) := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \lambda x.(\beta_x)_{z(x)})$ so that we can define:

$$\Pi(\alpha, \beta) := \mathsf{sup}((d, p_2), S(\alpha, G)).$$

Given the closure properties of k-types of Theorem 1.1.8 we have that $\Sigma(\alpha, \beta)$ and $\Pi(\alpha, \beta)$ are also terms in V_k.

It can be proved that $\Sigma(\alpha, \beta)$ is the disjoint union and $\Pi(\alpha, \beta)$ is the cartesian product of the family of sets $S(\alpha, \beta)$. See [Acz82, Theorem 6.4] for more details.

Now we have two key lemmas regarding $\Pi(\alpha, \beta)$ and $\Sigma(\alpha, \beta)$.

Lemma 2.4.7. *Let $\alpha : \mathsf{V}_k$ be injectively presented, given any $\beta\_ : \mathsf{El}(\alpha) \to \mathsf{V}_k$, consider $\beta := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \beta\_)$. If β_x is injectively presented for all $x : \mathsf{El}(\alpha)$, then $\Sigma(\alpha, \beta)$ is injectively presented.*

Proof. Let $\gamma := \Sigma(\alpha, \beta)$ and consider $z_1, z_2 : C := (\Sigma x : \mathsf{El}(\alpha))\mathsf{El}(\beta_x)$, such that $(\gamma_{z_1} \approx_\infty \gamma_{z_2})$ is inhabited. We want to prove that $\mathsf{Id}_C(z_1, z_2)$ is inhabited.

Consider the projections $x_1 := z_1.1$, and $x_2 := z_2.1$, and $y_1 := z_1.2$, and $y_2 := z_2.2$. By definition $(\gamma_{z_1} \approx_\infty \gamma_{z_2})$ equals $(\langle \alpha_{x_1}, (\beta_{x_1})_{y_1}\rangle \approx_\infty \langle \alpha_{x_2}, (\beta_{x_2})_{y_2}\rangle)$.
By definition of the set-theoretic pair we have terms inhabiting the types:

$$(\alpha_{x_1} \approx_\infty \alpha_{x_2}),$$

and

$$((\beta_{x_1})_{y_1} \approx_\infty (\beta_{x_2})_{y_2}). \tag{2.1}$$

Since α is injectively presented we have $p : \mathsf{Id}_{\mathsf{El}(\alpha)}(x_1, x_2)$, which implies that $\mathsf{Id}_{\mathsf{U}}(\mathsf{el}(\beta_{x_1}), \mathsf{el}(\beta_{x_2}))$ is inhabited.

Then, by applying the dependent case of Lemma 1.1.7 to the function$\lambda\delta.(\delta)_{_}$: $(\Pi\delta : \mathsf{V}_k)\mathsf{El}(\delta) \to \mathsf{V}_k$ we have a term inhabiting the type:

$$\mathsf{Id}_{\mathsf{El}(\beta_{x_2})\to\mathsf{V}_k}(p_*(\beta_{x_1})_{\_}, (\beta_{x_2})_{\_}).$$

Now we apply the two functions $p_*(\beta_{x_1})_{\_}$ and $(\beta_{x_2})_{\_}$ to the same term $p_*(y_1)$. Hence, we get a term in:

$$\mathsf{Id}_{\mathsf{V}_k}((\beta_{x_1})_{p_*^{-1}p_*(y_1)}, (\beta_{x_2})_{p_*(y_1)}).$$

Which in turn by the elimination rule for Id-types gives a term in:

$$((\beta_{x_1})_{y_1} \approx_\infty (\beta_{x_2})_{p_*(y_1)}).$$

By transitivity of $\approx_\infty$, this together with eq. (2.1) gives a term inhabiting:

$$((\beta_{x_2})_{y_2} \approx_\infty (\beta_{x_2})_{p_*(y_1)}).$$

Since by hypothesis β_{x_2} is injectively presented we have:

$$\mathsf{Id}_{\mathsf{El}(\beta_{x_2})}(p_*(y_1), y_2).$$

Combining this with the previously established $\mathsf{Id}_{\mathsf{El}(\alpha)}(x_1, x_2)$ we obtain that the identity type $\mathsf{Id}_C((x_1, y_1), (x_2, y_2))$ is inhabited by the characterisation of $\mathsf{Id}_{\mathsf{\Sigma}}$ we used in the proof of Theorem 1.1.8. Thus we have that $\mathsf{Id}_C(z_1, z_2)$ is inhabited.

□

Similarly, $\mathsf{\Pi}(\alpha, \beta)$ also preserves injectively presented sets.

Lemma 2.4.8. *Let $\alpha : \mathsf{V}_k$ be injectively presented, given any $\beta_{\_} : \mathsf{El}(\alpha) \to \mathsf{V}_k$, consider $\beta := \mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \beta_{\_})$. If β_x is injectively presented for all $x : \mathsf{El}(\alpha)$, then $\mathsf{\Pi}(\alpha, \beta)$ is injectively presented.*

Proof. Consider $\gamma := \mathsf{\Pi}(\alpha, \beta)$, and let z_1, z_2 be terms of $D := (\Pi x : \mathsf{El}(\alpha))\mathsf{El}(\beta_x)$ such that $(\gamma_{z_1} \approx_\infty \gamma_{z_2})$, we want to show that $\mathsf{Id}_D(z_1, z_2)$.

By definition of $\mathsf{\Pi}(\alpha, \beta)$ we have that γ_{z_i} equals $S(\alpha, G(z_i))$, which in turn equals:

$$\mathsf{sup}((\mathsf{el}(\alpha), \mathsf{p}_\alpha), \lambda x.\langle \alpha_x, G(z_i)_x\rangle),$$

where G has been defined together with $\mathsf{\Pi}(\alpha, \beta)$ in Section 2.4.

By hypothesis $[\![\gamma_{z_1} \subseteq \gamma_{z_2}]\!]_{k,\infty}$, which unfolds as:

$$(\Pi x_1 : \mathsf{El}(\alpha))(\Sigma x_2 : \mathsf{El}(\alpha))(\langle \alpha_{x_1}, G(z_1)_{x_1}\rangle \approx_\infty \langle \alpha_{x_2}, G(z_2)_{x_2}\rangle). \tag{2.2}$$

Since α is injectively presented we get $\mathsf{Id}_{\mathsf{El}(\alpha)}(x_1, x_2)$, which in turn implies:

$$\mathsf{Id}_{\mathsf{V}_k}(G(z_2)_{x_1}, G(z_2)_{x_2}).$$

Then by the elimination rule for Id-types:

$$(G(z_2)_{x_1} \approx_\infty G(z_2)_{x_2}).$$

This together with (2.2) implies by transitivity of $\approx_\infty$ that the following type is inhabited:

$$(G(z_1)_{x_1} \approx_\infty G(z_2)_{x_1}),$$

which is (definitionally) equal to:

$$((\beta_{x_1})_{z_1(x_1)} \approx_\infty (\beta_{x_1})_{z_2(x_1)}).$$

Since by hypothesis β_{x_1} is injectively presented we have that the following type is inhabited:

$$(\Pi x_1 : \mathsf{El}(\alpha))\mathsf{Id}_{\mathsf{El}(\beta_{x_1})}(z_1(x_1), z_2(x_1)).$$

Therefore we can conclude using function extensionality that z_1 and z_2 are propositionally equal.

□

Aczel uses the constructions of $\Pi(\alpha, \beta)$ and $\Sigma(\alpha, \beta)$ in [Acz82] to prove the following property of strong bases, which we can easily adapt to the case of the interpretations $[\![\cdot]\!]_{k,\infty}$.

Theorem 2.4.9. *In V_k the class of strong bases is $\Pi\Sigma$-closed.*

Proof. The adaptation of this proof is straightforward, see [Acz82, Theorem 6.7]. □

Corollary 2.4.10. *$\Pi\Sigma$-AC is valid in the interpretations $[\![\cdot]\!]_{k,\infty}$ into H_k.*

Proof. Recall that the class of $\Pi\Sigma$-generated sets is the smallest $\Pi\Sigma$-closed class. Since the class of strong bases is $\Pi\Sigma$-closed, we have that every $\Pi\Sigma$-generated set is a strong base. Finally, recall that every strong base is a base. □

Corollary 2.4.11. *Let ϕ be a CC sentence. If $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty \vdash t_\phi : [\![\phi]\!]$ for some term t_ϕ of $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$, then $\mathsf{H}_k \vdash t'_\phi : [\![\phi]\!]_{k,\infty}$, for some term t'_ϕ of H_k.*

Proof. Immediate from Corollary 2.4.10 and Theorem 2.2.2. □

2.5. The proof-theoretic characterisation

We can finally put together the results of Section 2.3 and Section 2.4 and give the full characterisation of the interpretations $[\![\cdot]\!]_{k,\infty}$ with respect to *CC* sentences.

Theorem 2.5.1. *Given a CC sentence ϕ of CZF, and $2 \leq k \leq \infty$. We have that $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC} \vdash \phi$ if and only if $\mathsf{H}_k \vdash t : [\![\phi]\!]_{k,\infty}$ for some term t of H_k.*

Proof. ($\Rightarrow$) This was proved in Corollary 2.4.10.
($\Leftarrow$) This was proved in Theorem 2.3.4. □

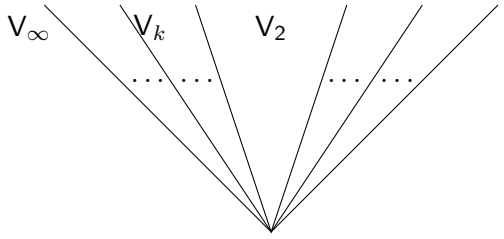
In particular we have the following corollary expressing precisely the intuition that for the purposes of interpreting set theory the homotopy levels interpreting sets do not matter.

Corollary 2.5.2. *Given a CC sentence of CZF and $2 \leq k_1, k_2 \leq \infty$ we have that $\mathsf{H}_{\mathsf{k}_1} \vdash [\![\phi]\!]_{k_1,\infty}$ if and only if $\mathsf{H}_{\mathsf{k}_2} \vdash [\![\phi]\!]_{k_2,\infty}$.*

Proof. Immediate consequence of Theorem 2.5.1. □

The interpretations $[\![\cdot]\!]_{k,\infty}$ interpret formulas in the same way, as arbitrary types. They differ in using the types V_k to interpret sets. Recall from Definition 1.2.1 that terms of V_k are constructed only from types of fixed homotopy level k. The idea is that the definition of V_k imposes a restriction on the amount of homotopical information of the type of elements $\mathsf{El}(\alpha)$ of a term $\alpha : \mathsf{V}_k$.

We can summarise the results of this chapter saying that we have a countable hierarchy of models of CZF in type theory defined by the homotopy level of the types used to interpret sets. These models approximate V_∞ which is the one originally developed by Aczel. However CZF is not able to distinguish any two of these models as they validate the same CC sentences of set theory. Hence we can argue that the essential set-theoretic content of Aczel's type-theoretic interpretation is already captured by V_2, where sets are interpreted as homotopy sets.



We leave open the question if CZF can distinguish the models via non-CC sentences.

Observe that adding the univalence axiom to H_k would break the proof presented here, since our proof relies on the interpretation $(\cdot)^*$ of H_k into $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$. The interpretation $(\cdot)^*$ is incompatible with univalence because it interprets the intensional identity type Id^i of H_k as the extensional identity type Id^e of $\mathsf{ML}_1^{\mathsf{e}}\mathsf{V}_\infty$. See

[Uni13, Example 3.1.9 and Section 7.2] for more details on the incompatibility of univalence with extensional identity types.

The interpretation $(\cdot)^*$ then allows us to use the proof-theoretic machinery of Rathjen and Tupailo, namely the interpretation $[\![\cdot]\!]^{cl}$ and the internal model $H(Y^*)$ (see point *(iii)* of 2.1). If a result similar to Theorem 2.5.1 holds for a type theory with the univalence axiom it is reasonable to expect that the proof-theoretic machinery developed in [RT06] would need to be rebuilt from scratch taking into account univalence.

2.6. A side question: ΠΣI-AC and hsets

A natural question arises regarding the role played throughout this section by the hypothesis of types having a fixed homotopy level.

There is an interesting remark regarding the $\Pi\Sigma I$ axiom of choice, which is an equivalent version of ΠΣ-AC. If we are working in V_2 with hsets, we can follow Aczel (see [Acz82, Lemma 6.6]) and give a direct proof that ΠΣI-AC is valid in V_2 making actual use of the restriction on the homotopy level.

Let us give the necessary preliminary definitions in order to introduce ΠΣI-AC.

Definition 2.6.1. In CZF, given a family of sets B indexed by a set A, we define $I(a, b) := \{z \in \{\emptyset\} \mid a \doteq b\}$.

Definition 2.6.2. In CZF we say that a class X is $\Pi\Sigma I$*-closed* if and only if it is ΠΣ-closed and also enjoys the property that $I(a, b) \in X$ for all $a, b \in A$ and all $A \in X$.

Theorem 2.6.3 (Aczel)**.** *In* CZF *there is a smallest* $\Pi\Sigma I$*-closed class, called the class of* $\Pi\Sigma I$-generated sets.

Proof. See [Acz82, Section 4.2]. □

The $\Pi\Sigma I$ axiom of choice has the same structure as the ΠΣ axiom of choice but it considers more general basic sets, i.e. the ones generated from ω via the set-theoretic operations of Π, Σ and I.

Axiom 2.6.4 (ΠΣI-AC)**.** every $\Pi\Sigma I$-generated set is a base.

However, this does not add strength to the axiom.

Theorem 2.6.5 (Aczel)**.** *In* CZF*,* ΠΣI-AC *is equivalent to* ΠΣ-AC

Proof. See [Acz86, Theorem 3.7].

□

Now we arrive at the direct proof that $\mathsf{\Pi\Sigma I\text{-}AC}$ is valid in $\mathsf{V_2}$, where we are taking advantage of the fact we are using hsets.

Definition 2.6.6. Given $\alpha : \mathsf{V_2}$ and $a, b : \mathsf{El}(\alpha)$, since $\mathsf{el}(\alpha)$ is a small hset the identity type $\mathsf{Id}_{\mathsf{El}(\alpha)}(a,b) = \mathsf{T}(\mathsf{id}(\mathsf{el}(\alpha), a, b))$ is also a small hset. Therefore we have a term witnessing this fact $p : \mathsf{isHSet_U}(\mathsf{Id}_{\mathsf{El}(\alpha)}(a,b))$. Thus we can define:

$$\hat{I}_\alpha(a,b) := \mathsf{sup}((\mathsf{id}(\mathsf{el}(\alpha)), p), \lambda x.\emptyset).$$

Lemma 2.6.7. *Given $\alpha : \mathsf{V_2}$ we have that $\hat{I}_\alpha(a,b)$ is always injectively presented.*

Proof. Let $\gamma := \hat{I}_\alpha(a,b)$ and consider $z_1, z_2 : \mathsf{Id}_{\mathsf{El}(\alpha)}(a,b)$ such that $(\gamma_{z_1} \approx_\infty \gamma_{z_2})$. Since $\mathsf{el}(\alpha)$ is a small hset we have that $\mathsf{Id}_{\mathsf{El}(\alpha)}(a,b)$ is a hproposition, and since it is inhabited by z_1, z_2 by hypothesis, it is contractible, i.e. $\mathsf{Id}_{\mathsf{Id}_{\mathsf{El}(\alpha)}(a,b)}(z_1, z_2)$ is inhabited. □

Note that Aczel in [Acz82, Lemma 6.6] can deduce that $\mathsf{Id}_{\mathsf{Id}_{\mathsf{El}(\alpha)}(a,b)}(z_1, z_2)$ is inhabited thanks to the properties of extensional identity types. Here we can do it using the properties of hsets for the type $\mathsf{V_2}$.

Theorem 2.6.8. *The class of strong bases is $\Pi\Sigma I$-closed. Therefore $\mathsf{\Pi\Sigma I\text{-}AC}$ is valid in $\mathsf{V_2}$.*

Proof. Given $\alpha : \mathsf{V_2}$ a strong base and $\beta_1, \beta_2 : \mathsf{V_2}$ elements of α, we want to show that there is a strong base $\gamma : \mathsf{V_2}$ such that for all $\eta : \mathsf{V_2}$ we have:

$$[\![\eta \in \gamma]\!]_{k,\infty} \leftrightarrow (\eta \approx_\infty \emptyset) \times (\beta_1 \approx_\infty \beta_2).$$

Without loss of generality we can assume α injectively presented, and that β_i is α_{a_i} for $a_i : \mathsf{El}(\alpha)$. Given Lemma 2.6.7, it is enough to show the following equivalence:

$$[\![\eta \in \hat{I}_\alpha(a_1, a_2)]\!]_{k,\infty} \leftrightarrow (\eta \approx_\infty \emptyset) \times (\alpha_{a_1} \approx_\infty \alpha_{a_2}).$$

Since α is injectively presented we have the (logical) equivalence:

$$(\alpha_{a_1} \approx_\infty \alpha_{a_2}) \leftrightarrow \mathsf{Id}_{\mathsf{El}(\alpha)}(a_1, a_2).$$

Hence, applying it to the definition $[\![\eta \in \hat{I}_\alpha(a_1, a_2)]\!]_{k,\infty} = \mathsf{Id}_{\mathsf{El}(\alpha)}(a_1, a_2) \times (\eta \approx_\infty \emptyset)$ we obtain the thesis. □

CHAPTER 3

Equivalence between some interpretations

This chapter presents definitions and results from [Gyl16b] and [Uni13] on the interpretations $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$ of Myhill's CST into the theories HoTT and $\mathsf{H}+\mathsf{V_H}$, with some adaptations and extensions on our part. The definition of Gylterud's interpretation $[\![\cdot]\!]_G$ and the one of the HoTT book $[\![\cdot]\!]_H$ are given in Section 3.2 and Section 3.5 respectively. They follow the same general structure as $[\![\cdot]\!]_{k,1}$, with few differences the main one being that set-theoretic equality is interpreted as the identity type Id of the type of sets.

In Gylterud's interpretation sets are interpreted as terms of the subtype of V_∞ given by the trees in which there are no repetitions of branches. Formally, $\mathsf{V_G} := (\Sigma x : \mathsf{V}_\infty)\mathsf{itset}(x)$, where itset is defined in Section 3.2. In the HoTT book interpretation sets are interpreted as terms of the higher inductive type $\mathsf{V_H}$ which is constructed in such a way that terms of $\mathsf{Id}_{\mathsf{V_H}}$ come from a truncated bisimulation relation on $\mathsf{V_H}$, see Section 3.3 for the definition.

The aim of the chapter is to prove that the interpretations $[\![\cdot]\!]_G, [\![\cdot]\!]_H$ and $[\![\cdot]\!]_{k,1}$ are equivalent for $2 \leq k \leq \infty$. The equivalence between the types $\mathsf{V_G}$ and $\mathsf{V}_\infty/\approx_1$ is stated in [Gyl16a, Proposition 8:5] and the one between $\mathsf{V}_\infty/\approx_1$ and $\mathsf{V_H}$ is stated in [Gyl16a, Remark 8:6]. However, the details of the proofs are left to the reader. We give details of the equivalence considering not only these types of sets, but also the setoids $(\mathsf{V}_\infty, \approx_1)$, $(\mathsf{V_G}, \mathsf{Id}_{\mathsf{V_G}})$ and $(\mathsf{V_H}, \mathsf{Id}_{\mathsf{V_H}})$, as well as the ones $(\mathsf{V}_k, \approx_1)$ for $2 \leq k < \infty$ in Section 3.4. Moreover, we prove that the equivalence between setoids induces a logical equivalence between the interpretations in Theorem 3.5.1, which is the main theorem of the chapter.

In this way we start to understand the relationships between some of the interpretations of the family $[\![\cdot]\!]_{k,h}$ and the ones already existing in the literature $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$. An interesting consequence of Theorem 3.5.1 is that Replacement and Exponentiation are valid in the interpretations $[\![\cdot]\!]_{k,1}$ for $2 \leq k \leq \infty$, see Corollary 3.5.3. Hence, for these values of k we can interpret CST into the type theory HoTT, which is a fact that we stated without proof in Lemma 1.3.13.

The equivalences proved in this chapter then form the basis for the analysis between the interpretations $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$ of chapter 5.

In this chapter, for notational convenience when dealing with projection functions, we use the notation $\pi_0(t)$ and $\pi_1(t)$ for the two projections of a Σ-type, rather than the notation $t.1$ and $t.2$ which are used in the rest of the thesis.

Unless stated otherwise, throughout this chapter we work in the type theory HoTT.

Outline. Section 3.1 recalls some notions and lemmas on setoids and on the epi-mono factorisation. Section 3.2 presents Gylterud's work from [Gyl16a]. The main theorem of the section is Theorem 3.2.9 which proves that there is an equivalence between the setoids $(\mathsf{V}_\infty, \approx_1)$ and $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$. Then Section 3.3 deals with the HoTT book interpretation and expands the details of [Gyl16a, Remark 8:6]. Theorem 3.3.2 proves that there is an equivalence between the setoids $(\mathsf{V}_\infty, \approx_1)$ and $(\mathsf{V}_\mathsf{H}, \mathsf{Id}_{\mathsf{V}_\mathsf{H}})$. Section 3.4 relates the other types of sets V_k for $2 \leq k < \infty$. Theorem 3.4.1 proves that there is an equivalence of setoids between $(\mathsf{V}_k, \approx_1)$ and $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$. Finally, Section 3.5, puts together the equivalences obtained in the previous sections, and Theorem 3.5.1 shows that the equivalence between setoids extends to a logical equivalence between the interpretations $[\![\cdot]\!]_G, [\![\cdot]\!]_H$ and $[\![\cdot]\!]_{k,1}$ for $2 \leq k \leq \infty$. The validity of Replacement and Exponentiation in $[\![\cdot]\!]_{k,1}$ are discussed in this section.

3.1. Preliminaries on setoids and the epi-mono factorisation

Before moving to the main content of the chapter we briefly recall from [PW14] some notions on setoids. We also recall the epi-mono factorisation of a type-theoretic map. In this chapter we use the notation $(\exists x : A)B(x)$ to refer to the truncation of the Σ-type $\left\|(\Sigma x : A)B(x)\right\|$ as doing so makes some expressions more readable.

Definition 3.1.1.

(a) A *setoid* is a pair (A, R), where A is a type and R an equivalence relation on A;

(b) A *map of setoids*, also called an *extensional function*, $F : (A, R) \to (B, Q)$ is a map $F : A \to B$ that is compatible with the relations, i.e.

$$(\Pi x, y : A)R(x, y) \to Q(F(x), F(y));$$

(c) given two setoids (A, R) and (B, Q) we say that they are *equivalent (as setoids)*, written $(A, R) \cong (B, Q)$, if and only if there are maps of setoids $F : A \to B$ and $G : B \to A$ such that for all $x : A$ we have $R(GF(x), x)$ and for all $y : B$ we have $Q(FG(y), y)$.

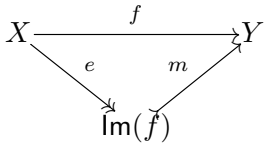
Notice that given setoids of the form (A, Id_A) and (B, Id_B), equivalence of setoids implies that the type $\mathsf{Equiv}(A, B)$ is inhabited.

Lemma 3.1.2. *Given setoids (A, R) and (B, Id_B) such that the type B is a hset, an equivalence between them induces an equivalence between the set-quotient $(A/R, \mathsf{Id}_{A/R})$ and (B, Id_B).*

Proof. The claim follows from the fact that an equivalence between the setoids (A, R) and (B, Q) induces an equivalence between the set quotients $(A/R, \mathsf{Id}_{A/R})$ and $(B/Q, \mathsf{Id}_{B/Q})$. Moreover, if B is a hset we have that the trivial set quotient B/Id_B is equivalent to B. □

Now recall the construction of the epi-mono factorisation in type theory.

Definition 3.1.3. Given a map $f : X \to Y$, its *epi-mono factorisation* is the factorisation of f as:



where the image is defined as:

$$\mathsf{Im}(f) := (\Sigma y : Y)(\exists x : X)\mathsf{Id}_Y(f(x), y),$$

and the maps e and m are:

$$e(x) := (f(x), |(x, \mathsf{refl}_{f(x)})|),$$

$$m(t) := \pi_0(t).$$

It is straightforward to check that e is an epi and m a mono in the sense that the following types are inhabited:

$$(\Pi x : \mathsf{Im}(f))(\exists y : X)\mathsf{Id}_{\mathsf{Im}(f)}(e(y), x),$$

and

$$(\Pi x, y : X)\mathsf{Id}_{\mathsf{Im}(f)}(f(x), f(y)) \to \mathsf{Id}_X(x, y).$$

Now we give a useful sufficient condition for $\mathsf{Im}(f)$ being equivalent to a small type, that is one of the key lemmas that allow to prove the equivalence between setoids in Theorem 3.2.9, Theorem 3.3.2 and Theorem 3.4.1. We briefly recall the notion of a locally small type that is used in the proof of Lemma 3.1.5.

Definition 3.1.4. A type A is *locally small* if and only if its identity types are equivalent to small types, i.e. for all $x, y : A$ there are $\mathsf{i}(x, y) : \mathsf{U}$ such that

$\mathsf{Id}_A(x,y) \simeq \mathsf{T}(\mathsf{i}(x,y))$.

The following lemma is folklore, we could not find a reference for it in the form we stated it, so we provide details for the proof. Our proof is based on the one of [Uni13, Lemma 10.5.6].

Lemma 3.1.5. *Assume* U *closed under set quotients. Let* $f : A \to B$ *with* A *small and* B *a locally small hset. Then the image factorisation* $\mathsf{Im}(f)$ *is equivalent to a small type.*

Proof. Since B is locally small, and $\mathsf{Im}(f) = (\Sigma b : B)(\exists a : A)\mathsf{Id}_B(f(a), b)$ then for any two terms $x, y : \mathsf{Im}(f)$ we have an equivalence with a small type $\mathsf{Id}_{\mathsf{Im}(f)}(x,y) \simeq \mathsf{T}(\mathsf{i}(x,y))$. Consider the small relation R on A given by:

$$x, y : A \vdash R(x,y) := \mathsf{T}(\mathsf{i}(e(x), e(y))),$$

where $e : A \to \mathsf{Im}(f)$ is the epi of the epi-mono factorisation.

Since B is a hset $R(x,y)$ is a family of hproposition, so that we can take the set-quotient A/R which is small by hypothesis. Let $q : A \to A/R$ be the projection onto the set quotient, see Section 1.1.

Now we prove that the image $\mathsf{Im}(f)$ is equivalent to A/R. We construct a map $\mathsf{Im}(f) \to A/R$ using the principle of unique choice (Lemma 1.1.10). One first proves that the following type is inhabited:

$$(\Pi z : \mathsf{Im}(f))(\exists! \alpha : A/R)(\exists x : A)\mathsf{Id}_{\mathsf{Im}(f)}(e(x), z) \times \mathsf{Id}_{A/R}(q(x), \alpha). \tag{3.1}$$

In this context the uniqueness condition $(\exists! x : A)B(x)$ is defined in terms of the identity type of A as:

$$(\exists x : A)B(x) \times (\Pi z_1, z_2 : A)B(z_1) \times B(z_2) \to \mathsf{Id}_A(z_1, z_2).$$

The uniqueness condition in eq. (3.1) comes from the definition of $\mathsf{Id}_{A/R}$ of the set-quotient. Then by defining:

$$P(z) := (\exists \alpha : A/R)(\exists x : A)\mathsf{Id}_{\mathsf{Im}(f)}(e(x), z) \times \mathsf{Id}_{A/R}(q(x), \alpha),$$

we can apply the principle of unique choice to the family $z : \mathsf{Im}(f) \vdash P(z)$ and have a function $\mathsf{Im}(f) \to A/R$.

Similarly for the other direction one proves:

$$(\Pi \alpha : A/R)(\exists! z : \mathsf{Im}(f))(\exists x : A)\mathsf{Id}_{\mathsf{Im}(f)}(e(x), z) \times \mathsf{Id}_{A/R}(q(x), \alpha),$$

by using the universal property of the quotient Lemma 1.1.13.

Then it is straightforward to check that this gives an equivalence of types. □

Given our definition of the type theory HoTT, given in Section 1.1, we can always perform the operation of set-quotients, and moreover the universe is closed under set-quotients.

3.2. Equivalence with Gylterud's interpretation

In this section we recall definitions and results from [Gyl16a] that lead to Theorem 3.2.9, also proved in [Gyl16a].

Observe that the bisimulation relation $\approx_\infty$ in V_∞ allows us to identify trees, recursively, up to permutation and repetition of their branches. This is the reason why $\approx_\infty$ is used to interpret set-theoretic equality, precisely because this makes the Extensionality axiom valid.

If we just consider the type V_∞ with its identity type $\mathsf{Id}_{\mathsf{V}_\infty}$ instead, we are not able to identify extensionally equal trees. However, in presence of univalence we have the following lemma.

Lemma 3.2.1 (Gylterud)**.** *Given $x, y : \mathsf{V}_\infty$, the identity type $\mathsf{Id}_{\mathsf{V}_\infty}(x, y)$, is equivalent to the type $(x \sim y)$, where the type $\sim$ is defined as follows:*

$$(\mathsf{sup}(a, f) \sim \mathsf{sup}(b, g)) := (\Sigma\alpha : \mathsf{T}(a) \simeq \mathsf{T}(b))(\Pi x : \mathsf{T}(a))(f(x) \sim g(\alpha(x))).$$

Proof. See [Gyl16b, Theorem 3.9], note that this proof uses the propositional η-rule. □

This lemma gives a description of the identity type $\mathsf{Id}_{\mathsf{V}_\infty}$ that thanks to the condition that asks for an equivalence $\alpha : \mathsf{T}(a) \simeq \mathsf{T}(b)$, it allows to identify trees up to a permutation of their branches. Indeed, given $\mathsf{sup}(a, f) : \mathsf{V}_\infty$, the type $\mathsf{T}(a)$ gives indices for the branches and the function $f : \mathsf{T}(a) \to \mathsf{V}_\infty$ assigns a sub-tree to every branch. Since there is an equivalence $\mathsf{T}(a) \simeq \mathsf{T}(b)$ on the indices for the branches, trees are identified up to permutations of their branches.

If we want to validate Extensionality we just need to deal with repetitions, and this can be done by simply restricting to the type V_G of trees that have no repetitions of branches. The following definition makes this idea precise.

Definition 3.2.2. Recall that a function $f : A \to B$ is an *embedding* if and only if the map induced on the identity types $\mathsf{ap}_f : \mathsf{Id}_A(x, y) \to \mathsf{Id}_B(f(x), f(y))$ is an equivalence. The type $\mathsf{Embedding}(f)$ is defined as follows:

$$(\Pi x, y : A)\mathsf{isequiv}(\mathsf{ap}_f).$$

With the notion of embedding we can require the branches of a tree $\mathsf{sup}(a, f)$ in V_∞ to have no repetitions, simply by asking that the function f selecting the branches is an embedding, iteratively. More formally, consider the family of types $\mathsf{itset}(x)$ constructed by recursion on $x : \mathsf{V}_\infty$ as:

$$\mathsf{itset}(\mathsf{sup}(a, f)) := \mathsf{Embedding}(f) \times (\Pi i : \mathsf{T}(a))\mathsf{itset}(f(a)).$$

Then the type $\mathsf{V_G}$ is defined as $(\Sigma x : \mathsf{V}_\infty)\mathsf{itset}(x)$.

Remark 3.2.3.

(i) If we assume FE function extensionality then $\mathsf{Embedding}(f)$ is always a hproposition. Thus $\mathsf{itset}(x)$ is a hproposition for any $x : \mathsf{V}_\infty$;

(ii) in the presence of FE embeddings have propositional fibres [Uni13, Lemma 7.6.2]. Recall that the fibre of $f : A \to B$ over $b : B$ is defined as the type $\mathsf{Fib_f}(b) := (\Sigma x : A)\mathsf{Id}_B(f(x), b)$.

Gylterud's interpretation of the membership relation in V_∞ and $\mathsf{V_G}$ is defined by recursion as follows. Consider $\alpha, \beta : \mathsf{V}_\infty$ with $\beta = \mathsf{sup}(b, g)$, and terms $b : \mathsf{U}$, $g : \mathsf{T}(a) \to \mathsf{V}_\infty$. Then one defines:

$$[\![\alpha \in \beta]\!]_G := (\Sigma i : \mathsf{T}(b)\mathsf{Id}_{\mathsf{V}_\infty}(g(i), \alpha)).$$

The interpretation $[\![\cdot]\!]_G$ is extended from V_∞ to $\mathsf{V_G}$, so given $x, y : \mathsf{V_G}$ one defines:

$$[\![x \in y]\!]_G := [\![\pi_0(x) \in \pi_0(y)]\!]_G;$$
$$[\![x \doteq y]\!]_G := \mathsf{Id}_{\mathsf{V_G}}(x, y).$$

Note that the Σ-type in the interpretation of $\in$ is not truncated, while that is the case for the interpretation $[\![\cdot]\!]_{\infty,1}$. However, as we see in Lemma 3.2.4 below, the type $[\![x \in y]\!]_G$ is always a hproposition.

The rest of the interpretation has the same structure as $[\![\cdot]\!]_{\infty,1}$:

$$[\![\bot]\!]_G := \mathsf{0};$$
$$[\![\phi \Rightarrow \psi]\!]_G := [\![\phi]\!]_G \to [\![\psi]\!]_G;$$
$$[\![\phi \wedge \psi]\!]_G := [\![\phi]\!]_G \times [\![\psi]\!]_G;$$
$$[\![\phi \vee \psi]\!]_G := \left\| [\![\phi]\!]_G + [\![\psi]\!]_G \right\|;$$
$$[\![\forall x\, \phi(x)]\!]_G := (\Pi \alpha : \mathsf{V_G})[\![\phi(\alpha)]\!]_G;$$
$$[\![\exists x\, \phi(x)]\!]_G := \left\| (\Sigma \alpha : \mathsf{V_G})[\![\phi(\alpha)]\!]_G \right\|.$$

We now present some lemmas that are useful in relating the types of sets V_∞ and V_G.

Lemma 3.2.4 (Gylterud)**.** *For $\alpha : \mathsf{V}_\infty$ and $x : \mathsf{V}_\mathsf{G}$ the type $[\![\alpha \in \pi_0(x)]\!]_G$ is a hproposition.*

Proof. Suppose $x = (\mathsf{sup}(a, f), p)$. By definition:

$$[\![\alpha \in \mathsf{sup}(a, f)]\!]_G = (\Sigma i : \mathsf{T}(a))\mathsf{Id}(f(i), \alpha).$$

Observe that the right hand side is $\mathsf{Fib}_\mathsf{f}(\alpha)$ i.e. the fibre of f over α. By Remark 3.2.3 embeddings have propositional fibres. □

Lemma 3.2.5 (Gylterud)**.** *We have the following facts about $\mathsf{Id}_{\mathsf{V}_\mathsf{G}}$:*

(i) *for all $x, y : \mathsf{V}_\mathsf{G}$ we have an equivalence:*

$$\mathsf{Id}_{\mathsf{V}_\mathsf{G}}(x, y) \simeq (\Pi z : \mathsf{V}_\mathsf{G})([\![z \in x]\!]_G \leftrightarrow [\![z \in y]\!]_G),$$

(ii) *V_G is a hset;*

(iii) *the Extensionality axiom is valid in $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}}, [\![\cdot]\!]_G)$.*

Proof. (i) See [Gyl16a, extensionality 6.1]. The proof uses [Gyl16b, Theorem 3:14] and Lemma 3.2.4.

(ii) Note that Lemma 3.2.4 and point (i) together imply that $\mathsf{Id}_{\mathsf{V}_\mathsf{G}}$ is equivalent to a hprop.

(iii) Immediate consequence of (i).

□

Lemma 3.2.6 (Gylterud)**.** *Given a small type $a : \mathsf{U}$ and a function $f : \mathsf{T}(a) \to \mathsf{V}_\mathsf{G}$ there is $\mathsf{image}(a, f) : \mathsf{V}_\mathsf{G}$ such that:*

(i) *for each $i : \mathsf{T}(a)$ we have a term in the type $[\![f(i) \in \mathsf{image}(a, f)]\!]_G$;*

(ii) *for any $P : \mathsf{V}_\mathsf{G} \to \mathsf{U}$ such that $P(\alpha)$ are all hprops, given $(\Pi i : \mathsf{T}(a))P(f(i))$ we have that $(\Pi z : \mathsf{V}_\mathsf{G})([\![z \in \mathsf{image}(a, f)]\!]_G \to P(z))$.*

Proof. To construct $\mathsf{image}(a, f)$, take the epi-mono factorisation of $f : \mathsf{T}(a) \to \mathsf{V}_\mathsf{G}$ as $e : \mathsf{T}(a) \to \mathsf{Im}(f)$ followed by $m : \mathsf{Im}(f) \to \mathsf{V}_\mathsf{G}$.

Since $\mathsf{itset}(x)$ is always a hproposition, we have that $\mathsf{Id}_{\mathsf{V}_\mathsf{G}}$ and $\mathsf{Id}_{\mathsf{V}_\infty}$ are equivalent, and that in presence of univalence the latter can be described as the following type (see [Gyl16b, Theorem 3.9]):

$$(\mathsf{sup}(a, f) \sim \mathsf{sup}(b, g)) := (\Sigma \alpha : \mathsf{T}(a) \simeq \mathsf{T}(b))(\Pi x : \mathsf{T}(a))(f(x) \sim g(\alpha(x))).$$

Hence V_G is locally small.

Then Lemma 3.1.5 guarantees that $\mathsf{Im}(f)$ is equivalent to a small type $\mathsf{T}(b)$. So we can define $\mathsf{image}(a, f) := (\mathsf{sup}(b, m'), p)$, where $m' := \pi_0 \circ m$, and $\pi_0 : \mathsf{V}_\mathsf{G} \to \mathsf{V}_\infty$ and $p : \mathsf{itset}(\mathsf{sup}(b, m'))$. It is straightforward to check the other properties of $\mathsf{image}(a, f)$. □

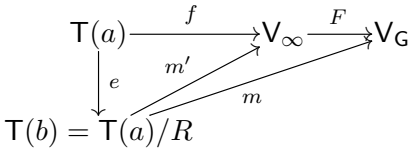
By recursion on V_∞ we construct the *iterative image* map $F : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{G}$. For a canonical term $\mathsf{sup}(a, f) : \mathsf{V}_\infty$ we define:

$$F(\mathsf{sup}(a, f)) := \mathsf{image}(a, F \circ f).$$

Lemma 3.2.7 and Lemma 3.2.8 are key in proving that the map $F : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{G}$ and the projection $\pi_0 : \mathsf{V}_\mathsf{G} \to \mathsf{V}_\infty$ give us an equivalence of setoids.

Lemma 3.2.7 (Gylterud)**.** *For each $x : \mathsf{V}_\infty$, the type $(x \approx_1 \pi_0 F(x))$ is inhabited.*

Proof. Same proof as [Gyl16a, Lemma 8.3]. Given $\mathsf{sup}(a, f) : \mathsf{V}_\infty$, we have that $F(\mathsf{sup}(a, f)) = (\mathsf{sup}(b, m'), p)$, where b and m are given by the epi-mono factorisation of f, and $p : \mathsf{itset}(\mathsf{sup}(b, m'))$, and $m' := \pi_0 \circ m$, as in the following diagram:



Then we can prove $(\Pi x : \mathsf{T}(a))(\Sigma y : \mathsf{T}(b))(f(x) \approx_1 m'(y))$ by simply taking $y := e(x)$. On the other hand we can prove $(\Pi y : \mathsf{T}(b))(\exists y : \mathsf{T}(a))(f(x) \approx_1 m'(y))$, by surjectivity of the projection $e : \mathsf{T}(a) \to \mathsf{T}(a)/R = \mathsf{T}(b)$. □

Lemma 3.2.8 (Gylterud)**.** *There is a function inhabiting the following type:*

$$(\Pi x, y : \mathsf{V}_\infty)\mathsf{itset}(x) \times \mathsf{itset}(y) \times (x \approx_1 y) \to \mathsf{Id}_{\mathsf{V}_\infty}(x, y).$$

Proof. We expand some details in the proof of [Gyl16a, Lemma 8.4].

By W-recursion suppose we have $(\mathsf{sup}(a, f) \approx_1 \mathsf{sup}(b, g))$ and $s : \mathsf{itset}(\mathsf{sup}(a, f))$ and $t : \mathsf{itset}(\mathsf{sup}(b, g))$.

In order to show that $\mathsf{Id}_{\mathsf{V}_\infty}(\mathsf{sup}(a, f), \mathsf{sup}(b, g))$ is inhabited it is enough to show that this type:

$$\mathsf{Id}_{\mathsf{V}_\mathsf{G}}((\mathsf{sup}(a, f), s), (\mathsf{sup}(b, g), t)),$$

is inhabited, because $\mathsf{itset}(x)$ is always a hproposition.

Since the extensionality axiom is valid in $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$ (see Lemma 3.2.5) in order to prove that there is an identity in V_G between x and y it is enough to show that for any $z : \mathsf{V}_\mathsf{G}$ we have that $[\![z \in x]\!]_G \leftrightarrow [\![z \in y]\!]_G$.

In one direction, given $[\![z \in \mathsf{sup}(a,f)]\!]_G$, by definition of $\in$, we have that the type $(\Sigma i : \mathsf{T}(a))\mathsf{Id}_{\mathsf{V}_\infty}(z, f(i))$ is inhabited. Since y is an iterative set, the type $[\![z \in y]\!]_G$ is a hproposition by Lemma 3.2.4. Hence, the hypothesis $(\mathsf{sup}(a,f) \approx_1 \mathsf{sup}(b,g))$ gives by the universal property of the propositional truncation that

$$(\Sigma j : \mathsf{T}(b))(f(i) \approx_1 g(j)),$$

is also inhabited. Then by inductive hypothesis $(f(i) \approx_1 g(j)) \to \mathsf{Id}_{\mathsf{V}_\infty}(f(i), g(j))$.

Similarly for the other direction.

□

We can finally state and prove the following equivalence.

Theorem 3.2.9. *The setoids* $(\mathsf{V}_\infty, \approx_1)$ *and* $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$ *are equivalent.*

Proof. We are going to prove that the function iterative image $F : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{G}$ and the projection $\pi_0 : \mathsf{V}_\mathsf{G} \to \mathsf{V}_\infty$ are mutually inverse.

One direction is given by Lemma 3.2.7. For the other direction we want to prove that $(\Pi t : \mathsf{V}_\mathsf{G})\mathsf{Id}_{\mathsf{V}_\mathsf{G}}(F(\pi_0 t), t)$. Recall that by definition $\mathsf{V}_\mathsf{G} = (\Sigma x : \mathsf{V}_\infty)\mathsf{itset}(x)$.

Given $t : \mathsf{V}_\mathsf{G}$, we apply Lemma 3.2.7 to $\pi_0(t) : \mathsf{V}_\infty$ and we get a term inhabiting the type:

$$(\pi_0 F \pi_0(t) \approx_1 \pi_0(t)).$$

Then we apply Lemma 3.2.8 to $x := \pi_0 F \pi_0(t)$ and $y := \pi_0(t)$ so that we have a term inhabiting:

$$\mathsf{Id}_{\mathsf{V}_\infty}(\pi_0 F \pi_0(t), \pi_0(t)).$$

Since $\mathsf{itset}(x)$ is a hproposition, an identity in V_∞ between iterative sets gives an identity in V_G. As a result we have a term in $\mathsf{Id}_{\mathsf{V}_\mathsf{G}}(F\pi_0(t), t)$ as desired. □

Corollary 3.2.10 (Gylterud)**.** *The setoids* $(\mathsf{V}_\infty/_{\approx_1}, \mathsf{Id}_{\mathsf{V}_\infty/_{\approx_1}})$ *and* $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$ *are equivalent.*

Proof. Immediate consequence of Lemma 3.1.2 and Theorem 3.2.9. □

3.3. Equivalence with the HoTT book interpretation

In this section we work in the type theory $\mathsf{HoTT} + \mathsf{V}_\mathsf{H}$. Let us recall the definition of the higher inductive type V_H used in [Uni13, Section 10.5]. We prove in Corollary 3.3.3 that the setoids $(\mathsf{V}_\mathsf{H}, \mathsf{Id}_{\mathsf{V}_\mathsf{H}})$ and $(\mathsf{V}_\infty/\approx_1, \mathsf{Id}_{\mathsf{V}_\infty/\approx_1})$ are equivalent (which is stated in [Gyl16a, Remark 8.6]).

In order to make the definition of the higher inductive type V_H more readable we introduce an auxiliary definition. Consider $\mathsf{set}(a,f), \mathsf{set}(b,g) : \mathsf{V}_\mathsf{H}$ two canonical terms of V_H, given by the introduction rule below. Then we define the following type:

$$\mathsf{Bisim\_Id}(\mathsf{set}(a,f),\mathsf{set}(b,g)) :=$$
$$(\Pi x:\mathsf{T}(a))\Big\|(\Sigma y:\mathsf{T}(b))\mathsf{Id}_{\mathsf{V_H}}(f(x),g(y))\Big\|\times(\Pi y:\mathsf{T}(b))\Big\|(\Sigma x:\mathsf{T}(a))\mathsf{Id}_{\mathsf{V_H}}(f(x),g(y))\Big\|.$$

The informal idea behind the definition of $\mathsf{V_H}$ is to start from the type V_∞ and to add identities corresponding to the terms of the truncated bisimulation $\approx_1$, and we also make $\mathsf{V_H}$ into a hset by adding terms into $\mathsf{Id}_{\mathsf{Id}_{\mathsf{V_H}}}$.

The type $\mathsf{V_H}$ is generated by the following constructors:

(i) for every $a:\mathsf{U}$ and $f:\mathsf{T}(a)\to\mathsf{V_H}$ a term $\mathsf{set}(a,f)$;
(ii) for every $a,b:\mathsf{U}$ and $f:\mathsf{T}(a)\to\mathsf{V_H}$ and $g:\mathsf{T}(b)\to\mathsf{V_H}$ such that $\mathsf{Bisim\_Id}(\mathsf{set}(a,f),\mathsf{set}(b,g))$, a path in $\mathsf{Id}_{\mathsf{V_H}}(\mathsf{set}(a,f),\mathsf{set}(b,g))$;
(iii) for all $x,y:\mathsf{V_H}$ and $p,q:\mathsf{Id}_{\mathsf{V_H}}(x,y)$ a term in $\mathsf{Id}_{\mathsf{Id}_{\mathsf{V_H}}}(p,q)$.

Given a family of hsets $P(x)$ for $x:\mathsf{V_H}$, the elimination rule gives a way to construct a dependent function $h:(\Pi x:\mathsf{V_H})P(x)$, provided that the following clauses are satisfied. We write the elimination rule, so that these clauses also express the computation rule.

(1) for any $a:\mathsf{U}$ and $f:\mathsf{T}(a)\to\mathsf{V_H}$ construct $h(\mathsf{set}(a,f))$ assuming given $h(f(x))$ for all $x:\mathsf{T}(a)$;
(2) verify that if we have $a,b:\mathsf{U}$ and $f:\mathsf{T}(a)\to\mathsf{V_H}$ and $g:\mathsf{T}(b)\to\mathsf{V_H}$ such that $\mathsf{Bisim\_Id}(\mathsf{set}(a,f),\mathsf{set}(b,g))$ holds, then:

$$\mathsf{Id}_{P(\mathsf{set}(b,g))}(q_*(h(\mathsf{set}(a,f))),h(\mathsf{set}(b,g))),$$

where q is the path arising from the second constructor (ii) and the type $\mathsf{Bisim\_Id}(\mathsf{set}(a,f),\mathsf{set}(b,g))$. Assuming inductively that $h(f(x))$ and $h(g(y))$ are defined for all $x:\mathsf{T}(a)$ and $y:\mathsf{T}(b)$, and that the following two conditions holds:

$$(3.1)\quad (\Pi x:\mathsf{T}(a))(\exists y:\mathsf{T}(b))(\exists p:\mathsf{Id}_{\mathsf{V_H}}(f(x),g(y))\,\mathsf{Id}_{P(h(g(y)))}(p_*h(f(x)),h(g(y))),$$

$$(3.2)\quad (\Pi y:\mathsf{T}(b))(\exists x:\mathsf{T}(a))(\exists p:\mathsf{Id}_{\mathsf{V_H}}(f(x),g(y))\,\mathsf{Id}_{P(h(g(y)))}(p_*h(f(x)),h(g(y))).$$

Remark 3.3.1. Observe that in the section on the interpretation of set theory of the HoTT book [Uni13, Section 10.5] univalence is used, but only in the proof of Lemma 10.5.6 and Lemma 10.5.9. We claim that these lemmas can be proved without univalence.

Lemma 10.5.6 of the HoTT book states that given $u:\mathsf{V_H}$ there is a small type $a_u:\mathsf{U}$ and a monic $m:\mathsf{T}(a_u)\to\mathsf{V_H}$ such that one has $\mathsf{Id}_{\mathsf{V_H}}(u,\mathsf{set}(a_u,m))$. The proof

describes a construction of the small image factorisation of a map $f : \mathsf{T}(a) \to \mathsf{V_H}$, which is unique up to equivalence and hence by univalence up to identity. This fact is used to construct explicit terms a_u and m. However, the construction itself provides explicitly the terms a_u and m as we have seen in Lemma 3.1.5 of the previous section and there is no need to prove their uniqueness.
Lemma 10.5.9 of the HoTT book shows that a Δ_0 class $C : \mathsf{V_H} \to \mathsf{HProp_U}$ is small. Univalence is used in the proof to deduce an identity from a logical equivalence between propositions. But there is no need for such identity if the lemma is rephrased directly in terms of a set-theoretic interpretations $[\![\cdot]\!]_H : \mathsf{CST} \to \mathsf{HoTT}$, stating that given ϕ a Δ_0 formula of CST its interpretation $[\![\phi]\!]_H$ is equivalent to a small type.

Now we construct the equivalence with the HoTT book interpretation. Recall from Definition 3.1.1 that in order to construct an equivalence between the setoids $(\mathsf{V}_\infty, \approx_1) \cong (\mathsf{V_H}, \mathsf{Id}_{\mathsf{V_H}})$ we need to construct a map out of $\mathsf{V_H}$, but for that we need to eliminate into a hset. So we first construct a map $K' : \mathsf{V_H} \to \mathsf{V_G}$, since $\mathsf{V_G}$ is a hset thanks to Lemma 3.2.5. Then we define $H : \mathsf{V}_\infty \to \mathsf{V_H}$ and prove in Theorem 3.3.2 that H and

$$K = \pi_0 \circ K' : \mathsf{V_H} \to \mathsf{V}_\infty,$$

give an equivalence of setoids. Here $\pi_0 : \mathsf{V_G} \to \mathsf{V}_\infty$ is the projection on the first component. To do this we use the equivalence of setoids $(\mathsf{V}_\infty, \approx_1) \cong (\mathsf{V_G}, \mathsf{Id}_{\mathsf{V_G}})$ proved in Theorem 3.2.9.

We define K' by recursion on $\mathsf{V_H}$: suppose we are given $\mathsf{set}(a, f)$ for a map $f : \mathsf{T}(a) \to \mathsf{V_H}$ and the map $K' : \mathsf{V_H} \to \mathsf{V_G}$ already defined on the terms $f(x)$ for $x : \mathsf{T}(a)$.

Recall from Theorem 3.2.9 that we have maps $F : \mathsf{V}_\infty \to \mathsf{V_G}$ and $\pi_0 : \mathsf{V_G} \to \mathsf{V}_\infty$ giving an equivalence of setoids. We define the map $K' : \mathsf{V_H} \to \mathsf{V_G}$ as follows:

$$K'(\mathsf{set}(a, f)) := F(\mathsf{sup}(a, \pi_0 K' f)).$$

This definition respects the clauses 3.1 and 3.2. Indeed, suppose given a term in $\mathsf{Bisim\_Id}(\mathsf{set}(a, f), \mathsf{set}(b, g))$, we want to construct a term inhabiting:

$$\mathsf{Id}_{\mathsf{V_G}}(F(\mathsf{sup}(a, \pi_0 K' f)), F(\mathsf{sup}(b, \pi_0 K' g))),$$

recalling that F forms an equivalence of setoids, it is enough to show that we have: $\mathsf{sup}(a, \pi_0 K' f) \approx_1 \mathsf{sup}(b, \pi_0 K' g)$ which follows from the hypothesis that the type $\mathsf{Bisim\_Id}(\mathsf{set}(a, f), \mathsf{set}(b, g))$ is inhabited.

The map $H : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{H}$ is readily defined by recursion as follows:

$$H(\mathsf{sup}(a, f)) := \mathsf{set}(a, H \circ f),$$

for $f : \mathsf{T}(a) \to \mathsf{V}_\infty$. Note that an induction on V_∞ gives that H is a map of setoids $H : (\mathsf{V}_\infty, \approx_1) \to (\mathsf{V}_\mathsf{H}, \mathsf{Id}_{\mathsf{V}_\mathsf{H}})$.

Theorem 3.3.2. *The functions $H : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{H}$ and $K = \pi_0 \circ K' : \mathsf{V}_\mathsf{H} \to \mathsf{V}_\infty$ give an equivalence of setoids $(\mathsf{V}_\infty, \approx_1) \cong (\mathsf{V}_\mathsf{H}, \mathsf{Id}_{\mathsf{V}_\mathsf{H}})$.*

Proof. First, we want to prove that $(KH(\alpha) \approx_1 \alpha)$ for α. By induction on $\alpha = \mathsf{sup}(a, f)$ we have that:

$$\begin{aligned} KH(\mathsf{sup}(a, f)) &= \pi_0 \circ K' \circ H(\mathsf{sup}(a, f)) && \text{(by definition of } K) \\ &= \pi_0 \circ K'(\mathsf{set}(a, H \circ f)) && \text{(by definition of } H) \\ &= \pi_0 \circ F(\mathsf{sup}(a, \pi_0 \circ K' \circ H \circ f)). && \text{(by definition of } K') \end{aligned}$$

Since π_0 and F form an equivalence of setoids we have:

$$(\pi_0 \circ F(\mathsf{sup}(a, \pi_0 \circ K' \circ H \circ f)) \approx_1 \mathsf{sup}(a, \pi_0 \circ K' \circ H \circ f)),$$

and we conclude using the inductive hypothesis.

Then we want to construct a term $d(x) : \mathsf{Id}_{\mathsf{V}_\mathsf{H}}(HK(x), x)$ for $x : \mathsf{V}_\mathsf{H}$, so that we can conclude that H and K form an equivalence. Since $\mathsf{Id}_{\mathsf{V}_\mathsf{H}}$ is a hprop it is also a hset, we can apply the elimination rule of V_H to prove it by induction.

So, consider $\mathsf{set}(a, f)$ for $f : \mathsf{T}(a) \to \mathsf{V}_\mathsf{H}$. We have:

$$HK(\mathsf{set}(a, f)) = H \circ \pi_0 \circ K'(\mathsf{set}(a, f)) \qquad \text{(by definition of } K) \tag{3.3}$$

$$= H \circ \pi_0 \circ F(\mathsf{sup}(a, \pi_0 \circ K' \circ f)), \text{ (by definition of } K') \tag{3.4}$$

then recall that $(\pi_0 \circ F(\alpha)) \approx_1 \alpha$ for any $\alpha : \mathsf{V}_\infty$ and since H is a map of setoids we have a term in the identity type:

$$\mathsf{Id}_{\mathsf{V}_\mathsf{H}}(H \circ \pi_0 \circ F(\alpha), H(\alpha)). \tag{3.5}$$

Recall that by definition of H we have the definitional equality:

$$H(\mathsf{sup}(a, \pi_0 \circ K' \circ f)) = \mathsf{set}(a, H \circ \pi_0 \circ K' \circ f). \tag{3.6}$$

Concatenating the previous identities eq. (3.4), eq. (3.5) and eq. (3.6) and applying the inductive hypothesis we obtain $d(a, f) : \mathsf{Id}_{\mathsf{V}_\mathsf{H}}(HK(\mathsf{set}(a, f)), \mathsf{set}(a, f))$ as desired.

Lastly, we need to check that this induction respected the clauses 3.1 and 3.2 of the elimination rule of V_H. But this is immediate since given $\mathsf{set}(a, f), \mathsf{set}(b, g) : \mathsf{V}_\mathsf{H}$ the terms $d(a, f)$ and $d(b, g)$ are terms in $\mathsf{Id}_{\mathsf{V}_\mathsf{H}}$ and V_H is a hset. □

Corollary 3.3.3 (Gylterud)**.** *There is an equivalence of setoids between* $(\mathsf{V}_{\mathsf{H}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{H}}})$ *and* $(\mathsf{V}_\infty/_{\approx_1}, \mathsf{Id}_{\mathsf{V}_\infty})$.

Proof. Applying Lemma 3.1.2 to the equivalence of setoids $(\mathsf{V}_\infty, \approx_1) \cong (\mathsf{V}_{\mathsf{H}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{H}}})$ of Theorem 3.3.2 we obtain the equivalence $(\mathsf{V}_\infty/_{\approx_1}, \mathsf{Id}_{\mathsf{V}_\infty/_{\approx_1}}) \cong (\mathsf{V}_{\mathsf{H}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{H}}})$.

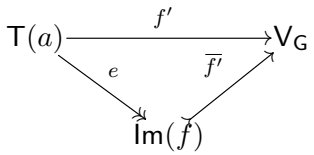
□

3.4. Equivalence with the interpretations $[\![\cdot]\!]_{k,1}$

In this section we prove the equivalence between the setoids $(\mathsf{V}_{\mathsf{G}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{G}}})$ and $(\mathsf{V}_k, \approx_1)$. This section was inspired by the comment in [Led14, Section 3.2] that taking the epi-mono factorisation of the map $f : \mathsf{T}(a) \to \mathsf{V}_{\mathsf{H}}$ produces a small hset $\mathsf{Im}(f)$. The results of this section combined with the ones of Theorem 3.3.2 on the HoTT book interpretation make precise and fully explicit the link between V_{H} and hsets observed in [Led14].

Given $2 \leq k < \infty$ we can easily define a map $F_k : \mathsf{V}_k \to \mathsf{V}_{\mathsf{G}}$ by recursion on V_k by simply forgetting the extra information on the hlevel: $F_k(\mathsf{sup}((a,p),f)) := \mathsf{sup}(a, F_k \circ f)$.

In order to define a map in the other direction consider a canonical term $(\mathsf{sup}(a,f),i) : \mathsf{V}_{\mathsf{G}}$ and consider the function $f' : \mathsf{T}(a) \to \mathsf{V}_{\mathsf{G}}$ induced by f that maps $x : \mathsf{T}(a)$ to $(f(x), i(f(x)))$. Then take its epi-mono factorisation:



Due to Lemma 3.1.5, the type $\mathsf{Im}(f)$ is equivalent to $\mathsf{T}(a)/R$ which is a small hset, i.e. there is a code $h(a) : \mathsf{U}$ such that $\mathsf{T}(h(a)) = \mathsf{T}(a)/R$, and $p : \mathsf{isHSet}(\mathsf{T}(h(a)))$. So we can define the map $G_k : \mathsf{V}_{\mathsf{G}} \to \mathsf{V}_k$ as:

$$G_k(\mathsf{sup}(a,f)) := \mathsf{sup}((h(a),p), G_k \circ \bar{f}').$$

Note that for $k = \infty$ we have $F_\infty = F$ and $G_\infty = \pi_0$.

Theorem 3.4.1. *For any given integer* $2 \leq k < \infty$ *the functions* F_k *and* G_k *give an equivalence of setoids between* $(\mathsf{V}_{\mathsf{G}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{G}}})$ *and* $(\mathsf{V}_k, \approx_1)$.

Proof. Given a canonical term $\mathsf{sup}((a,p),f) : \mathsf{V}_k$, and supposing by inductive hypothesis that we already have $G_k \circ (F_k \circ f)(y) \approx_1 f(y)$, we want to prove that:

$$G_k F_k(\mathsf{sup}((a,p),f)) \approx_1 \mathsf{sup}((a,p),f). \tag{3.1}$$

Looking at the left-hand side and expanding the definitions of G_k and F_k, we have $G_k F_k(\mathsf{sup}(a,p),f) = \mathsf{sup}((h(a),p'), G_k \circ \overline{(F_k \circ f)})$. The truncated bisimulation relation $\approx_1$ of eq. (3.1) is made of two conjuncts, the first one is:

$$(\Pi x : \mathsf{T}(h(a)))(\exists y : \mathsf{T}(a)) G_k \circ \overline{(F_k \circ f)}(x) \approx_1 f(y), \tag{3.2}$$

which is what we want to prove. Since the function $e : \mathsf{T}(a) \to \mathsf{T}(h(a))$ is epi (see the remark after Definition 3.1.3) we have a term inhabiting the type:

$$(\Pi x : \mathsf{T}(h(a)))(\exists y : \mathsf{T}(a)) \mathsf{Id}_{\mathsf{T}(h(a))}(x, e(y)).$$

We now apply the function $G_k \circ (\overline{F_k \circ f})$ to the terms x and $e(y)$. By applying a function to two propositionally equal terms we get propositionally equal terms, thanks to Lemma 1.1.7. Moreover, Id is the smallest reflexive relation, so we get a term inhabiting the type:

$$G_k \circ (\overline{F_k \circ f})(x) \approx_1 G_k \circ (\overline{F_k \circ f})(e(y)).$$

Then notice that $G_k \circ (\overline{F_k \circ f})(e(y))$ is definitionally equal to $G_k \circ (F_k \circ f)(y)$ by construction of the epi-mono factorisation. And the inductive hypothesis:

$$G_k \circ (F_k \circ f)(y) \approx_1 f(y),$$

gives the desired (3.2).

The second conjunct of the bisimulation is:

$$(\Pi y : \mathsf{T}(a))(\exists x : \mathsf{T}(h(a))) G_k \circ \overline{(F_k \circ f)}(x) \approx_1 f(y),$$

which is easier to prove, since for a given $y : \mathsf{T}(a)$ it is enough to consider the term $e(y)$ as a witness for the existential quantifier.

For the other direction, we are given a term $(\mathsf{sup}(a,f),i) : \mathsf{V}_\mathsf{G}$, where we have $i : \mathsf{itset}(\mathsf{sup}(a,f))$. Our aim is to prove that the following type is inhabited:

$$\mathsf{Id}_{\mathsf{V}_\mathsf{G}}(F_k \circ G_k(\mathsf{sup}(a,f),i), (\mathsf{sup}(a,f),i)).$$

Then observe that in order to construct a path in V_G it is enough to construct a path in V_∞ between the corresponding projections since itset is always a hprop. So it is enough to prove that:

$$\mathsf{Id}_{\mathsf{V}_\mathsf{G}}(\pi_0 \circ F_k \circ G_k(\mathsf{sup}(a,f),i), \mathsf{sup}(a,f)), \tag{3.3}$$

is inhabited. The two terms that we want to prove identical are terms in V_∞ that are iterative sets, so thanks to Lemma 3.2.8 it is enough to prove that the following type is inhabited:

$$\pi_0 \circ F_k \circ G_k((\mathsf{sup}(a,f),i)) \approx_1 \mathsf{sup}(a,f).$$

This proof is analogous to the proof of eq. (3.1). □

Corollary 3.4.2. *The setoids* $(\mathsf{V}_k/_{\approx_1}, \mathsf{Id}_{\mathsf{V}_k/_{\approx_1}})$ *and* $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$ *are equivalent.*

Proof. Immediate consequence of Lemma 3.1.2 and Theorem 3.4.1. □

3.5. Logical aspects

In Section 3.2, Section 3.3 and Section 3.4 we showed that the setoids $(\mathsf{V}_\infty, \approx_1)$, $(\mathsf{V}_k, \approx_1)$, $(\mathsf{V}_\mathsf{G}, \mathsf{Id}_{\mathsf{V}_\mathsf{G}})$ and $(\mathsf{V}_\mathsf{H}, \mathsf{Id}_{\mathsf{V}_\mathsf{H}})$ are all equivalent, for any $k \geq 2$. In this section we relate the interpretations $[\![\cdot]\!]_{k,1}, [\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$ of formulas of CST, for $2 \leq k \leq \infty$. We show that the equivalence of setoids induces a logical equivalence of the interpretations.

We then look in more detail at the Replacement and Exponentiation axioms, and what are the key facts that allow to prove their validity in each of the three interpretations.

Firstly, recall the interpretation of formulas in the HoTT book. Set-theoretic equality is interpreted using the identity type:

$$[\![x \doteq y]\!]_H := \mathsf{Id}_{\mathsf{V}_\mathsf{H}}(x, y),$$

for $\alpha : \mathsf{V}_\mathsf{H}$ and $a : \mathsf{U}$, $f : \mathsf{T}(a) \to \mathsf{V}_\mathsf{H}$ the interpretation of the membership relation is defined by recursion on V_H as:

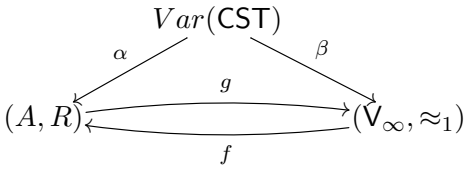
$$[\![\alpha \in \mathsf{set}(a, f)]\!]_H := (\exists i : \mathsf{T}(a))\mathsf{Id}_{\mathsf{V}_\mathsf{H}}(f(i), \alpha)).$$

This definition is by recursion on $\mathsf{set}(a, f) : \mathsf{V}_\mathsf{H}$ so one has to check the validity of the clauses 3.1 and 3.2 which is straightforward. The rest of the interpretation is as usual:

$$\begin{aligned}
[\![\bot]\!]_H &:= 0;\\
[\![\phi \Rightarrow \psi]\!]_H &:= [\![\phi]\!]_H \to [\![\psi]\!]_H;\\
[\![\phi \wedge \psi]\!]_H &:= [\![\phi]\!]_H \times [\![\psi]\!]_H;\\
[\![\phi \vee \psi]\!]_H &:= \left\| [\![\phi]\!]_H + [\![\psi]\!]_H \right\|;\\
[\![\forall x\, \phi(x)]\!]_H &:= (\Pi\alpha : \mathsf{V}_\mathsf{H})[\![\phi(\alpha)]\!]_H;\\
[\![\exists x\, \phi(x)]\!]_H &:= \left\| (\Sigma\alpha : \mathsf{V}_\mathsf{H})[\![\phi(\alpha)]\!]_H \right\|.
\end{aligned}$$

The equivalence of setoids induces an equivalence of the interpretations, as we see in the following theorem.

Theorem 3.5.1. *Given assignments of terms to variables α and β:*



where the setoid (A, R) is either $(\mathsf{V_G}, \mathsf{Id_{V_G}}), (\mathsf{V_H}, \mathsf{Id_{V_H}})$ or $(\mathsf{V}_k, \approx_1)$, and the maps f, g the respective equivalences of setoids (see Theorem 3.2.9, Theorem 3.3.2 and Theorem 3.4.1).

Suppose that the assignments α and β are compatible with the equivalence of setoids, in the sense that we have terms inhabiting $(g(\alpha(\underline{x})) \approx_1 \beta(\underline{x}))$ and $R(\alpha(\underline{x}), f(\beta(\underline{x})))$.

Then for all formulas $\phi(\underline{x})$ of CST, we have a logical equivalence between the interpretations:

$$[\![\phi(\alpha(\underline{x}))]\!]_* \leftrightarrow [\![\phi(\beta(\underline{x}))]\!]_{\infty,1},$$

where $[\![\cdot]\!]_*$ is either $[\![\cdot]\!]_G, [\![\cdot]\!]_H$ or $[\![\cdot]\!]_{k,1}$.

Proof. We prove the equivalence for $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_G$, the other cases are similar. The proof is a straightforward induction on the structure of ϕ:

- if ϕ is $x \doteq y$ then the logical equivalence follows from the fact that π_0, F, H, K are maps of setoids, so for example if $\mathsf{Id_{V_G}}(\alpha(x), \alpha(y))$, then also:
$$(\pi_0(\alpha(x)) \approx_1 \pi_0(\alpha(y))),$$
and we conclude using the compatibility between α and β.
- If ϕ is $x \in y$, notice that by definition of $[\![x \in y]\!]_G$ we have that $[\![x \in y]\!]_G$ is a Σ-type not truncated, whereas $[\![x \in y]\!]_{\infty,1}$ truncates the Σ-type (see the definition of $[\![\cdot]\!]_{\infty,1}$ in Section 1.2). However, since $[\![x \in y]\!]_G$ is always a hproposition due to Lemma 3.2.4, we can remove the truncation when proving the logical equivalence. The equivalence follows from inspecting the definitions and using the hypothesis of compatibility between assignments α, β, γ, e.g. $(\pi_0(\alpha(\underline{x})) \approx_1 \beta(\underline{x}))$ and $\mathsf{Id_{V_G}}(\alpha(\underline{x}), F(\beta(\underline{x})))$.
- The cases for $\wedge, \vee, \to$ are straightforward.
- If ϕ is $\forall x\, \psi(x)$, the aim is to prove that there are functions in both directions between the type $(\Pi x : \mathsf{V_G})[\![\psi(x)]\!]_G$ and $(\Pi x : \mathsf{V}_\infty)[\![\psi(x)]\!]_{\infty,1}$. We construct a function of type:
$$(\Pi t : \mathsf{V_G})[\![\psi(t)]\!]_G \to (\Pi s : \mathsf{V}_\infty)[\![\psi(s)]\!]_{\infty,1},$$
by taking a dependent function f in the domain and mapping it to one in the codomain as follows. Given $s : \mathsf{V}_\infty$ we map it to $\mathsf{V_G}$ using the

map $F : \mathsf{V}_\infty \to \mathsf{V}_\mathsf{G}$, then we can apply the given dependent function f, obtaining $f(F(s)) : [\![\psi(F(s))]\!]_G$. Now notice that we have a map:

$$I : [\![\psi(F(s))]\!]_G \to [\![\psi(\pi_0 F(s))]\!]_{\infty,1},$$

given by the first conjunct of the logical equivalence coming from the inductive hypothesis. Applying I to our term to get:

$$I \circ f \circ F(s) : [\![\psi(\pi_0 F(s))]\!]_{\infty,1},$$

and recalling that the interpretations of formulas satisfy the Leibniz rule with respect to $\approx_1$, we have a function $\mathsf{Leib} : [\![\psi(\pi_0 F(s))]\!]_{\infty,1} \to [\![\psi(s)]\!]_{\infty,1}$.

Combining these steps together gives the desired term:

$$\lambda s.\mathsf{Leib} \circ I \circ f \circ F(s) : (\Pi s : \mathsf{V}_\infty)[\![\psi(s)]\!]_{\infty,1}.$$

- If ϕ is $\exists x\, \psi(x)$, let us prove that $[\![\exists x\, \psi(x)]\!]_{\infty,1}$ is logically equivalent to $[\![\exists x\, \psi(x)]\!]_G$. Since both these types are truncated, in order to construct a logical equivalence it is enough to consider untruncated types. So we want a function of type:

$$(\Sigma s : \mathsf{V}_\infty)[\![\psi(s)]\!]_{\infty,1} \to (\Sigma t : \mathsf{V}_\mathsf{G})[\![\psi(t)]\!]_G.$$

Given a term p in the domain, we get a term $F(\pi_0(p)) : \mathsf{V}_\mathsf{G}$. By inductive hypothesis there is a function $I : [\![\psi(\pi_0(p))]\!]_{\infty,1} \to [\![\psi(F(\pi_0(p)))]\!]_G$, which we can apply to the second projection obtaining $I(\pi_1(p)) : [\![\psi(F(\pi_0(p)))]\!]_G$, so that we have $(F(\pi_0(p)), I(\pi_1(p)))$ the desired term. The other direction of the logical equivalence is similar.

□

Observe that we could have stated the previous theorem with only one assignment of terms to variables, say β, and stated the logical equivalence as:

$$[\![\phi(\beta(\underline{x}))]\!]_{\infty,1} \leftrightarrow [\![\phi(F(\beta(\underline{x})))]\!]_G \leftrightarrow [\![\phi(H(\beta(\underline{x})))]\!]_H \leftrightarrow [\![\phi(N(\beta(\underline{x})))]\!]_{k,1}.$$

A consequence of the previous Theorem 3.5.1 is that the Replacement and Exponentiation axioms that are valid in $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$ are also valid in $[\![\cdot]\!]_{k,1}$. We now give a brief sketch of the proof of their validity in the first two interpretations.

The key point is that in $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$, since set-theoretic equality is interpreted as the identity type, we can use the uniqueness conditions contained in the Replacement and Exponentiation axioms to infer information about the homotopy levels

of certain types and then apply the principle of unique choice (see Lemma 1.1.10) to extract a type-theoretic function.

Theorem 3.5.2. *The Replacement and the Exponentiation axioms are valid in the interpretations* $[\![\cdot]\!]_H$ *and* $[\![\cdot]\!]_G$.

Proof. We prove the statement for $[\![\cdot]\!]_H$, the case of $[\![\cdot]\!]_G$ being similar.

- Recall the statement of Replacement:

$$(\forall x \in \alpha)\exists! y\, \phi(x,y) \Rightarrow (\exists \beta)(\forall x \in \alpha)(\exists! y \in \beta)\phi(x,y).$$

  Suppose $(\Pi x : \mathsf{El}(\alpha))[\![\exists! y\, \phi(x,y)]\!]_H$. Then for all $x : \mathsf{El}(\alpha)$ the Σ-type $(\Sigma y : \mathsf{V_H})[\![\phi(x,y)]\!]_H$ is a hprop, since by definition of the quantifier $\exists!$ we have that $[\![\exists! y\, \phi(x,y)]\!]_H$ is definitionally equal to:

$$(\Sigma y : \mathsf{V_H})[\![\phi(x,y)]\!]_H \times (\Pi s,t : \mathsf{V_H})[\![\phi(x,s)]\!]_H \times [\![\phi(x,t)]\!]_H \to \mathsf{Id}_{\mathsf{V_H}}(s,t),$$

  and $[\![\phi(x,y)]\!]_H$ is always a hproposition.

  We apply the principle of unique choice (see Lemma 1.1.10) to the family of hpropositions $(\Sigma y : \mathsf{V_H})[\![\phi(x,y)]\!]_H$. By hypothesis we have a term inhabiting the type $(\Pi x : \mathsf{El}(\alpha))[\![\exists y\, \phi(\alpha_x, y)]\!]_H$, hence we have a function:

$$f : (\Pi x : \mathsf{El}(\alpha))(\Sigma y : \mathsf{V_H})[\![\phi(\alpha_x, y)]\!]_H,$$

  which we can use to construct the term $\mathsf{set}(\mathsf{el}(\alpha), f) : \mathsf{V_H}$. It is then straightforward to check that this term provides a witness for the existential quantifier in the conclusion of Replacement.
- Recall the statement of Exponentiation:

$$\forall \alpha, \beta\, \exists \gamma[\forall f(\alpha \xrightarrow{f} \beta) \Rightarrow f \in \gamma].$$

  Given $\alpha, \beta : \mathsf{V_H}$ we construct γ by first considering the type $\mathsf{El}(\alpha) \to \mathsf{El}(\beta)$ which is definitionally equal to $\mathsf{T}(\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)))$, where exp is the term representing in U the function type. Then we define the function $\gamma_{\_} := \lambda z.\mathsf{set}(\mathsf{el}(\alpha), \lambda x.\langle \alpha_x, \beta_{z(x)} \rangle)$, which inhabits the type:

$$\mathsf{T}(\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta))) \to \mathsf{V_H},$$

  so that we can define the term $\gamma := \mathsf{set}(\mathsf{exp}(\mathsf{el}(\alpha), \mathsf{el}(\beta)), \gamma_{\_})$.

  In order to validate the axiom we are given $f : \mathsf{V_H}$ and a term witnessing the premiss of the implication and our thesis is to validate:

$$[\![f \in \gamma]\!]_H = (\Sigma t : \mathsf{El}(\alpha) \to \mathsf{El}(\beta))[\![f \doteq \gamma_t]\!]_H.$$

Similarly to what we did for the Replacement axiom, consider the premiss:

$$[\![(\forall x \in \alpha)(\exists! y \in \beta)(\langle \alpha, \beta \rangle \in f)]\!]_H,$$

which allows us to apply the principle of unique choice (see Lemma 1.1.10). Thus we obtain the desired type-theoretic function $t : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$. It is then straightforward to check that $[\![f \in \gamma]\!]_H$ follows, which validates Exponentiation.

□

Corollary 3.5.3. *The Replacement and Exponentiation axioms are valid in the interpretations* $[\![\cdot]\!]_{k,1}$ *for* $2 \leq k \leq \infty$.

Proof. Immediate consequence of Theorem 3.5.2 and Theorem 3.5.1 on the logical equivalence of the interpretations. Note that univalence is used in proving the equivalence between the setoids and the interpretations.

In the proof of Theorem 3.5.2 we have that the type $(\Sigma y : \mathsf{V}_\infty)[\![\phi(x, y)]\!]_{k,1}$, although not being necessarily a hproposition, is logically equivalent to the type $(\Sigma y : \mathsf{V}_\mathsf{H})[\![\phi(x, y)]\!]_H$. On the other hand $(\Sigma y : \mathsf{V}_\mathsf{H})[\![\phi(x, y)]\!]_H$ is both a Σ-type and a hproposition.

Using the principle of unique choice and the logical equivalence of Theorem 3.5.1 once again, we can validate the interpretation of Replacement in $[\![\cdot]\!]_{k,1}$. A similar argument proves the Exponentiation axiom. □

Recall that we used Corollary 3.5.3 in chapter 1 to establish Lemma 1.3.13 and Theorem 1.3.14.

Remark 3.5.4. A case worth a specific mention is when $k = 2$, where we have $[\![\cdot]\!]_{2,1}$ and the set-quotient $\mathsf{V}_2/_{\approx_1}$, interpreting Myhill's Constructive Set Theory. Sets are interpreted as hsets and formulas as hpropositions, the type $\mathsf{V}_2/_{\approx_1}$ is itself a hset, and set-theoretic equality is interpreted as the identity type. Moreover $\mathsf{V}_2/_{\approx_1}$ is defined as a set-quotient of a W-type, and hence is a standard and relatively well understood higher inductive type. So we can see this model as combining many of the desirable features one might want in a model of set theory in homotopy type theory.

The only drawback of this model is that being a set-quotient some technicalities may arise in working within that model. Especially when one wants to apply the elimination rule one has to check that the type in which one eliminates is a hset. Therefore one may look at the type $\mathsf{V}^k_G := (\Sigma x : \mathsf{V}_k)\mathsf{itset}(x)$ in general and more specifically at the type V^2_G.

Observe that thanks to Theorem 3.4.1 we have, as in the case of $[\![\cdot]\!]_{k,\infty}$ a hierarchy of models of CST given by $(\mathsf{V}_k, \approx_1, [\![\cdot]\!]_{k,1})$ for $2 \leq k \leq \infty$. Indeed, the proof of the validity of the axioms of CST in the interpretations $[\![\cdot]\!]_{k,1}$ is not affected by the hlevel of the indices of the W-types V_k. This is another indication that set theories do not detect any difference when sets are interpreted as k-types.

Note that among these equivalent interpretations $[\![\cdot]\!]_H, [\![\cdot]\!]_G, [\![\cdot]\!]_{k,1}$ and $[\![\cdot]\!]_{\infty,1}$, the first two interpret set-theoretic equality as the identity type, which allowing one to validate easily Replacement and Exponentiation. On the other hand in $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{k,1}$ equality is interpreted as a truncated bisimulation relation and the validity of Replacement and Exponentiation is established using the equivalence with the other two interpretations. A direct proof would face the issues described in Remark 1.3.12.

However, among them the interpretation $[\![\cdot]\!]_{\infty,1}$ is more amenable to an analysis using a logic-enriched type theory, a line of investigation that we pursue in chapter 5. Indeed, the interpretation of set-theoretic equality follows closely Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$.

CHAPTER 4

A characterisation of the $[\![\cdot]\!]_{PHP}$ interpretation

This chapter and chapter 5 are devoted to a more in depth analysis of the interpretations considered in chapter 1 and chapter 3. We are interested in understanding not only how different foundational settings such as constructive set theories or type theories relate via different interpretations, but also how these interpretations themselves relate to each other. Following [GA06], in this chapter we introduce a tool for the study of the interpretations, namely logic-enriched type theories. These are type theories that have primitive judgements to express logical formulas (see Section 4.1). In chapter 5 we use logic-enriched type theories to perform a comparative analysis of the interpretation $[\![\cdot]\!]_{\infty,1} : \mathsf{CST} \to \mathsf{HoTT}$ and Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty} : \mathsf{CZF} \to \mathsf{ML_1W}$.

The purpose of this chapter is twofold. The first is to introduce the propositions-as-hpropositions interpretation $[\![\cdot]\!]_{PHP}$ from an appropriate logic-enriched type theory LEH (see Section 4.1) into the homotopy type theory H. This interpretation $[\![\cdot]\!]_{PHP}$ captures one of the key ingredients of the interpretations $[\![\cdot]\!]_{k,1}, [\![\cdot]\!]_G$ and $[\![\cdot]\!]_H$, namely the fact that propositions are interpreted as hprop by means of propositional truncations.

The main result of this chapter is Theorem 4.3.4, in which we characterise the formulas valid in $[\![\cdot]\!]_{PHP}$ as those provable in the logic-enriched type theory LEH using the axiom of unique choice AUC (see table 4.5). This characterisation parallels the one of [AG00, Theorem 2] where it is shown that a formula is valid in the propositions-as-types interpretation if and only if it is provable using the axiom of choice AC (see table 4.4). In this way we have a perfectly parallel characterisation of the propositions-as-types and the propositions-as-hprops interpretations in terms of their corresponding choice principles. This is an example of how one can use logic-enriched type theories to clearly isolate key aspects of a proof or a construction allowing for a more careful and detailed analysis.

It is worth noting that the logic-enriched type theory LEH comprises of both the identity types Id_A and equality propositions Eq_A, in a fashion similar to [PL15] but with some differences since our propositions do not have proof terms inhabiting

them. Theorem 4.3.4 is an example of application of a notion of equality proposition that may be fruitful in other contexts.

Secondly, since the full axiom of unique choice AUC is unnecessarily strong for the analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$ of chapter 5, we introduce new principles for the logic-enriched type theory LE. Namely, we introduce a propositions-as-types principle (PU) and two forms of the axiom of unique choice ($\mathsf{AUC}_{\mathsf{V}}, \mathsf{AUC}_{\mathsf{El}(\beta)}$) for specific types V and $\mathsf{El}(\beta)$ for $\beta : V$, with specific equality propositions. See table 4.6 on page 87, table 4.7 on page 89 and table 4.8 on page 89. These principles are then used in chapter 5 to provide an analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$. We show in Theorem 4.4.1 and Theorem 4.4.2 that these new principles are valid in the propositions-as-hproposition $[\![\cdot]\!]_{PHP}$ interpretation of LE into HoTT.
An important part of the work in providing the comparative analysis of chapter 5 consists in isolating the right principles that factor the interpretation $[\![\cdot]\!]_{\infty,1}$. This has to be done in the clearest and most informative way, and at the same time highlight the similarities and differences between the two interpretations $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$.

This and the next chapter are indebted to [Acz16] where Aczel suggests logic-enriched dependent type theories as frameworks for the analysis of different philosophical foundations of mathematics.

Outline. In Section 4.1 we give a concise introduction to logic-enriched type theories and review some of the basic notions introduced in [GA06]. In Section 4.2 we define the propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP}$ of the logic-enriched type theory LEH into the type theory H. Section 4.3 is devoted to the characterisation of the interpretation $[\![\cdot]\!]_{PHP}$. In Section 4.4 we introduce the logic-enriched type theory $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC}_{\mathsf{V}} + \mathsf{AUC}_{\mathsf{El}(\beta)})$, which is then used in chapter 5 to provide an analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$.

4.1. Logic-enriched type theories

In a logic-enriched type theory, in addition to its pure type-theoretic part, there are new forms of judgements for logical formulas:

$$\Gamma \vdash \phi : \mathsf{prop}$$

$$\Gamma \vdash \phi_1, \ldots, \phi_n \Rightarrow \phi.$$

where Γ is an ordinary type-theoretic context. We use $\Gamma \vdash \phi$ as a shorthand for $\Gamma \vdash\Rightarrow \phi$. Note that the symbol '$\Rightarrow$' appearing here is a structural symbol for the judgement and is not to be confused with the logical implication, which in this

chapter and chapter 5 we denote as $\supset$. Accordingly, in this chapter and chapter 5 we use the symbol $\equiv$ for logical equivalence between formulas.

There are standard natural deduction rules in sequent calculus style for logical connectives and quantifiers. We consider only logic-enriched type theories with intuitionistic logic. For example the rules for the universal quantifier are:

$$\frac{A : \mathsf{type} \qquad x : A \vdash \phi(x) : \mathsf{prop}}{(\forall x : A)\phi(x) : \mathsf{prop}}$$

$$\frac{x : A \vdash \phi(x)}{(\forall x : A)\phi(x)} \qquad \frac{(\forall x : A)\phi(x) \qquad t : A}{\phi(t)}$$

TABLE 4.1. Rules for the universal quantifier

Logic-enriched type theories are rich systems that share the advantages of type theory with the extra flexibility of not necessarily fixing an interpretation of logic. This is a feature that may or may not be added. For more examples of the uses of logic-enriched type theories see [PL15], where a logic-enriched type theory is used to give a construction of semi simplicial types in homotopy type theory.

The base logic-enriched type theory that we consider, which we name LE, is given by the following:

- $\mathsf{ML_1W}$ as its pure part;
- $\mathsf{IL_1}$, i.e. intuitionistic logic with a type of small propositions P, with an elimination rule that given $p : \mathsf{P}$ returns a proposition $\tau(p) : \mathsf{prop}$;
- IND, i.e. induction rules for the inductive types $\mathsf{0}, \mathsf{1}, \mathsf{2}, \mathsf{N}$, and $\Sigma, +$ and W-types formulated using the logic.

See appendix B for the rules. This logic-enriched type-theory was originally introduced in [GA06].

We consider also other logic-enriched type theories that have other rules in addition to the ones of LE. We introduce an equality proposition $\mathsf{Eq}_A(x, y)$ for $x, y : A$. Its rules are simply the reformulation of the usual rules for intensional identity types $\mathsf{Id}_A(x, y)$ in the context of the logic-enriched type theory.

$$\frac{A : \mathsf{type} \qquad x, y : A}{\mathsf{Eq}_A(x, y) : \mathsf{prop}} \qquad \frac{A : \mathsf{type} \qquad x : A}{\mathsf{Eq}_A(x, x)}$$

$$\frac{a, b : A \qquad \mathsf{Eq}_A(a, b) \qquad x : A \vdash \phi(x, x)}{\phi(a, b)}$$

TABLE 4.2. Rules for Eq

The paper [PL15] introduces a similar notion of equality proposition for a logic-enriched type theory, and the papers [MS05] and [Mai08] for a minimalist type theory. However, in both approaches propositions are inhabited by proof terms, so that their theories are more alike two-level type theories than logic-enriched type theories.

We also consider the theory LEH, which can be thought of as a logic-enriched type theory for homotopy levels. It is defined as $\mathsf{LEH} := \mathsf{H} + \mathsf{IL}_1^{\mathsf{Eq}} + \mathsf{IND}$. Where its components are:

- $\mathsf{H} = \mathsf{ML}_1^{\mathsf{Id}}\mathsf{W} + \| \cdot \| + \mathsf{FE}$ as its pure part;
- $\mathsf{IL}_1^{\mathsf{Eq}}$, i.e. intuitionistic logic with one type of propositions P and equality proposition Eq;
- IND, i.e. inductive rules for inductive types, including the intensional identity type Id_A and propositional truncations $\| \cdot \|$ (see table A17).

Definition 4.1.1. For a type A we define the formula:

$$\mathsf{isHProp}^{\mathsf{Eq}}(A) := (\forall x, y : A)\mathsf{Eq}_A(x, y).$$

We say that the type A is an Eq-*hprop* if and only if the formula $\mathsf{isHProp}^{\mathsf{Eq}}(A)$ is provable.

We always refer to the usual notion of Id-hprops unless stated explicitly.

The two notions of hproposition are related by the following lemma.

Lemma 4.1.2. *In* LEH*, the existence of a term inhabiting* $\mathsf{isHProp}^{\mathsf{Id}}(A)$ *implies the formula* $\mathsf{isHProp}^{\mathsf{Eq}}(A)$*. More formally, we can derive the following judgement:*

$$(\exists t : \mathsf{isHProp}^{\mathsf{Id}}(A))\top \Rightarrow \mathsf{isHProp}^{\mathsf{Eq}}(A)$$

Proof. Consider that by definition $\mathsf{isHProp}^{\mathsf{Id}}(A)$ is the type $(\Pi x, y : A)\mathsf{Id}_A(x, y)$, and that $\mathsf{isHProp}^{\mathsf{Eq}}(A)$ is the formula $(\forall x, y : A)\mathsf{Eq}_A(x, y)$. Therefore, it is enough to show that given a term in $\mathsf{Id}_A(x, y)$ one can deduce the formula $\mathsf{Eq}_A(x, y)$. This follows from the induction rule $\mathsf{IND}_{\mathsf{Id}}$ applied to the formula $\mathsf{Eq}_A(x, y)$:

$$\frac{x, y : A, z : \mathsf{Id}_A(x, y) \vdash \phi(x, y, z) : \mathsf{prop} \qquad e : \mathsf{Id}_A(a, b) \qquad w : A \vdash \phi(w, w, \mathsf{refl}_{\mathsf{w}})}{\phi[a, b, e/x, y, z]}$$

□

Remark 4.1.3. Observe that the identity type Id_A and the equality proposition Eq_A interact. Indeed, we can form the usual tower of identity types: Id_A, $\mathsf{Id}_{\mathsf{Id}_A}$, $\mathsf{Id}_{\mathsf{Id}_{\mathsf{Id}_A}}$, ..., but also for each type we have an equality proposition, so that we have Eq_A , $\mathsf{Eq}_{\mathsf{Id}_A}$, $\mathsf{Eq}_{\mathsf{Id}_{\mathsf{Id}_A}}$, ...

In the following sections we consider a propositions-as-types principles that provide a map $\mathsf{pu}(\_) : \mathsf{P} \to \mathsf{U}$, which then generates an infinite binary tree of the iterations of Id and Eq, since at any given point we have a type so we can form either its identity type or its equality proposition.

Note that if the propositions-as-types map $\mathsf{pu}(\_) : \mathsf{P} \to \mathsf{U}$ by construction always gives a hproposition, then iterated towers of Id over an equality proposition Eq become trivial, hence part of the structure simplifies. This is the case for the propositions-as-hprops interpretation.

4.2. The propositions-as-hprops interpretation

Here we introduce the propositions-as-hpropositions interpretation:

$$[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}.$$

In formulating this interpretation we look at $[\![\cdot]\!]_H$, $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_{k,1}$ and we isolate one aspect that they share, namely the way in which they interpret formulas. Then we give a definition of $[\![\cdot]\!]_{PHP}$ with the same structure as $[\![\cdot]\!]_H$, $[\![\cdot]\!]_G$ and $[\![\cdot]\!]_{k,1}$, but in the more general context of the logic-enriched type theory LEH, so that we can better analyse this aspect of the interpretations.

The interpretation $[\![\cdot]\!]_{PHP}$ leaves intact the pure part of the logic-enriched type theory. Logical connectives and quantifiers are interpreted as the corresponding type-theoretic operators, the equality proposition is interpreted as the identity type. Additionally the type constructors that do not necessarily preserve hpropositions, namely $+$ and Σ, are propositionally truncated using $\| \cdot \|$:

$$
\begin{aligned}
[\![\bot]\!]_{PHP} &:= \mathsf{0} \\
[\![\phi \supset \psi]\!]_{PHP} &:= [\![\phi]\!]_{PHP} \to [\![\psi]\!]_{PHP} \\
[\![\phi \wedge \psi]\!]_{PHP} &:= [\![\phi]\!]_{PHP} \times [\![\psi]\!]_{PHP} \\
[\![\phi \vee \psi]\!]_{PHP} &:= \left\| [\![\phi]\!]_{PHP} + [\![\psi]\!]_{PHP} \right\| \\
[\![(\forall x : A)\phi(x)]\!]_{PHP} &:= (\Pi x : A)[\![\phi(x)]\!]_{PHP} \\
[\![(\exists x : A)\phi(x)]\!]_{PHP} &:= \left\| (\Sigma x : A)[\![\phi(x)]\!]_{PHP} \right\| \\
[\![\mathsf{Eq}_A(x, y)]\!]_{PHP} &:= \mathsf{Id}_A(x, y) \\
[\![\mathsf{P}]\!]_{PHP} &:= \mathsf{HProp}_{\mathsf{U}}
\end{aligned}
$$

For this, recall that $\mathsf{HProp}_{\mathsf{U}}$ in the type theory H is the type of small hprops, formally defined as $(\Sigma x : \mathsf{U})\mathsf{isHProp}^{\mathsf{Id}}(x)$. For the interpretation of small proposition we define:

$$[\![\tau(p)]\!]_{PHP} := \mathsf{T}([\![p]\!]_{PHP}) \text{ for } p : \mathsf{P}.$$

Here the interpretation $[\![\cdot]\!]_{PHP}$ associates to $p : \mathsf{P}$ a small type $[\![p]\!]_{PHP} : \mathsf{U}$ following the same structure as in the interpretation of large propositions. The operators τ and T are the decodings for small propositions and small types respectively (see appendix B).

This interpretation is studied in more depth in Section 4.3.

In Section 4.4 and in chapter 5 we consider the propositions-as-hprops interpretation $[\![\cdot]\!]'_{PHP} : \mathsf{LE} \to \mathsf{HoTT}$ which has the same inductive structure on formulas as $[\![\cdot]\!]_{PHP}$, but different domain and codomain theories. Notice that the characterisation of Theorem 4.3.4 is stated only for the interpretation $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$.

4.3. Characterisation of the formulas valid in $[\![\cdot]\!]_{PHP}$

We proceed to characterise the formulas valid in the propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$. The proof of the characterisation follows the same structure of the characterisation of the propositions-as-type interpretation $[\![\cdot]\!]_{PT} : \mathsf{LE} \to \mathsf{ML_1W}$ of [AG00, Theorem 2]. Firstly, we introduce a propositions-as-hprop principle PaHP (see table 4.3) that is crafted in such a way that a formula is valid in $[\![\cdot]\!]_{PHP}$ if and only if it is provable in $\mathsf{LEH} + \mathsf{PaHP}$, as detailed in Lemma 4.3.1.

Having moved from the interpretation $[\![\cdot]\!]_{PHP}$ to a single logical principle PaHP within the logic-enriched type theory, it is then easier to characterise it more explicitly using other logical principles. We show in Theorem 4.3.4 that over LEH the principle PaHP is equivalent to an axiom of unique choice AUC (see table 4.5).

From the interpretation $[\![\cdot]\!]_{PHP}$ to the PaHP principle. The principle PaHP states internally in the logic-enriched type theory that a formula ϕ holds if and only if its interpretation under $[\![\cdot]\!]_{PHP}$ is valid, i.e. if and only if the type $[\![\phi]\!]_{PHP}$ is inhabited.

$$\frac{\phi : \mathsf{prop}}{\phi \equiv (\exists\_ : [\![\phi]\!]_{PHP})\top}$$

TABLE 4.3. Propositions-as-HPropositions (PaHP)

The following theorem is essentially a reformulation in the context of LEH of [AG00, Theorem 2].

We write $\mathsf{H} \vdash [\![\phi]\!]_{PHP}$ meaning that $\mathsf{H} \vdash t : [\![\phi]\!]_{PHP}$ for some term t of H.

Lemma 4.3.1. *For a judgement $\mathcal{J}$ of* LEH *we have the following characterisation of the formulas valid in the $[\![\cdot]\!]_{PHP}$ interpretation:*

$$\mathsf{H} \vdash [\![\mathcal{J}]\!]_{PHP} \qquad \textit{if and only if} \qquad \mathsf{LEH} + \mathsf{PaHP} \vdash \mathcal{J}.$$

Proof. The direction from right to left is straightforward: given a proof of $\mathcal{J}$ we apply the propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP}$ to every step in that proof. Since every logical rule of the logic-enriched type theory, as well as PaHP, translates into a derived rule of H we obtain a term inhabiting $[\![\mathcal{J}]\!]_{PHP}$.

For the direction from left to right, suppose we have derived in the pure type theory H the judgement $y_1 : [\![\phi_1]\!]_{PHP}, \ldots, y_n : [\![\phi_n]\!]_{PHP} \vdash t : [\![\phi]\!]_{PHP}$. Since H is a subtheory of LEH we can perform the same derivation in LEH, then by applying the rules for $\exists$ and $\top$ we derive the following judgement:

$$(\exists y_1 : [\![\phi_1]\!]_{PHP})\top, \ldots, (\exists y_n : [\![\phi_n]\!]_{PHP})\top \Rightarrow (\exists y : [\![\phi]\!]_{PHP})\top.$$

Now recall that PaHP asserts that $\phi \equiv (\exists y : [\![\phi]\!]_{PHP})\top$, so that we obtain the judgement $\phi_1, \ldots, \phi_n \Rightarrow \phi$. □

Characterisation of the principle PaHP. Now we make the characterisation of Lemma 4.3.1 more explicit and informative by proving that PaHP is equivalent to the axiom of unique choice AUC over the logic-enriched type theory LEH.

First recall the axiom of choice AC for a logic-enriched type theory. The axiom of choice is a simple translation of the usual statement using the language of the logic-enriched type theory and the quantification over types.

$$\frac{A : \mathsf{type} \qquad x : A \vdash B(x) : \mathsf{type} \qquad x : A, y : B \vdash \phi(x,y) : \mathsf{prop}}{(\forall x : A)(\exists y : B)\phi(x,y) \supset (\exists f : (\Pi x : A)B(x))(\forall x : A)\phi(x, f(x))}$$

TABLE 4.4. Type-theoretic axiom of choice (AC)

The axiom of unique choice stated in table 4.5 has the same structure of the axiom of choice but it strengthens the premiss of the implication by requiring the formula $\phi(x,y)$ to be functional in x and y. This involves the unique existential quantifier $(\exists! y : B(x))\phi(x,y)$, which is defined using the equality proposition Eq as:

$$(\exists y : B(x))\phi(x,y) \wedge (\forall z_1, z_2 : B(x))\phi(x,z_1) \wedge \phi(x,z_2) \supset \mathsf{Eq}_{B(x)}(z_1, z_2).$$

The conclusion of the implication is the same.

$$\frac{A : \mathsf{type} \qquad x : A \vdash B(x) : \mathsf{type} \qquad x : A, y : B(x) \vdash \phi(x,y) : \mathsf{prop}}{(\forall x : A)(\exists! y : B(x))\phi(x,y) \supset (\exists f : (\Pi x : A)B(x))(\forall x : A)\phi(x, f(x))}$$

TABLE 4.5. Axiom of unique choice (AUC)

We proceed to characterise the propositions-as-hprops principle PaHP.

Lemma 4.3.2. *Over the base theory* LEH *the principle* PaHP *implies* AUC. *Thus, the rule* AUC *is valid in the interpretation* $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$.

Proof. Suppose we have the premiss $(\forall x : A)(\exists! y : B(x))\phi(x, y)$, which by PaHP is equivalent to:

$$(\exists_{-} : [\![(\forall x : A)(\exists! y : B(x))\phi(x, y)]\!]_{PHP})\top. \tag{4.1}$$

Unfolding the definition of the interpretation $[\![\cdot]\!]_{PHP}$ in the previous formula 4.1 we have on the one hand $(\Pi x : A)\left\|(\Sigma y : B(x))[\![\phi]\!]_{PHP}\right\|$. On the other hand, from the interpretation of the uniqueness condition we have $\mathsf{isHProp}^{\mathsf{Id}}((\Sigma y : B(x))[\![\phi]\!]_{PHP})$, since $[\![\phi]\!]_{PHP}$ is always a hprop by construction.

So now we can apply the principle of unique choice (see Lemma 1.1.10) to the family of types $P(x) := (\Sigma y : B(x))[\![\phi(x, y)]\!]_{PHP}$. Note that the principle of unique choice holds in LEH because it holds in H, which is contained in LEH. So we get a term:

$$f : (\Pi x : A)(\Sigma y : B(x))[\![\phi]\!]_{PHP},$$

from which the conclusion of AUC follows.

For the second claim, since PaHP is valid in the $[\![\cdot]\!]_{PHP}$ interpretation, the principle AUC is also valid. □

Lemma 4.3.3. *Over the base theory* LEH *we have that* AUC *implies* PaHP.

Proof. Recall that PaHP states that $\phi \equiv (\exists_{-} : [\![\phi]\!]_{PHP})\top$. We prove PaHP by induction on the structure of the formula ϕ. The rule IND_0 is used for the case of $\bot$, the axiom of unique choice AUC for the cases of $\supset$ and $\forall$, and the induction rules $\mathsf{IND}_+, \mathsf{IND}_\Sigma$ and $\mathsf{IND}_{\|\cdot\|}$ for the cases of $\vee$ and $\exists$.

For $\bot$, notice that from $\bot$ we can deduce everything. Conversely applying the inductive rule IND_0 (see table A17) we have that from the existence of a term $t : 0$ we can deduce $\bot$. For $\top$ the required logical equivalence is immediate.

For $\phi \wedge \psi$, by inductive hypothesis we have the equivalences $\phi \equiv (\exists_{-} : [\![\phi]\!]_{PHP})\top$ and $\psi \equiv (\exists_{-} : [\![\psi]\!]_{PHP})\top$. The result follows from the fact that the formula:

$$(\exists_{-} : [\![\phi]\!]_{PHP})\top \wedge (\exists_{-} : [\![\psi]\!]_{PHP})\top,$$

is equivalent to $(\exists_{-} : [\![\phi]\!]_{PHP} \times [\![\psi]\!]_{PHP})\top$.

For $\phi \supset \psi$, we need to show:

$$(\exists_{-} : [\![\phi]\!]_{PHP} \to [\![\psi]\!]_{PHP})\top \equiv (\exists_{-} : [\![\phi]\!]_{PHP})\top \supset (\exists_{-} : [\![\psi]\!]_{PHP})\top.$$

The implication left-to-right follows easily from the elimination rules for $\exists$ and $\supset$. For the other direction assume $(\exists_{-} : [\![\phi]\!]_{PHP})\top \supset (\exists_{-} : [\![\psi]\!]_{PHP})\top$. So given any $x : [\![\phi]\!]_{PHP}$ we can prove $(\exists_{-} : [\![\phi]\!]_{PHP})\top$ from which $(\exists_{-} : [\![\psi]\!]_{PHP})\top$ follows. Thus we have derived $(\forall x : [\![\phi]\!]_{PHP})(\exists_{-} : [\![\psi]\!]_{PHP})\top$. Now notice that since $[\![\psi]\!]_{PHP}$ is always an Id-hproposition by definition hence it is an Eq-hproposition by

Lemma 4.1.2. Thus we have that $(\forall x : [\![\phi]\!]_{PHP})(\exists!_{\_} : [\![\psi]\!]_{PHP})\top$. Hence applying the AUC rule to the last formula we deduce: $(\exists_{\_} : [\![\phi]\!]_{PHP} \to [\![\psi]\!]_{PHP})\top$.

For $(\forall x : A)\phi(x)$, the proof is similar to the one for $\supset$ with the use of the axiom of unique choice AUC.

For $(\exists x : A)\phi(x)$, we want to prove the judgement:

$$(\exists x : A)(\exists_{\_} : [\![\phi(x)]\!]_{PHP})\top \equiv \left(\exists_{\_} : \left\|(\Sigma x : A)[\![\phi(x)]\!]_{PHP}\right\|\right)\top.$$

For the implication left-to-right, note that in order to derive the conclusion it suffices to do so assuming one has terms $a : A$ and $t : [\![\phi(a)]\!]_{PHP}$. We can then use them to construct a witness $|(a,t)| : \left\|(\Sigma x : A)[\![\phi(x)]\!]_{PHP}\right\|$ so that we can conclude by applying the introduction rule for $\exists$.

For the other direction we want to prove:

$$\left(\exists_{\_} : \left\|(\Sigma x : A)[\![\phi(x)]\!]_{PHP}\right\|\right)\top \Rightarrow (\exists x : A)(\exists_{\_} : [\![\phi(x)]\!]_{PHP})\top.$$

In order to show the conclusion it suffices to do so assuming one has a term:

$$t : \left\|(\Sigma x : A)[\![\phi(x)]\!]_{PHP}\right\|.$$

We use the induction rule for truncation $\mathsf{IND}_{\|\cdot\|}$ (see table A17) to get another term $t' : (\Sigma x : A)[\![\phi(x)]\!]_{PHP}$, which we use to provide the witnessing terms $t'.1 : A$ and $t'.2 : [\![\phi(t.1)]\!]_{PHP}$ so that we can apply the introduction rule for $\exists$ twice.

For $\phi \vee \psi$, the proof is analogous to the one of $\exists$ with the use of the induction rules $\mathsf{IND}_{\|\cdot\|}$ and IND_{+} (see table A17). □

Theorem 4.3.4. *Given a judgement $\mathcal{J}$ of LEH, we have the following characterisation of the interpretation $[\![\cdot]\!]_{PHP}$ of LEH into H:*

$$\mathsf{H} \vdash [\![\mathcal{J}]\!]_{PHP} \qquad \textit{if and only if} \qquad \mathsf{LEH} + \mathsf{AUC} \vdash \mathcal{J}.$$

Proof. Immediate combining Lemma 4.3.1, Lemma 4.3.2 and Lemma 4.3.3. □

Theorem 4.3.4 is not only of interest in its own right, but also for the analysis of the interpretations of chapter 5. Indeed, two of the principles used to analyse the interpretation $[\![\cdot]\!]_{\infty,1}$ are $\mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$, which are two particular cases of AUC (see table 4.7 and table 4.8). Moreover, Theorem 5.3.5 isolates the propositions-as-hprops and the propositions-as-types interpretations as the key difference between the interpretations $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$.

The proof of the characterisation of PaHP that we have obtained is similar to the proof of the corresponding characterisation for the propositions-as-types interpretation. In that context it is proved in [AG00, Theorem 2] that, for a judgement, $\mathcal{J}$ we have: $\mathsf{LE} + \mathsf{AC} \vdash \mathcal{J}$ if and only if $\mathsf{ML_1W} \vdash [\![\mathcal{J}]\!]_{PT}$.

The statement in [AG00] uses an additional principle ($\mathsf{0}\bot$) asserting that we can deduce $\bot$ from the existence of a term $t : \mathsf{0}$. However, the principle ($\mathsf{0}\bot$) follows from the rule $\mathsf{IND}_{\mathsf{0}}$ which is one of the rules of the logic-enriched type theories LE and LEH (see table A17).

Thus we have a parallel characterisation of the propositions-as-hprops and the propositions-as-types interpretations in terms of the choice principles AUC and AC.

4.4. A logic-enriched type theory for $[\![\cdot]\!]_{\infty,1}$

In this section we introduce other principles PU, $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ (see table 4.6, table 4.7 and table 4.8) for the logic-enriched type theory LE that we use in chapter 5 to provide an analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$. They are designed to obtain a factorisation of that interpretation through an intermediate step by isolating the key steps of the proof of Theorem 1.3.14 of the soundness of the interpretation $[\![\cdot]\!]_{\infty,1} : \mathsf{CST} \to \mathsf{HoTT}$.

The principles $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ are inspired by the characterisation of Theorem 4.3.4 of Section 4.3, as well as from the analysis of Aczel's original interpretation in [GA06]. The principle PU is a propositions-as-types principle whereas $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ are principles that have the form of the axiom of unique choice for specific types and for specific equality propositions.

Other principles for a logic-enriched type theory. The propositions-as-types principle PU gives a function assigning a small type to every small proposition $\mathsf{pu}(\_) : \mathsf{P} \to U$. Then it further requires that the proposition $\tau(p) : \mathsf{prop}$ corresponding to $p : \mathsf{P}$ is equivalent to the corresponding type being inhabited. This principle is introduced in [GA06, Section Propositions-as-types of §2 and Lemma 2.1], where it is used in the analysis of Aczel's interpretation.

$$\frac{p : \mathsf{P}}{\mathsf{pu}(p) : \mathsf{U}} \qquad \frac{p : \mathsf{P}}{\tau(p) \equiv (\exists\_ : \mathsf{pu}(p))\top}$$

TABLE 4.6. Propositions-as-types principle (PU)

Notice that the principle PaHP we introduced in table 4.3 implies PU and is actually stronger since it applies to arbitrary propositions and not only small ones.

Moreover PaHP gives information about the map $\mathsf{pu}(\_) : \mathsf{P} \to \mathsf{U}$, which in that case is induced by the propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$. One could see PaHP as requiring further computation rules for the map $\mathsf{pu}(\_)$ on small propositions.

Also observe that in Lemma 4.3.3 we proved PaHP using the rules AUC and IND_0, and hence constructed a map $\mathsf{P} \to \mathsf{U}$. However, in the context of this section and of chapter 5 the full axiom of unique choice AUC is unnecessarily strong so we restrict to the special cases AUC_V, $\mathsf{AUC}_{\mathsf{El}(\beta)}$ and its consequence PU.

Another reason to prefer this formulation of PU is that by not imposing any other condition on the map $\mathsf{pu}(\_)$ we gain in flexibility. Indeed, we can use the same principle for the analysis of both the interpretations $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$, and possibly others.

The other two principles AUC_V and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ have the general form of the axiom of unique choice AUC (see table 4.5) but for specific types and with specific propositions taking the role of the equality proposition Eq. The principle AUC_V applies only to the type of sets V and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ only for the types of elements of a set $\mathsf{El}(\beta)$ for $\beta : \mathsf{V}$.

Before presenting the principle AUC_V we define the type of sets V. The same construction of the type $\mathsf{V} := (Wx : \mathsf{U})\mathsf{T}(x)$ of Martin-Löf type theory carries over in the logic-enriched type theory LE. Quantification over a set $\alpha : \mathsf{V}$ is defined as:

$$(\nabla x \in \alpha)\phi(x) := (\nabla x : \mathsf{El}(\alpha))\phi(\alpha_x),$$

here ∇ is either $\forall$ or $\exists$. However, when defining equality for V as a bisimulation relation, we do not define it as a type, but as a formula of the logic. One can define for $\alpha, \beta : \mathsf{V}$, a formula satisfying:

$$\alpha \approx_\mathsf{V} \beta \equiv \Big(\forall\exists \frac{x \in \alpha}{y \in \beta}\Big) x \approx_\mathsf{V} y. \tag{4.1}$$

Where the notation $(\forall\exists\frac{x\in\alpha}{y\in\beta})\phi(x,y)$ is a shorthand for

$$(\forall x \in \alpha)(\exists y \in \beta)\phi(x,y) \wedge (\forall y \in \beta)(\exists x \in \alpha)\phi(x,y).$$

For the details of the definition of $\approx_\mathsf{V}$ in LE see [GA06, Lemma 3.10].

The principles AUC_V and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ both make use of a unique existential quantification. For AUC_V, the formula $(\exists! y : \mathsf{V})\psi(y)$ expresses the unique quantification for $y : \mathsf{V}$. It is defined using the bisimulation relation $\approx_\mathsf{V}$ as follows:

$$(\exists y : \mathsf{V})\psi(y) \wedge (\forall z, z' : \mathsf{V})\psi(z) \wedge \psi(z') \supset z \approx_\mathsf{V} z'. \tag{4.2}$$

On the other hand, for $\mathsf{AUC}_{\mathsf{El}(\beta)}$, the formula $(\exists! y \in \beta)\psi(y)$ expressing the unique existential quantification for $\beta : \mathsf{V}$ is defined as:

$$(\exists y : \mathsf{El}(\beta))\psi(\beta_y) \wedge (\forall z, z' : \mathsf{El}(\beta))\psi(\beta_z) \wedge \psi(\beta_{z'}) \supset \beta_z \approx_{\mathsf{V}} \beta_{z'}. \tag{4.3}$$

We now present the principle $\mathsf{AUC}_{\mathsf{V}}$. Consider a formula $\phi(x, y)$ with $x : A$ and $y : \mathsf{V}$ such that ϕ satisfies a functionality condition in x and y expressed using the unique existential quantification of (4.2). Then there exists a type-theoretic function $f : A \to \mathsf{V}$ that for $x : A$ 'chooses' the unique element $y : \mathsf{V}$ satisfying the formula $\phi(x, y)$.

$$\frac{(\forall x : A)(\exists! y : \mathsf{V})\phi(x, y)}{(\exists f : A \to \mathsf{V})(\forall x : A)\phi(x, f(x))}$$

TABLE 4.7. Axiom of unique choice for V ($\mathsf{AUC}_{\mathsf{V}}$)

The third principle $\mathsf{AUC}_{\mathsf{El}(\beta)}$ is similar to $\mathsf{AUC}_{\mathsf{V}}$. Given a formula $\phi(x, y)$ with $x : A$ and $y : \mathsf{V}$, if ϕ satisfies a functional condition with respect to a set $\beta : \mathsf{V}$, i.e.

$$(\forall x : A)(\exists! y \in \beta)\phi(x, y),$$

where the unique existential quantifier is defined in (4.3). Then we can deduce the existence of a type-theoretic function $f : A \to \mathsf{El}(\beta)$ 'choosing' the unique terms determined by the formula.

$$\frac{\beta : \mathsf{V} \qquad (\forall x : A)(\exists! y \in \beta)\phi(x, y)}{(\exists f : A \to \mathsf{El}(\beta))(\forall x : A)\phi(x, \beta_{f(x)})}$$

TABLE 4.8. Axiom of unique choice for terms of V ($\mathsf{AUC}_{\mathsf{El}(\beta)}$)

Despite the similarities between $\mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ we had to formulate them as separate principles since the functions that are obtained in the conclusions have different codomains, namely V and $\mathsf{El}(\beta)$. Furthermore, the proofs of their validity in HoTT are also different (see Theorem 4.4.1 and Theorem 4.4.2).

Nevertheless, they are both instances of AUC for specific type families and specific reinterpretations of the equality proposition Eq as detailed in the table below.

	family of types $x : A \vdash B(x)$	equality proposition $\mathsf{Eq}_{B(x)}(a, b)$
$\mathsf{AUC}_{\mathsf{V}}$	V	$a \approx_{\mathsf{V}} b$
$\mathsf{AUC}_{\mathsf{El}(\beta)}$	$\mathsf{El}(\beta)$	$\beta_a \approx_{\mathsf{V}} \beta_b$

Validity of the principles under $[\![\cdot]\!]'_{PHP}$. Now we consider the other propositions-as-hprops interpretation $[\![\cdot]\!]'_{PHP} : \mathsf{LE} \to \mathsf{HoTT}$. With the exception of the interpretation of equality $[\![\cdot]\!]'_{PHP}$ has the same inductive definition on formulas as the interpretation $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$ of Section 4.2, this is because LE does not have the equality proposition Eq. The crucial difference between $[\![\cdot]\!]_{PHP}$ and $[\![\cdot]\!]'_{PHP}$ is that they have different domain and codomain theories.

Since we have introduced the principles $\mathsf{PU}, \mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$, we wish to establish their relative consistency with respect to the type theory HoTT via the propositions-as-hprops interpretation $[\![\cdot]\!]'_{PHP} : \mathsf{LE} \to \mathsf{HoTT}$. In Lemma 4.3.2 we have already proved that AUC is valid in the other propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP} : \mathsf{LEH} \to \mathsf{H}$, but the interpretation of the equality propositions differs in the two propositions-as-hprops interpretations.

Note that the proofs of the following Theorem 4.4.1 and Theorem 4.4.2 use the equivalence between the setoids $(\mathsf{V}_\infty, \approx_1) \cong (\mathsf{V}_{\mathsf{G}}, \mathsf{Id}_{\mathsf{V}_{\mathsf{G}}})$ of Theorem 3.2.9 that we proved in chapter 3.

Theorem 4.4.1. *The principles* PU *and* $\mathsf{AUC}_{\mathsf{V}}$ *are valid in the propositions-as-hprops interpretation* $[\![\cdot]\!]'_{PHP} : \mathsf{LE} \to \mathsf{HoTT}$.

Proof. The rules of PU are an immediate consequence of the definition of the interpretation $[\![\cdot]\!]'_{PHP}$.

The proof of the validity of $\mathsf{AUC}_{\mathsf{V}}$ is essentially a reformulation in this context of Theorem 3.5.2 of the validity of Replacement in $[\![\cdot]\!]_H$ and $[\![\cdot]\!]_G$ of chapter 3. Note that the type V of LE is interpreted as the type V_∞ of HoTT under $[\![\cdot]\!]'_{PHP}$. Given the premiss of $\mathsf{AUC}_{\mathsf{V}}$, unfolding the definition of its interpretation we have:

$$(\Pi x : A) \left\| (\Sigma y : \mathsf{V}_\infty)[\![\phi(x, y)]\!]'_{PHP} \right\|,$$

together with the uniqueness condition which is

$$(\Pi z, z' : \mathsf{V}_\infty)[\![\phi(x, z)]\!]'_{PHP} \times [\![\phi(x, z')]\!]'_{PHP} \to (z \approx_1 z').$$

We claim that from these two conditions we can derive the corresponding ones for $[\![\cdot]\!]_G$ eq. (4.5) and eq. (4.4) below. For this observe that the inductive definition

of $[\![\cdot]\!]'_{PHP}$ (see Section 4.2) is the same as the one of $[\![\cdot]\!]_G$ (see Section 3.2). Hence following the same reasoning of Theorem 3.5.1 we have a logical equivalence between the interpretations $[\![\cdot]\!]'_{PHP}$ and $[\![\cdot]\!]_G$.

Therefore we have

$$(\Pi x : A) \left\| (\Sigma y : \mathsf{V_G})[\![\phi(x,y)]\!]_G \right\|, \tag{4.4}$$

and

$$(\Pi z, z' : \mathsf{V_G})[\![\phi(x,z)]\!]_G \times [\![\phi(x,z')]\!]_G \to \mathsf{Id}_{\mathsf{V_G}}(z,z'). \tag{4.5}$$

Analogously to Theorem 3.5.2, the uniqueness condition in (4.5) implies that the type $(\Sigma y : \mathsf{V_G})[\![\phi(x,y)]\!]_G$ is a hproposition so that we can apply the principle of unique choice (Lemma 1.1.10) obtaining a dependent function:

$$(\Pi x : A)(\Sigma y : \mathsf{V_G})[\![\phi(x,y)]\!]_G.$$

Then applying the type-theoretic axiom of choice and using again the logical equivalence between the interpretations $[\![\cdot]\!]'_{PHP}$ and $[\![\cdot]\!]_G$ we have the interpretation of the conclusion of the rule $\mathsf{AUC_V}$. □

Theorem 4.4.2. *The principle* $\mathsf{AUC}_{\mathsf{El}(\beta)}$ *is valid in the propositions-as-hprops interpretation* $[\![\cdot]\!]'_{PHP} : \mathsf{LE} \to \mathsf{HoTT}$.

Proof. The proof for $\mathsf{AUC}_{\mathsf{El}(\beta)}$ is similar to the one for $\mathsf{AUC_V}$. Since the inductive definition of $[\![\cdot]\!]'_{PHP}$ is the same as the one of $[\![\cdot]\!]_G$ Theorem 3.5.1 extends to $[\![\cdot]\!]'_{PHP}$ giving a logical equivalence between the interpretations $[\![\cdot]\!]'_{PHP}$ and $[\![\cdot]\!]_G$.

Let us consider the interpretation of the premiss $[\![(\forall x : A)(\exists! y \in \beta)\phi(x,y)]\!]'_{PHP}$, which is logically equivalent to $(\Pi x : A)[\![(\exists! y \in \beta)\phi(x,y)]\!]_G$.

Unfolding the interpretation we have

$$(\Pi x : A)\left\| (\Sigma y : \mathsf{El}(\beta))[\![\phi(x,\beta_y)]\!]_G \right\|,$$

together with the uniqueness condition:

$$(\Pi z, z' : \mathsf{El}(\beta))[\![\phi(x,\beta_z)]\!]_G \times [\![\phi(x,\beta_{z'})]\!]_G \to \mathsf{Id}_{\mathsf{V_G}}(\beta_z,\beta_{z'}).$$

Now recall that, by definition of $\mathsf{V_G}$, given a term $\beta : \mathsf{V_G}$ we have that the map $\beta_\_ : \mathsf{El}(\beta) \to \mathsf{V_G}$ is an embedding, therefore from the uniqueness condition we obtain a term inhabiting the type $\mathsf{Id}_{\mathsf{El}(\beta)}(z,z')$. Hence, an application of the principle of unique choice gives:

$$(\Pi x : A)(\Sigma y : \mathsf{El}(\beta))[\![\phi(x,\beta_y)]\!]_G.$$

Applying the type-theoretic axiom of choice and the logical equivalence between $[\![\cdot]\!]_G$ and $[\![\cdot]\!]'_{PHP}$ we have the conclusion of the rule $\mathsf{AUC}_{\mathsf{El}(\beta)}$, as required. □

CHAPTER 5

Analysis of the interpretations via logic-enriched type theories

In chapter 3 we showed that for $2 \leq k \leq \infty$ the interpretations:

$$[\![\cdot]\!]_H, [\![\cdot]\!]_G, [\![\cdot]\!]_{k,1} : \mathsf{CST} \to \mathsf{HoTT} + \mathsf{V_H},$$

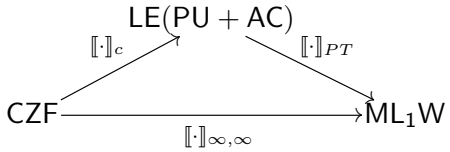
are equivalent, in the sense that there are equivalences of setoids

$$(\mathsf{V}_k, \approx_1) \cong (\mathsf{V_H}, \mathsf{Id}_{\mathsf{V_H}}) \cong (\mathsf{V_G}, \mathsf{Id}_{\mathsf{V_G}}),$$

which induce a logical equivalence of the interpretations of formulas, as detailed in Theorem 3.5.1.

A natural question is then to relate these interpretations to Aczel's original interpretation $[\![\cdot]\!]_{\infty,\infty}$ of CZF. The aim of this chapter is to answer this question by giving a uniform account of these interpretations via logic-enriched type theories. For this purpose we focus on the interpretation $[\![\cdot]\!]_{\infty,1}$ which is the only one among the ones above where the type interpreting sets is the same as Aczel's V_∞. This makes the comparative analysis technically easier.

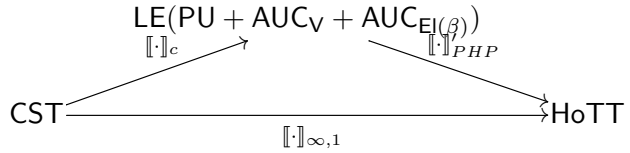
Our analysis parallels the one done in [GA06] for Aczel's interpretation of CZF into Martin-Löf type theory. There, the interpretation $[\![\cdot]\!]_{\infty,\infty}$ is factored through the logic-enriched type theory $\mathsf{LE}(\mathsf{AC} + \mathsf{PU})$[1], as the combinatorial interpretation $[\![\cdot]\!]_c$ followed by the propositions-as-types interpretation $[\![\cdot]\!]_{PT}$ (see [GA06] for more details). In the factorisation the type-theoretic axiom of choice AC (see table 4.4) and the propositions-as-types principle PU (see table 4.6) allow one to derive in the logic-enriched type theory the Collection axioms, as in the following diagram:



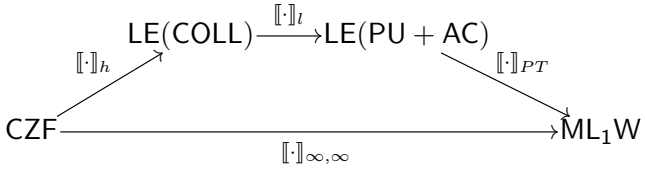
We factor the interpretation $[\![\cdot]\!]_{\infty,1}$ in two steps. The first step the combinatorial interpretation $[\![\cdot]\!]_c$ of CST into $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC_V} + \mathsf{AUC}_{\mathsf{El}(\beta)})$ that we introduced in

[1]Note that we use a slightly different notation: in [GA06] our theory $\mathsf{LE}(\mathsf{PU} + \mathsf{AC})$ is denoted as $\mathsf{ML}(\mathsf{PU} + \mathsf{AC})$.

Section 4.4. The second step is the propositions-as-hprops interpretation $[\![\cdot]\!]'_{PHP}$ of Section 4.4 into HoTT.



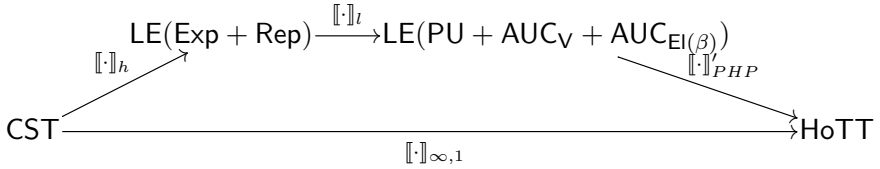
As an additional part of the analysis, in [GA06], the combinatorial interpretation of CZF into LE(PU + AC) is factored further through another logic-enriched type theory LE(COLL) where the Collection axioms are validated using the Collection Principles COLL (see tables 5.1 and 5.2). Then the logic-enriched type theory LE(COLL) is interpreted into LE(AC + PU), where the axiom of choice AC (see table 4.4) and the propositions-as-types principle PU (see table 4.6) allow to derive the Collection principles COLL.



The interpretation $[\![\cdot]\!]_l$ of one logic-enriched type theory into the other is defined in (5.1) in Section 5.3, and $[\![\cdot]\!]_h$ is the hybrid interpretation that we present in Section 5.2.

Similarly, we further factor the interpretation of CST into the logic-enriched type theory LE(PU + AUC$_V$ + AUC$_{El(\beta)}$). First, CST is interpreted via the hybrid interpretation $[\![\cdot]\!]_h$ in a different logic-enriched type theory LE(Rep + Exp) with two rules Rep and Exp (see tables 5.3 and 5.4) that directly allow to validate Replacement and Exponentiation. This is followed by the interpretation $[\![\cdot]\!]_l$ of LE(Rep + Exp) into LE(PU + AUC$_V$ + AUC$_{El(\beta)}$), which is induced by Lemma 5.3.2. This further step clarifies what is the role of the principles PU, AUC$_V$ and AUC$_{El(\beta)}$ in the interpretation of set theory. Indeed, Theorem 5.3.4 and Theorem 5.3.3 show that in presence of PU the principles Rep and Exp follow from AUC$_V$ and AUC$_{El(\beta)}$, respectively.

The main result of this chapter is Theorem 5.3.5 which summarises the situation with the following diagram of interpretations:



In this way we have a step-by-step parallel analysis of the two interpretations $[\![\cdot]\!]_{\infty,\infty}$ and $[\![\cdot]\!]_{\infty,1}$.

Outline. In Section 5.1 we develop the combinatorial interpretation of CST into the logic-enriched type theory LE(PU + AUC$_V$ + AUC$_{El(\beta)}$) which factors the interpretation $[\![\cdot]\!]_{\infty,1}$ through the propositions-as-hprops interpretation $[\![\cdot]\!]'_{PHP}$. In Section 5.2 we further factor the combinatorial interpretation through the hybrid interpretation of CST into the logic-enriched type theory LE(Rep+Exp). Section 5.3 relates the different logic-enriched type theories considered in this chapter. Finally, Section 5.4 summarises the comparative analysis and the results obtained.

5.1. The combinatorial interpretation

In LE the logical and type-theoretic structure is sufficiently rich for us to define different notions of 'collection of objects of a given type'. See [GA06, Section 3] for a detailed discussion of different notions of collection. In this section we focus on the case where collections are given by *families*, i.e. a family of terms of a type A indexed by a small type. This notion gives rise to the combinatorial interpretation of CST, whereas the notion of *subclass* (see Definition 5.2.1) gives rise to the hybrid interpretation of CST of Section 5.2.

The interpretation $[\![\cdot]\!]_c$ is called combinatorial interpretation since it is based on the combinatorial notion of family. The terminology is chosen to stress the difference with the logical notion of collection given by $\mathsf{Cla}(A) := A \to \mathsf{P}$. For more details on the combinatorial, logical and hybrid notions of collection we refer the reader to [GA06, Section 3].

Definition 5.1.1. Given a type A, a *family over* A is a term of the type:

$$\mathsf{Fam}(A) := (\Sigma a : \mathsf{U})\mathsf{T}(a) \to A.$$

Given a family $\alpha : \mathsf{Fam}(A)$, we define $\mathsf{El}(\alpha) := \mathsf{T}(\alpha.1)$ and for $x : \mathsf{El}(\alpha)$ we define $\alpha_x := \alpha.2(x) : A$.

We define quantification over families in the expected way. Namely, given a family $\alpha : \mathsf{Fam}(A)$ and a judgement $x : A \vdash \phi(x) : \mathsf{prop}$ with a free variable ranging

over A we define quantification over the family α as:

$$(\forall x \in \alpha)\phi(x) := (\forall x : \mathsf{El}(\alpha))\phi[\alpha_x],$$

$$(\exists x \in \alpha)\phi(x) := (\exists x : \mathsf{El}(\alpha))\phi[\alpha_x].$$

Using these and the decoding function τ for the proposition universe P, we define quantification over α for a family of small propositions $x : A \vdash p : \mathsf{P}$ in the evident way.

We develop the combinatorial interpretation of CST into the logic-enriched type theory $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC}_{\mathsf{V}} + \mathsf{AUC}_{\mathsf{El}(\beta)})$. We show that using PU, the notion of family allows us to validate the Bounded Separation axiom. Moreover, the principles $\mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ imply the validity of the Replacement and Exponentiation axioms respectively.

The combinatorial interpretation $[\![\cdot]\!]_c$ of CST into $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC}_{\mathsf{V}} + \mathsf{AUC}_{\mathsf{El}(\beta)})$ interprets sets as terms of the type V, set-theoretic equality as the bisimulation relation $\approx_{\mathsf{V}}$ (introduced in (4.1)) and the logical structure of set theory with the corresponding structure of the logic-enriched type theory, where quantification is defined in terms of families.

$$\begin{aligned}
[\![\alpha \doteq \beta]\!]_c &:= (\alpha \approx_{\mathsf{V}} \beta) \\
[\![\phi \star \psi]\!]_c &:= [\![\phi]\!]_c \star [\![\psi]\!]_c \\
[\![(\nabla x \in \alpha)\phi(x)]\!]_c &:= (\nabla x \in \alpha)[\![\phi(x)]\!]_c \\
[\![\nabla x\, \phi(x)]\!]_c &:= (\nabla x : \mathsf{V})[\![\phi(x)]\!]_c
\end{aligned}$$

Where ∇ is a quantifier and $\star$ a logical connective.

The proof of the following theorem does not involve the principles $\mathsf{PU}, \mathsf{AUC}_{\mathsf{V}}$ nor $\mathsf{AUC}_{\mathsf{El}(\beta)}$.

Theorem 5.1.2 (Gambino and Aczel)**.** *The following axioms of* CST *are valid in the combinatorial interpretation: Extensionality, Set-Induction, Pairing, Union and Infinity.*

Proof. For the proof see [GA06, Theorem 4.5]. □

Now we see how the Bounded Separation, Replacement and Exponentiation axioms follow respectively from $\mathsf{PU}, \mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$.

Theorem 5.1.3 (Gambino and Aczel)**.** *The rule* PU *implies the validity of Bounded Separation in the combinatorial interpretation.*

Proof. Given a type A, a family $\alpha : \mathsf{Fam}(A)$ and a proposition $x : A \vdash p(x) : \mathsf{P}$ with a free variable ranging over A, then we can define $\{x : A \,|\, p(x)\}$ as the family $\gamma : \mathsf{Fam}(A)$ determined by

$$\mathsf{el}(\gamma) := (\sigma x : \mathsf{el}(\alpha))\mathsf{pu}(p(\alpha_x)),$$

$$\gamma_z := \alpha_{z.1} : A.$$

Checking that γ has the desired properties is straightforward.

This definition is possible thanks to PU (see table 4.6), that for any $p : \mathsf{P}$ gives $\mathsf{pu}(p) : \mathsf{U}$. □

Theorem 5.1.4. *The rule* $\mathsf{AUC_V}$ *implies the validity of Replacement in the combinatorial interpretations.*

Proof. We want to validate the Replacement axiom:

$$(\forall x \in \alpha)\exists! y \phi(x, y) \Rightarrow \exists \beta (\forall x \in \alpha)(\exists y \in \beta)\phi(x, y).$$

Consider the combinatorial interpretation of the premiss of Replacement. Unfolding the definition of the combinatorial interpretation we have:

$$(\forall x : \mathsf{El}(\alpha))(\exists! y : \mathsf{V})[\![\phi(\alpha_x, y)]\!]_c.$$

Then we can apply the rule $\mathsf{AUC_V}$ (see table 4.7) obtaining:

$$(\exists f : \mathsf{El}(\alpha) \to \mathsf{V})(\forall x : \mathsf{El}(\alpha))[\![\phi(\alpha_x, f(x))]\!]_c. \tag{5.1}$$

Now that we have the function $f : \mathsf{El}(\alpha) \to \mathsf{V}$ we can define β by $\mathsf{El}(\beta) := \mathsf{El}(\alpha)$ and $\beta_x := f(x)$. Then the goal is to prove the interpretation of the conclusion of Replacement, i.e.

$$(\forall x \in \alpha)(\exists y \in \beta)[\![\phi(x, y)]\!]_c. \tag{5.2}$$

Unfolding the definition of the bounded quantifiers and the definition of β, (5.2) becomes:

$$(\forall x : \mathsf{El}(\alpha))(\exists y : \mathsf{El}(\alpha))[\![\phi(\alpha_x, f(y)]\!]_c.$$

For a given $x : \mathsf{El}(\alpha)$ the witness for the existential quantifier is then provided by x itself, using the conclusion of the rule $\mathsf{AUC_V}$. □

Theorem 5.1.5. *The rule* $\mathsf{AUC}_{\mathsf{El}(\beta)}$ *implies the validity of the Exponentiation axiom in the combinatorial interpretation.*

Proof. For Exponentiation, recall that in CST we define $\alpha \xrightarrow{F} \beta$ as a shorthand for the formula stating that F is a set of pairs with the first component in α and the second in β and that F is functional, i.e.

$$(\forall z \in F)(\exists x \in \alpha)(\exists y \in \beta)(z \doteq \langle x, y \rangle) \wedge (\forall x \in \alpha)(\exists! y \in \beta)(\langle x, y \rangle \in F). \tag{5.3}$$

Recall the Exponentiation axiom: $\forall\alpha,\beta\ \exists\gamma\ \forall F\ [(\alpha \overset{F}{\to} \beta) \Rightarrow F \in \gamma]$. So given two terms $\alpha, \beta : \mathsf{V}$ we define a term $\gamma : \mathsf{V}$ playing the role of the set of functions, so we let $\mathsf{El}(\gamma) := \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$. The map $\gamma_{\_} : (\mathsf{El}(\alpha) \to \mathsf{El}(\beta)) \to \mathsf{V}$ is defined by mimicking type-theoretically a set-theoretic function as a set of pairs. Given a function $z : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$, its image is given by

$$\mathsf{El}(\gamma_z) := \mathsf{El}(\alpha),$$

$$\gamma_{z_x} := \langle \alpha_x, \beta_{z(x)} \rangle.$$

Now given an $F : \mathsf{V}$ satisfying $[\![\alpha \overset{F}{\to} \beta]\!]_c$ we want to prove that $F \in \gamma$. For this purpose consider part of the hypothesis: $(\forall x \in \alpha)(\exists! y \in \beta)(\langle x, y\rangle \in F)$, recalling the definition of the universal quantifier in the combinatorial interpretation, it unfolds as:

$$(\forall x : \mathsf{El}(\alpha))(\exists! y \in \beta)\, \langle \alpha_x, y\rangle \in F.$$

Here we can apply the rule $\mathsf{AUC}_{\mathsf{El}(\beta)}$ (see table 4.8) obtaining:

$$(\exists f : \mathsf{El}(\alpha) \to \mathsf{El}(\beta))(\forall x : \mathsf{El}(\alpha))\, \langle \alpha_x, \beta_{f(x)}\rangle \in F. \tag{5.4}$$

Now we want to use f to prove that F is an element of γ, i.e. $[\![F \in \gamma]\!]_c$, which by definition is equal to $(\exists x : \mathsf{El}(\gamma))(F \approx_{\mathsf{V}} \gamma_x)$. The term $f : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$ provides the witness to prove this existential quantifier. Then to prove the equality $F \approx_{\mathsf{V}} \gamma_f$ we have to prove the two conjuncts of the bisimulation relation $F \approx_{\mathsf{V}} \gamma_f$. The first is:

$$(\forall s \in F)(\exists t \in \gamma_x)(F_s \approx_{\mathsf{V}} \gamma_{f_t}). \tag{5.5}$$

Recall from (5.3) that F is a set made of pairs, i.e.

$$(\forall z \in F)(\exists x \in \alpha)(\exists y \in \beta)\, z \approx_{\mathsf{V}} \langle x, y\rangle, \tag{5.6}$$

therefore combining (5.6) with the uniqueness condition

$$(\forall x \in \alpha)(\exists! y \in \beta)\langle x, y\rangle \in F,$$

we have that given a term $x : \mathsf{El}(\alpha)$ the function $f : \mathsf{El}(\alpha) \to \mathsf{El}(\beta)$ provides the second component of the pair $f(\alpha_x)$:

$$(\forall z : \mathsf{El}(F))(\exists x : \mathsf{El}(\alpha))\, z \approx_{\mathsf{V}} \langle \alpha_x, f(\alpha_x)\rangle.$$

This gives the first conjunct of the bisimulation relation of (5.5).

The second conjunct of the bisimulation relation is:

$$(\forall v \in \gamma_f)(\exists w \in F) F_v \approx_{\mathsf{V}} \gamma_{f_w}. \tag{5.7}$$

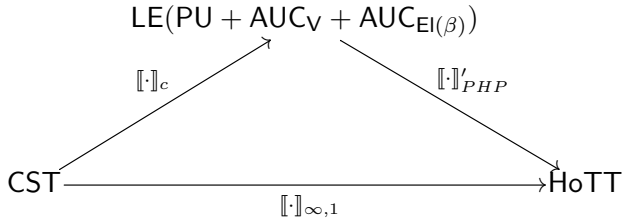
then using the definition of γ_f, (5.7) is equivalent to:

$$(\forall v : \mathsf{El}(\alpha))(\exists w \in F) F_v \approx_{\mathsf{V}} \langle \alpha_w, \beta_{f(w)} \rangle.$$

This follows from (5.4), which is the conclusion of the rule $\mathsf{AUC}_{\mathsf{El}(\beta)}$. □

We conclude this first part of our analysis of the interpretation $[\![\cdot]\!]_{\infty,1}$ with the following theorem.

Theorem 5.1.6. *The interpretation* $[\![\cdot]\!]_{\infty,1} : \mathsf{CST} \to \mathsf{HoTT}$ *factors as the combinatorial interpretation* $[\![\cdot]\!]_c$ *followed by the propositions-as-hprops interpretation* $[\![\cdot]\!]'_{PHP}$.



Proof. This follows by induction on the formula of CST that one is interpreting, and inspecting the definitions of the three interpretations: $[\![\cdot]\!]_{\infty,1}$ in Section 1.2, and $[\![\cdot]\!]_c$ in Section 5.1 and $[\![\cdot]\!]'_{PHP}$ in Section 4.2.

To illustrate, we consider the case of set-theoretic equality $\doteq$. It is interpreted under $[\![\cdot]\!]_c$ as the bisimulation relation $\alpha \approx_{\mathsf{V}} \beta$. This relation is then interpreted under $[\![\cdot]\!]'_{PHP}$ as the bisimulation relation $\approx_1$ on the type V_∞, which is the interpretation of set-theoretic equality under $[\![\cdot]\!]_{\infty,1}$.

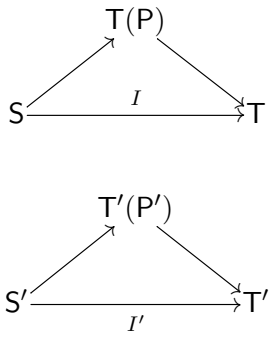
Similarly for the other formulas.

□

Remark 5.1.7. The method and guiding principles that we have used in this chapter are quite general and could be applied to other contexts.

Suppose one has an interpretation (or a model) I of the system S into the theory T, and a similar interpretation I' of system S' into the theory T'. By a careful analysis of the proofs one can isolate the key principles P and P' used in

these interpretations. Then one can factor the interpretations as:



One can strive to formulate the principles P and P' so that the factorisation of the interpretations I and I' isolate all the aspects that are similar, which are then treated in exactly the same way by the arrows on the left hand side of the factorisation. What is different between I and I' is neatly isolated on the right hand side of the factorisation.

Then one can analyse further the interpretations $\mathsf{T}(\mathsf{P}) \to \mathsf{T}$ and $\mathsf{T}'(\mathsf{P}') \to \mathsf{T}'$ that encode the differences between I and I'.

To a certain extent, this is what we have done in the characterisation of the propositions-as-hprops interpretation $[\![\cdot]\!]_{PHP}$ of Theorem 4.3.4, with the caveat that the interpretations $[\![\cdot]\!]_{PHP}$ and $[\![\cdot]\!]'_{PHP}$ have different domain and codomain theories.

Moreover, if in the interpretations I and I' more constructions are happening at the same time one can factor the interpretations step by step, formulating principles that encode what is used at each stage. This is what we do in Section 5.2 where we factor further the combinatorial interpretation.

5.2. The hybrid interpretation

In this section we follow [GA06, §4] and refine the analysis of Section 5.1 factoring further the combinatorial interpretation $[\![\cdot]\!]_c$. First, we define the hybrid interpretation of CST into the logic-enriched type theory $\mathsf{LE}(\mathsf{Rep} + \mathsf{Exp})$, then an interpretation of $\mathsf{LE}(\mathsf{Rep} + \mathsf{Exp})$ into $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC}_{\mathsf{V}} + \mathsf{AUC}_{\mathsf{El}(\beta)})$. The Replacement and Exponentiation rules Rep and Exp (see tables 5.3 and 5.4) are formulated so that they are natural principles for a logic-enriched type theory and also allow us to validate the Replacement and Exponentiation axioms in the hybrid interpretation.

One reason for our interest in this further step is that in the combinatorial interpretation the validity of Bounded Separation uses crucially the propositions-as-types principle PU. Instead, the hybrid interpretation $[\![\cdot]\!]_h$ uses a different notion of collection, the one of *subclass* (see Definition 5.2.1 below). A subclass of a type A consists of a family of terms of A indexed by a small type $i : \mathsf{U}$ with the extra

structure of a formula $x : \mathsf{T}(i) \vdash \phi(x) : \mathsf{prop}$. The objects of a subclass are the terms in A indexed by $x : \mathsf{T}(i)$ such that $\phi(x)$ holds. The name 'hybrid' comes precisely from the idea of combining the notion of family with logical aspects.

Using the hybrid interpretation, the Bounded Separation axiom follows directly from the structure of the interpretation without the need of any additional principle.

Before moving to the new features that we introduce in order to factor the combinatorial interpretation $[\![\cdot]\!]_c$, we first present what is in common between $\mathsf{LE}(\mathsf{Rep} + \mathsf{Exp})$ and $\mathsf{LE}(\mathsf{COLL})$ of [GA06, §3] and give a review of their rules COLL.

Definition 5.2.1. Given a type A, a *subclass of* A is a term of the type:

$$\mathsf{Sub}(A) := (\Sigma x : \mathsf{U})[(\mathsf{T}(x) \to \mathsf{P}) \times (\mathsf{T}(x) \to A)].$$

Given a subclass $\alpha : \mathsf{Sub}(A)$ we use the following notation: $\mathsf{El}(\alpha) := \mathsf{T}(\alpha.1)$, and $\alpha_x := \alpha.2.2(x) : A$, and $\mathsf{dom}(\alpha, x) := \alpha.2.1(x) : \mathsf{P}$.

We can then introduce quantification over a subclass. Given a subclass $\alpha : \mathsf{Sub}(A)$ and a family of propositions $x : A \vdash \phi(x) : \mathsf{prop}$, quantification over α means:

$$(\forall x \in \alpha)\phi(x) := (\forall x : \mathsf{El}(\alpha))(\mathsf{dom}(\alpha, x) \supset \phi[\alpha_x/x]\,),$$
$$(\exists x \in \alpha)\phi(x) := (\exists x : \mathsf{El}(\alpha))(\mathsf{dom}(\alpha, x) \wedge \phi[\alpha_x/x]\,).$$

Similarly for a family of small propositions $x : A \vdash p : \mathsf{P}$.

The definition of the type of iterative sets has to be adapted to match the new notion of subclass:

$$\mathbb{V} := (W z : (\Sigma x : \mathsf{U})(\mathsf{T}(x) \to \mathsf{P}))\mathsf{T}(z.1). \tag{5.1}$$

As a consequence of this definition we have a correspondence between the terms of $\mathbb{V}$ and $\mathsf{Sub}(\mathbb{V})$. Firstly, we define the function $\mathsf{set} : \mathsf{Sub}(\mathbb{V}) \to \mathbb{V}$ that given a subclass $(a, (p, f))$ with $a : \mathsf{U}$ and $p : \mathsf{T}(a) \to \mathsf{P}$ and $f : \mathsf{T}(a) \to \mathbb{V}$ returns $\mathsf{sup}((a, p), f) : \mathbb{V}$.

Secondly, we define the function $\mathsf{sub} : \mathbb{V} \to \mathsf{Sub}(\mathbb{V})$ that given a set $\mathsf{sup}((a, p), f)$ returns the subclass $(a, (p, f))$.

Lemma 5.2.2 (Gambino and Aczel)**.** *Given two formulas* $x : \mathbb{V} \vdash \phi(x) : \mathsf{prop}$ *and* $y : \mathsf{Sub}(\mathbb{V}) \vdash \psi(y) : \mathsf{prop}$ *we have the following judgements:*

$$(\nabla x : \mathbb{V})\phi(x) \equiv (\nabla y : \mathsf{Sub}(\mathbb{V}))\phi(\mathsf{set}(y)),$$
$$(\nabla y : \mathsf{Sub}(\mathbb{V}))\psi(y) \equiv (\nabla x : \mathbb{V})\psi(\mathsf{sub}(x)).$$

Where ∇ *is* $\forall$ *or* $\exists$*.*

Proof. Straightforward, see [GA06, Lemmas 3.8 and 3.9] for details. □

Using the rules for W-types we can define a bisimulation relation $\approx_{\mathbb{V}}$ on the type of sets $\mathbb{V}$ such that the following judgement is derivable (see [GA06, equation 25] for details):

$$\alpha \approx_{\mathbb{V}} \beta \equiv \Big(\forall\exists \frac{x \in \alpha}{y \in \beta}\Big) x \approx_{\mathbb{V}} y.$$

The inductive structure of the hybrid interpretation and the bisimulation $\approx_{\mathbb{V}}$ are the same as the ones for the combinatorial ones, but the definition of quantification is different, being defined in terms of subclasses rather than families.

$$\begin{aligned}
[\![\alpha \doteq \beta]\!]_h &:= (\alpha \approx_{\mathbb{V}} \beta) \\
[\![\phi \star \psi]\!]_h &:= [\![\phi]\!]_h \star [\![\psi]\!]_h \\
[\![(\nabla x \in \alpha)\phi(x)]\!]_h &:= (\nabla x \in \alpha)[\![\phi(x)]\!]_h \\
[\![\nabla x\, \phi(x)]\!]_h &:= (\nabla x : \mathbb{V})[\![\phi(x)]\!]_h
\end{aligned}$$

Where ∇ is a quantifier and $\star$ a logical connective.

Note that in the following theorem Bounded Separation can be validated without any need for a propositions-as-types principle.

Theorem 5.2.3 (Gambino and Aczel)**.** *The basic axioms of* CST*, namely Extensionality, Set-Induction, Pairing, Union, Infinity and Bounded Separation are valid in the hybrid interpretation in* LE*.*

Proof. For the proof see [GA06, Lemmas 3.14 and 3.15]. □

In [GA06, §3] CZF is interpreted in the logic-enriched type theory LE(COLL) via the hybrid interpretation. Here we recall the Strong Collection and Subset Collection rules COLL. These two rules recast in the context of logic-enriched type theory and subclasses the familiar Collection axioms. More precisely the axioms are refined while maintaining their logical structure: instead of a set we have either a type A, a subclass $\mathsf{Sub}(A)$ or a subclass of subclass of types $\mathsf{Sub}^2(A)$ depending on the role played by that set within the axiom.

$$\frac{A, B : \mathsf{type} \qquad \alpha : \mathsf{Sub}(A) \qquad x : A, y : B \vdash \phi(x, y) : \mathsf{prop}}{(\forall x \in \alpha)(\exists y : B)\phi(x, y) \supset (\exists v : \mathsf{Sub}(B))(\forall\exists \tfrac{x\in\alpha}{y\in v}\phi(x, y))}$$

TABLE 5.1. Strong Collection rule (StrCOLL)

$$\frac{A, B, C : \mathsf{type} \qquad \alpha : \mathsf{Sub}(A) \qquad \beta : \mathsf{Sub}(B) \qquad x : A, y : B, z : C \vdash \phi : \mathsf{prop}}{(\exists u : \mathsf{Sub}^2(B))(\forall z : C)((\forall x \in \alpha)(\exists y \in \beta)\phi(x, y) \supset (\exists v \in u)(\forall \exists_{y \in v}^{x \in \alpha})\phi(x, y))}$$

TABLE 5.2. Subset Collection rule (SubCOLL)

Now we move to our logic-enriched type theory where we introduce a Replacement and Exponentiation rules that allow us to validate the corresponding axiom of Myhill's Constructive Set Theory CST. The Replacement rule Rep is designed like the Collection rules, maintaining the structure of the axiom while adapting it to the logic-enriched type theory.

$$\frac{A : \mathsf{type} \qquad \alpha : \mathsf{Sub}(A) \qquad x : A, y : \mathbb{V} \vdash \phi(x, y) : \mathsf{prop}}{(\forall x \in \alpha)(\exists! y : \mathbb{V})\phi(x, y) \supset (\exists \beta : \mathbb{V})(\forall x \in \alpha)(\exists y \in \beta)\phi(x, y)}$$

TABLE 5.3. Replacement rule (Rep)

Notice that in the premiss of the implication we insist in using the type $\mathbb{V}$ in $(\exists! y : \mathbb{V})$, defined using $\approx_{\mathbb{V}}$, rather than allowing an arbitrary type B. This is because we have a uniqueness quantification that we can express in terms of $\approx_{\mathbb{V}}$ for the type $\mathbb{V}$, but do not have for arbitrary types. Using the identity types or the equality propositions of LEH is not the desired option since our overall aim is to factor the interpretation $[\![\cdot]\!]_{\infty,1}$ in which set-theoretic equality is interpreted as a truncated bisimulation relation.

The Exponentiation rule Exp is a direct reformulation of the Exponentiation axiom in this new context.

Let us define the formula $(\alpha \xrightarrow{F} \beta)$ for given sets $\alpha, \beta : \mathbb{V}$, within the logic-enriched type theory, stating that F is a functional relation from α to β.

$$(\forall x \in \alpha)(\exists! y \in \beta)\, \langle x, y \rangle \in F \ \wedge\ (\forall z \in F)(\exists x \in \alpha)(\exists y \in \beta)\, z \approx_{\mathbb{V}} \langle x, y \rangle$$

The Exponentiation rule can now be stated more easily.

$$\frac{\alpha, \beta : \mathbb{V}}{(\exists \gamma : \mathbb{V})(\forall F : \mathbb{V})[(\alpha \xrightarrow{F} \beta) \supset F \in \gamma]}$$

TABLE 5.4. Exponentiation rule (Exp)

This rule is a direct reformulation of the Exponentiation axiom with only the type $\mathbb{V}$ appearing. It could not have been more general since in the formula $(\alpha \xrightarrow{F} \beta)$ we have the uniqueness quantification $(\exists! y \in \beta)$ which forces $\beta : \mathbb{V}$, and the equality $(z \approx_{\mathbb{V}} \langle \alpha, \beta \rangle)$ which also forces $F, \alpha : \mathbb{V}$. Moreover, in order to have $F \in \gamma$ being well-typed we also have to have $\gamma : \mathbb{V}$.

Having stated these rules we can finally use them to interpret CST into LE(Rep+ Exp).

Theorem 5.2.4. *The Replacement rule* Rep *implies the validity of the Replacement axiom in the hybrid interpretation. Similarly, the Exponentiation rule* Exp *implies the validity of the Exponentiation axiom in the hybrid interpretation.*

Proof. For Replacement let us apply the rule Rep of table 5.3 to the case $A = \mathbb{V}$. Using the correspondence between $\mathbb{V}$ and $\mathsf{Sub}(\mathbb{V})$ of Lemma 5.2.2 without loss of generality we can consider $\alpha : \mathbb{V}$ in the premiss. Then the conclusion of Rep gives the interpretation of the Replacement axiom.

For the Exponentiation axiom it is enough to observe that the Exponentiation rule Exp is equivalent to the hybrid interpretation of the Exponentiation axiom. □

5.3. Relating logic-enriched type theories

In this section we have three goals. First we relate the notions of family and subclass via the propositions-as-hprops principle PU and relate the two types of sets V and $\mathbb{V}$. Recall that $\mathsf{V} = (Wx : \mathsf{U})\mathsf{T}(x)$, and that $\mathbb{V}$ is defined in (5.1) as an adaptation of V for the hybrid interpretation. Secondly, we show that the logic-enriched type theory LE(Rep + Exp) can be reinterpreted in $\mathsf{LE}(\mathsf{PU} + \mathsf{AUC}_{\mathsf{V}} + \mathsf{AUC}_{\mathsf{El}(\beta)})$. These first two parts are an adaptation to this context of the analogous results of [GA06, §4]. Finally, we compare the principles $\mathsf{Rep}, \mathsf{Exp}, \mathsf{AUC}_{\mathsf{V}}$ and $\mathsf{AUC}_{\mathsf{El}(\beta)}$ used here for CST with the principles COLL and AC introduced in [GA06] for CZF.

As a first step we recall some notions and results from [GA06, §4] on the relationship between families and subclasses. We use PU to introduce two comparison

maps between families and subclasses of a given type. These allow us to relate the combinatorial and the hybrid interpretations of set theory.

We need to introduce a preliminary construction. For a subclass $\alpha : \mathsf{Sub}(A)$, the construction gives a subtype of $\mathsf{el}(\alpha)$ on which the formula $\mathsf{dom}(\alpha)$ holds, as follows:

$$\mathsf{comp}(\alpha) := (\sigma x : \mathsf{el}(\alpha))\mathsf{pu}(\mathsf{dom}(\alpha, x)) : \mathsf{U},$$

and

$$\mathsf{Comp}(\alpha) := \mathsf{T}(\mathsf{comp}(\alpha)).$$

We then define $i : \mathsf{Sub}(A) \to \mathsf{Fam}(A)$ which simply forgets the logical structure, by

$$i(\alpha) := (\mathsf{comp}(\alpha), \lambda z.\alpha_{z.1}),$$

for $\alpha : \mathsf{Sub}(A)$.

On the other hand, $j : \mathsf{Fam}(A) \to \mathsf{Sub}(A)$, is defined by

$$j(\sigma) := (\mathsf{el}(\sigma), (\lambda\_.\top, \lambda x.\sigma_x)).$$

Using these comparison maps we can state and prove the following lemma.

Lemma 5.3.1 (Gambino and Aczel)**.** *Given a proposition $x : A \vdash \phi(x) : \mathsf{prop}$, a subclass $\alpha : \mathsf{Sub}(A)$ and a family $\sigma : \mathsf{Fam}(A)$, the rule* PU *implies the validity of the following judgements:*

$$(\nabla x \in \alpha)\phi(x) \equiv (\nabla x \in i(\alpha))\phi(x),$$

$$(\nabla x \in \sigma)\phi(x) \equiv (\nabla x \in j(\sigma))\phi(x).$$

Where ∇ is $\exists$ or $\forall$.

Proof. See [GA06, Lemma 4.6] for details. □

We have similar comparison maps $J : \mathsf{V} \to \mathbb{V}$ and $I : \mathbb{V} \to \mathsf{V}$ for the types of sets of the combinatorial and hybrid interpretations. Given a canonical term $\mathsf{sup}(a, f) : \mathsf{V}$ with $f : \mathsf{T}(a) \to \mathsf{V}$ we define:

$$J(\mathsf{sup}(a, f)) := \mathsf{sup}((a, \lambda\_.\top), J \circ f).$$

In the other direction we have a canonical term $\alpha = \mathsf{sup}((a, p), f)$ where the terms p and f have types $p : \mathsf{T}(a) \to \mathsf{P}$ and $f : \mathsf{T}(a) \to \mathbb{V}$. The set α gives rise to its corresponding subclass $\mathsf{sub}(\alpha) : \mathsf{Sub}(\mathbb{V})$. Then we consider $\mathsf{comp}(\mathsf{sub}(\alpha))$, which is the code in the universe U corresponding to the Σ-type $(\Sigma x : \mathsf{T}(a))\mathsf{T}(\mathsf{pu}(p(x)))$. Finally, we define the function I on canonical terms as:

$$I(\mathsf{sup}((a, p), f)) := \mathsf{sup}(\mathsf{comp}(\mathsf{sub}(\alpha)), I \circ f).$$

Then we have the following comparison lemma.

Lemma 5.3.2 (Gambino and Aczel)**.** *Assuming* PU*, the comparison maps* $J : \mathsf{V} \to \mathbb{V}$ *and* $I : \mathbb{V} \to \mathsf{V}$ *give an isomorphism of setoids* $(\mathsf{V}, \approx_{\mathsf{V}}) \cong (\mathbb{V}, \approx_{\mathbb{V}})$*. Thus, given two judgements* $x : \mathbb{V} \vdash \phi(x) : \mathsf{prop}$ *and* $x : \mathsf{V} \vdash \psi(x) : \mathsf{prop}$*, we have logical equivalences:*

$$(\nabla x : \mathbb{V})\phi(x) \equiv (\nabla x : \mathsf{V})\phi(J(x)),$$
$$(\nabla x : \mathsf{V})\psi(x) \equiv (\nabla x : \mathbb{V})\psi(I(x)).$$

Where ∇ *is* $\exists$ *or* $\forall$*.*

In particular, assuming PU*, we have that for a given sentence* ϕ *of* CST *there is a logical equivalence of the combinatorial and hybrid interpretations* $[\![\phi]\!]_c \equiv [\![\phi]\!]_h$*.*

Proof. It is straightforward to check by recursion that the functions I and J are inverses of each other in the sense of $\approx_{\mathsf{V}}$ and $\approx_{\mathbb{V}}$.

The logical equivalence between formulas quantified over V and $\mathbb{V}$ follows directly from the isomorphism of setoids. □

The comparison maps i, j, I and J induce an interpretation of the logic-enriched type theory $\mathsf{LE}(\mathsf{Rep}+\mathsf{Exp})$ into $\mathsf{LE}(\mathsf{PU}+\mathsf{AUC}_{\mathsf{V}}+\mathsf{AUC}_{\mathsf{El}(\beta)})$ that interprets the type $\mathbb{V}$ as the type V, and quantification over a subclass of $\mathbb{V}$, as quantification over the corresponding family of V.

$$[\![\cdot]\!]_l : \mathsf{LE}(\mathsf{Rep}+\mathsf{Exp}) \longrightarrow \mathsf{LE}(\mathsf{PU}+\mathsf{AUC}_{\mathsf{V}}+\mathsf{AUC}_{\mathsf{El}(\beta)}) \tag{5.1}$$

Note that no other part of the logic-enriched type theory is changed by this interpretation. In other words, $[\![\cdot]\!]_l$ acts only on the type $\mathbb{V}$ and on formulas quantified over $\mathbb{V}$.

Now we use the previous lemmas to prove that this interpretation is sound.

Theorem 5.3.3. *Assuming* PU *we have that* $\mathsf{AUC}_{\mathsf{El}(\beta)}$ *implies* Exp*.*

Proof. Note that validating Exp is equivalent to validate the Exponentiation axiom in the hybrid interpretation. Since the Exponentiation axiom is a sentence of CST its hybrid interpretation is equivalent to its combinatorial interpretation thanks to Lemma 5.3.2 (which uses PU). So the result follows from Theorem 5.1.5 where we proved that the rule $\mathsf{AUC}_{\mathsf{El}(\beta)}$ implies the combinatorial interpretation of the Exponentiation axiom. □

The case of the Replacement rule is not as easy since the rule involves arbitrary types, hence it is more general than the hybrid interpretation of the Replacement axiom.

Theorem 5.3.4. *Assuming* PU *we have that* $\mathsf{AUC_V}$ *implies* Rep.

Proof. The proof is similar to the proof of Theorem 5.1.4 where we validated the Replacement axiom in the combinatorial interpretation. The premiss of the Rep rule gives $\alpha : \mathsf{Sub}(A)$ and a formula $x : A, y : \mathbb{V} \vdash \phi(x, y) : \mathsf{prop}$. The conclusion is an implication, so let us consider its antecedent. We have the following chain of logical equivalences:

$$\begin{aligned}
(\forall x \in \alpha)(\exists! y : \mathbb{V})\phi(x, y) &\equiv \\
(\forall x \in i(\alpha))(\exists! y : \mathbb{V})\phi(x, y) &\equiv && \text{(by Lemma 5.3.1)} \\
(\forall x \in i(\alpha))(\exists! y : \mathsf{V})\phi(x, J(y)) &\equiv && \text{(by Lemma 5.3.2)} \\
(\forall x : \mathsf{Comp}(\alpha))(\exists! y : \mathsf{V})\phi(\alpha_{x.1}, J(y)) && \text{(expanding the definition of } i(\alpha)).
\end{aligned}$$

Then applying $\mathsf{AUC_V}$ we have the following formula:

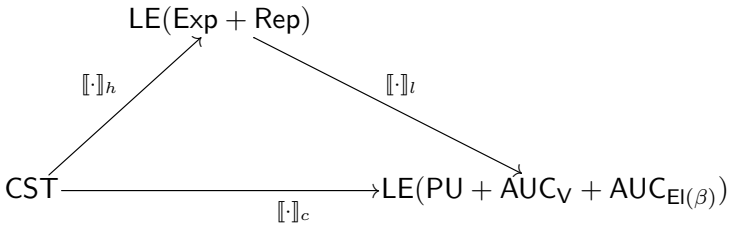
$$(\exists f : \mathsf{Comp}(\alpha) \to \mathsf{V})(\forall x : \mathsf{Comp}(\alpha))\ \phi(\alpha_{x.1}, J \circ f(x)).$$

Next we construct a witness β for the existential quantifier in the consequent of the implication of Exponentiation as $J(\mathsf{sup}(\mathsf{comp}(\alpha), f)) : \mathbb{V}$. This term can be made more explicit expanding the definition of J as $\mathsf{sup}((\mathsf{comp}(\alpha), \lambda\_.\top), J \circ f)$. So we get that

$$(\exists \beta : \mathbb{V})(\forall x : \mathsf{Comp}(\alpha))\phi(\alpha_{x.1}, J \circ f(x)),$$

which implies $(\exists \beta : \mathbb{V})(\forall x \in i(\alpha))(\exists y \in i(\beta))\phi(x, y)$, as desired. □

Theorem 5.3.5. *The combinatorial interpretation* $[\![\cdot]\!]_c$ *factors as the hybrid interpretation* $[\![\cdot]\!]_h$ *followed by the interpretation* $[\![\cdot]\!]_l$. *The situation can be summarised by the following diagram:*



Proof. The claim follows using Lemma 5.3.1 and Lemma 5.3.2 and inspecting the definitions of the three interpretations. Recall that the combinatorial interpretation is given in Section 5.1, the hybrid interpretation is given in Section 5.2 and the interpretation $[\![\cdot]\!]_l$ in Section 5.3 in (5.1). □

In the remainder of this section we look at the relationships between the rules $\mathsf{AUC_V}, \mathsf{AUC}_{\mathsf{El}(\beta)}, \mathsf{Rep}, \mathsf{Exp}$ introduced here to analyse the interpretation $[\![\cdot]\!]_{\infty,1}$ and the related rules $\mathsf{AC}, \mathsf{COLL}$ of [GA06] used to analyse Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$.

Theorem 5.3.6. *In the basic logic-enriched type theory* LE *we have the following implications:*

(i) $\mathsf{StrCOLL}$ *implies* Rep*;*
(ii) $\mathsf{SubCOLL}$ *implies* Exp*;*
(iii) AC *implies both* $\mathsf{AUC_V}$ *and* $\mathsf{AUC_{El(\beta)}}$.

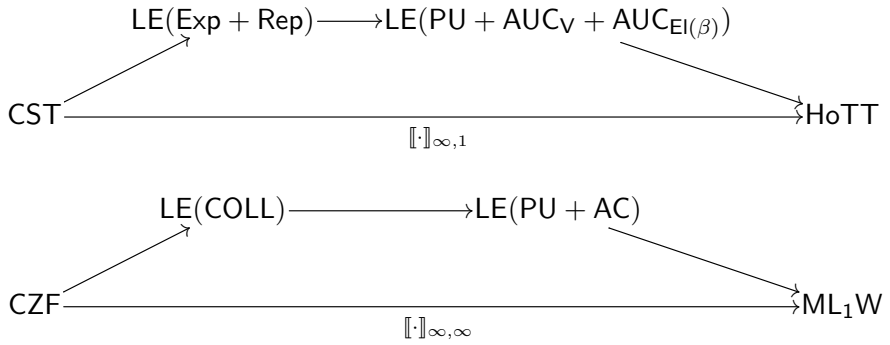
Proof. (i) Immediate taking $B := \mathbb{V}$ in the Strong Collection rule, and recalling the correspondence between $\mathbb{V}$ and $\mathsf{Sub}(\mathbb{V})$ of Lemma 5.2.2.

(ii) By construction, and using the correspondence between $\mathbb{V}$ and $\mathsf{Sub}(\mathbb{V})$, the Subset Collection rule $\mathsf{SubCOLL}$ implies the hybrid interpretation of the Subset Collection axiom. Recall that the Subset Collection axiom implies the Fullness axiom which implies Exponentiation (see Theorem A.2). Hence, the hybrid interpretation of Exponentiation follows, which is in turn equivalent to the rule Exp.

(iii) Both $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ are immediate consequences of AC where we just ignore the uniqueness condition for the existential quantifiers. □

5.4. Conclusions

We can summarise the results obtained in this chapter in the following diagram of interpretations.



We can see that $[\![\cdot]\!]_{\infty,1}$ interprets a weaker set theory into a stronger type theory when compared with Aczel's interpretation $[\![\cdot]\!]_{\infty,\infty}$. However, in $[\![\cdot]\!]_{\infty,1}$ propositions of the language of set theory are interpreted as hpropositions in homotopy type theory, an additional requirement on the structure of the interpretation itself.

Our analysis in this chapter gives a precise and clear comparison between the principles used to factor the interpretations $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$ at the level of the hybrid interpretation as well as the combinatorial interpretation. The propositions-as-types principle PU and the two axioms of unique choice $\mathsf{AUC_V}$ and $\mathsf{AUC_{El(\beta)}}$ are natural principles for a logic-enriched type theory that we have distilled from the proof of the equivalence between interpretations of Theorem 3.5.2.

Since the interpretations $[\![\cdot]\!]_H, [\![\cdot]\!]_G$ as well as the whole family of interpretations $[\![\cdot]\!]_{k,1}$ for $2 \leq k \leq \infty$ are logically equivalent, the comparative analysis of $[\![\cdot]\!]_{\infty,1}$ and $[\![\cdot]\!]_{\infty,\infty}$ provided in this chapter relates all the known interpretations of constructive set theory in homotopy type theory to Aczel's interpretation.

CHAPTER 6

Feasible ordinals

This chapter contributes to the area of 'feasible mathematics', since we are concerned with reworking classical areas of mathematics focusing on the computational content of their notions and proofs. This means we also take into account the computational complexity of the algorithms involved rather than merely reducing the notion of computational content to the one of computable function.

More specifically, we deal with a notion of feasible ordinals, and this chapter can be seen as a step in the direction of a study of feasible ordinals reminiscent of the description of the computable ordinals as the ones smaller than the Church-Kleene ordinal ω_1^{CK}. For a reference on this classical result see [Gir87, annex 5.A].

In order to study feasible ordinals one has to find a suitable setting and a suitable characterisation of polynomial time functions, which are used to formalise the notion of feasible computation. Given the wide variety of characterisations and formalisms one can expect quite diverse proposals for feasible ordinals.

One of the most important characterisations of polytime functions is the one given by Bellantoni and Cook [BC92] in terms of predicative recursion. However, many natural polytime algorithms like insertion sort do not fall into that characterisation. The key observation, dating back to Caseiro [Cas97], is that those algorithms are non-size-increasing. With the aim to represent those algorithms Hofmann developed a type theory called Linear Functional Programming Language, LFPL for short [Hof03].

LFPL is an affine linear type theory with a special resource type $\diamond$ called 'diamond' used to control the typing of size-increasing functions. The combination of linearity and the resource type restricts the typable closed terms of type $\mathsf{N} \multimap \mathsf{N}$ to be polytime non-size-increasing. We review the theory in Section 6.1.

One of the main reasons for choosing LFPL, in addition to being a technically neat and expressive type theory, is that the question of definable ordinals in the theory is posed by Hofmann himself as a comment on future directions for research, he conjectured that the upper bound for the definable ordinals of LFPL

is ω^2. However, note that in [Hof03, Section 10] the conjecture is not formulated in technical terms. Our definition of $\mathcal{O}$ in table 6.9 provides a precise formulation of the conjecture.

The main contribution of this chapter consists of Theorem 6.3.7 which disproves Hofmann's conjecture[1] showing that for any $k \in \mathbb{N}$ the ordinal ω^k is definable. We leave open the problem to finding a sharp upper bound for the definable ordinals, and the characterisation of the definable functions in the theory $\mathsf{LFPL} + \mathcal{O}$.

We remark that what we call definable ordinals in a type theory can actually be called provably well-founded ordinals as it is more customarily done in the context of ordinal analysis [Rat99]. Indeed, the definition of an ordinal in type theory is nothing more than the construction of a term which itself provides a proof of well-foundedness.

Outline. In Section 6.1 we review the syntax of the system LFPL and introduce a linear version of Kleene's $\mathcal{O}$, next in Section 6.2 we study some of the relationships between variants of the system. Section 6.3 deals with definable ordinals and presents the main result. Finally, in Section 6.4 we discuss some approaches for characterising the definable functions and for proving upper bounds for the definable ordinals of $\mathsf{LFPL} + \mathcal{O}$.

6.1. Ordinal notations in LFPL

LFPL is an affine linear type theory, meaning that it is a linear type theory with the structural rule of weakening, but not contraction.
It is given by a core theory $\multimap, \otimes$ and $\&$, and by a collection of base types namely Booleans and naturals and a collection of type operators namely lists and trees which involve the resource type $\diamond$. Variations of the system may involve different base types and type constructors.

Since the system is used to study polytime algorithms, the rules for the natural numbers correspond to the ones of binary lists. Indeed, there are algorithms that are polytime in their unary input but exponential in their binary input. So the type of natural numbers N has two successor functions, appending respectively $\mathsf{0}$ and $\mathsf{1}$ at the end of the list.

Note that we use Hofmann's LFPL as presented in [Hof03] in which there is a redundancy since the types of naturals N and the one of binary lists $\mathsf{L}(\mathsf{2})$ have the

[1] We thank Martin Hofmann for a useful discussion that helped to disprove the conjecture.

same rules.

A distinctive feature of this type theory is the presence of a resource type $\diamond$ which has no rules on its own but appears in the rules for other types. Its intended meaning is to provide 'pointers to free memory' which have to be used every time we want to type a size-increasing function. For example the successor functions for binary lists are typed as follows: $\mathsf{s}_0, \mathsf{s}_1 : \diamond \multimap \mathsf{N} \multimap \mathsf{N}$. On the other hand a function that increases the binary length of its input by 2 has to be typed as $\diamond \multimap \diamond \multimap \mathsf{N} \multimap \mathsf{N}$, and so on.

We add to the theory a type of 'linear ordinal notations' in the style of Kleene's $\mathcal{O}$. We remark that although we colloquially refer to it as a type of ordinals, we are actually dealing with ordinal notations throughout this chapter, i.e. the limit ordinals come with a specified fundamental sequence as part of their structure.
We refer to the type theory as $\mathsf{LFPL} + \mathcal{O}$. We give to $\mathcal{O}$ the same structure of the natural numbers type, with two successor operators, working on binary lists, since doing so gives immediately the natural embedding map from N to $\mathcal{O}$, for the details see the proof of Lemma 6.3.1.
The binary digits appearing in an ordinal representation give the binary encoding of natural numbers only, so for example the string $\mathsf{sup}(f)10$ represents the ordinal $\mathsf{sup}(f) + 2$.

Syntax of $\mathsf{LFPL} + \mathcal{O}$. For the syntax we follow the style of the original article where LFPL was introduced [Hof03], rather than the ones of [DLH05] or [AS02].

[DLH05] presents a version of LFPL with polymorphic quantification and the type operations + and !, but it has the shortcoming of using a notion of type dependency that is not fully formalised (see the last lines of Section 7.2). Whereas [AS02] provides a syntax optimised for a normalisation proof.

Note that whenever we write Γ, Δ to mean the union of contexts we implicitly assume that Γ and Δ are disjoint.
We conventionally assume that bracketing of function types associates on the right, i.e. $A_1 \multimap A_2 \multimap \cdots \multimap A_n := A_1 \multimap (A_2 \multimap \ldots (A_{n-1} \multimap A_n))$.

The rules of the system are as follows:

$$\frac{}{x : A \vdash x : A}\,(Ax) \qquad \frac{\Gamma \vdash \mathcal{J}}{\Gamma, y : B \vdash \mathcal{J}}\,(W) \qquad \frac{\Gamma, x : A, y : B, \Delta \vdash \mathcal{J}}{\Gamma, y : B, x : A, \Delta \vdash \mathcal{J}}\,(Ex)$$

TABLE 6.1. Structural rules

$$\frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x.t : A \multimap B}\,(\multimap^{+}) \qquad \frac{\Gamma \vdash t : A \multimap B \qquad \Delta \vdash s : A}{\Gamma, \Delta \vdash ts : B}\,(\multimap^{-})$$

$$\frac{\Gamma \vdash \lambda x.t : A \multimap B \qquad \Delta \vdash s : A}{\Gamma, \Delta \vdash (\lambda x.t)s = t[s/x] : B}\,(\multimap^{=})$$

TABLE 6.2. Rules for linear implication

Notice the difference between the definitions of the two kinds of conjunction. Additive conjunction & is defined using projections, whereas multiplicative conjunction $\otimes$ is defined via an elimination rule that follows the same pattern of elimination rules for inductive types.

Also observe the difference in the handling of contexts in the rules for the two conjunctions: in the rule $(\&^{+})$ the same context Γ appears in both the branches of the premiss and in the conclusion. On the other hand, in all rules for $\otimes$ we have two disjoint contexts Γ and Δ for each of the two branches of the premiss, and their disjoint union Γ, Δ in the conclusion.

The effect of these differences is that one can access both components of a tensor product, whereas one can access only one of an additive pair.

$$\frac{\Gamma \vdash t : A \qquad \Delta \vdash s : B}{\Gamma, \Delta \vdash t \otimes s : A \otimes B}\,(\otimes^{+}) \qquad \frac{\Gamma \vdash t : A \otimes B \qquad \Delta, x : A, y : B \vdash r : C}{\Gamma, \Delta \vdash \mathsf{let} t \;\mathsf{be}\; x \otimes y \;\mathsf{in}\; r : C}\,(\otimes^{-})$$

$$\frac{\Gamma \vdash t \otimes s : A \otimes B \qquad \Delta, x : A, y : B \vdash r : C}{\Gamma, \Delta \vdash (\mathsf{let}\; t \otimes s \;\mathsf{be}\; x \otimes y \;\mathsf{in}\; r) = r[t, s/x, y] : C}\,(\otimes^{=})$$

TABLE 6.3. Rules for multiplicative conjunction

$$\frac{\Gamma \vdash t_1 : A_1 \qquad \Gamma \vdash t_2 : A_2}{\Gamma \vdash \langle t_1, t_2 \rangle : A_1 \& A_2}\,(\&^+)$$

$$\frac{\Gamma \vdash t : A_1 \& A_2}{\Gamma \vdash t.1 : A_1}\,(\&_1^-) \qquad \frac{\Gamma \vdash t : A_1 \& A_2}{\Gamma \vdash t.2 : A_2}\,(\&_2^-)$$

$$\frac{\Gamma \vdash \langle t_1, t_2 \rangle : A_1 \& A_2}{\Gamma \vdash \langle t_1, t_2 \rangle.1 = t_1 : A_1}\,(\&_1^=) \qquad \frac{\Gamma \vdash \langle t_1, t_2 \rangle : A_1 \& A_2}{\Gamma \vdash \langle t_1, t_2 \rangle.2 = t_2 : A_2}\,(\&_2^=)$$

TABLE 6.4. Rules for additive conjunction

The rules for Booleans are straightforward:

$$\frac{}{\mathsf{ff} : 2}\,(2_0^+) \qquad \frac{}{\mathsf{tt} : 2}\,(2_1^+) \qquad \frac{}{\mathsf{if} : 2 \multimap (A \& A) \multimap A}\,(2^-)$$

$$\frac{}{\mathsf{if}(\mathsf{ff}, e) = e.1 : A}\,(2_0^=) \qquad \frac{}{\mathsf{if}(\mathsf{tt}, e) = e.2 : A}\,(2_1^=)$$

TABLE 6.5. Rules for Booleans

Now we move to the presentation of the rules for lists, trees, natural numbers and ordinals. They may appear complicated at first sight, but the elimination and computation rules follow the structure of inductive types stating that the type in consideration (lists, trees, naturals, ordinals) is the initial type with appropriate structure.
Moreover, notice that for a given type the premisses of the elimination and computation rules are the same.

Observe that these elimination and computation rules need to have closed terms in the premiss, i.e. the terms appearing in the premiss are typed in the empty context. This is required in order to prevent the duplication of resources (i.e. terms of type $\diamond$) in the premiss of an elimination rule, which would lead beyond the class of polytime functions.

$$\frac{}{\mathsf{nil} : L(X)}\,(L(X)_0^+) \qquad \frac{}{\mathsf{cons} : \diamond \multimap X \multimap L(X) \multimap L(X)}\,(L(X)_1^+)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap X \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{L(X)}(a,h) : L(X) \multimap A}\,(L(X)^-)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap X \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{L(X)}(a,h)(\mathsf{nil}) = a : A}\,(L(X)_0^=)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap X \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{L(X)}(a,h)(\mathsf{cons}(d,x,l)) = h(d,x,\mathsf{it}^A_{L(X)}(l)) : A}\,(L(X)_1^=)$$

TABLE 6.6. Rules for lists

$$\frac{}{\mathsf{leaf} : X \multimap T(X)}\,(T(X)_0^+)$$

$$\frac{}{\mathsf{node} : \diamond \multimap X \multimap T(X) \multimap T(X) \multimap T(X)}\,(T(X)_1^+)$$

$$\frac{\emptyset \vdash g : X \multimap A \qquad \emptyset \vdash k : \diamond \multimap X \multimap A \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{T(X)}(g,k) : T(X) \multimap A}\,(T(X)^-)$$

$$\frac{\emptyset \vdash g : X \multimap A \qquad \emptyset \vdash k : \diamond \multimap X \multimap A \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{T(X)}(g,k)(\mathsf{leaf}(x)) = g(x) : A}\,(T(X)_0^=)$$

$$\frac{\emptyset \vdash g : X \multimap A \qquad \emptyset \vdash k : \diamond \multimap X \multimap A \multimap A \multimap A}{\emptyset \vdash \mathsf{it}^A_{T(X)}(g,k)(\mathsf{node}(d,x,l,r)) = k(d,x,\mathsf{it}^A_{T(X)}(g,k)(l),\mathsf{it}^A_{T(X)}(g,k)(r)) : A}\,(T(X)_1^=)$$

TABLE 6.7. Rules for binary labelled trees

The rules for natural numbers correspond to the ones of binary lists $L(2)$ where instead of one function $\mathsf{cons} : \diamond \multimap 2 \multimap L(2) \multimap L(2)$ we have two functions $\mathsf{s}_0, \mathsf{s}_1 : \diamond \multimap \mathsf{N} \multimap \mathsf{N}$ corresponding to $\mathsf{cons}(\mathsf{tt})$ and $\mathsf{cons}(\mathsf{ff})$. And similarly for the elimination and computation rules.

$$\frac{}{0 : \mathsf{N}}\,(\mathsf{N}_0^+) \qquad \frac{}{\mathsf{s}_0, \mathsf{s}_1 : \diamond \multimap \mathsf{N} \multimap \mathsf{N}}\,(\mathsf{N}_1^+)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^A(a, h_0, h_1) : \mathsf{N} \multimap A}\,(\mathsf{N}^-)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^A(a, h_0, h_1)(0) = a : A}\,(\mathsf{N}_0^=)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^A(a, h_0, h_1)(s_i(d, n)) = h_i(d, \mathsf{it}_{\mathsf{N}}^A(n)) : A}\,(\mathsf{N}_1^=)$$

TABLE 6.8. Rules for the type of natural numbers

The rules for $\mathcal{O}$ are the same as the ones for N except for one extra piece of structure, namely the supremum operation sup.

$$\frac{}{0 : \mathcal{O}}\,(\mathcal{O}_0^+) \qquad \frac{}{\mathsf{S}_0, \mathsf{S}_1 : \diamond \multimap \mathcal{O} \multimap \mathcal{O}}\,(\mathcal{O}_1^+) \qquad \frac{}{\mathsf{sup} : \diamond \multimap (\mathsf{N} \multimap \mathcal{O}) \multimap \mathcal{O}}\,(\mathcal{O}_2^+)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}_{\mathcal{O}}^A(a, h_0, h_1, \sigma) : \mathcal{O} \multimap A}\,(\mathcal{O}^-)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}_{\mathcal{O}}^A(a, h_0, h_1, \sigma)(0) = a : A}\,(\mathcal{O}_0^=)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}_{\mathcal{O}}^A(a, h_0, h_1, \sigma)(\mathsf{S_i}(d, \alpha)) = h_i(d, \mathsf{it}_{\mathcal{O}}^A(\alpha)) : A}\,(\mathcal{O}_1^=)$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}_{\mathcal{O}}^A(a, h_0, h_1, \sigma)(\mathsf{sup}(d, x)) = \sigma(d, \mathsf{it}_{\mathcal{O}}^A \circ x) : A}\,(\mathcal{O}_2^=)$$

TABLE 6.9. Rules for the type of ordinal notations

6.2. Comparing systems

In this section we study the relation between three systems: Gödel's system T augmented with a type of ordinal notations, the type theory $\mathsf{LFPL} + \mathcal{O}$ and its variant $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$. In the latter theory the types of natural numbers and ordinals are defined with one successor instead of two, which corresponds to a unary representation of natural numbers instead of a binary one.

We show that, as expected, adding the structural rule of contraction to $\mathsf{LFPL}+\mathcal{O}$ and $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$ makes these systems bi-interpretable with the system $\mathsf{T} + \mathcal{O}$. However, the direct relationship between the two linear type theories is more complex: $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$ is trivially interpretable in $\mathsf{LFPL} + \mathcal{O}$, but it doesn't seem the case for the converse. We leave this last question open.

Throughout this section we use the notation $L(2)$ for the type of natural numbers of the system $\mathsf{LFPL} + \mathcal{O}$ as defined in table 6.8, and we use $L(1)$ for the type of natural numbers of $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$.

We abuse notation by denoting with the same symbol $\mathcal{O}$ the type of ordinal notation in the three different systems. The rules for the type of ordinal notations of $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$ are as follows:

$$\frac{}{0 : \mathcal{O}} \qquad \frac{}{\mathsf{S} : \diamond \multimap \mathcal{O} \multimap \mathcal{O}} \qquad \frac{}{\mathsf{sup} : \diamond \multimap (\mathsf{N} \multimap \mathcal{O}) \multimap \mathcal{O}}$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}^A_{\mathcal{O}}(a, h, \sigma) : \mathcal{O} \multimap A}$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(0) = a : A}$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(\mathsf{S}(d, \alpha)) = h(d, \mathsf{it}^A_{\mathcal{O}}(\alpha)) : A}$$

$$\frac{\emptyset \vdash a : A \qquad \emptyset \vdash h : \diamond \multimap A \multimap A \qquad \emptyset \vdash \sigma : \diamond \multimap (\mathsf{N} \multimap A) \multimap A}{\emptyset \vdash \mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(\mathsf{sup}(d, x)) = \sigma(d, \mathsf{it}^A_{\mathcal{O}} \circ x) : A}$$

TABLE 6.10. Rules for the type of ordinal notations for $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$

We adapt to our context Feferman's definition of proof-theoretical reducibility. See [Fef88, Definition 2.1] for the details.

Definition 6.2.1. We say that a type theory T_1 is *interpretable* in a type theory T_2 if and only if for each type and term of T_1 we can associate a type and a term of T_2 such that the rules of T_1 are admissible in T_2 after having interpreted each type and term.

For Gödel's system T we consider a type of ordinal notations with the same rules as the ones presented in table 6.9, but where the linear arrow $\multimap$ has been replaced by the standard intuitionistic one $\to$ and the resource type $\diamond$ has been removed.

$$\frac{}{0 : \mathcal{O}} \qquad \frac{}{\mathsf{S} : \mathcal{O} \to \mathcal{O}} \qquad \frac{}{\mathsf{sup} : (\mathsf{N} \to \mathcal{O}) \to \mathcal{O}}$$

$$\frac{a : A \qquad h : A \to A \qquad \sigma : (\mathsf{N} \to A) \to A}{\mathsf{it}^A_{\mathcal{O}}(a, h, \sigma) : \mathcal{O} \to A}$$

$$\frac{a : A \qquad h : A \to A \qquad \sigma : (\mathsf{N} \to A) \to A}{\mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(0) = a : A}$$

$$\frac{a : A \qquad h : A \to A \qquad \sigma : (\mathsf{N} \to A) \to A}{\mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(\mathsf{S}(\alpha)) = h(\mathsf{it}^A_{\mathcal{O}}(\alpha)) : A}$$

$$\frac{a : A \qquad h : A \to A \qquad \sigma : (\mathsf{N} \to A) \to A}{\mathsf{it}^A_{\mathcal{O}}(a, h, \sigma)(\mathsf{sup}(x)) = \sigma(\mathsf{it}^A_{\mathcal{O}} \circ x) : A}$$

TABLE 6.11. Rules for the type of ordinal notations for $\mathsf{T} + \mathcal{O}$

Let CON stand for the contraction rule.

$$\frac{\Gamma, x : A, x : A \vdash \mathcal{J}}{\Gamma, x : A \vdash \mathcal{J}}$$

TABLE 6.12. Contraction rule

Before the main theorem that compares the systems we represent ordinal addition in the theory $T + \mathcal{O}$.

Note that the set-theoretic semantics $[\![\cdot]\!]$ of $\mathsf{LFPL}+\mathcal{O}$ defined in appendix C also gives a semantics for the systems $\mathsf{T}+\mathcal{O}$, $\mathsf{LFPL}+\mathcal{O}+\mathsf{CON}$ and $(\mathsf{LFPL}+\mathcal{O})^{\mathsf{succ}}+\mathsf{CON}$.

Lemma 6.2.2. *In the theory $\mathsf{T} + \mathcal{O}$ there is a term $+ : \mathcal{O} \to \mathcal{O} \to \mathcal{O}$ representing ordinal addition (for which we use the infix notation), i.e. such that given $\alpha, \beta : \mathcal{O}$ we have $[\![\alpha + \beta]\!] = [\![\beta]\!] + [\![\alpha]\!]$.*

Proof. Notice that we can lift the structure of successor and supremum from $\mathcal{O}$ to the function type $X := \mathcal{O} \to \mathcal{O}$ by defining it point-wise. So that we have $\lambda x.x : X$, and $\mathsf{S}' := \lambda f.\mathsf{S} \circ f : X \to X$ and $\mathsf{sup}' := \lambda u.\lambda x.\mathsf{sup}(\lambda n.u(n)(x)) : (\mathsf{N} \to X) \to X$.

Ordinal sum is definable by the following application of the elimination rule:

$$\frac{\lambda x.x : X \qquad \mathsf{S}' : X \to X \qquad \mathsf{sup}' : (\mathsf{N} \to X) \to X}{+ : \mathcal{O} \to (\mathcal{O} \to \mathcal{O})}$$

It is straightforward to check that the computation rules give the recursive definition of ordinal sum. □

Theorem 6.2.3. *The systems $\mathsf{LFPL} + \mathcal{O} + \mathsf{CON}$ and $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}} + \mathsf{CON}$ are both bi-interpretable with $\mathsf{T} + \mathcal{O}$.*

Proof. Notice that we can interpret the resource type $\diamond$ as the unit type $\mathsf{1}$. The contraction rule allows to ignore the restriction in the elimination and computation rules that the contexts in the premisses has to be empty.

So our task is simply to check that Gödel's T with $L(\mathsf{2})$ and $\mathcal{O}$ defined with two successors is bi-interpretable with T with $L(\mathsf{1})$ and $\mathcal{O}$ with one successor.

We can interpret the rules for unary lists $L(\mathsf{1})$ if we are given the ones for binary lists $L(\mathsf{2})$. Indeed, it is enough to encode a unary list representing n as the binary list given by the concatenation of n zeros. The case of $\mathcal{O}$ is analogous.

Conversely, since the semantics of the two binary successors is $[\![\mathsf{s}_0(x)]\!] = 2[\![x]\!]$ and $[\![\mathsf{s}_1(x)]\!] = 2[\![x]\!] + 1$, is is enough to define the functions $\mathsf{s}_0, \mathsf{s}_1 : L(\mathsf{1}) \to L(\mathsf{1})$.

For the elimination and computation rules, observe that the semantics of $\mathsf{it}_{L(\mathsf{2})}$ is:

$$[\![\mathsf{it}^A_{L(\mathsf{2})}(a, h_0, h_1)]\!](n) = \begin{cases} [\![a]\!] & \text{if } n = 0 \\ [\![h_0]\!]([\![\mathsf{it}^A_{L(\mathsf{2})}(a, h_0, h_1)]\!](x)) & \text{if } n = 2x \\ [\![h_1]\!]([\![\mathsf{it}^A_{L(\mathsf{2})}(a, h_0, h_1)]\!](x)) & \text{if } n = 2x + 1 \end{cases}$$

So that we need to type a function $f : L(\mathsf{1}) \to A$ such that $[\![f]\!] = [\![\mathsf{it}^A_{L(2)}(a, h_0, h_1)]\!]$ and f satisfies the computation rules of $\mathsf{it}^A_{L(2)}(a, h_0, h_1)$. Such a function f can be represented in Gödel's system T using double recursion, which in turn can be defined using the predecessor function and case distinctions.

The treatment of $\mathcal{O}$ reduces to the case of natural numbers after few simple definitions, as we see now. In Lemma 6.2.2, we have established that ordinal addition is representable in the system $\mathsf{T} + \mathcal{O}$.

We can then define two functions $L : \mathcal{O} \to \mathcal{O}$ and $N : \mathcal{O} \to L(\mathsf{1})$ that given α respectively return the greatest limit ordinal smaller or equal than α and the difference between the two, which is a natural number. These functions are easily defined by recursion as: $L(0) = 0$, $L(\mathsf{S}(\alpha)) = L(\alpha)$ and $L(\mathsf{sup}(f)) = \mathsf{sup}(f)$. The definition of the other function is as follows: $N(0) = 0$, $N(\mathsf{S}(\alpha)) = \mathsf{s}(N(\alpha))$ and $N(\mathsf{sup}(f)) = 0$.

Using the functions L and N we can extend the definitions of predecessor, case distinction and the double recursion to $\mathcal{O}$ in order to interpret the elimination rule with two successors in the theory where $\mathcal{O}$ is defined with one successor. For example the binary predecessor of an ordinal can be defined as $P_{\mathcal{O}}(\alpha) := L(\alpha) + P(N(\alpha))$ where P is the usual binary predecessor for natural numbers. □

As we can see from the first lines of the proof of Theorem 6.2.3, $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$ is trivially interpretable in $\mathsf{LFPL} + \mathcal{O}$. The approach used for the converse interpretation in the proof of Theorem 6.2.3 does not work in this case. Indeed, we would need to type the binary successor function $\mathsf{S}_0 : \diamond \multimap L(\mathsf{1}) \multimap L(\mathsf{1})$, but one term $d : \diamond$ is not enough since $lg(2x) = 2 \cdot lg(x)$ where lg is the unary length, opposed to the situation with binary lengths $|2x| = |x| + 1$.

At present we cannot prove that there is no interpretation of $\mathsf{LFPL} + \mathcal{O}$ into $(\mathsf{LFPL} + \mathcal{O})^{\mathsf{succ}}$ but it seems likely that linearity together with the resource type prevents any such interpretation if the type of natural numbers in one system is interpreted as the type of natural numbers in the other.

6.3. Definable ordinals

In Theorem 6.3.7 we show that for any ordinal $\alpha < \omega^\omega$ there is a term t_α of $\mathcal{O}$ which has free variables (if any) of type $\diamond$, such that $[\![t_\alpha]\!] = \alpha$. Thus we disprove Hofmann's conjecture.

The set-theoretic semantics $[\![\cdot]\!]$ is defined in appendix C. Our methodology follows in the context of linear type theory the one of [CHS] for Gödel's system T.

We present a sequence of lemmas that show how to represent the ordinals: k, ω, $\omega \cdot k$, ω^2, ω^k, for $k \in \mathbb{N}$ a natural number given externally. Along the way we construct the operation of ordinal addition for terms of type $\mathcal{O}$.

The question if the upper bound ω^ω is sharp is left open. Theorem 6.3.8 gives a conditional result for the definability of ω^ω assuming the typability of the function that performs the ordinal multiplication of an ordinal with a natural number.

For the rest of this section we work in $\mathsf{LFPL} + \mathcal{O}$. We use the notation A^k for the tensor product of A iterated k times $A \otimes \cdots \otimes A$.

Lemma 6.3.1.

(i) *For any natural number $n \in \mathbb{N}$ with binary expansion $i_1 \dots i_k$ there is a term of the form $t_n := \lambda d_1, \dots, d_k.\mathsf{S}_{i_1}(d_1, \mathsf{S}_{i_2}(d_2, \dots \mathsf{S}_{i_k}(d_k, 0))) : \diamond^k \multimap \mathcal{O}$ that represents that number, i.e. $[\![t_n]\!] = n$.*

(ii) *There is a term t_ω representing ω, i.e. such that $[\![t_\omega]\!] = \omega$. It is given by $t_\omega = \lambda d.\mathsf{sup}(d, \mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \mathsf{S}_0, \mathsf{S}_1)) : \diamond \multimap \mathcal{O}$.*

Proof. The claim for natural numbers is obvious.

We represent ω as the supremum of the function $\mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \mathsf{S}_0, \mathsf{S}_1) : \mathsf{N} \multimap \mathcal{O}$ that sends each natural number to itself, just seen as an ordinal. We obtain this function applying the elimination rule for N:

$$\frac{\emptyset \vdash 0 : \mathcal{O} \qquad \emptyset \vdash \mathsf{S}_0, \mathsf{S}_1 : \diamond \multimap \mathcal{O} \multimap \mathcal{O}}{\emptyset \vdash \mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \mathsf{S}_0, \mathsf{S}_1) : \mathsf{N} \multimap \mathcal{O}}$$

So that taking the supremum and λ-abstracting on d gives the desired term $t_\omega := \lambda d.\mathsf{sup}(d, \mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \mathsf{S}_0, \mathsf{S}_1)) : \diamond \multimap \mathcal{O}$. Therefore we have:

$$\begin{aligned} [\![t_\omega]\!] &= [\![\mathsf{sup}(d, \mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \mathsf{S}_0, \mathsf{S}_1))]\!] \\ &= sup_n(f), \end{aligned}$$

where $f : \mathbb{N} \to [\![\mathcal{O}]\!]$ is given by $n \mapsto n$. □

Lemma 6.3.2. *For all $k \in \mathbb{N}$, given externally, we can construct a term $t_{\omega \cdot k}$ of type $\diamond^k \multimap \mathcal{O}$ such that $[\![t_{\omega \cdot k}]\!] = \omega \cdot k$.*

Proof. By induction on k in the meta-theory. We have just shown the case $k = 1$ in Lemma 6.3.1. For $k > 1$, suppose we have a term $t_{\omega \cdot k} : \diamond^k \multimap \mathcal{O}$ with $[\![t_{\omega \cdot k}]\!] = \omega \cdot k$, we want to construct $t_{\omega \cdot (k+1)}$.

Notice that we have a function $\hat{\mathsf{S}}_0 : \diamond \multimap (\diamond^k \multimap \mathcal{O}) \multimap (\diamond^k \multimap \mathcal{O})$ induced by S_0. For $d : \diamond$, $\alpha : \diamond^k \multimap \mathcal{O}$ and $D : \diamond^k$ we define $\hat{\mathsf{S}}_0 := \lambda d.\lambda\alpha.\lambda D.\mathsf{S}_0(d, \alpha(D))$. Notice that we can discard d since we have allowed the rule of weakening in the theory. Similarly for $\hat{\mathsf{S}}_1$.

We construct a function by induction as in Lemma 6.3.1 but rather than starting from 0 we start from $t_{\omega\cdot k}$.

$$\frac{\emptyset \vdash t_{\omega\cdot k} : \diamond^k \multimap \mathcal{O} \qquad \emptyset \vdash \hat{\mathsf{S}}_0, \hat{\mathsf{S}}_1 : \diamond \multimap (\diamond^k \multimap \mathcal{O}) \multimap (\diamond^k \multimap \mathcal{O})}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^{\diamond^k \multimap \mathcal{O}}(t_{\omega\cdot k}, \hat{\mathsf{S}}_0, \hat{\mathsf{S}}_1) : \mathsf{N} \multimap (\diamond^k \multimap \mathcal{O})}$$

Then we rearrange the types N and $\diamond^k$ of the function just defined by recursion $F := \lambda d_1, \dots d_k.\lambda n.\mathsf{it}_{\mathsf{N}}^{\diamond^k \multimap \mathcal{O}}(t_{\omega\cdot k}, \hat{\mathsf{S}}_0, \hat{\mathsf{S}}_1)(n, d_1, \dots, d_k) : \diamond^k \multimap (\mathsf{N} \multimap \mathcal{O})$. And we take the supremum obtaining the desired term:

$$t_{\omega\cdot(k+1)} := \lambda d_1, \dots, d_{k+1}.\mathsf{sup}(d_{k+1}, F(d_1, \dots, d_k)).$$

The semantics of the term is $[\![t_{\omega\cdot(k+1)}]\!] = sup_n(\omega \cdot k + n) = \omega \cdot (k+1)$. □

Before we move on to the construction of the term representing ω^2 we first construct the function representing ordinal addition of an ordinal and a natural number.

Lemma 6.3.3. *There is a function $+ : \mathsf{N} \multimap \mathcal{O} \multimap \mathcal{O}$ representing ordinal addition of an ordinal and a natural number (for which we use the infix notation), i.e. given $n : \mathsf{N}$ and $\alpha : \mathcal{O}$ it satisfies $[\![n + \alpha]\!] = [\![\alpha]\!] + [\![n]\!]$.*

Proof. For $i = 0, 1$ we define the functions that are used in the iteration:

$$h_i := \lambda d\lambda F\lambda x.\mathsf{S}_{\mathsf{i}}(d, F(x)) : \diamond \multimap (\mathcal{O} \multimap \mathcal{O}) \multimap (\mathcal{O} \multimap \mathcal{O}).$$

We obtain $+$ as the term in the conclusion of this application of the elimination rule for N:

$$\frac{\emptyset \vdash \lambda x.x : \mathcal{O} \multimap \mathcal{O} \qquad \emptyset \vdash h_0, h_1 : \diamond \multimap (\mathcal{O} \multimap \mathcal{O}) \multimap (\mathcal{O} \multimap \mathcal{O})}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^{\mathcal{O} \multimap \mathcal{O}}(\lambda x.x, h_0, h_1) : \mathsf{N} \multimap (\mathcal{O} \multimap \mathcal{O})}$$

Given $n : \mathsf{N}$ and $\alpha : \mathcal{O}$ it is straightforward to prove by induction on $[\![n]\!] \in \mathbb{N}$ that the term $+$ represents ordinal addition, i.e. $[\![n + \alpha]\!] = [\![\alpha]\!] + [\![n]\!]$. □

Lemma 6.3.4. *There is a term $t_{\omega^2} : \diamond \multimap \mathcal{O}$ such that $[\![t_{\omega^2}]\!] = \omega^2$.*

Proof. The idea is to define the fundamental sequence $\{\omega \cdot |n|\}_n$ for ω^2, where $|n|$ is the binary length of n. We begin by defining $H : \mathsf{N} \multimap \mathcal{O}$ by recursion on N as follows:

$$\frac{\emptyset \vdash 0 : \mathcal{O} \qquad \emptyset \vdash \lambda d\lambda x.\mathsf{sup}(d, \lambda m.x + m) : \diamond \multimap \mathcal{O} \multimap \mathcal{O}}{\emptyset \vdash H := \mathsf{it}_{\mathsf{N}}^{\mathcal{O}}(0, \lambda d.\lambda x.\mathsf{sup}(d, \lambda m.x + m)) : \mathsf{N} \multimap \mathcal{O}}$$

By construction we have that $H(0) = 0$ and $H(\mathsf{S}_{\mathsf{i}}(d, n)) = \mathsf{sup}(d, \lambda m.H(n)+m)$, therefore its interpretation is $[\![H(\mathsf{S}_{\mathsf{i}}(d, n))]\!] = sup_m([\![H(n)]\!] + m) = [\![H(n)]\!] + \omega$. A straightforward induction on $k \in \mathbb{N}$ then gives $[\![H]\!](k) = \omega \cdot |k|$.

Finally, $t_{\omega^2} = \lambda d.\mathsf{sup}(d, H) : \diamond \multimap \mathcal{O}$, which is interpreted as $[\![t_{\omega^2}]\!] = sup_k([\![H]\!](k)) = sup_k(\omega \cdot |k|) = \omega^2$. □

Now we represent the operation of ordinal addition between two arbitrary ordinals in $\mathcal{O}$. We shall abuse notation by denoting both terms by $+$.

Lemma 6.3.5. *There is a term $+ : \mathcal{O} \multimap \mathcal{O} \multimap \mathcal{O}$ representing ordinal addition (for which we use the infix notation), i.e. such that given $\alpha, \beta : \mathcal{O}$ we have $[\![\alpha+\beta]\!] = [\![\beta]\!] + [\![\alpha]\!]$.*

Proof. Let $X = \mathcal{O} \multimap \mathcal{O}$. We define $+$ via one application of the elimination rule for $\mathcal{O}$.

Given $i = 0, 1$, consider the following terms:

$$\mathsf{S}'_{\mathsf{i}} := \lambda d^{\diamond} \lambda F^{X} \lambda x^{\mathcal{O}}.\mathsf{S}_{\mathsf{i}}(d, F(x)) : \diamond \multimap X \multimap X,$$

and

$$\mathsf{sup}' := \lambda d^{\diamond} \lambda u^{\mathsf{N} \multimap X} \lambda x^{\mathcal{O}}.\mathsf{sup}(d, \lambda n.u(n)(x)) : \diamond \multimap (\mathsf{N} \multimap X) \multimap X.$$

Then we can apply the elimination rule for $\mathcal{O}$ over the structure $(X, \mathsf{S}'_{\mathsf{i}}, \mathsf{sup}')$:

$$\frac{\emptyset \vdash \lambda x.x : X \qquad \emptyset \vdash \mathsf{S}'_{\mathsf{i}} : \diamond \multimap X \multimap X \qquad \emptyset \vdash \mathsf{sup}' : \diamond \multimap (\mathsf{N} \multimap X) \multimap X}{\emptyset \vdash \mathsf{it}^{X}_{\mathcal{O}}(\lambda x.x, \mathsf{S}'_{\mathsf{i}}, \mathsf{sup}') : \mathcal{O} \multimap X}$$

The computation rules give $0 + \alpha = \alpha$, and $\mathsf{S}_{\mathsf{i}}(d, \beta) + \alpha = \mathsf{S}_{\mathsf{i}}(d, \beta + \alpha)$, and $\mathsf{sup}(d, u) + \alpha = \mathsf{sup}(d, \lambda n.u(n) + \alpha)$, which correspond under $[\![\cdot]\!]$ to the definition of ordinal sum by transfinite recursion, so the claim follows. □

Finally we prove that all ordinals ω^k for $k \in \mathbb{N}$ are definable.

Lemma 6.3.6. *For any integer $k \geq 2$ there is a function $H_k : \mathsf{N} \multimap \mathcal{O}$ and a term $t_{\omega^k} = \lambda d.\mathsf{sup}(d, H_k) : \diamond \multimap \mathcal{O}$ such that $[\![H_k]\!](n) = \omega^{k-1} \cdot |n|$, hence $[\![t_{\omega^k}]\!] = \omega^k$.*

Proof. We prove the claim by induction on k (externally). The base case $k = 2$ has been proven in Lemma 6.3.4. Inductive step: suppose we have already constructed t_{ω^k} by means of $H_k(n)$, we shall construct a function H_{k+1} such that $[\![H_{k+1}]\!](n) = \omega^k \cdot |n|$ then $t_{\omega^{k+1}}$ is its supremum.

So we appropriately apply the elimination rule for N:

$$\frac{\emptyset \vdash 0 : \mathsf{N} \qquad \emptyset \vdash \lambda d.\lambda x.\mathsf{sup}(d, \lambda m.x + H_k(m)) : \diamond \multimap \mathcal{O} \multimap \mathcal{O}}{\emptyset \vdash \mathsf{it}^{\mathcal{O}}_{\mathsf{N}}(0, \lambda d.\lambda x.\mathsf{sup}(d, \lambda m.x + H_k(m))) : \mathsf{N} \multimap \mathcal{O}}$$

then we can define our term $t_{\omega^{k+1}} := \lambda d.\mathsf{sup}(d, H_{k+1})$.

The computation rules for H_{k+1} give $H_{k+1}(0) = 0$ and for the successors $H_{k+1}(\mathsf{S_i}(d,n)) = \mathsf{sup}(d, \lambda m.H_{k+1}(n) + H_k(m))$. Therefore its interpretation satisfies:

$$[\![H_{k+1}(\mathsf{S_i}(d,n))]\!] = sup_m([\![H_{k+1}(n)]\!] + \omega^{k-1} \cdot |m|) = [\![H_{k+1}(n)]\!] + \omega^k$$

therefore $[\![H_{k+1}]\!](m) = \omega^k \cdot |m|$, which in turn gives that the interpretation of $t_{\omega^{k+1}}$ is:

$$[\![t_{\omega^{k+1}}]\!] = sup_m([\![H_{k+1}]\!](m)) = sup_m(\omega^k \cdot |m|) = \omega^{k+1}$$

□

The following theorem summarises the lemmas of this section and disproves Hofmann's conjecture that the supremum of the definable ordinals of $\mathsf{LFPL} + \mathcal{O}$ is ω^2.

Theorem 6.3.7. *For every ordinal $\alpha < \omega^\omega$ there is a natural number $n \in \mathbb{N}$ and a term $t_\alpha : \diamond^n \multimap \mathcal{O}$ such that $[\![t_\alpha]\!] = \alpha$.*

We are left with the open question if the lower bound ω^ω is sharp.

Let us consider possible definitions of ω^ω in $\mathsf{LFPL} + \mathcal{O}$. It can be seen easily that a general definition of exponentiation of ordinals, e.g. as it is done in [Sim00, Definition 9.22] violates linearity. However, different constructions may a priori work. For example one can try to define a function $K : \mathsf{N} \multimap \mathcal{O}$ such that $[\![K]\!](n) = \omega^{|n|}$. The best we can offer in this direction is the following conditional result.

Theorem 6.3.8. *Let (a_n) be a diverging sequence of natural numbers. If there is a typable function $\mu : \mathsf{N} \multimap \mathcal{O} \multimap \mathcal{O}$ such that given $n \in \mathbb{N}$ and $\alpha \in [\![\mathcal{O}]\!]$ we have $[\![\mu]\!](n, \alpha) = \alpha \cdot a_n$. Then there is a term $t_{\omega^\omega} : \diamond^2 \multimap \mathcal{O}$ representing ω^ω.*

Proof. We want to represent the function $\mathbb{N} \to [\![\mathcal{O}]\!]$ given by $n \mapsto \omega^{|n|}$, the idea is to iterate the operation $sup_m(\alpha \cdot a_m)$ on the length of n starting from the base 1 which is represented as $\lambda d.\mathsf{S_1}(d,0) : \diamond \multimap \mathcal{O}$.
Consider the following term:

$$\hat{\mathsf{S}} := \lambda d_0^\diamond \lambda \alpha^{\diamond \multimap \mathcal{O}} \lambda d_1^\diamond.\mathsf{sup}(d_0, \lambda n.\mu(\alpha(d_1), n)).$$

We can apply the elimination rule for N over the structure $\diamond \multimap \mathcal{O}$, with successor operator $\hat{\mathsf{S}}$:

$$\frac{\emptyset \vdash \lambda d.\mathsf{S_1}(d,0) : \diamond \multimap \mathcal{O} \qquad \emptyset \vdash \hat{\mathsf{S}} : \diamond \multimap (\diamond \multimap \mathcal{O}) \multimap (\diamond \multimap \mathcal{O})}{\emptyset \vdash \mathsf{it}_{\mathsf{N}}^{\diamond \multimap \mathcal{O}}(\lambda d.\mathsf{S_1}(d,0), \hat{\mathsf{S}}) : \mathsf{N} \multimap (\diamond \multimap \mathcal{O})}$$

Then we rearrange the types N and $\diamond$ of the function just defined by recursion to get the function K:

$$K := \lambda d_0.\lambda n.\mathsf{it}^{\diamond \multimap \mathcal{O}}_{\mathsf{N}}(\lambda d_1.\mathsf{S}_1(d_1,0), \hat{\mathsf{S}})(n,d) : \diamond \multimap (\mathsf{N} \multimap \mathcal{O}).$$

Then the desired term is constructed by taking the supremum and then a λ-abstraction on the tokens utilised, i.e. $t_{\omega^\omega} := \lambda d_1, d_2.\mathsf{sup}(d_2, K(d_1)) : \diamond^2 \multimap \mathcal{O}$.

Now we prove that t_{ω^ω} represents ω^ω. By construction $[\![t_{\omega^\omega}]\!] = sup_n([\![K]\!](n))$. The only thing to prove is that $[\![K]\!](n) = \omega^{|n|}$.

We claim that $[\![K]\!](n) = \omega^{|n|}$, which we prove by induction on the binary length of n. The base case is $[\![K]\!](0) = [\![\lambda d.\mathsf{S}_1(d,0)]\!] = 1$. Inductive step: we write ni for the number given by writing n in binary notation and adding $i \in \{0,1\}$ at the right of the binary representation. If $[\![K]\!](n) = \omega^{|n|}$ then:

$$\begin{aligned}
[\![K]\!](ni) &= sup_m([\![\mu]\!]([\![K]\!](n), a_m)) && \text{by definition of } K \\
&= sup_m(\omega^{|n|} \cdot a_m) && \text{by inductive hypothesis} \\
&= \omega^{|n|+1}.
\end{aligned}$$

□

To investigate if the function $\mu : \mathsf{N} \multimap \mathcal{O} \multimap \mathcal{O}$ of Theorem 6.3.8 is representable, a strategy may be to generalise to the context of ordinals what is done to represent the multiplication function on naturals. However, the usual recursive definition of multiplication cannot be reproduced since it involves the duplication of a variable in the recursive step. A general result of [Hof02] guarantees that every polytime non-size-increasing function is representable in the system, so is the multiplication function.

We do not pursue this question here, leaving it for future investigations.

6.4. Conclusions

We conclude this chapter by discussing the problems left open and some approaches for solving them. We leave open the problem of characterising the definable function in $\mathsf{LFPL} + \mathcal{O}$, for which we consider the possibility to try to generalise the results of [Hof03] and [AS02] on the definable functions of LFPL.

The other main question left open is to prove sharp upper bounds for the definable ordinals. In this section we discuss approaches based on:

(i) formalising a realizability model in a weak theory of arithmetic;
(ii) using a hierarchy of functions like Hardy's hierarchy;
(iii) adapting the ordinal analysis of Gödel's T to the case of $\mathsf{LFPL} + \mathcal{O}$.

For all of these strategies we highlight the issues we encountered when trying to implement them.

At the moment we cannot settle the question of whether adding the type of ordinals $\mathcal{O}$ increases the computational power to the theory, i.e. if it is possible to represent functions that are not computable in polynomial time. By *definable function* we always mean a closed term of type $\mathsf{N} \multimap \mathsf{N}$.
Giving a negative answer to this question would properly justify the name 'feasible ordinals'. A characterisation of the definable functions could also help to determine an upper bound for the definable ordinals of the theory.

A possible approach for this question is to try to extend the proof of [Hof03, Theorem 5.3], in which the characterisation of the definable functions is obtained via a resource sensitive realizability model. The key case is the construction of the realizer for the term $\mathsf{it}_{\mathcal{O}}$ of the elimination rule of $\mathcal{O}$.

However, the construction of such realizer does not seem straightforward when it comes to the case of limit ordinals $\mathsf{sup}(d, f)$. In that case one knows by inductive hypothesis that all computations $f(n)$ for $n : \mathsf{N}$ terminate in polynomial time, but one needs to find a *uniform bound* for the family of polynomials in order to extract the desired realizer.

Another possibility consists in taking the syntactic point of view of Aehlig and Schwichtenberg [AS02] and trying to extend their result from LFPL to $\mathsf{LFPL} + \mathcal{O}$. They associate a polynomial $\theta(t)(x)$ to any term t. Then for each term t one considers the natural number $N(t) := \theta(t)(|FV(t)|) + |t|$, where $|t|$ is the syntactic length of t, and $FV(t)$ the set of its free variables.

Their main result is [AS02, Theorem 4.8], which states that if t reduces to t' then $N(t) > N(t')$. Therefore each term normalises in a finite number of steps bounded by $N(t)$ and applying this result to an application $f(n) : \mathsf{N}$ for a natural number $n : \mathsf{N}$ and $f : \mathsf{N} \multimap \mathsf{N}$ provides a proof that the definable functions are polytime (non-size-increasing) functions.

Again, we have a problem in giving the right definition of bounding polynomial $\theta(\mathsf{sup}(d, f))$ for a limit ordinal. Indeed, in this case the problem boils down to finding a definition for $\theta(\mathsf{sup}(d, f))$ which is bigger than $\theta((fn))$ for all $n : \mathsf{N}$.

Regarding the problem of proving upper bounds for the definable ordinals, we have explored the strategy of formalising Hofmann's realisability model of [Hof03] in a weak theory of arithmetic, hoping to extract information on the strength of the

theory. For such a strategy one would also needs to solve the problem of extending the realizability model to the case of $\mathcal{O}$.

The basic type-theoretic structure used to interpret LFPL, with the exception of the elimination rules for inductive types, can be easily formalised in Elementary Arithmetic. See [SW11, chapter 3] for the definition of the theory, where it is called $\mathsf{I}\Delta_0(\mathsf{exp})$.

However, when it comes to interpret the elimination rules for inductive types, the proof of [Hof03, Theorem 5.3] proceeds by induction on formulas of arbitrary logical complexity. Therefore it cannot be formalised in Elementary Arithmetic, but only in PA, which does not give new information on the strength of LFPL. Indeed, we formalise the function type in arithmetic as the formula $[\![A \multimap B]\!]_{Ar}(x)$ defined as:

$$\forall n\, \exists m\, [\![A]\!]_{Ar}(n) \wedge [\![B]\!]_{Ar}(m) \wedge \phi_x(n) = m,$$

where ϕ_x is the x-th recursive function, and $[\![\cdot]\!]_{Ar}$ the interpretation of types as formulas of arithmetic. Hence, the formulas formalising the interpretation of iterated function types have arbitrary logical complexity.

The second approach consists in trying to characterise the definable functions first and then study how the types $\mathcal{O}$ and $\mathsf{N} \multimap \mathsf{N}$ interact. The simplest way to study their interaction is to look at the type $\mathcal{O} \multimap (\mathsf{N} \multimap \mathsf{N})$ which can be seen as a type of 'hierarchies of functions' indexed by the ordinal notations in $\mathcal{O}$.
The hope would be to type in $\mathcal{O} \multimap (\mathsf{N} \multimap \mathsf{N})$ an appropriate hierarchy of functions like the slow growing hierarchy or the Hardy hierarchy. Then one could use it to prove that if a certain ordinal is definable then one can type in $\mathsf{N} \multimap \mathsf{N}$ a function that grows too fast to be polytime computable.

There are some issues related to this approach in addition to the obvious one to characterise the definable functions in the first place. The definitions of the slow growing and Hardy hierarchies for limit ordinals are done by diagonalisation, e.g. for the latter one:

$$H_\alpha(n) = \begin{cases} n & \text{if } \alpha = 0 \\ H_\beta(n+1) & \text{if } \alpha = \beta + 1 \\ H_{\lambda_n}(n) & \text{if } \alpha = sup_n(\lambda_n) \end{cases}$$

Defining H using this scheme is not possible in $\mathsf{LFPL}+\mathcal{O}$ because of the duplication of the variable n in the limit step.

In the third approach one tries to generalise the characterisation of ordinals definable in Gödel's T to the case of $\mathsf{LFPL} + \mathcal{O}$.

In the ordinal analysis of Gödel's T one proves that ϵ_0 is an upper bound for the definable ordinals of T. It has been carried out by various authors, e.g. [How70], [Tai65], [Vod97], [Wei98]. The neater and conceptually clearer in our opinion is Martin-Löf's paper [ML72].

The idea in [ML72] is to embed T in an infinitary system T_∞ where it is possible to unwind terms defined by primitive recursion and give an ordinal analysis of the normalisation process from which the upper bound follows. The system T_∞ is given by some unspecified base types among which there is a type for natural numbers N with no elimination nor computation rules and two type constructors: $\to$ the usual function type constructor and Π_n a type corresponding to an infinitary conjunction.

One can represent ordinals in T even without a specific type of ordinal notations, using what are sometimes called *Church's ordinals* (see [Sim00, chapter 9]): given any type α ordinals can be represented as functions that map 'ordinal notation structures on A' to terms of A.
So, given any type A we form the type:

$$\mathcal{O}(A) := [A \to (A \to A) \to ((N \to A) \to A)] \to A,$$

in which we can type ordinals less than ϵ_0.

However, note that the same strategy of interpreting the system in an infintary one cannot work for linear type theories in the same way as the usual one does. Indeed, we would type the number 2 in $\mathcal{O}(A)$ as $\lambda x, s, l.s(s(x))$ which is not an affine λ-term. So we would need a proper type of ordinal notations $\mathcal{O}$ rather than the relativised version $\mathcal{O}(A)$. Then we would need to interpret $\mathsf{LFPL} + \mathcal{O}$ into an appropriate infinitary system, and it is not obvious how one would interpret the elimination rule for $\mathcal{O}$ without having already characterised the definable ordinals.

Remark 6.4.1. An obvious consequence of the ordinal analysis of Gödel's T is that ϵ_0 is also an upper bound for the definable ordinals of $\mathsf{LFPL} + \mathcal{O}$ since the latter can be easily interpreted in the former.

In conclusion, we disproved Hofmann's conjecture, showing that ω^2 can be reached by the rather slow fundamental sequence $\{\omega \cdot |n|\}_n$ in Lemma 6.3.4, and similarly we showed in Lemma 6.3.6 that for any given $k \in \mathbb{N}$ the ordinal ω^k ca be defined. We have also determined a sufficient condition for carrying on the same idea to ω^ω in Theorem 6.3.8.

The question of what is the sharp upper bound for the definable ordinals is still open. At this point our opinion is that the upper bound is either ω^ω or ϵ_0.

If there is a way to code an ordinal multiplication function $\mu(\alpha, n) = \alpha \cdot n$ as in Theorem 6.3.8 using a construction analogous to the one detailed in [Hof02] for ordinary multiplication, then ω^ω would be definable. In such a situation we expect all finite towers of powers of ω to be definable in a similar way.

If it is not possible to construct μ and ω^ω is indeed the supremum of the definable ordinals, this may be proved by using the ideas exposed in this section.

CHAPTER 7

Future research

Some questions are still left open and may be object of future work. In chapter 1, Theorem 1.3.14 leaves open the question if stronger set-theoretic axioms can be validated in the interpretations $[\![\cdot]\!]_{k,h}$, possibly using stronger type theories. Moreover, in the interpretations $[\![\cdot]\!]_{k,h}$ of chapter 1 we used closure properties of k-types to prove that they can validate set-theoretic axioms. One may try to show that this proof works for any locally cartesian closed category of types containing $\mathsf{N}, \mathsf{0}, \mathsf{1}$ and $\mathsf{2}$. We expect this to be the easiest among the questions left open for future research.

Recall that Whitehead's principle asserts that if a continuous map between CW-complexes $f : X \to Y$ induces an isomorphism on all homotopy groups, then it is an equivalence. The translation of this principle in homotopy type theory is not provable. One may then introduce in homotopy type theory a notion of ∞-truncation that given a type X returns a type $\|X\|_\infty$ for which Whitehead's principle holds. Then, this notion of truncation may be used to construct variants of the interpretations $[\![\cdot]\!]_{k,h}$[1].

For chapter 2, note that Theorem 2.5.1 uses the fact that the type theory H_k can be interpreted into the extensional theory $\mathsf{ML}^{\mathsf{e}}_1\mathsf{V}_\infty$. This interpretation cannot take place for a type theory with the univalence axiom, hence a possibility to extend Theorem 2.5.1 to homotopy type theory is to try to construct the groupoid model internally in the theory of classes of $\mathsf{CZF} + \Pi\Sigma\text{-}\mathsf{AC}$ so that one can reconstruct the proof of Rathjen and Tupailo in this context.

Regarding chapter 3, one may want to look at the type $\mathsf{V}^k_G := (\Sigma x : \mathsf{V}_k)\mathsf{itset}(x)$ for the interpretation of sets as a hybrid version between Gylterud's type V_G and our V_k.

Chapter 6 leaves open a number of questions. Primarily, the characterisation of the definable functions of type $\mathsf{N} \multimap \mathsf{N}$ in the system $\mathsf{LFPL} + \mathcal{O}$. Proving that the definable functions are polytime is needed in order to properly justify the claim that this is the study of feasible ordinals.
Then, some investigation is required to see if techniques of encoding similar to the ones of [Hof02] can be used to define ordinal multiplication between an ordinal and

[1]We thank Nicolai Kraus for suggesting this question.

a natural number. This would allow one to define the ordinal ω^ω in $\mathsf{LFPL} + \mathcal{O}$, as detailed in Theorem 6.3.8. If that is not the case we believe it is likely that the upper bound for the definable ordinals of $\mathsf{LFPL} + \mathcal{O}$ is ω^ω. Then one may try use the observations laid out in Section 6.4 to prove an upper bound for the definable ordinals.

Appendix

A. Constructive set theories

Here we introduce BCS, recall the axioms of CST and CZF and some basic properties of CZF. All the set theories we consider in this thesis are constructive theories, meaning that the underlying logic is intuitionistic. We use the symbol $\doteq$ for set-theoretic equality.

Axioms of BCS. We call *basic constructive set theory*, BCS for short, the theory given by the following axioms: Extensionality, $\in$-Induction, Pairing, Union, Bounded Separation and Infinity.

Extensionality:

$$\forall\alpha\,\forall\beta\,[\forall x\,(x \in \alpha \Leftrightarrow x \in \beta) \Rightarrow (\alpha \doteq \beta)]$$

Set-Induction, or $\in$-Induction:

$$\forall\alpha\,[(\forall x \in \alpha)\phi(x) \Rightarrow \phi(\alpha)] \Rightarrow \forall x\,\phi(x)$$

Pairing:

$$\forall\alpha\,\forall\beta\,\exists z\,(\alpha \in z \wedge \beta \in z)$$

Union:

$$\forall\alpha\,\exists z\,(\forall x \in \alpha)(\forall y \in x)(y \in z)$$

Bounded Separation:

$$\forall\alpha\,\exists\gamma[(\forall x \in \gamma)(x \in \alpha \wedge \phi(x)) \wedge (\forall x \in \alpha)(\phi(x) \Rightarrow x \in \gamma)]$$

for all bounded formulas $\phi(x)$, in which y is not free in $\phi(x)$. Recall that a bounded formula is one such that all quantifiers are bounded, i.e. $(\forall x \in \alpha)\phi(x)$ and $(\exists x \in \alpha)\phi(x)$.

Infinity:

$$\exists z[(\exists x \in z)Zero(x) \wedge (\forall y \in z)(\exists x \in z)Succ(y,x) \wedge (\forall x \in z)(Zero(x) \vee (\exists y \in x)Succ(y,x))]$$

where the expression $Zero(x)$ is the formula $(\forall y \in x)\bot$, and the expression $Succ(\alpha, \beta)$ is the following formula:

$$(\forall x \in \alpha)(x \in \beta) \wedge (\alpha \in \beta) \wedge (\forall x \in \beta)(x \in \alpha \vee x \doteq \alpha).$$

Axioms of CST. Myhill's CST consists of the axioms of BCS plus Exponentiation and Replacement. The Exponentiation axiom guarantees that we can form the set of functions between two given sets.
Replacement:

$$(\forall x \in \alpha)\exists! y\, \phi(x, y) \Rightarrow \exists z(\forall x \in \alpha)(\exists y \in z)\phi(x, y)$$

for all formulas $\phi(x, y)$ in which z is not free in $\phi(x, y)$.
Exponentiation:

$$\forall \alpha, \beta\, \exists \gamma[\forall f(\alpha \xrightarrow{f} \beta) \Rightarrow f \in \gamma],$$

where the expression $(\alpha \xrightarrow{f} \beta)$ is a shorthand for the formula stating that f is a set-theoretic function, i.e. a set of pairs satisfying a functional relation:

$$(\forall x \in \alpha)(\exists! y \in \beta)\, \langle x, y\rangle \in f \ \wedge\ (\forall z \in f)(\exists x \in \alpha)(\exists y \in \beta)\, z \doteq \langle x, y\rangle.$$

Axioms of CZF. Aczel's CZF consists of the axioms of BCS plus Strong Collection and Subset Collection. Strong Collection strengthens Replacement since it has a weaker premiss and a stronger conclusion. The Subset Collection axiom strictly implies the Exponentiation axiom and is strictly implied by the Power Set axiom.

We use the notation $\forall \exists \frac{x \in \alpha}{y \in \beta} \phi(x, y)$ to abbreviate the following formula:

$$(\forall x \in \alpha)(\exists y \in \beta)\phi(x, y) \wedge (\forall y \in \beta)(\exists x \in \alpha)\phi(x, y).$$

Strong Collection:

$$(\forall x \in \alpha)\exists y \phi(x, y) \Rightarrow \exists z\Big(\forall \exists \frac{x \in \alpha}{y \in z} \phi(x, y)\Big)$$

for all formulas $\phi(x, y)$ in which z is not free in $\phi(x, y)$.
Subset Collection:

$$\exists \gamma \forall u\Big[(\forall x \in \alpha)(\exists y \in \beta)\phi(x, y, u) \Rightarrow (\exists z \in \gamma)\Big(\forall \exists \frac{x \in \alpha}{y \in z} \phi(x, y, u)\Big)\Big].$$

Note that the universal quantification over u ensures that the construction of the set γ is uniform with respect to the variable u appearing in ϕ. This is a technical condition that is used for example in Theorem A.2 to show that Subset Collection implies Exponentiation.

The Fullness axiom. The formulation of the Subset Collection axiom may obscure its content, for this reason we recall that Subset Collection is equivalent

to the Fullness axiom. The following definition gives some notions used in the formulation of the Fullness axiom.

Definition A.1.

(a) We denote by $mv(\beta^\alpha)$ the class of total relations (i.e. multi-valued functions) from α to β. Formally:

$$mv(\beta^\alpha) := \{R \subseteq \alpha \times \beta \mid \forall x \in \alpha\, \exists y \in \beta\, \langle x, y\rangle \in R\};$$

(b) given sets α and β, and a set $\gamma \subseteq mv(\beta^\alpha)$ we say that γ is *full in α and β* if and only if every total relation from α to β has an approximation in γ, i.e. $\forall R \in mv(\beta^\alpha)\, \exists S \in \gamma :\ S \subseteq R$.

Note that neither CZF nor CST prove that $mv(\beta^\alpha)$ is a set [AR10, Proposition 5.1.6(ii)].

Fullness:

$$\forall\alpha\, \forall\beta \exists z[z \text{ is full in } \alpha \text{ and } \beta]$$

The Fullness axiom is related to the Subset Collection and the Exponentiation axioms as follows.

Theorem A.2 (Aczel)**.** *Consider a basic constructive set theory consisting of the following axioms: Extensionality, Pairing, Union, Replacement, and Bounded Separation. In such a theory one can prove that Subset Collection implies Fullness and Fullness in turn implies Exponentiation.*

Proof. See [AR10, Theorem 5.1.2] for the details, here we provide a sketch of the proof. For the first implication the idea is that given sets α, β, to find a set γ full in α and β one can consider the formula $\phi(x, y, u)$ given by $y \in u \wedge (\exists z \in \beta)(y \doteq \langle x, z\rangle)$, and apply Subset Collection. The implication from Fullness to Exponentiation is straightforward. □

Remark A.3.

(i) In a base theory consisting of Extensionality, Pairing, Union, Replacement, Bounded Separation and Strong Collection, we have that the Fullness axiom implies the Subset Collection axiom. See [AR10, Theorem 5.1.2(ii)].

(ii) Fullness does not imply Power Set, see [AR10, Proposition 5.1.6].

(iii) Exponentiation does not imply Fullness, see [Lub05].

B. Type-theoretic rules

Martin-Löf type theory. Here we recall the rules of Martin-Löf type theory, this is the theory that we referred as $\mathsf{ML_1W}$ in the thesis. The rules for $\mathsf{ML_1W}$ can be found in [ML84], with the exception of the ones for intensional identity types Id.

In order to simplify the presentation of the rules we omit mention of a context that is common to both the premisses and the conclusion.

We assume the propositional η-rule for Π-types, i.e. a term inhabiting the type:

$$\mathsf{Id}_{(\Pi x:A)B(x)}(f, \lambda x.fx). \tag{B.1}$$

We start with the presentation of the structural rules and the rules governing definitional equality.

$$\frac{A : \mathsf{type}}{x : A \vdash x : A}\ \text{Variable declaration} \qquad \frac{x : A, \Delta \vdash \mathcal{J} \qquad a : A}{\Delta[a/x] \vdash \mathcal{J}[a/x]}\ \text{Substitution}$$

$$\frac{\Gamma, \Delta \vdash \mathcal{J} \qquad A : \mathsf{type}}{\Gamma, x : A, \Delta \vdash \mathcal{J}}\ \text{Weakening} \qquad \frac{\Gamma, x : A, x : A, \Delta \vdash \mathcal{J}}{\Gamma, x : A, \Delta \vdash \mathcal{J}}\ \text{Contraction}$$

$$\frac{\Gamma, x : A, y : B, \Delta \vdash \mathcal{J}}{\Gamma, y : B, x : A, \Delta \vdash \mathcal{J}}\ \text{Exchange}$$

TABLE A1. Structural rules

$$\frac{A : \mathsf{type}}{A = A : \mathsf{type}} \qquad \frac{A_1 = A_2 : \mathsf{type}}{A_2 = A_1 : \mathsf{type}} \qquad \frac{A_1 = A_2 : \mathsf{type} \qquad A_2 = A_3 : \mathsf{type}}{A_1 = A_3 : \mathsf{type}}$$

$$\frac{a : A}{a = a : A} \qquad \frac{a_1 = a_2 : A}{a_2 = a_1 : A} \qquad \frac{a_1 = a_2 : A \qquad a_2 = a_3 : A}{a_1 = a_3 : A}$$

$$\frac{a : A_1 \qquad A_1 = A_2}{a : A_2} \qquad \frac{a_1 = a_2 : A_1 \qquad A_1 = A_2}{a_1 = a_2 : A_2}$$

TABLE A2. Rules for the definitional equality

$$\frac{x : A, \Delta \vdash C(x) : \mathsf{type} \qquad a_1 = a_2 : A}{\Delta[a_1/x] \vdash C[a_1/x] = C[a_2/x] : \mathsf{type}}$$

$$\frac{x : A, \Delta \vdash c(x) : C(x) \qquad a_1 = a_2 : A}{\Delta[a_1/x] \vdash c[a_1/x] = c[a_2/x] : C[a_1/x]}$$

$$\frac{x : A \vdash B_1(x) = B_2(x) : \mathsf{type}}{(\nabla x : A)B_1(x) = (\nabla x : A)B_2(x) : \mathsf{type}}$$

$$\frac{x : A \vdash b_1(x) = b_2(x) : B(x)}{\lambda x.b_1(x) = \lambda x.b_2(x) : (\Pi x : A)B(x)} \qquad \dots$$

TABLE A3. Congruence and conversion rules

In the rule of table A3 the symbol ∇ stands for any type constructor: Π, Σ, W. There are also analogous congruence and conversion rules for the identity types and for all canonical terms. We omit them for brevity.

Here we present the rules for the base types: $\mathsf{0}$, $\mathsf{1}$, $\mathsf{2}$ and N.

$$\frac{}{\mathsf{0} : \mathsf{type}} \qquad \frac{c : \mathsf{0} \qquad C : \mathsf{type}}{\mathsf{R_0}(c) : C}$$

TABLE A4. Rules for $\mathsf{0}$

$$\frac{}{\mathsf{1} : \mathsf{type}} \qquad \frac{}{0 : \mathsf{1}} \qquad \frac{c : \mathsf{1} \qquad c_0 : C(0)}{\mathsf{R_1}(c, c_0) : C(c)} \qquad \frac{c : \mathsf{1} \qquad c_0 : C(0)}{\mathsf{R_1}(0, c_0) = c_0 : C(0)}$$

TABLE A5. Rules for $\mathsf{1}$

$$\frac{}{2 : \mathsf{type}} \qquad \frac{}{0, 1 : 2} \qquad \frac{c : 2 \qquad c_0 : C(0) \qquad c_1 : C(1)}{\mathsf{R}_2(c, c_0, c_1) : C(c)}$$

$$\frac{c : 2 \qquad c_0 : C(0) \qquad c_1 : C(1)}{\mathsf{R}_2(i, c_0, c_1) = c_i : C(0) \quad \text{for } i = 0, 1}$$

TABLE A6. Rules for 2

$$\frac{}{\mathsf{N} : \mathsf{type}} \qquad \frac{}{0 : \mathsf{N}} \qquad \frac{n : \mathsf{N}}{\mathsf{s}(n) : \mathsf{N}}$$

$$\frac{d : C(0) \qquad x : \mathsf{N}, y : C(x) \vdash e(x, y) : C(\mathsf{s}(x)) \qquad c : \mathsf{N}}{\mathsf{R}_\mathsf{N}(c, d, e(x, y)) : C(c)}$$

$$\frac{d : C(0) \qquad x : \mathsf{N}, y : C(x) \vdash e(x, y) : C(\mathsf{s}(x))}{\mathsf{R}_\mathsf{N}(0, d, e(x, y)) = d : C(0)}$$

$$\frac{a : \mathsf{N} \qquad d : C(0) \qquad x : \mathsf{N}, y : C(x) \vdash e(x, y) : C(\mathsf{s}(x))}{\mathsf{R}_\mathsf{N}(\mathsf{s}(a), d, e(x, y)) = e(a, \mathsf{R}_\mathsf{N}(a, d, e(x, y))) : C(\mathsf{s}(a))}$$

TABLE A7. Rules for natural numbers N

Here we present the rules for the type constructors: $\Pi, \Sigma, +, W$, and Id.

$$\frac{A : \mathsf{type} \qquad x : A \vdash B(x) : \mathsf{type}}{(\Pi x : A)B(x) : \mathsf{type}} \qquad \frac{x : A \vdash b(x) : B(x)}{\lambda x.b(x) : (\Pi x : A)B(x)}$$

$$\frac{a : A \qquad c : (\Pi x : A)B(x)}{c(a) : B(a)} \qquad \frac{a : A \qquad x : A \vdash b(x) : B(x)}{(\lambda x.b(x))(a) = b[a/x] : B(a)}$$

TABLE A8. Rules for Π-types

The function type constructor $A \to B$ is obtained as the non dependent version of Π, i.e. for a constant family of types $x : A \vdash B : \mathsf{type}$.

Note that we abuse notation for the application of a type-theoretic function, so that we write $c(a)$, rather than $\mathsf{ap}(c, a)$. We do so in order to simplify the notation.

$$\frac{A : \mathsf{type} \qquad x : A \vdash B(x) : \mathsf{type}}{(\Sigma x : A)B(x) : \mathsf{type}} \qquad \frac{a : A \qquad b : B(a)}{(a, b) : (\Sigma x : A)B(x)}$$

$$\frac{x : A, y : B(x) \vdash d(x, y) : C((x, y)) \qquad c : (\Sigma x : A)B(x)}{\mathsf{E}(c, d(x, y)) : C(c)}$$

$$\frac{a : A \qquad b : B(a) \qquad x : A, y : B(x) \vdash d(x, y) : C((x, y))}{\mathsf{E}((a, b), d(x, y)) = d(a, b) : C(c)}$$

TABLE A9. Rules for Σ-types

The cartesian product constructor $A \times B$ is obtained as the non dependent version of Σ, i.e. for a constant family of types $x : A \vdash B : \mathsf{type}$.

From the rules for Σ-types one can obtain the ones for projections as derived rules. For $t : (\Sigma x : A)B(x)$ we use the notation $t.1 : A$ and $t.2 : B(t.1)$ for the projections.

$$\frac{A : \mathsf{type} \qquad B : \mathsf{type}}{A + B : \mathsf{type}} \qquad \frac{a : A}{\mathsf{i}(a) : A + B} \qquad \frac{b : B}{\mathsf{j}(b) : A + B}$$

$$\frac{x : A \vdash d(x) : C(\mathsf{i}(x)) \qquad y : B \vdash e(y) : C(\mathsf{j}(y)) \qquad c : A + B}{\mathsf{D}(c, d(x), e(y)) : C(c)}$$

$$\frac{a : A \qquad x : A \vdash d(x) : C(\mathsf{i}(x)) \qquad y : B \vdash e(y) : C(\mathsf{j}(y))}{\mathsf{D}(\mathsf{i}(a), d(x), e(y)) = d(a) : C(c)}$$

$$\frac{b : B \qquad x : A \vdash d(x) : C(\mathsf{i}(x)) \qquad y : B \vdash e(y) : C(\mathsf{j}(y))}{\mathsf{D}(\mathsf{j}(b), d(x), e(y)) = e(b) : C(c)}$$

TABLE A10. Rules for $+$-types

$$\frac{A : \mathsf{type} \qquad x : A \vdash B(x) : \mathsf{type}}{(Wx : A)B(x) : \mathsf{type}} \qquad \frac{a : A \qquad b : B(a) \to (Wx : A)B(x)}{\mathsf{sup}(a, b) : (Wx : A)B(x)}$$

$$\frac{\begin{array}{c} c : (Wx : A)B(x) \\ x : A, y : B(x) \to (Wx : A)B(x), z : (\Pi v : B(x))C(y(v)) \vdash d : C(\mathsf{sup}(x, y)) \end{array}}{\mathsf{R}_W(c, d(x, y, z)) : C(c)}$$

$$\frac{\begin{array}{c} a : A \qquad b : B(a) \to (Wx : A)B(x) \\ x : A, y : B(x) \to (Wx : A)B(x), z : (\Pi v : B(x))C(y(v)) \vdash d : C(\mathsf{sup}(x, y)) \end{array}}{\mathsf{R}_W(\mathsf{sup}(a, b), d(x, y, z)) = d(a, b, \lambda v.\mathsf{R}_W(b(v), d(x, y, z))) : C(\mathsf{sup}(a, b))}$$

TABLE A11. Rules for W-types

$$\frac{A : \mathsf{type} \qquad a, b : A}{\mathsf{Id}_A(a, b) : \mathsf{type}} \qquad \frac{A : \mathsf{type} \qquad a : A}{\mathsf{refl}_a : \mathsf{Id}_A(a, a)}$$

$$\frac{x, y : A, u : \mathsf{Id}_A(x, y) \vdash C(x, y, u) : \mathsf{type} \qquad z : A \vdash d(z) : C(z, z, \mathsf{refl}_z)}{\mathsf{J}_{z,d}(x, y, u) : C(x, y, u)}$$

$$\frac{x, y : A, u : \mathsf{Id}_A(x, y) \vdash C(x, y, u) : \mathsf{type} \qquad z : A \vdash d(z) : C(z, z, \mathsf{refl}_z)}{\mathsf{J}_{z,d}(x, x, \mathsf{refl}_x) = d(x) : C(x, x, \mathsf{refl}_x)}$$

TABLE A12. Rules for Id-types

$$\frac{}{\mathsf{U} : \mathsf{type}} \qquad \frac{a : \mathsf{U}}{\mathsf{T}(a) : \mathsf{type}}$$

$$\frac{a : \mathsf{U} \qquad x : \mathsf{T}(a) \vdash b(x) : \mathsf{U}}{\pi(a, b(x)) : \mathsf{U}} \qquad \frac{a : \mathsf{U} \qquad x : \mathsf{T}(a) \vdash b(x) : \mathsf{U}}{\mathsf{T}(\pi(a, b(x))) = (\Pi x : \mathsf{T}(a))\mathsf{T}(b(x))}$$

$$\frac{}{\mathsf{n} : \mathsf{U}} \qquad \frac{}{\mathsf{T}(\mathsf{n}) = \mathsf{N}} \qquad \ldots$$

TABLE A13. Rules for the universe U

Table A13 gives only the rules for Π-types and natural numbers in the universe. There are similar rules for all other base types and constructors of the theory. We omit them for brevity. In our notation the base types and type constructors $0, 1, 2, \mathsf{N}, \Pi, \to, \Sigma, \times, +, W, \mathsf{Id}_A$ are represented in the universe using the corresponding terms and term constructors: $0, 1, 2, \mathsf{n}, \pi, \mathsf{exp}, \sigma, \mathsf{times}, \mathsf{plus}, \mathsf{w}, \mathsf{i}_a$.

Logic-enriched type theories. The rules of a logic-enriched type theory consist in rules for its pure type-theoretic part, plus logical rules for formulas. We refer to IL_1 as the rules for intuitionistic predicate logic plus a type of small propositions P. The rules for intuitionistic logic are the usual natural deduction rules in sequent calculus style. We give the rules for the quantifiers as an illustration.

Recall that in the context of logic-enriched type theories we use the notation $\Rightarrow$ for the structural symbol in a logical context $\phi_1, \ldots, \phi_n \Rightarrow \phi$, and we use the symbol $\supset$ for material implication.

$$\frac{A : \mathsf{type} \qquad x : A \vdash \phi(x) : \mathsf{prop}}{(\forall x : A)\phi(x) : \mathsf{prop}}$$

$$\frac{x : A \vdash \phi(x)}{(\forall x : A)\phi(x)} \qquad \frac{(\forall x : A)\phi(x) \qquad t : A}{\phi(t)}$$

TABLE A14. Rules for the universal quantifier

$$\frac{A : \mathsf{type} \qquad x : A \vdash \phi(x) : \mathsf{prop}}{(\exists x : A)\phi(x) : \mathsf{prop}}$$

$$\frac{x : A \vdash \phi(x) : \mathsf{prop} \qquad a : A \qquad \phi[a/x]}{(\exists x : A)\phi(x)}$$

$$\frac{(\exists x : A)\phi(x) \qquad \psi : \mathsf{prop} \qquad x : A \vdash \phi(x) \Rightarrow \psi}{\psi}$$

TABLE A15. Rules for the existential quantifier

Next we recall the rules for the type of small propositions P. Note that rather than a definitional equality, the computation rules for propositions ask for a logical equivalence $\equiv$.

$$\frac{}{\mathsf{P} : \mathsf{type}} \qquad \frac{p : \mathsf{P}}{\tau(p) : \mathsf{prop}}$$

$$\frac{p_1 : \mathsf{P} \qquad p_2 : \mathsf{P}}{p_1 \wedge p_2 : \mathsf{P}} \qquad \frac{p_1 : \mathsf{P} \qquad p_2 : \mathsf{P}}{\tau(p_1 \wedge p_2) \equiv \tau(p_1) \wedge \tau(p_2)}$$

$$\frac{a : \mathsf{U} \qquad x : \mathsf{T}(a) \vdash p(x) : \mathsf{P}}{(\forall x : a)p(x) : \mathsf{P}} \qquad \frac{a : \mathsf{U} \qquad x : \mathsf{T}(a) \vdash p(x) : \mathsf{P}}{\tau((\forall x : a)p(x)) \equiv (\forall x : \mathsf{T}(a))\tau(p(x))}$$

$$\ldots$$

TABLE A16. Rules for the type of small propositions P

Following [GA06], in the logic-enriched type theories we consider any inductive type C is endowed with its induction rule IND_C.

$$\frac{z : C \vdash \phi(z) : \mathsf{prop} \qquad e : C \qquad \mathsf{Ind}_C}{\phi[e/z]}$$

TABLE A17. Induction rule for inductive types IND_C

Here Ind_C is the induction premiss, as given in the following table.

C	Ind_C
$\mathsf{0}$	
$\mathsf{1}$	$\phi(0)$
$\mathsf{2}$	$\phi(0) \quad \phi(1)$
N	$\phi(0) \quad x : \mathsf{N} \vdash \phi(x) \Rightarrow \phi(\mathsf{s}(x))$
$(\Sigma x : A)B(x)$	$x : A, y : B \vdash \phi((x, y))$
$A + B$	$x : A \vdash \phi(\mathsf{i}(x)) \quad y : B \vdash \phi(\mathsf{j}(y))$
$(W x : A)B(x)$	$x : A, u : B \to C \vdash (\forall y : B)\phi(u(y)) \Rightarrow \phi(\mathsf{sup}(x, u))$
$\|A\|$	$x : A \vdash \phi(\|x\|)$

The induction rule for identity types $\mathsf{IND}_{\mathsf{Id}}$ is slightly more complex.

$$\frac{x, y : A, p : \mathsf{Id}_A(x, y) \vdash \phi(x, y, p) : \mathsf{prop} \qquad e : \mathsf{Id}_A(a, b) \qquad z : A \vdash \phi(z, z, \mathsf{refl}_{\mathsf{z}})}{\phi[a, b, e/x, y, p]}$$

TABLE A18. Induction rule for identity types $\mathsf{IND}_{\mathsf{Id}}$

C. Set-theoretic semantics of LFPL + $\mathcal{O}$

Here we recall from [Hof03, Section 2.1] a set-theoretic semantics for terms and types of LFPL and we extend it to LFPL+$\mathcal{O}$, giving the definition in full details. We work informally in Kripke-Platek set theory (KPω, [Poh09, Section 11.4]) which is more than enough for our purposes. The definition proceeds by induction on their derivation.

- $[\![\mathsf{2}]\!] := \{0, 1\}$;
- $[\![\mathsf{N}]\!] := \mathbb{N}$;
- $[\![\mathcal{O}]\!] := \{\alpha \in On \mid \alpha < \epsilon_0\}$, where On is a set of ordinals;
- $[\![A \otimes B]\!] := [\![A]\!] \times [\![B]\!]$;
- $[\![A \& B]\!] := [\![A]\!] \times [\![B]\!]$;
- $[\![A \multimap B]\!] := [\![A]\!] \to [\![B]\!]$;
- $[\![L(X)]\!] := \bigcup_{n \in \mathbb{N}} [\![X]\!]^n$;
- $[\![T(X)]\!] := \{T \mid T \text{ is a tree with leaves and nodes labelled by elements of } X\}$;
- $[\![\diamond]\!] := \{0\}$.

Notice that by Remark 6.4.1 we know that ϵ_0 is an upper bound for the interpretation of the ordinals of $\mathsf{LFPL} + \mathcal{O}$.

Given a context $\Gamma = \{x_1 : A_1, \ldots, x_n : A_n\}$ we define an *environment* to be a function, say η, mapping variables from $\{x_1, \ldots, x_n\}$ to elements $\eta(x_i) \in [\![A_i]\!]$, equivalently an environment is an n-tuple $\langle \eta(x_1), \ldots, \eta(x_n)\rangle \in [\![A_1]\!] \times \cdots \times [\![A_n]\!]$. Then the interpretation of a judgement $\Gamma \vdash e : A$ is a function $[\![e]\!]$ mapping environments for Γ to elements of $[\![A]\!]$.

The set-theoretic semantics for terms is given by induction on the derivation of the term in the natural way.

Affine type theory:

- $(\multimap^+)$**:** if $e = [\![x : A \vdash t : B]\!]$ is a function mapping environments for the context $x : A$ to elements of $[\![B]\!]$, then e itself is the interpretation of the λ-abstraction $[\![\lambda x.t]\!] := e : [\![A]\!] \to [\![B]\!]$;
- $(\multimap^-)$**:** if $f : A \multimap B$ and $a : A$ then $[\![fa]\!] := [\![f]\!]([\![a]\!]) \in [\![B]\!]$;
- $(\otimes^+)$**:** $[\![t_1 \otimes t_2]\!] := \langle [\![t_1]\!], [\![t_2]\!]\rangle \in [\![A \otimes B]\!]$;
- $(\otimes^-)$**:** given $t : A \otimes B$ and $e = [\![x : A, y : B \vdash r : C]\!]$ a function mapping environments for the context $x : A, y : B$ to elements of $[\![C]\!]$, then e can be seen as a function $[\![A]\!] \times [\![B]\!] \to [\![C]\!]$. So we define $[\![\mathsf{let}\, t\, \mathsf{be}\, x \otimes y\, \mathsf{in}\, r]\!] := e([\![t]\!]) \in [\![C]\!]$;
- $(\&^+)$**:** $[\![\langle t_1, t_2\rangle]\!] = \langle [\![t_1]\!], [\![t_2]\!]\rangle \in [\![A \& B]\!]$;
- $(\&^-)$**:** given $t : A \& B$ we define $[\![t.1]\!] = \pi_1([\![t]\!]) \in [\![A]\!]$, where π_1 is the first projection, and similarly for $[\![t.2]\!] = \pi_2([\![t]\!]) \in [\![B]\!]$.

Booleans:

- (2_0^+)**:** $[\![\mathsf{ff}]\!] := 0$;
- (2_1^+)**:** $[\![\mathsf{tt}]\!] := 1$;
- (2^-)**:** the function $[\![\mathsf{if}]\!] : \{0, 1\} \to [\![A]\!] \to [\![A]\!] \to [\![A]\!]$ is defined by $[\![\mathsf{if}]\!](0, a, b) = a$ and $[\![\mathsf{if}]\!](1, a, b) = b$.

Natural numbers:

- (N_0^+)**:** $[\![0]\!] = 0 \in \mathbb{N}$;
- (N_1^+)**:** given $n \in [\![\mathsf{N}]\!]$ then $[\![\mathsf{s}_0]\!](*, n) = 2n$ and $[\![\mathsf{s}_1]\!](*, n) = 2n + 1$;
- (N^-)**:** given $a \in [\![A]\!]$ and $u_0, u_1 : \{*\} \to [\![A]\!] \to [\![A]\!]$ we write:

$$It_N(a, u_0, u_1) : \mathbb{N} \to [\![A]\!],$$

for the function defined by recursion using a, u_0 and u_1:

$$It_N(a,u_0,u_1)(m) := \begin{cases} a & \text{if } m = 0, \\ u_0(*, It_N(a,u_0,u_1)(n)) & \text{if } m = 2n, \\ u_1(*, It_N(a,u_0,u_1)(n)) & \text{if } m = 2n+1. \end{cases}$$

Then given $t : A$ and $h_0, h_1 : \diamond \multimap A \multimap A$ we define $[\![\mathsf{it}^A_{\mathsf{N}}(t,h_0,h_1)]\!] := It_N([\![t]\!],[\![h_0]\!],[\![h_1]\!])$.

Ordinals:

($\mathcal{O}^+_0$)**:** $[\![0]\!] = 0 \in [\![\mathcal{O}]\!]$;

($\mathcal{O}^+_1$)**:** if $\alpha = \lambda + n \in [\![\mathcal{O}]\!]$ where λ is a limit ordinal or zero and n a finite ordinal then $[\![\mathsf{S_0}]\!](*,\alpha) = \lambda + 2n$ and $[\![\mathsf{S_1}]\!](*,\alpha) = \lambda + 2n + 1$;

($\mathcal{O}^+_2$)**:** if $f : [\![\mathsf{N}]\!] \to [\![\mathcal{O}]\!]$ then $[\![\mathsf{sup}]\!](*,f) = sup_{n\in\mathbb{N}}\{f(n)\}$ where sup is the set-theoretic supremum of ordinals;

($\mathcal{O}^-$)**:** given $a \in [\![A]\!]$ and functions $u_0, u_1 : \{*\} \to [\![A]\!] \to [\![A]\!]$ and a function $v : \{*\} \to (\mathbb{N} \to [\![A]\!]) \to [\![A]\!]$ we write $It_{\mathcal{O}}(a,u_0,u_1,v) : [\![\mathcal{O}]\!] \to [\![A]\!]$ for the function defined by recursion using a, u_0, u_1 and v:

$$It_{\mathcal{O}}(a,u_0,u_1,v)(\alpha) := \begin{cases} a & \text{if } \alpha = 0, \\ u_0(*, It_{\mathcal{O}}(a,u_0,u_1,v)(\lambda+n)) & \text{if } \alpha = \lambda + 2n \text{ and } \lambda \text{ limit}, \\ u_1(*, It_{\mathcal{O}}(a,u_0,u_1,v)(\lambda+n)) & \text{if } \alpha = \lambda + 2n + 1 \text{ and } \lambda \text{ limit}, \\ v(*, n \mapsto It_{\mathcal{O}}(a,u_0,u_1,v)(f(n))) & \text{if } \alpha = sup_n(f(n)). \end{cases}$$

Then given $t : A$ and $h_0, h_1 : \diamond \multimap A \multimap A$ and $l : \diamond \multimap (\mathsf{N} \multimap A) \multimap A$ we define $[\![\mathsf{it}^A_{\mathcal{O}}(t,h_0,h_1,l)]\!] := It_{\mathcal{O}}([\![t]\!],[\![h_0]\!],[\![h_1]\!],[\![l]\!])$.

Lists:

($L(X)^+_0$)**:** $[\![\mathsf{nil}]\!] := \emptyset \in [\![X]\!]^0$;

($L(X)^+_1$)**:** if $x \in [\![X]\!]$ and $l \in [\![X]\!]^n$, then $[\![\mathsf{cons}]\!](*,x,l) := \langle x, l\rangle$;

($L(X)^-$)**:** given $a \in [\![A]\!]$ and $u : \{*\} \to [\![X]\!] \to [\![A]\!] \to [\![A]\!]$ we write $It_{L(X)}(a,u) : [\![L(X)]\!] \to [\![A]\!]$ for the function defined by recursion using a and u:

$$It_{L(X)}(a,u)(l) := \begin{cases} a & \text{if } l = \emptyset, \\ u(*, x, It_{L(X)}(a,u)(l')) & \text{if } l = \langle x, l'\rangle. \end{cases}$$

Then given $t : A$ and $h : \diamond \multimap A \multimap A$ we define $[\![\mathsf{it}^A_{L(X)}(t,h)]\!] := It_{L(X)}([\![t]\!],[\![h]\!])$.

Trees:

($T(X)_0^+$)**:** $[\![\mathsf{leaf}]\!] : [\![X]\!] \to [\![T(X)]\!]$ is the function that given $x \in [\![X]\!]$ returns the tree consisting of only one leaf labelled by x. We denote such a tree by $\langle x \rangle$;

($T(X)_1^+$)**:** $[\![\mathsf{node}]\!] : \{*\} \to [\![X]\!] \to [\![T(X)]\!] \to [\![T(X)]\!]$ is the function that given an $x \in [\![X]\!]$, and two trees t_1, t_2 returns the binary tree with root labelled by x, and t_1 and t_2 as the subtrees with the root as parent. We denote such a tree by $\langle x, t_1, t_2 \rangle$;

($T(X)^-$)**:** given a function $f : [\![X]\!] \to [\![A]\!]$ and a function:

$$u : \{*\} \to [\![X]\!] \to [\![A]\!] \to [\![A]\!] \to [\![A]\!],$$

we write $It_{T(X)}(f, u) : [\![T(X)]\!] \to [\![A]\!]$ defined by recursion using f and h as follows:

$$It_{T(X)}(f,u)(t) := \begin{cases} f(x) & \text{if } t = \langle x \rangle, \\ u(*, x, It_{T(X)}(f,u)(t_1), It_{T(X)}(f,u)(t_2)) & \text{if } t = \langle x, t_1, t_2 \rangle. \end{cases}$$

Then given terms $g : X \multimap A$ and $k : \diamond \multimap X \multimap A \multimap A \multimap A$ we define $[\![\mathsf{it}^A_{T(X)}(g,k)]\!] := It_{T(X)}([\![g]\!], [\![k]\!])$.

Given the construction of the semantics $[\![\cdot]\!]$, it is immediate to check that if we have a definitional equality $t_1 = \ t_2 : \ A$ then $[\![t_1]\!] = [\![t_2]\!] \in [\![A]\!]$.

Bibliography

[Acz78] Peter Aczel. The type theoretic interpretation of constructive set theory. In *Logic colloquium*, volume 77, pages 55–66, 1978.

[Acz82] Peter Aczel. The type theoretic interpretation of constructive set theory: choice principles. In *Studies in Logic and the Foundations of Mathematics*, volume 110, pages 1–40. Elsevier, 1982.

[Acz86] Peter Aczel. The type theoretic interpretation of constructive set theory: inductive definitions. In *Studies in Logic and the Foundations of Mathematics*, volume 114, pages 17–49. Elsevier, 1986.

[Acz16] Peter Aczel. On type theory and the philosophy of mathematics. Slides of the talk given at the Leeds Logic Colloquium, 2016.

[AG00] Peter Aczel and Nicola Gambino. Collection principles in dependent type theory. In *International Workshop on Types for Proofs and Programs*, pages 1–23. Springer, 2000.

[AR10] Peter Aczel and Michael Rathjen. CST book draft, 2010.

[AS02] Klaus Aehlig and Helmut Schwichtenberg. A syntactical analysis of non-size-increasing polynomial time computation. *ACM Transactions on Computational Logic (TOCL)*, 3(3):383–401, 2002.

[Awo11] Steve Awodey. From sets to types, to categories, to sets. In *Foundational Theories of Classical and Constructive Mathematics*, pages 113–125. Springer, 2011.

[BC92] Stephen Bellantoni and Stephen Cook. A new recursion-theoretic characterization of the polytime functions. *Computational complexity*, 2(2):97–110, 1992.

[BS13] Samuel R. Buss and Philip J. Scott. *Feasible Mathematics: A Mathematical Sciences Institute Workshop, Ithaca, New York, June 1989*, volume 9. Springer Science & Business Media, 2013.

[Cas97] Vuokko-Helena Caseiro. Equations for defining poly-time functions. PhD Thesis, University of Oslo, 1997.

[CHS] Thierry Coquand, Peter Hancock, and Anton Setzer. Ordinals in type theory. `www.cse.chalmers.se/~coquand/ordinal.ps`. Slides of the talk at CSL'97, Aarhus.

[DLH05] Ugo Dal Lago and Martin Hofmann. Quantitative models and implicit complexity. In *International Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 189–200. Springer, 2005.

[Fef75] Solomon Feferman. A language and axioms for explicit mathematics. In *Algebra and logic*, pages 87–139. Springer, 1975.

[Fef88] Solomon Feferman. Hilbert's program relativized; proof-theoretical and foundational reductions. *The Journal of Symbolic Logic*, 53(2):364–384, 1988.

[GA06] Nicola Gambino and Peter Aczel. The generalised type-theoretic interpretation of constructive set theory. *The Journal of Symbolic Logic*, 71(1):67–103, 2006.

[Gir87] Jean-Yves Girard. *Proof theory and logical complexity*, volume 1. Bibliopolis, 1987.

[GSS92] Jean-Yves Girard, Andre Scedrov, and Philip J. Scott. Bounded linear logic: a modular approach to polynomial-time computability. *Theoretical computer science*, 97(1):1–66, 1992.

[Gyl16a] Håkon Robbestad Gylterud. From multisets to sets in hotmotopy type theory. *arXiv preprint arXiv:1612.05468*, 2016.

[Gyl16b] Håkon Robbestad Gylterud. Multisets in type theory. *arXiv preprint arXiv:1610.08027*, 2016.

[Hof02] Martin Hofmann. The strength of non-size increasing computation. *ACM SIGPLAN Notices*, 37(1):260–269, 2002.

[Hof03] Martin Hofmann. Linear types and non-size-increasing polynomial time computation. *Information and Computation*, 183(1):57–85, 2003.

[How70] William Alvin Howard. Assignment of ordinals to terms for primitive recursive functionals of finite type. In *Studies in Logic and the Foundations of Mathematics*, volume 60, pages 443–458. Elsevier, 1970.

[HS94] Martin Hofmann and Thomas Streicher. The groupoid model refutes uniqueness of identity proofs. In *Logic in Computer Science, 1994. LICS'94. Proceedings., Symposium on*, pages 208–212. IEEE, 1994.

[HS98] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. *Twenty-five years of constructive type theory (Venice, 1995)*, 36:83–111, 1998.

[Hub16] Simon Huber. Canonicity for cubical type theory. *arXiv preprint arXiv:1607.04156*, 2016.

[Laf04] Yves Lafont. Soft linear logic and polynomial time. *Theoretical Computer Science*, 318(1-2):163–180, 2004.

[Led14] Jérémy Ledent. Modeling set theory in homotopy type theory. Internship report, 2014.

[Lub05] Robert S. Lubarsky. Independence results around constructive ZF. *Annals of Pure and Applied Logic*, 132(2-3):209–225, 2005.

[Lum09] Peter LeFanu Lumsdaine. Weak ω-categories from intensional type theory. In *International Conference on Typed Lambda Calculi and Applications*, pages 172–187. Springer, 2009.

[Mai08] Maria Emilia Maietti. A minimalist two-level foundation for constructive mathematics. *arXiv preprint arXiv:0811.2774*, 2008.

[ML72] Per Martin-Löf. Infinite terms and a system of natural deduction. *Compositio Mathematica*, 24(1):93–103, 1972.

[ML75] Per Martin-Löf. An intuitionistic theory of types: Predicative part. In *Studies in Logic and the Foundations of Mathematics*, volume 80, pages 73–118. Elsevier, 1975.

[ML84] Per Martin-Löf. *Intuitionistic Type Theory. Notes by Giovanni Sambin of a series of lectures given in Padova 1980*. Bibliopolis, 1984.

[MP02] Ieke Moerdijk and Erik Palmgren. Type theories, toposes and constructive set theory: predicative aspects of AST. 2002.

[MS05] Maria Emilia Maietti and Giovanni Sambin. Toward a minimalist foundation for constructive mathematics. *From Sets and Types to Topology and Analysis: Practicable Foundations for Constructive Mathematics*, 48:91–114, 2005.

[Myh75] John Myhill. Constructive set theory. *The Journal of Symbolic Logic*, 40(3):347–382, 1975.

[NPS90] Bengt Nordström, Kent Petersson, and Jan M Smith. *Programming in Martin-Löf's type theory*, volume 200. Oxford University Press Oxford, 1990.

[PL15] Fedor Part and Zhaohui Luo. Semi-simplicial types in logic-enriched homotopy type theory. *arXiv preprint arXiv:1506.04998*, 2015.

[Poh09] Wolfram Pohlers. *Proof theory: an introduction*, volume 1407 of *Lecture Notes in Mathematics*. Springer, 2009.

[PW14] Erik Palmgren and Olov Wilander. Constructing categories and setoids of setoids in type theory. *arXiv preprint arXiv:1408.1364*, 2014.

[Rat99] Michael Rathjen. The realm of ordinal analysis. *London Mathematical Society Lecture Note Series*, 258:219–280, 1999.

[Rat06] Michael Rathjen. The formulae-as-classes interpretation of constructive set theory. *Proof Technology and Computation (IOS Press, Amsterdam, 2006)*, pages 279–322, 2006.

[RS15] Egbert Rijke and Bas Spitters. Sets in homotopy type theory. *Mathematical Structures in Computer Science*, 25(5):1172–1202, 2015.

[RT06] Michael Rathjen and Sergei Tupailo. Characterizing the interpretation of set theory in Martin-Löf type theory. *Annals of Pure and Applied Logic*, 141(3):442–471, 2006.

[Sim00] Harold Simmons. *Derivation and Computation: taking the Curry-Howard correspondence seriously*, volume 51. Cambridge University Press, 2000.

[Sim09] Stephen George Simpson. *Subsystems of second order arithmetic*, volume 1. Cambridge University Press, 2009.

[SU06] Morten Heine Sørensen and Pawel Urzyczyn. *Lectures on the Curry-Howard isomorphism*. Elsevier, 2006.

[SW11] Helmut Schwichtenberg and Stanley Wainer. *Proofs and computations*. Cambridge University Press, 2011.

[Tai65] William Tait. Infinitely long terms of transfinite type. In *Studies in Logic and the Foundations of Mathematics*, volume 40, pages 176–185. Elsevier, 1965.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. `https://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[Vod97] Paul J. Voda. A simple ordinal recursive normalization of Gödel's T. In *International Workshop on Computer Science Logic*, pages 491–509. Springer, 1997.

[Voe15] Vladimir Voevodsky. An experimental library of formalized mathematics based on the univalent foundations. *Mathematical Structures in Computer Science*, 25(5):1278–1294, 2015.

[Wei98] Andreas Weiermann. How is it that infinitary methods can be applied to finitary mathematics? Gödel's T: a case study. *The Journal of Symbolic Logic*, 63(4):1348–1370, 1998.