Hereditarily Finite Sets in Constructive Type Theory

Gert Smolka and Kathrin Stark

Saarland University
May 29, 2016

To appear in Proc. of ITP 2016, Nancy, France, Springer LNCS

We axiomatize hereditarily finite sets in constructive type theory and show that all models of the axiomatization are isomorphic. The axiomatization takes the empty set and adjunction as primitives and comes with a strong induction principle. Based on the axiomatization, we construct the set operations of ZF and develop the basic theory of finite ordinals and cardinality. We construct a model of the axiomatization as a quotient of an inductive type of binary trees. The development is carried out in Coq.

1 Introduction

An HF set (hereditarily finite set) is a finite and well-founded set whose elements are HF sets. The class of HF sets may be defined inductively:

- The empty set is an HF set.
- If x and y are HF sets, then $\{x\} \cup y$ is an HF set.

We call the operation $x.y := \{x\} \cup y$ adjunction. Set membership can be expressed with adjunction and equality: $x \in y \leftrightarrow x.y = y$. Ackermann [1] discovered that the natural numbers are in one-to-one correspondence with the HF sets, and that the class of HF sets satisfies all axioms of ZF set theory but infinity.

We present an axiomatization of HF sets in a constructive type theory without inductive types and obtain the following results:

- All models of the axiomatization are isomorphic.
- The usual set operations, including separation, replacement, union, power, and transitive closure, can be constructed.
- A cardinality operation mapping sets to equipotent ordinals can be constructed.

- A model of the axiomatization can be constructed as a quotient of an inductive type of binary trees.

Our axiomatization of HF sets assumes a type X of sets and constants for the empty set and adjunction. There are four basic axioms

- $x.(x.y) = x.y$
- $x.(y.z) = y.(x.z)$
- $x.y \neq \emptyset$
- $x.(y.z) = y.z \;\to\; x = y \;\vee\; x.z = z$

and a strong induction principle:

- $\forall p : X \to \mathsf{Type}.\;\; p\emptyset \to (\forall xy.\; px \to py \to p(x.y)) \to \forall x.\, px$

We speak of a strong induction principle since it applies to functions into Type rather than just functions into Prop (i.e., predicates). The strong induction principle provides for the recursive definition of functions and ensures that all sets are finite and well-founded. In contrast to recursors for inductive types, the induction principle does not come with equations.

Related work. Several axiomatizations of hereditarily finite sets appear in the literature: Takahashi 1977 [8], Givant and Tarski 1977 [2], Previale 1994 [6], Świerczkowski 2003 [7], and Kirby 2009 [3]. All of them are formulated as first-order theories, and all of them employ the empty set, adjunction, and an adjunction-based induction principle as ingredients. Except for Kirby's axiomatization, which uses no additional constant, the existing axiomatizations are formulated with an additional constant for set membership. Except for Previale's axiomatization, which is studied in an intuitionistic setting, the existing axiomatizations are studied in a classical setting. Previale's axiomatization employs both membership and its transitive closure as additional constants. Previale derives the decidability of equality and membership. Our axiomatization extends Kirby's axiomatization by strengthening the induction principle to types.

A type of HF sets is available in Isabelle/HOL. It is realized with Ackermann's [1] encoding. Paulson [4, 5] makes essential use of the type of HF sets in his formalizations of finite automata and Gödel's incompleteness theorems in Isabelle/HOL. Interestingly, Isabelle/HOL can also define a type of HF sets by recursion through a type constructor for finite sets over a given base type.

Contribution of the paper. The paper explores for the first time an axiomatization of HF sets in constructive type theory. We show that the axiomatization is categorical, a result that has not been shown before for any of the existing axiomatizations. Our axiomatization extends Kirby's axiomatization by strengthening the induction principle from predicates to general functions.

We construct a model of the axiomatization as a quotient of an inductive type of binary trees. This natural model construction (from the perspective of constructive type theory) does not appear in the literature. It complements a conventional model construction based on numbers and Ackermann's encoding. To obtain the quotient with minimal assumptions, we base the construction on a normalizing sorting function for the lexical tree ordering.

Organization of the paper. We start with a section recalling the underlying type theory and basic notions like decidability. We also recall how quotients can be obtained as subtypes based on normalizers. We then introduce HF structures and establish basic results including extensionality, decidability, and strong epsilon induction. In Section 4 we show that all HF structures are isomorphic. We then construct the basic set operations using the strong induction principle and membership-based specifications. In Sections 6 and 7 we consider ordinals and define equipotence of sets. Based on an inductively defined cardinality relation, we show that every equipotence class contains exactly one ordinal, and that equipotence is a decidable equivalence relation. We use the strong induction principle to construct a cardinality operator. In Section 8 we extend the underlying type theory with an inductive type of binary trees and show that every HF structure is a quotient of the tree type. In Section 9 we define tree equivalence and construct a normalizing sorting function for the lexical tree order. Based on the sorting function, we obtain an HF structure, thus showing consistency of the axiomatization of HF sets.

Accompanying Coq development. The development of the paper is formalized in Coq. The Coq development is available at `http://www.ps.uni-saarland.de/extras/hfs` and contains additional results that for space reasons could not be included in the paper.

2 Preliminaries

We assume a constructive type theory with dependent function types, dependent pair types, sum types, and an impredicative universe Prop of propositions. We do not use inductive types except for the model construction in Section 9, which requires an inductive type of binary trees and an inductive proposition $\top$ with exactly one proof.

We will frequently use inductively defined predicates, which are always obtained as impredicatively defined intersection predicates. The logical operations and the equality predicate are also defined impredicatively.

We write $P \barwedge Q$ for strong disjunctions (sums $P+Q$ in Coq) and $\bar{\exists} x.px$ for strong existentials (sigT p in Coq). A proposition P is **decidable** if $P \barwedge \neg P$.

A **decidable predicate** on a type X is a pair consisting of a predicate $p : X \to \mathrm{Prop}$ and a function $\forall x.\, px \veebar \neg px$. Decidable binary predicates are defined analogously.

A **discrete type** is a pair of a type X and a function $\forall xy.\, x = y \veebar x \neq y$.

In Section 9 we will construct a quotient of an inductive tree type. The quotient will be obtained as a subtype of the tree type consisting of the fixed points of a normalizer for the underlying equivalence relation. The quotient construction will be an instance of the abstract subtype construction described in the following.

We assume a proposition $\top$ such that $\forall AB : \top.\, A = B$.

A predicate $p : X \to \mathsf{Prop}$ is **pure** if $\forall x \forall AB : px.\, A = B$.

Fact 1 For every decidable predicate there is an equivalent pure predicate.

Proof Let p be a decidable predicate. Then $\lambda x.\, \mathsf{if}\ px\ \mathsf{then}\ \top\ \mathsf{else}\ \bot$ is an equivalent pure predicate ($\bot := \forall P : \mathrm{Prop}.\, P$ is falsity). ∎

Fact 2 (Subtype) Let f be an idempotent function on a discrete type A. Then there are a discrete type X and functions $S : A \to X$ and $I : X \to A$ such that $S(Ix) = x$ and $I(Sa) = fa$ for all x and all a.

Proof The assumptions suffice to construct a pure predicate p such that $pa \leftrightarrow fa = a$ and a function $F : \forall a.\, p(fa)$. We define $X := \exists a.pa$, $Sa := (fa, Fa)$, and $I(a, \phi) := a$. ∎

We may see the type X established by Fact 2 in several ways:

1. X is the subtype of A consisting of all fixed points of f.
2. X is the subtype of A consisting of all points in the range of f.
3. X is the quotient of A under the equivalence relation $\lambda ab.\, fa{=}fb$ induced by f.

A **normalizer** for a relation $\sim$ on a type X is an idempotent function $f : X \to X$ such that $x \sim y \leftrightarrow fx = fy$ for all x and y. Obviously, a relation is an equivalence relation if it has a normalizer. Moreover, a relation on a discrete type is decidable if it has a normalizer.

3 HF Structures

We axiomatize HF sets with a type of sets, a constant $\emptyset$ for the empty set, and a binary operation $x.y$ on sets we call adjunction. Informally, $x.y$ is the set $\{x\} \cup y$.

Formally, an **HF structure** consists of the following:

- A type X. The elements of X are called **sets**.
- A set $\emptyset$ called **empty set**.

- A function $X \to X \to X$ called **adjunction**. We write $x.y$ for the adjunction of two sets x and y.
- A function $\forall p : X \to \mathsf{Type}.\ \ p\emptyset \to (\forall xy.\ px \to py \to p(x.y)) \to \forall x.\, px$ called **strong induction principle**.
- The following laws:
  - $x.(x.y) = x.y$ **cancellation law**
  - $x.(y.z) = y.(x.z)$ **swap law**
  - $x.y \neq \emptyset$ **discrimination law**
  - $x.(y.z) = y.z \ \to\ x = y \ \vee\ x.z = z$ **membership law**

We write $x.y.z$ for $x.(y.z)$.

Given an HF structure, we define **membership** and **inclusion**:

$$x \in y \ :=\ (x.y = y)$$
$$x \subseteq y \ :=\ \forall z.\, z \in x \to z \in y$$

Using the notation for membership, we may write the membership law more suggestively as $x \in y.z \to x = y \vee x \in z$.

Example 3 We prove $(\emptyset.\emptyset).\emptyset \neq \emptyset.\emptyset$. Suppose $A : (\emptyset.\emptyset).\emptyset = \emptyset.\emptyset$. Cancellation gives us $(\emptyset.\emptyset).(\emptyset.\emptyset).\emptyset = \emptyset.\emptyset$. Thus $\emptyset.\emptyset \in \emptyset.\emptyset$ using A. Thus either $\emptyset.\emptyset = \emptyset$ or $\emptyset.\emptyset \in \emptyset$ with the membership law. In either case we have a contradiction by the discrimination law.

We assume an HF structure X and use the letters x, y, z, a, and b to denote sets in X.

Fact 4 (Decomposition) $x = \emptyset \ \bar{\vee}\ \bar{\exists} a \bar{\exists} y.\ x = a.y$.

Proof Immediate consequence of the strong induction principle. ∎

Fact 5

1. $z \notin \emptyset$.
2. $z \in x.y \ \leftrightarrow\ z = x \vee z \in y$.
3. $x.y \subseteq z \ \leftrightarrow\ x \in z \ \wedge\ y \subseteq z$.
4. $x \subseteq \emptyset \leftrightarrow x = \emptyset$.
5. $a \notin x \to x \subseteq a.y \to x \subseteq y$.

Proof Straightforward. ∎

The sets of an HF structure are extensional in that two sets are equal if they have the same elements. Proving this basic fact constructively is not straightforward. We

employ a nested HF induction and interleave the extensionality proof with proofs of the decidability of membership, inclusion, and equality of HF sets. The proof is organized in three lemmas.

Lemma 6

1. $\emptyset \subseteq x$ and $x \subseteq \emptyset$ and $x \in \emptyset$ and $x = \emptyset$ are decidable.
2. If $x = a$ and $x \in y$ are decidable, $x \in a.y$ is decidable.
3. If $a \in y$ and $x \subseteq y$ are decidable, $a.x \subseteq y$ is decidable.
4. $\emptyset \in x$ is decidable.

Proof Claim (1) follows with Fact 4. Claims (2) and (3) follow with Claims (2) and (3) of Fact 5. Claim (4) follows by induction on x using Claims (2) and (1). ∎

Lemma 7 (Partition) Let $a \in x$. Then there strongly exists a set u such that $x = a.u$ and $a \notin u$, provided the propositions $a \in z$ and $a = z$ are decidable for all sets z.

Proof By induction on x. The case $x = \emptyset$ is contradictory. Let $x = b.x$. By assumption, $a \in x$ is decidable. If $a \notin x$, the claim follows with $u = x$ and Fact 5 (2). Otherwise, let $a \in x$. By the inductive hypothesis we have a set u such that $x = a.u$ and $a \notin u$. By assumption, $a = b$ is decidable. If $a = b$, the claim follows with $u = x$. If $a \neq b$, the claim follows with $u = b.x$. ∎

Lemma 8 For all sets x and y:

1. $x \subseteq y$ and $y \subseteq x$ are decidable.
2. $x \in y$ and $y \in x$ are decidable.
3. $x \subseteq y \to y \subseteq x \to x = y$.
4. $x = y$ is decidable.

Proof We prove the claims simultaneously by nested induction on x and y. If $x = \emptyset$ or $y = \emptyset$, the claims follow with Lemma 6. Otherwise, we have $x = a.x$ and $y = b.y$ and inductive hypotheses for a, x, b, and y.

1. $a.x \subseteq b.y$ and $b.y \subseteq a.x$ are decidable. Follows by Lemma 6 (3) and the inductive hypotheses for a, x, b, and y.
2. $a.x \in b.y$ and $b.y \in a.x$ are decidable. Follows by Lemma 6 (2) and the inductive hypotheses for b, y, a, and x.
3. $a.x \subseteq b.y \to b.y \subseteq a.x \to a.x = b.y$. Let $a.x \subseteq b.y$ and $b.y \subseteq a.x$. We show $a.x = b.y$. By the inductive hypothesis for x we know that $a \in x$ is decidable. Case analysis.

a) $a \in x$. Then $a.x = x$ and the claim follows by Claim (3) of the inductive hypothesis for x.

b) $a \notin x$. We have $a \in b.y$. By Lemma 7 we have a set u such that $b.y = a.u$ and $a \notin u$. Thus it suffices to show $x = u$, which follows by Claim (3) of the inductive hypothesis for x provided we have $x \subseteq u$ and $u \subseteq x$. The two inclusions hold since $a.x \subseteq a.u$, $a \notin x$, $a.u \subseteq a.x$, and $a \notin u$.

4. $a.x = b.y$ is decidable. Case analysis based on (1).

a) $a.x \subseteq b.y$ and $b.y \subseteq a.x$. Then $a.x = b.y$ by (3).

b) $a.x \nsubseteq b.y$ or $b.y \nsubseteq a.x$. Then $a.x \neq b.y$. ∎

Theorem 9 (Extensionality) $(\forall z.\, z \in x \leftrightarrow z \in y) \to x = y$.

Proof Follows with Lemma 8 (3). ∎

Corollary 10 Set inclusion is a partial ordering on sets.

Theorem 11 (Decidability) Equality, membership, and inclusion of HF sets are decidable.

Proof Follows with Lemma 8. ∎

Fact 12 (Decidability) The propositions $\exists z.\, z \in x \wedge pz$ and $\forall z.\, z \in x \to pz$ are decidable if p is a decidable predicate.

Proof Follows by induction on x. ∎

Fact 13 (Partition) $a \in x \to \bar{\exists} u.\, x = a.u \wedge a \notin u$.

Proof Follows with Lemmas 7 and 8. ∎

Fact 14 (Strong Epsilon Induction)
$\forall p : X \to \mathsf{Type}.\ (\forall x.\, (\forall z \in x.\, pz) \to px) \to \forall x.\, px$.

Proof Assume $p : X \to \mathsf{Type}$ and $A : \forall x.\, (\forall z \in x.\, pz) \to px$. By A it suffices to prove $\forall z \in x.\, pz$. We prove this claim by induction on x. The case for $x = \emptyset$ is obvious. Let $x = a.x'$ and $z \in a.x'$. It suffices to show pz. If $z = a$, then pa follows by A and the inductive hypothesis for a. Otherwise, $z \in x'$ and the claim follows by the inductive hypothesis for x'. ∎

Fact 15 $x \notin x$.

Proof By ϵ-induction we have $z \notin z$ for all $z \in x$. The claim follows. ∎

Corollary 16 There is no set that contains all sets.

Fact 17 $x \in y \to y \notin x$.

Proof By ϵ-induction. ∎

The operation $\lambda x.x.x$ of self-adjunction is known as **successor operation**. The successor of x contains the elements of x plus one additional element, which is x itself. That x is in fact a new element is asserted by Fact 15.

The successor operation is injective.

Fact 18 (Successor Injectivity) $x.x = y.y \to x = y$.

Proof Let $x.x = y.y$. Then $x \in y.y$ and $y \in x.x$. By the membership law, we have either $x = y$ or $y = x$ or $x \in y \in x$. The third case is impossible by Fact 17. ∎

4 Categoricity

We show that all HF structures are isomorphic. In fact, given two HF structures X and Y, there is exactly one homomorphism from X to Y. We obtain the homomorphism with the recursion principle from an inductively defined relational version of the homomorphism.

We start with the definition of an inductive predicate[1]

$$R : \forall X\, Y : \mathrm{HF}.\ X \to Y \to \mathsf{Prop}$$

homomorphically relating the sets of two HF structures:

$$\frac{}{R\emptyset\emptyset} \qquad\qquad \frac{Rab \qquad Rxy}{R(a.x)(b.y)}$$

Given HF structures X and Y, we will show that R_{XY} is a bijection.

Fact 19 (Symmetry) $R_{XY}\, xy \to R_{YX}\, yx$.

Proof By induction on Rxy. ∎

Fact 20 (Strong Totality) $\forall x \exists y.\ Rxy$.

Proof By induction on x. ∎

[1] An impredicative definition of R looks as follows: $\lambda(XY : \mathrm{HF})(x : X)(y : Y)$.
$\forall S : X \to Y \to \mathsf{Prop}.\ S\emptyset\emptyset \to (\forall axby.\ Sab \to Sxy \to S(a.x)(b.y)) \to Sxy$.

Proving that R is functional requires some effort. The key ingredients are extensionality and a simulation lemma for membership.

Lemma 21 (Simulation) $Rxy \to a \in x \to \exists b.\, b \in y \wedge Rab$.

Proof By induction on Rxy. The case for the first rule is trivial. For the second rule, we have $Ra'b'$, $Rx'y'$, and $a \in a'.x'$ and need a set $b \in b'.y'$ such that Rab. If $a = a'$, $b := b'$ does the job. Otherwise, we have $a \in x'$. By the inductive hypothesis we obtain $b \in y'$ with Rab. The claim follows since $y' \subseteq b'.y'$. ∎

Fact 22 (Functionality) $Rxy \to Rxy' \to y = y'$.

Proof By ϵ-induction on x. We show $y = y'$ using extensionality (Theorem 9). Let $b \in y$. We show $b \in y'$. By the facts for symmetry and simulation we obtain an $a \in x$ such that Rab. By the simulation lemma we obtain $b' \in y'$ such that Rab'. By the inductive hypothesis we have $b = b'$. Thus $b \in y'$. We now have $y \subseteq y'$. The other direction $y' \subseteq y$ follows analogously. ∎

A **homomorphism** from an HF structure X to an HF structure Y is a function $f : X \to Y$ such that $f\emptyset = \emptyset$, and $f(a.x) = fa.fx$ for all a and x. Two HF structures X and Y are **isomorphic** if there are homomorphisms $f : X \to Y$ and $g : Y \to X$ such that $g(fx) = x$ and $f(gy) = y$ for all x and y.

Fact 23 Let f be a homomorphism from an HF structure X to an HF structure Y. Then $Rx(fx)$ for all x.

Proof By induction on x. ∎

Fact 24 All homomorphisms between two HF structures are equivalent.

Proof Follows with Facts 23 and 22. ∎

Theorem 25 (Categoricity) All HF structures are isomorphic.

Proof Follows with Facts 20, 19, and 22. ∎

5 Set Operations

We now construct basic set operations known from ZF for HF structures using the strong induction principle and preexisting specifications of the desired operations. The specifications are needed since the induction principle does not come with equations (in contrast to a full recursor). Suitable specifications for the basic set operations are easily obtained using to the extensionality property of sets.

Fact 26 (Binary Union) There is a function $x \cup y$ from sets to sets as follows:

1. $z \in x \cup y \leftrightarrow z \in x \vee z \in y$.
2. $\emptyset \cup y = y$.
3. $(a.x) \cup y = a.(x \cup y)$.

Proof We fix y and define $Uxu := \forall z.\ z \in u \leftrightarrow z \in x \vee z \in y$. We construct a function $F : \forall x \bar{\exists} u.\ Uxu$ using the strong induction principle. The base case follows with $U\emptyset y$. For the adjunction case it suffices to prove $\forall axu.\ Uxu \to U(a.x)(a.u)$, which is straightforward. We define $x \cup y := \pi_1(Fx)$. We have $Ux(\pi_1(Fx))$ and thus Claim 1. Claims 2 and 3 follow with extensionality from Claim 1.

Note that the two cases of the inductive construction of F choose their witnesses according to Claims 2 and 3. This is by design. Claims 2 and 3 explicate the ideas behind the construction of F. ∎

The following constructions all follow the scheme used for binary union.

Fact 27 (Big Union) There is a function $\bigcup x$ from sets to sets as follows:

1. $z \in \bigcup x \leftrightarrow \exists y \in x.\ z \in y$.
2. $\bigcup \emptyset = \emptyset$.
3. $\bigcup (a.x) = a \cup \bigcup x$.

A set x is **transitive** if every element of x is a subset of x. For transitive sets, big union undoes the successor operation.

Fact 28 (Predecessor) Let x be transitive. Then $\bigcup (x.x) = x$.

Fact 29 (Separation) For every decidable predicate p on sets there is a function $x \,|\, p$ from sets to sets as follows:

1. $z \in x \,|\, p \leftrightarrow z \in x \wedge pz$.
2. $\emptyset \,|\, p = \emptyset$.
3. $(a.x) \,|\, p = \mathsf{if}\ pa\ \mathsf{then}\ a.(x \,|\, p)\ \mathsf{else}\ x \,|\, p$.

Constructions and correctness proofs for the remaining set operations of ZF can be found in the accompanying Coq development, which also covers a transitive closure operation.

6 Ordinals

We define the class of **ordinals** as an inductive predicate on sets:

$$\frac{}{\mathcal{O}\,\emptyset} \qquad\qquad \frac{\mathcal{O}x}{\mathcal{O}(x.x)}$$

The ordinals represent the natural numbers as HF sets, where a number n is represented as the unique ordinal having n elements. The ordinal for n can be obtained by applying the successor function n-times to the empty set.

We use the letters α and β for sets that should be thought of as ordinals.

Fact 30 (Transitivity) Ordinals are transitive sets whose elements are ordinals.

Proof By induction on $\mathcal{O}\alpha$. ∎

Fact 31 (Empty Ordinal) Let α be an ordinal. Then $\alpha = \emptyset \;\bar{\vee}\; \emptyset \in \alpha$.

Proof Show $\alpha \neq \emptyset \to \emptyset \in \alpha$ by induction on $\mathcal{O}\alpha$. ∎

Fact 32 (Predecessor Ordinal) Let α be an ordinal. Then $\bigcup\alpha$ is an ordinal. Moreover, $\alpha = (\bigcup\alpha).(\bigcup\alpha)$ if $\alpha \neq \emptyset$.

Proof Both claims follow by induction on $\mathcal{O}\alpha$ using Facts 28 and 30. ∎

Fact 33 (Inversion) Let α be an ordinal. Then $\alpha = \emptyset \;\bar{\vee}\; \bar{\exists}\gamma.\, \mathcal{O}\gamma \wedge \alpha = \gamma.\gamma$.

Proof Follows with Fact 32. ∎

Fact 34 (Strong Ordinal Induction)
$\forall p : X \to \mathsf{Type}.\;\; p\emptyset \to (\forall\alpha.\, \mathcal{O}\alpha \to p\alpha \to p(\alpha.\alpha)) \to \forall\alpha.\, \mathcal{O}\alpha \to p\alpha$.

Proof Follows by strong epsilon induction (Fact 14) using Fact 33. ∎

7 Cardinality

Given an HF structure, we would expect that we can construct a model of the natural numbers by taking the subtype of the ordinals as type for the numbers. In constructive type theory, however, the subtype of ordinals does not come for free. We need an idempotent function on sets whose fixed points are the ordinals. A natural choice for this function is a cardinality function mapping every set to the unique equipotent ordinal. Two sets are equipotent if they have same number of elements.

We define **equipotence of sets** with an inductive predicate $x \sim y$:

$$\frac{}{\emptyset \sim \emptyset} \qquad\qquad \frac{a \notin x \qquad b \notin y \qquad x \sim y}{a.x \sim b.y}$$

Our definition of equipotence is tuned for finite sets. From the definition of equipotence it is not obvious that equipotence is an equivalence relation.

We will construct a function Γ from sets to sets such that Γx is the unique ordinal equipotent to x. Similar to what we did in the section on categoricity, we obtain Γ with the strong induction principle from an inductively defined predicate $Cx\alpha$:

$$\frac{}{C\emptyset\emptyset} \qquad\qquad \frac{a \notin x \qquad Cx\alpha}{C(a.x)(\alpha.\alpha)}$$

We will show that the relation C is strongly total and functional. Γ will be defined as the function accompanying C.

Fact 35 (Soundness) Let Cxy. Then $x \sim y$ and y is an ordinal.

Proof By induction on Cxy using Fact 15. ∎

Fact 36 (Strong Totality) $\forall x \dot{\exists} \alpha.\ Cx\alpha$.

Proof By induction on x. ∎

Fact 37 (Idempotence) Let α be an ordinal. Then $C\alpha\alpha$.

Proof By induction on $\mathcal{O}\alpha$ using Fact 15. ∎

Sets related to the same ordinal by C are equipotent.

Fact 38 (Injectivity) $Cx\alpha \to Cy\alpha \to x \sim y$.

Proof By induction on $Cx\alpha$. The case for the first rule is straightforward. For the second rule we have $x = a.x'$, $a \notin x'$, $Cx'\alpha'$, and $\alpha = \alpha'.\alpha'$. By inversion of $Cy\alpha$ we obtain b, y', and β such that $y = b.y'$, $b \notin y'$, $\alpha = \beta.\beta$, and $Cy'\beta$. By Fact 18 we have $\alpha' = \beta$. Thus $x' \sim y'$ by the inductive hypothesis for $Cx'\alpha'$. Hence $x \sim y$. ∎

Proving that C is functional takes effort. We need an inversion lemma whose proof requires a further lemma involving an instance of separation (Fact 29). We use the notation $x \div y := x\,|\,(\lambda z.z \neq y)$ (read x without y).

Lemma 39 $Cx\alpha \to a \in x \to \exists \beta.\ \alpha = \beta.\beta \wedge C(x \div a)\beta$.

Proof By induction on $Cx\alpha$. The case for the first rule is straightforward. For the second rule we have $x = a'.x'$, $a' \notin x'$, $Cx'\alpha'$, and either $a = a'$ or $a \in x'$. It suffices to show that $C((a'.x') \div a)\alpha'$.

Let $a = a'$. Then the claim follows with $(a'.x') \div a = x'$.

Let $a \neq a'$ and $a \in x'$. The inductive hypothesis for $Cx'\alpha'$ gives us some β such that $\alpha' = \beta.\beta$ and $C(x' \div a)\beta$. The claim follows since $(a'.x') \div a = a'.(x' \div a)$ and $a' \notin (x' \div a)$. ∎

Fact 40 (Inversion) $a \notin x \to C(a.x)\alpha \to \exists\beta.\ \alpha = \beta.\beta \wedge C x \beta$.

Proof Follows with Lemma 39. ∎

Fact 41 (Functionality) $Cx\alpha \to Cx\beta \to \alpha = \beta$.

Proof By induction on $Cx\alpha$. The case for the first rule is straightforward, and the case for the second rule follows with Fact 40 and the inductive hypothesis. ∎

Fact 42 (Invariance) $x \sim y \to Cx\alpha \to Cy\alpha$.

Proof By induction on $x \sim y$ using Fact 40. ∎

Fact 43 (Canonicity) Equipotent ordinals are equal.

Proof Let α and β be equipotent ordinals. Then $C\alpha\alpha$ and $C\beta\beta$ by Fact 37. Thus $C\beta\alpha$ by Fact 42, and $\alpha = \beta$ by Fact 41. ∎

We define Γ as $\Gamma x := \pi_1(Tx)$ where T is the function established by Fact 36.

Fact 44 Γ is an idempotent function such that $Cx(\Gamma x)$, $x \sim \Gamma x$, and Γx is an ordinal for every set x.

Proof $Cx(\Gamma x)$ holds by definition for all x. Thus $x \sim \Gamma x$ and Γx is an ordinal by Fact 35.

To show the idempotence of Γ, we fix some x. We have $C(\Gamma x)(\Gamma(\Gamma x))$. Since Γx is an ordinal, we also have $C(\Gamma x)(\Gamma x)$ by Fact 37. Hence $\Gamma(\Gamma x) = \Gamma x$ by Fact 41. ∎

Fact 45 (Coincidence) $x \sim y \leftrightarrow \Gamma x = \Gamma y$.

Proof Let $x \sim y$. We have $Cx(\Gamma x)$ and $Cy(\Gamma y)$ by Fact 44. Hence $Cy(\Gamma x)$ by Fact 42. Thus $\Gamma x = \Gamma y$ by Fact 41.

Let $\Gamma x = \Gamma y$. Then $x \sim y$ by Facts 38 and 44. ∎

Corollary 46 (Equipotence) Equipotence is a decidable equivalence relation.

Fact 47 (Fixed Point) A set x is an ordinal if and only if $\Gamma x = x$.

Proof Let α be an ordinal. Then $C\alpha\alpha$ by Fact 37 and $C\alpha(\Gamma\alpha)$ by Fact 44. Thus $\Gamma\alpha = \alpha$ by Fact 41. The other direction follows by Fact 44. ∎

Corollary 48 It is decidable whether a set is an ordinal.

8 Binary Trees

We now strengthen the type theory by adding an inductive type $\mathbf{T}$ of binary trees:

$$\mathbf{T} \;:=\; 0 \mid \mathbf{T}.\mathbf{T}$$

We will construct an HF structure in the strengthened type theory.

The letters s, t, and u will range over binary trees. We write $s.t.u$ for $s.(t.u)$.

Fact 49 $\mathbf{T}$ is a discrete type.

Proof We obtain a decision function $\forall st.\ s = t \;\bar{\vee}\; s \neq t$ by induction on s using the strong induction principle for trees. ∎

We assume an HF structure X and define a function $S : \mathbf{T} \to X$ mapping trees to sets:

$$\begin{aligned} S0 &:= \emptyset \\ S(s.t) &:= Ss.St \end{aligned}$$

We may see trees as expressions describing sets and S as a function evaluating expressions to sets.

Fact 50 S has a right inverse. That is, there is a function $I : X \to \mathbf{T}$ such that $S(Ix) = x$ for every set x. Consequently, we have $x = y \leftrightarrow Ix = Iy$ for all sets x and y.

Proof With the strong induction principle of X we obtain a certifying function $F : \forall x \bar{\exists} s.\ Ss = x$. We define $Ix := \pi_1(Fx)$. ∎

Lemma 51 (Transfer of Induction Principle) Let X be a structure that is an HF structures except that it does not come with an induction principle. Let $S : \mathbf{T} \to X$ and $I : X \to \mathbf{T}$ be functions such that $S(Ix) = x$, $S0 = \emptyset$, and $S(s.t) = Ss.St$ for all sets x and all trees s and t. Then X can be extended to an HF structure.

Proof Let $p : X \to \mathsf{Prop}$. The strong induction principle for X and p can be obtained from the strong induction principle for $\mathbf{T}$ and $\lambda s.p(Ss)$. ∎

9 Tree Model

We now construct an HF structure as a quotient of the tree type under an equivalence generated by cancellation and swapping. We define this equivalence as an

inductive predicate $s \approx t$ and call it **tree equivalence**:

$$\frac{}{s.s.t \approx s.t} \qquad \frac{}{s.t.u \approx t.s.u} \qquad \frac{s \approx s' \qquad t \approx t'}{s.t \approx s'.t'}$$

$$\frac{}{s \approx s} \qquad \frac{s \approx t}{t \approx s} \qquad \frac{s \approx t \qquad t \approx u}{s \approx u}$$

Tree equivalence satisfies the cancellation and swapping law by definition. It also satisfies the discrimination law.

Fact 52 $s \approx t \to (s = 0 \leftrightarrow t = 0)$.

Proof By induction on $s \approx t$. We prove $s = 0 \to t = 0$ and $t = 0 \to s = 0$ together so that we can accommodate the symmetry rule. ∎

Fact 53 (Discrimination) $s.t \not\approx 0$.

Proof Follows with Fact 52. ∎

We define $s \in t := (s.t \approx t)$. Proving that tree equivalence satisfies the membership law takes a little effort. We need an inductive auxiliary predicate $s \dot\in t$ providing a restricted form of membership:

$$\frac{}{s \dot\in s.t} \qquad \frac{s \dot\in u}{s \dot\in t.u}$$

Fact 54 $u \dot\in s.t \leftrightarrow u = s \vee u \dot\in t$.

We also need an auxiliary inclusion predicate $s \preceq t := \forall u \dot\in s\, \exists v \dot\in t.\, u \approx v$.

Lemma 55 $s \approx t \to s \preceq t \wedge t \preceq s$.

Proof By induction on $s \approx t$ using Fact 54. We prove $s \preceq t$ and $t \preceq s$ together so that we can accommodate the symmetry rule. ∎

Lemma 56 $s \dot\in t \to s \in t$.

Proof By induction on $s \dot\in t$. ∎

Fact 57 (Membership) $u \in s.t \to u \approx s \vee u \in t$.

Proof Let $u \in s.t$. Then $u.s.t \approx s.t$. Thus $u.s.t \preceq s.t$ by Lemma 55. Since $u \dot\in u.s.t$, we have $u \approx v \dot\in s.t$ for some v. Thus either $u \approx v = s$ or $u \approx v \dot\in t$ by Fact 54. The claim follows with Lemma 56. ∎

The quotient of the tree type for tree equivalence will be obtained with a normalizer for tree equivalence using Fact 2. The normalizer will be an idempotent function $\sigma : \mathbf{T} \to \mathbf{T}$ such that $s \approx t \leftrightarrow \sigma s = \sigma t$. Given that tree equivalence is generated by cancellation and swapping, we can obtain a normalizer for tree equivalence as a sorting function for some linear ordering on trees. There is a natural linear ordering on trees based on the idea of lexical ordering:

$$\frac{}{0 < s.t} \qquad \frac{s < s'}{s.t < s'.t'} \qquad \frac{t < t'}{s.t < s.t'}$$

We speak of the **lexical tree ordering**.

Fact 58 The lexical tree ordering is irreflexive and transitive.

Proof Follows with induction on $s < t$. ∎

Fact 59 (Trichotomy) $s < t \;\bar{\vee}\; s = t \;\bar{\vee}\; t < s$.

Proof By nested induction on s and t. ∎

We shall obtain the normalizer by duplicate-eliminating insertion sort. We define a function $\alpha : \mathbf{T} \to \mathbf{T} \to \mathbf{T}$ for order-observing and duplicate-avoiding **insertion** based on the case analysis provided by Fact 59:

$$\begin{aligned} \alpha s 0 &:= s.0 \\ \alpha s (t.u) &:= \text{case } s < t \Rightarrow s.t.u \mid s = t \Rightarrow t.u \mid t < s \Rightarrow t.\alpha s u \end{aligned}$$

Fact 60 $\alpha s t \approx s.t$.

Proof By induction on t using Fact 59. ∎

We finally define a duplicate-eliminating **sorting** function $\sigma : \mathbf{T} \to \mathbf{T}$:

$$\begin{aligned} \sigma 0 &:= 0 \\ \sigma (s.t) &:= \alpha (\sigma s)(\sigma t) \end{aligned}$$

Fact 61 $\sigma s \approx s$.

Proof By induction on s using Fact 60. ∎

Next we show that that σ normalizes equivalent trees to identical trees. The key insight behind this result is the fact that insertion respects the cancellation and swap law with respect to equality.

Fact 62 $\alpha s(\alpha st) = \alpha st$ and $\alpha s(\alpha tu) = \alpha t(\alpha su)$.

Proof The first claim follows by induction on t and the second claim follows by induction on u. Both proofs do case analysis according to Fact 59 and eliminate inconsistent cases with Fact 58. There are many cases to consider. The following facts are useful for the case analysis:

- $\alpha s0 = s.0$ and $\alpha s(s.t) = s.t$.
- If $s < t$, then $\alpha s(t.u) = s.t.u$.
- If $t < s$, then $\alpha s(t.u) = t.\alpha su$. ∎

Fact 63 $s \approx t \to \sigma s = \sigma t$.

Proof By induction on $s \approx t$ using Fact 62. ∎

Fact 64

1. $s \approx t \leftrightarrow \sigma s = \sigma t$.
2. σ is idempotent; that is, $\sigma(\sigma s) = \sigma s$.
3. Tree equivalence is decidable.

Proof The claims follow with Facts 61 and 63. ∎

Theorem 65 (Model Existence) There exist an HF structure X and two functions $S : \mathbf{T} \to X$ and $I : X \to \mathbf{T}$ such that:

1. $S(Ix) = x$ and $I(Ss) \approx s$.
2. $Ss = St \leftrightarrow s \approx t$ and $Ix = Iy \leftrightarrow x = y$.
3. $S(s.t) = Ss.St$ and $I(x.y) \approx Ix.Iy$.
4. $S0 = \emptyset$ and $I\emptyset = 0$.

Proof By Facts 2, 49, and 64 we have a discrete type X and functions $S : \mathbf{T} \to X$ and $I : X \to \mathbf{T}$ such that $S(Ix) = x$ and $I(Ss) = \sigma s$ for all x and s. We define $\emptyset := S0$ and $x.y := S(Ix.Iy)$. The claims (1)–(4) follow with Fact 64. By Lemma 51 it suffices to show that the definitions of $\emptyset$ and adjunction satisfy the cancellation, swap, discrimination, and membership law.

We show the discrimination law. Let $x.y = \emptyset$. Then $S(Ix.Iy) = S0$ by definition. Thus $Ix.Iy \approx 0$ by (2). Contradiction by Fact 53.

We show the swap law. We have $Ix.Iy.Iz \approx Iy.Ix.Iz$ by definition of tree equivalence. Thus $Ix.I(y.z) \approx Iy.I(x.z)$ by (3) and $S(Ix.I(y.z)) = S(Iy.I(x.z))$ by (2). Hence $x.y.z = y.x.z$ by the definition of adjunction.

The cancellation law follows analogously.

We show the membership law. Let $x.y.z = y.z$. By the definition of adjunction and (2) we have $Ix.I(y.z) \approx Iy.Iz$. By (3) we have $Ix.Iy.Iz \approx Iy.Iz$. Hence either $Ix \approx Iy$ or $Ix.Iz \approx Iz$ by Fact 57. Thus either $x = y$ or $x.z = z$ by (2), (1), and the definition of adjunction. ∎

We can now transfer results for HF structures to tree equivalence.

Corollary 66 (Extensionality of Tree Equivalence)
$(\forall u.\, u \in s \leftrightarrow u \in t) \to s \approx t$.

Proof Let X, S, and I be the objects provided by Theorem 65. Then $s \in t \leftrightarrow Ss \in St$ for all s and t with (2) and (3). Suppose $\forall u.\, u \in s \leftrightarrow u \in t$. Then $\forall x.\, Ix \in s \leftrightarrow Ix \in t$. Thus $\forall x.\, x \in Ss \leftrightarrow x \in St$. Hence $Ss = St$ since X is extensional (Fact 9). Thus $s \approx t$. ∎

10 Conclusion

We have studied finite set theory in constructive type theory. In contrast to a general set theory, finite set theory has a unique model that can be constructed in constructive type theory. We have presented a categorial axiomatization of finite set theory providing for a constructive development of the theory, including the usual set operations, finite ordinals, and cardinality.

We have constructed a model of the axiomatization as a quotient of an inductive type of binary trees. The tree model gives us a natural realization of the type of finite sets in constructive type theory. The operations and results obtained on top of the axiomatization apply to the tree model and all other models. Seen from the perspective of programming, the axiomatization provides an abstraction layer.

We have been careful in spelling out the type theoretic resources needed for the development. For the study of the axiomatization, we work in a type theory with dependent function and pair types, with sum types, and with an impredicative universe of propositions. For the model construction, we add an inductive type of binary trees and a single proof proposition $\top$.

We see finite set theory as a constructive subtheory of general set theory. We believe that the study of finite sets in constructive type theory is instructive for students and also prepares them well for the study of general set theory.

There are many possibilities for future work: Prove that the axiomatization of HF sets is minimal; Find a recursor constructing functions on HF sets in the style of primitive recursion (step functions will have to be provided with admissibility proofs); Study the Peano axiomatization of numbers with strong induction and show that it enables the construction of a model of HF (following Ackermann [1]); Establish categorial axiomatizations for flat finite sets over a base type, for finite

multisets, and for finite sets also including non-wellfounded sets; develop a Coq library supporting the construction of the mentioned inductive quotient types.

The accompanying Coq development follows the presentation of the paper. We wrote a tactic supporting membership-based reasoning in HF structures. With this tactic the proofs of the abstract results turn out to be pleasantly compact. Unexpectedly, some of the proofs for tree sorting took effort because there are so many cases to consider (Lemma 55 and Fact 62). We arrived at compact proofs by devising special-purpose tactics.

Acknowledgement. Denis Müller contributed to the study of tree equivalence during his Bachelor's thesis project on finitary sets.

References

[1] Wilhelm Ackermann. Die Widerspruchsfreiheit der allgemeinen Mengenlehre. *Mathematische Annalen*, 114(1):305–315, 1937.

[2] Steven Givant and Alfred Tarski. Peano arithmetic and the Zermelo-like theory of sets with finite ranks. Abstract. *Notices of the American Mathematical Society*, 77T-E51:A–437, 1977.

[3] Laurence Kirby. Finitary set theory. *Notre Dame Journal of Formal Logic*, 50(3):227–244, 2009.

[4] Lawrence C. Paulson. A formalisation of finite automata using hereditarily finite sets. In *25th International Conference on Automated Deduction (CADE-25), Berlin, Germany*, volume 9195 of *LNCS*, pages 231–245. Springer, 2015.

[5] Lawrence C. Paulson. A mechanised proof of Gödel's incompleteness theorems using Nominal Isabelle. *Journal of Automated Reasoning*, 55(1):1–37, 2015.

[6] Flavio Previale. Induction and foundation in the theory of hereditarily finite sets. *Archive for Mathematical Logic*, 33(3):213–241, 1994.

[7] S. Świerczkowski. *Finite sets and Gödel's incompleteness theorems*, volume 422 of *Dissertationes Mathematicae*. Polish Academy of Sciences, Institute of Mathematics, 2003.

[8] Moto-o Takahashi. A foundation of finite mathematics. *Publications of the Research Institute for Mathematical Sciences, Kyoto University*, 12(3):577–708, 1977.