Cardinals in Isabelle/HOL

Jasmin Christian Blanchette[1], Andrei Popescu[1,2], and Dmitriy Traytel[1]

[1] Fakultät für Informatik, Technische Universität München, Germany
[2] Institute of Mathematics Simion Stoilow of the Romanian Academy, Bucharest, Romania

Abstract. We report on a formalization of ordinals and cardinals in Isabelle/HOL. A main challenge we faced is the inability of higher-order logic to represent ordinals canonically, as transitive sets (as done in set theory). We resolved this into a "decentralized" representation that identifies ordinals with wellorders, with all concepts and results proved to be invariant under order isomorphism. We also discuss two applications of this general theory in formal developments.

1 Introduction

Set theory is the traditional framework for ordinals and cardinals. Axiomatizations such as Zermelo–Fraenkel (ZF) and von Neumann–Bernays–Gödel (NBG) permit the definition of ordinals as transitive sets well-ordered by membership as the strict relation and by inclusion as the nonstrict counterpart. Ordinals form a class Ord which is itself well-ordered by membership. Basic constructions and results in the theory of ordinals and cardinals make heavy use of Ord, employing definitions and proofs by transfinite recursion and induction. In short, Ord conveniently captures the notion of wellorder.

The situation is quite different in higher-order logic (HOL, Section 2). There is no support for infinite transitive sets, since the type system permits only finite iterations of the powerset. Consequently, membership cannot be used to implement ordinals and cardinals. Another difficulty is that there is no single type that can host a complete collection of canonical representatives for wellorders.

A natural question to ask is: Can we still develop in HOL a theory of cardinals? The answer depends on the precise goals. Our criterion for the affirmative answer is the possibility to prove general-purpose theorems on cardinality for the working mathematician, such as: Given any two types, one can be embedded into the other; given any infinite type, the type of lists over it has the same cardinality; and so on.

We present a formalization in Isabelle/HOL [13] that provides such general-purpose theorems, as well as some more specialized results and applications. We follow a "decentralized" approach, identifying ordinals with arbitrary wellorders and developing all the concepts up to (order-preserving) isomorphism (Section 3). Cardinals are defined, again up to isomorphism, as the minimum ordinals on given underlying sets (Section 4).

The concepts are more abstract than in set theory: Ordinal equality is replaced by a polymorphic relation $=_o$ stating the existence of an order isomorphism, and membership is replaced by a polymorphic operator $<_o$ stating the existence of a strict order embedding (with a nonstrict counterpart $\leq_o$). This abstract view takes more effort to maintain than the concrete implementation from set theory (Section 5), since all the defined operations need to be shown compatible with the new equality and most of them

need to be shown monotonic with respect to the new ordering. For example, $|A|$, the cardinal of A, is defined as *some* cardinal order on A and then proved to be isomorphic to *any* cardinal order on A; and $r_1 +_c r_2$, the sum of cardinals r_1 and r_2, is defined as the cardinal of the sum of r_1's and r_2's fields, before it is proved compatible with $=_o$ and $\leq_o$. Moreover, since the collection of all ordinals does not fit in one type, we must predict the size of the constructed objects and choose suitably large support types.

Our development validates the following thesis:

> *The basics of cardinals can be developed independently of membership-based implementation details and the existence of large classes from set theory.*

This was not clear to us when we started formalizing, since we could not find any textbook or formal development that takes this abstract approach. All introductions to cardinals rely quite heavily on set theory, diving at will into the homogeneous ether provided by the class of all ordinals.

The initial infrastructure and general-purpose theorems were incorporated in the *Archive of Formal Proofs* [18] in 2009, together with thorough documentation, but was not otherwise published. Since then, the formalization has evolved to help specific applications: Cofinalities and regular cardinals were added for a formalization of syntax with bindings [19] (Section 6), and cardinal arithmetic was developed to support Isabelle's new (co)datatype package [24] (Section 7). Moreover, Breitner employed our cardinal infrastructure to formalize free group theory [3].

The theory of cardinals is included with Isabelle starting with the 2012 edition. Some of the features described here are present only in Isabelle's development repository; they are expected to be part of the forthcoming 2014 release. Supplemental formalized material discussed in this paper is publicly available [2].

Related Work. Ordinals, unlike cardinals, have been formalized in HOL before. Harrison [8] formalized ordinals in HOL88 and proved theorems such as Zermelo, Zorn, and transfinite induction. Huffman [11] formalized countable ordinals in Isabelle/HOL, including arithmetic and the Veblen hierarchies—the countability assumption made it possible to fix a type of ordinals. Recently, Norrish and Huffman [14] independently redeveloped in HOL4 much of our theory of ordinals. But while they focus on establishing ordinals as quotients of wellorders under isomorphism and develop some deeper ordinal arithmetic including Cantor normal form, we see the ordinals mostly as a stepping stone toward the cardinals and concentrate on these.

Beyond HOL, Paulson and Grabczewski [16] have formalized some ordinal and cardinal theory in Isabelle/ZF following the usual set-theoretic recipe, via the class of ordinals with membership. Their main objective was to formalize several alternative statements of the axiom of choice, and hence they preferred constructive arguments for most of the cardinal theory. In our development, Hilbert's choice operator (effectively enforcing a bounded version of the axiom of choice) is pervasive.

Outside the realm of mechanized reasoning, there seems to be little interest in developing ordinals and cardinals in a weaker setting than ZF. An exception is Taylor [23], who proposes a foundation for ordinals that avoids membership and meshes well with category theory. His Remark 1.12 mentions the bounded nature of the introduced concepts, which is crucial to express them in HOL.

2 Higher-Order Logic

Isabelle/HOL implements classical higher-order logic with Hilbert choice, the axiom of infinity, and rank-1 polymorphism. HOL is based on Church's simple type theory [5]. It is the logic of Gordon's system of the same name [6] and of its many successors. HOL is roughly equivalent to ZF without support for classes and with the axiom of comprehension taking the place of the axiom of replacement.

Types in HOL are either atomic types (e.g., unit, nat, and bool), type variables α, β, or fully applied type constructors (e.g., nat list and nat set). The binary type constructors $\alpha \to \beta$, $\alpha + \beta$, and $\alpha \times \beta$ for function space, disjoint sum, and product are written in infix notation. All types are nonempty. New types can be introduced by carving out nonempty subsets of existing types. A constant c of type τ is indicated as $c : \tau$.

The following types and constants from the Isabelle library are heavily used in our formalization. $\mathsf{UNIV} : \alpha\ \mathsf{set}$ is the universe set, the set of all elements of type α. 0 and Suc are the constructors of the type nat. Elements of the sum type are constructed by the two embeddings $\mathsf{Inl} : \alpha \to \alpha + \beta$ and $\mathsf{Inr} : \beta \to \alpha + \beta$.

The function $\mathsf{id} : \alpha \to \alpha$ is the identity. $f \bullet A$ is the image of $A : \alpha\ \mathsf{set}$ through $f : \alpha \to \beta$, i.e., the set $\{f\,a.\ a \in A\}$. The predicates inj_on $f\,A$ and bij_betw $f\,A\,B$ state that $f : \alpha \to \beta$ is an injection on $A : \alpha\ \mathsf{set}$ and that $f : \alpha \to \beta$ is a bijection between $A : \alpha\ \mathsf{set}$ and $B : \beta\ \mathsf{set}$.

The type $(\alpha \times \alpha)\ \mathsf{set}$ of binary relations on α is abbreviated to $\alpha\ \mathsf{rel}$. $\mathsf{Id} : \alpha\ \mathsf{rel}$ is the identity relation. Given $r : \alpha\ \mathsf{rel}$, $\mathsf{Field}\ r : \alpha\ \mathsf{set}$ is the field (underlying set) of r, i.e., the union between its domain and its codomain: $\{a.\ \exists b.\ (a,b) \in r\} \cup \{b.\ \exists a.\ (a,b) \in r\}$.

The following predicates operate on relations, where $A : \alpha\ \mathsf{set}$ and $r : \alpha\ \mathsf{rel}$:

REFLEXIVE	$\mathsf{refl\_on}\ A\ r \equiv r \subseteq A \times A \wedge \forall x \in A.\ (x,x) \in r$
TRANSITIVE	$\mathsf{trans}\ r \equiv \forall a\,b\,c.\ (a,b) \in r \wedge (b,c) \in r \to (a,c) \in r$
ANTISYMMETRIC	$\mathsf{antisym}\ r \equiv \forall a\,b.\ (a,b) \in r \wedge (b,a) \in r \to a = b$
TOTAL	$\mathsf{total\_on}\ A\ r \equiv \forall (a \in A)\ (b \in A).\ a \neq b \to (a,b) \in r \vee (b,a) \in r$
WELLFOUNDED	$\mathsf{wf}\ r \equiv \forall P.\ (\forall a.\ (\forall b.\ (b,a) \in r \to P\ b) \to P\ a) \to (\forall a.\ P\ a)$
PARTIAL ORDER	$\mathsf{partial\_order\_on}\ A\ r \equiv \mathsf{refl\_on}\ A\ r \wedge \mathsf{trans}\ r \wedge \mathsf{antisym}\ r$
LINEAR ORDER	$\mathsf{linear\_order\_on}\ r \equiv \mathsf{partial\_order\_on}\ A\ r \wedge \mathsf{total\_on}\ A\ r$
WELLORDER	$\mathsf{well\_order\_on}\ A\ r \equiv \mathsf{linear\_order\_on}\ A\ r \wedge \mathsf{wf}\ (r - \mathsf{Id})$

If r is a partial order, then $r - \mathsf{Id}$ is its associated strict partial order. Some of the above definitions are slightly nonstandard, but can be proved equivalent to standard ones. For example, well-foundedness is given here a higher-order definition useful in proofs as an induction principle, while it is usually equivalently defined as the nonexistence of infinite chains $a : \mathsf{nat} \to \alpha$ with $(a\,(\mathsf{Suc}\ i), a\,i) \in r$ for all i.

Note that $\mathsf{refl\_on}\ A\ r$ (and hence $\mathsf{well\_order\_on}\ A\ r$) implies $\mathsf{Field}\ r = A$. We abbreviate $\mathsf{well\_order\_on}\ (\mathsf{Field}\ r)\ r$ to $\mathsf{Wellorder}\ r$ and $\mathsf{well\_order\_on}\ \mathsf{UNIV}\ r$ to $\mathsf{wellorder}\ r$.

3 Ordinals

This section presents some highlights of our formalization of ordinals. In a break with tradition, we work with abstract ordinals—i.e., with wellorders—making no assumption about their underlying implementation.

3.1 Infrastructure

We represent a wellorder as a relation $r : \tau\ \mathsf{rel}$, where τ is some type. Although some of the lemmas below hold for arbitrary relations, we generally assume that r, s, and t range over wellorders. The following operators are pervasive in our constructions: $\mathsf{under}\ r\ a$ is the set of all elements less than or equal to a, or "under" a, with respect to r; $\mathsf{underS}\ r\ a$ gives the elements strictly under a. We call these *under*- and *strict-under*-intervals:

$$\begin{array}{ll} \mathsf{under} : \alpha\ \mathsf{rel} \to \alpha \to \alpha\ \mathsf{set} & \mathsf{underS} : \alpha\ \mathsf{rel} \to \alpha \to \alpha\ \mathsf{set} \\ \mathsf{under}\ r\ a \equiv \{b \mid (b,a) \in r\} & \mathsf{underS}\ r\ a \equiv \{b \mid (b,a) \in r \wedge b \neq a\} \end{array}$$

A wellorder is a linear order relation r such that its strict version, $r - \mathsf{Id}$, is a well-founded relation. Well-founded induction and recursion are well supported by Isabelle's library. We define slight variations of these notions tailored for wellorders.

Lemma 1. *If* $\forall a \in \mathsf{Field}\ r.\ (\forall a' \in \mathsf{underS}\ r\ a.\ P\ a')$ $\to P\ a$, *then* $\forall a \in \mathsf{Field}\ r.\ P\ a$.

When proving a property P for all elements of r's field, wellorder induction allows us to show P for fixed $a \in \mathsf{Field}\ r$, assuming P holds for elements strictly r-smaller than a.

Wellorder recursion is similar, except that it allows us to define a function f on $\mathsf{Field}\ r$ instead of to prove a property. For each $a \in \mathsf{Field}\ r$, we assume f already defined on $\mathsf{underS}\ r\ a$ and specify $f\ a$. This is technically achieved by a "wellorder recursor" operator $\mathsf{wo\_rec}_r : ((\alpha \to \beta) \to \alpha \to \beta) \to \alpha \to \beta$ and an admissibility predicate $\mathsf{adm\_wo}_r : ((\alpha \to \beta) \to \alpha \to \beta) \to \mathsf{bool}$ defined by

$$\mathsf{adm\_wo}_r\ H \equiv \forall f\ g\ a.\ (\forall a' \in \mathsf{underS}\ r\ a.\ f\ a' = g\ a') \to H\ f\ a = H\ g\ a$$

A recursive definition is represented by a function $H : (\alpha \to \beta) \to \alpha \to \beta$, where $H\ f$ maps a to a value based on the values of f on $\mathsf{underS}\ r\ a$. A more precise type for H would be $\prod_{a \in \mathsf{Field}\ r} (\mathsf{underS}\ r\ a \to \beta) \to \beta$, but this is not possible in HOL. Instead, H is required to be admissible, i.e., not dependent on the values of f outside $\mathsf{underS}\ r\ a$. The defined function $\mathsf{wo\_rec}\ H$ is then a fixpoint of H on $\mathsf{Field}\ r$.

Lemma 2. *If* $\mathsf{adm\_wo}_r\ H$, *then* $\forall a \in \mathsf{Field}\ r.\ \mathsf{wo\_rec}_r\ H\ a = H\ (\mathsf{wo\_rec}_r\ H)\ a$.

An (*order*) *filter* on r, also called an *initial segment* of r if r is a wellorder, is a subset A of r's field such that whenever A contains a, it also contains all elements under a:

$$\begin{array}{l} \mathsf{ofilter} : \alpha\ \mathsf{rel} \to \alpha\ \mathsf{set} \to \mathsf{bool} \\ \mathsf{ofilter}\ r\ A \equiv A \subseteq \mathsf{Field}\ r \wedge (\forall a \in A.\ \mathsf{under}\ r\ a \subseteq A) \end{array}$$

Both the under- and the strict-under-intervals are filters of r. Moreover, every filter of r is either its whole field or a strict-under-interval.

Lemma 3. (1) $\mathsf{ofilter}\ r\ (\mathsf{under}\ r\ a) \wedge \mathsf{ofilter}\ r\ (\mathsf{underS}\ r\ a)$;
(2) $\mathsf{ofilter}\ r\ A \longleftrightarrow A = \mathsf{Field}\ r \vee (\exists a \in \mathsf{Field}\ r.\ A = \mathsf{underS}\ r\ a)$.

3.2 Embedding and Isomorphism

Wellorder embeddings, strict embeddings, and isomorphisms are defined as follows:

$$\begin{array}{l} \mathsf{embed}, \mathsf{embedS}, \mathsf{iso} : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha \to \beta) \to \mathsf{bool} \\ \mathsf{embed}\quad r\ s\ f \equiv \forall a \in \mathsf{Field}\ r.\ \mathsf{bij\_betw}\ f\ (\mathsf{under}\ r\ a)\ (\mathsf{under}\ s\ (f\ a)) \end{array}$$

$$\mathsf{embedS}\ r\ s\ f \equiv \mathsf{embed}\ r\ s\ f \wedge \neg\ \mathsf{bij\_betw}\ f\ (\mathsf{Field}\ r)\ (\mathsf{Field}\ s)$$
$$\mathsf{iso}\quad\ r\ s\ f \equiv \mathsf{embed}\ r\ s\ f \wedge \mathsf{bij\_betw}\ f\ (\mathsf{Field}\ r)\ (\mathsf{Field}\ s)$$

We read $\mathsf{embed}\ r\ s\ f$ as "f embeds r into s." It is defined by stating that for all $a \in \mathsf{Field}\ r$, f establishes a bijection between the under-intervals of a in r and those of $f\,a$ in s. The more conventional definition (stating that f is injective, order preserving, and maps $\mathsf{Field}\ r$ into a filter of s) is derived as a lemma:

Lemma 4. $\mathsf{embed}\ r\ s\ f \longleftrightarrow \mathsf{compat}\ r\ s\ f \wedge \mathsf{inj\_on}\ f\ (\mathsf{Field}\ r) \wedge \mathsf{ofilter}\ s\ (f \bullet \mathsf{Field}\ r)$, *where* $\mathsf{compat}\ r\ s\ f$ *expresses order preservation of* f $(\forall a\ b.\ (a,b) \in r \rightarrow (f\,a, f\,b) \in s)$.

Every embedding is either an (order) isomorphism or a strict embedding (i.e., $\mathsf{iso}\ r\ s\ f \vee \mathsf{embedS}\ r\ s\ f$), depending on whether f is a bijection. These notions yield the following relations between wellorders:

$$\leq_o, <_o, =_o : (\alpha\ \mathsf{rel} \times \beta\ \mathsf{rel})\ \mathsf{set}$$
$$\leq_o \equiv \{(r,s).\ \mathsf{Wellorder}\ r \wedge \mathsf{Wellorder}\ s \wedge (\exists f.\ \mathsf{embed}\ r\ s\ f)\}$$
$$<_o \equiv \{(r,s).\ \mathsf{Wellorder}\ r \wedge \mathsf{Wellorder}\ s \wedge (\exists f.\ \mathsf{embedS}\ r\ s\ f)\}$$
$$=_o \equiv \{(r,s).\ \mathsf{Wellorder}\ r \wedge \mathsf{Wellorder}\ s \wedge (\exists f.\ \mathsf{iso}\ r\ s\ f)\}$$

We abbreviate $(r,s) \in \leq_o$ to $r \leq_o s$, and similarly for $<_o$ and $=_o$. These notations are fairly intuitive; for example, $r \leq_o s$ means that r is smaller than or equal to s, in that it can be embedded in s. The relations are also well behaved.

Theorem 5. *The following properties hold:*

(1) $r =_o r$
(2) $r =_o s \rightarrow s =_o r$
(3) $r =_o s \wedge s =_o t \rightarrow r =_o t$
(4) $r \leq_o r$
(5) $r \leq_o s \wedge s \leq_o t \rightarrow r \leq_o t$
(6) $\neg\ r <_o r$
(7) $r <_o s \wedge s <_o t \rightarrow r <_o t$
(8) $r \leq_o s \longleftrightarrow r <_o s \vee r =_o s$
(9) $r =_o s \longleftrightarrow r \leq_o s \wedge s \leq_o r$

If we restrict the types of these relations from $(\alpha\ \mathsf{rel} \times \beta\ \mathsf{rel})\ \mathsf{set}$ to $(\alpha\ \mathsf{rel})\ \mathsf{rel}$ (by taking $\beta = \alpha$), we obtain that $=_o$ is an equivalence (1–3) and $\leq_o$ is a preorder (4–5). Moreover, $<_o$ is the strict version of $\leq_o$ with respect to $=_o$ (6–8). If we think of $=_o$ as the equality, $\leq_o$ becomes a partial order (9) and $<_o$ a strict partial order.

The above relations establish an order between the wellorders similar to the standard one on the class of ordinals but distributed across types and, as a consequence, only up to isomorphism. What is still missing is a result corresponding to the class of ordinals being itself well ordered. To this end, we first show $\leq_o$ to be total.

Theorem 6. $r \leq_o s \vee r \leq_o s$.

Proof idea. In textbooks, totality of $\leq_o$ follows from the fact that every wellorder is isomorphic to an ordinal and that the class of ordinals Ord is totally ordered. To show the former, one starts with a wellorder r and provides an embedding of r into Ord.

In our distributed setting, we must start with two wellorders $r : \alpha\ \mathsf{rel}$ and $s : \beta\ \mathsf{rel}$, without a priori knowing which one is larger, hence which should embed which. Our proof proceeds by defining a function by transfinite recursion on r that embeds r into s if $r \leq_o s$ and that is the inverse of an embedding of s into r otherwise. □

This total order is a wellorder. Equivalently, its strict counterpart $<_o$ is well founded.

Theorem 7. $\mathsf{wf}\ (<_\mathsf{o} : (\alpha\ \mathsf{rel})\ \mathsf{rel})$.

Theorems 5, 6, and 7 yield that for any fixed type, its wellorders are themselves well ordered up to isomorphism. This paves the way for introducing cardinals.

3.3 Ordinal Arithmetic

Most textbooks define operations on ordinals (sum, product, exponentiation) by transfinite recursion. Yet these operations admit direct, nonrecursive definitions, which are particularly suited to arbitrary wellorders. In Holz et al. [10], these direct definitions are presented as "visual" descriptions.

We define the ordinal sum $+_\mathsf{o}$ by concatenating the two argument wellorders r and s such that elements of $\mathsf{Field}\ r$ come below those of $\mathsf{Field}\ s$:

$$+_\mathsf{o} : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha+\beta)\ \mathsf{rel}$$
$$r +_\mathsf{o} s \equiv (\mathsf{Inl} \otimes \mathsf{Inl}) \bullet r \cup (\mathsf{Inr} \otimes \mathsf{Inr}) \bullet s \cup \{(\mathsf{Inl}\ x, \mathsf{Inr}\ y).\ x \in \mathsf{Field}\ r \wedge y \in \mathsf{Field}\ s\}$$

In the above, the operator $\otimes : (\alpha_1 \to \beta_1) \to (\alpha_2 \to \beta_2) \to (\alpha_1 \times \alpha_2 \to \beta_1 \times \beta_2)$ is the map function for products: $(f_1 \otimes f_2)\ (a_1, a_2) = (f_1\, a_1, f_2\, a_2)$.

Similarly, ordinal multiplication $\times_\mathsf{o}$ is defined as the anti-lexicographic ordering on the product type:

$$\times_\mathsf{o} : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha+\beta)\ \mathsf{rel}$$
$$r \times_\mathsf{o} s \equiv \{((x_1,y_1),(x_2,y_2)).\ x_1,x_2 \in \mathsf{Field}\ r \wedge y_1,y_2 \in \mathsf{Field}\ s \wedge (y_1 \neq y_2 \wedge (y_1,y_2) \in s \vee y_1 = y_2 \wedge (x_1,x_2) \in r)\}$$

For ordinal exponentiation $r \mathbin{\hat{}}_\mathsf{o} s$, the underlying set consists of the functions of finite support from $\mathsf{Field}\ s$ to $\mathsf{Field}\ r$. Assuming $f \neq g$, the finite support ensures that there exists a maximum $z \in \mathsf{Field}\ s$ (with respect to s) such that $f\, z \neq g\, z$. We make f smaller than g if $(f\, z, g\, z) \in r$:

$$\mathbin{\hat{}}_\mathsf{o} : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\beta \to \alpha)\ \mathsf{rel}$$
$$r \mathbin{\hat{}}_\mathsf{o} s \equiv \{(f,g).\ f,g \in \mathsf{FinFunc}\ (\mathsf{Field}\ s)\ (\mathsf{Field}\ r) \wedge (f = g \vee (\mathsf{let}\ z = \mathsf{max}_s\{x \in \mathsf{Field}\ s.\ f\, x \neq g\, x\}\ \mathsf{in}\ (f\, z, g\, z) \in r))\}$$

The definition rests on the auxiliary notion of a function of finite support from $B : \alpha\ \mathsf{set}$ to $A : \beta\ \mathsf{set}$. $\mathsf{FinFunc}\ B\ A$ carves out a suitable subspace of the total function space $\beta \to \alpha$ by requiring that functions are equal to a particular unspecified value $\bot$ outside their intended domains. In addition, finite support means that only finitely many elements of B are mapped to elements other than the minimal element 0_r of the wellorder r:

$$\mathsf{Func}, \mathsf{FinFunc} : \beta\ \mathsf{set} \to \alpha\ \mathsf{set} \to (\beta \to \alpha)\ \mathsf{set}$$
$$\mathsf{Func}\ B\ A \equiv \{f.\ f \bullet B \subseteq A \wedge (\forall x \notin B.\ f\, x = \bot)\}$$
$$\mathsf{FinFunc}\ B\ A \equiv \mathsf{Func}\ B\ A \cap \{f.\ \mathsf{finite}\ \{x \in B.\ f\, x \neq 0_r\}\}$$

All three constructions yield wellorders. Moreover, they satisfy various arithmetic properties, including those listed below.

Theorem 8. (1) $\mathsf{Wellorder}\ (r +_\mathsf{o} s)$; (2) $\mathsf{Wellorder}\ (r \times_\mathsf{o} s)$; (3) $\mathsf{Wellorder}\ (r \mathbin{\hat{}}_\mathsf{o} s)$.

Lemma 9 (Lemma 1.4.3 in Holz et al. [10]). *Let* 0 *be the empty wellorder and* 1 *be the singleton wellorder. The following properties hold:*

(1) $0 +_o r =_o r =_o r +_o 0$
(2) $s \leq_o r +_o s$
(3) $s <_o t \to r +_o s <_o r +_o t$
(4) $(r +_o s) +_o t =_o r +_o (s +_o t)$
(5) $r \leq_o s \to r +_o t \leq_o s +_o t$

(6) $0 \times_o r =_o 0 =_o r \times_o 0$
(7) $(r \times_o s) \times_o t =_o r \times_o (s \times_o t)$
(8) $r \leq_o s \to r \times_o t \leq_o s \times_o t$
(9) $1 \times_o r =_o r =_o r \times_o 1$
(10) $r \times_o (s +_o t) =_o r \times_o s +_o r \times_o t$
(11) $0 <_o r \wedge s <_o t \to r \times_o s <_o r \times_o t$

(12) $0 <_o r \to 0 \wedge_o r =_o 0$
(13) $(r \wedge_o s) \wedge_o t =_o r \wedge_o (s \times_o t)$
(14) $r \leq_o s \to r \wedge_o t \leq_o s \wedge_o t$
(15) $1 <_o r \to s \leq_o r \wedge_o s$
(16) $1 \wedge_o r =_o 1$
(17) $r \wedge_o s +_o t =_o r \wedge_o s \times_o r \wedge_o t$
(18) $1 <_o r \wedge s <_o t \to r \wedge_o s <_o r \wedge_o t$

An advantage of the standard definitions of these operations by transitive recursion is that the above arithmetic facts can then be nicely proved by corresponding transfinite induction. With direct definitions, we aim as much as possible at direct proofs via the explicit indication of suitable isomorphisms or embeddings, as in the definitions of $=_o$, $\leq_o$, and $<_o$. This approach works well for the equations ($=_o$-identities) and for right-monotonicity properties of the operators (where one assumes equality on the left arguments and ordering of the right arguments). For example, to prove $0 <_o r \wedge s <_o t \to r \times_o s <_o r \times_o t$, we use the definition of $<_o$ to obtain from $s <_o t$ a strict embedding f of s into t. The desired strict embedding of $r \times_o s$ into $r \times_o t$ is then $\mathsf{id} \otimes f$.

In contrast, left-monotonicity properties such as $r \leq_o s \to r \times_o t \leq_o s \times_o t$ no longer follow smoothly, because it is not clear how to produce an embedding of $r \times_o t$ into $s \times_o t$ from one of r into s. An alternative characterization of $\leq_o$ is called for:

Lemma 10. $r \leq_o s \leftrightarrow \mathsf{Wellorder}\ r \wedge \mathsf{Wellorder}\ s \wedge (\exists f.\ \forall a \in \mathsf{Field}\ r.\ f\,a \in \mathsf{Field}\ s \wedge f \bullet \mathsf{underS}\ r\,a \subseteq \mathsf{underS}\ s\,(f\,a))$.

Thus, to show $r \leq_o s$, it suffices to provide an order embedding, which need not be a wellorder embedding (an embedding of $\mathsf{Field}\ r$ as a filter of s). This dramatically simplifies the proof. To show the left-monotonicity property $r \times_o t \leq_o s \times_o t$ assuming an embedding f of r into s, the obvious order embedding $f \otimes \mathsf{id}$ meets the requirements. Surprisingly, this technique is not mentioned in the textbooks.

Right-monotonicity holds for both $<_o$ and $\leq_o$, wheres left-monotonicity holds only for $\leq_o$. This is fortunate in a sense, because Lemma 10 is not adaptable to $<_o$.

4 Cardinals

With the ordinals in place, we can develop a theory of cardinals, which endows HOL with many conveniences of cardinality reasoning, including basic cardinal arithmetic.

4.1 Bootstrapping

We define cardinal orders on a set (or cardinals) as those wellorders that are minimal with respect to $=_o$. This is our HOL counterpart of the standard definition of cardinals as "ordinals that cannot be mapped one-to-one onto smaller ordinals" [10, p. 42]:

$$\mathsf{card\_order\_on}\ A\ r \equiv \mathsf{well\_order\_on}\ A\ r \wedge (\forall s.\ \mathsf{well\_order\_on}\ A\ s \to r \leq_o s)$$

We abbreviate card_order_on (Field r) r to Card_order r and card_order_on UNIV r to card_order r. By definition, card_order_on A r implies $A = \mathsf{Field}\ r$, allowing us to write Card_order r when we want to omit A.

Cardinals are useful to measure sets. There exists a cardinal on every set, and it is unique up to isomorphism.

Theorem 11. (1) $\exists r.$ card_order_on A r;
(2) card_order_on A $r \wedge$ card_order_on A $s \rightarrow r =_o s$.

We define the cardinality of a set $|\_| : \alpha\ \mathsf{set} \rightarrow \alpha\ \mathsf{rel}$ using Hilbert's choice operator to pick an arbitrary cardinal order on A: $|A| \equiv \varepsilon r.$ card_order_on A r. The order exists and is irrelevant by Theorem 11. We can prove that the cardinality operator behaves as expected; in particular, it is monotonic. We can also connect it to the more elementary comparisons in terms of functions.

Lemma 12. *The following properties hold:*

(1) card_order_on A $|A|$
(2) $\mathsf{Field}\ |A| = A$
(3) $A \subseteq B \rightarrow |A| \leq_o |B|$
(4) $r \leq_o s \rightarrow |\mathsf{Field}\ r| \leq_o |\mathsf{Field}\ s|$

Lemma 13. *The following equivalences hold:*

(1) $|A| =_o |B| \leftrightarrow (\exists f.$ bij_betw f A $B)$
(2) $|A| \leq_o |B| \leftrightarrow (\exists f.$ inj_on f $A \wedge f \bullet A \subseteq B)$
(3) $A \neq \emptyset \rightarrow (|A| \leq_o |B| \leftrightarrow (\exists g.\ g \bullet B \subseteq A))$

Lemma 13, in conjunction with Theorem 6, allows us to prove the following interesting order-free fact for the working mathematician, mentioned in Section 1.

Theorem 14. *Given any two types σ and τ, one is embeddable in the other: There exists an injection either from σ to τ or from τ to σ.*

4.2 Cardinality of Set and Type Constructors

We analyze the cardinalities of several standard type constructors: $\alpha + \beta$ (disjoint sum), $\alpha \times \beta$ (binary product), $\alpha\ \mathsf{set}$ (powertype), and $\alpha\ \mathsf{list}$ (lists). In the interest of generality, we consider the homonymous set-based versions of these constructors, which take the form of polymorphic constants:

$+ : \alpha\ \mathsf{set} \rightarrow \beta\ \mathsf{set} \rightarrow (\alpha + \beta)\ \mathsf{set}$
$A + B \equiv \{\mathsf{Inl}\ a \mid a \in A\} \cup \{\mathsf{Inr}\ b \mid b \in B\}$

$\times : \alpha\ \mathsf{set} \rightarrow \beta\ \mathsf{set} \rightarrow (\alpha \times \beta)\ \mathsf{set}$
$A \times B \equiv \{(a,b) \mid a \in A \wedge b \in B\}$

$\mathsf{Pow} : \alpha\ \mathsf{set} \rightarrow (\alpha\ \mathsf{set})\ \mathsf{set}$
$\mathsf{Pow}\ A \equiv \{X \mid X \subseteq A\}$

$\mathsf{lists} : \alpha\ \mathsf{list} \rightarrow (\alpha\ \mathsf{list})\ \mathsf{set}$
$\mathsf{lists}\ A \equiv \{as \mid \mathsf{set}\ as \subseteq A\}$

(Such operators can be generated automatically from the specification of a (co)datatype, as we will see in Section 7.) The cardinalities of these operators are compatible with isomorphism and embedding.

Lemma 15. *Let K be any of $+$, $\times$, Pow, and* lists*, let $n \in \{1,2\}$ be its arity, and let θ be either $=_o$ or $\leq_o$. If $\forall i \in \{1,\ldots,n\}$. $|A_i|\ \theta\ |B_i|$, then $|K\ A_1\ \ldots\ A_n|\ \theta\ |K\ B_1\ \ldots\ B_n|$.*

Lemma 16. *The following orderings between cardinalities hold:*

(1) $|A| \leq_o |A+B|$
(2) $|A+B| \leq_o |A \times B|$ *if both A and B have at least two elements*
(3) $|A| <_o |\mathsf{Pow}\ A|$
(4) $|A| \leq_o |\mathsf{lists}\ A|$

If one of the involved sets is infinite, some embeddings collapse to isomorphisms.

Lemma 17. *Assuming* infinite *A, the following equalities between cardinals hold:*

(1) $|A \times A| =_o |A|$
(2) $|A| =_o |\mathsf{lists}\ A|$
(3) $|A+B| =_o (\text{if } A \leq_o B \text{ then } |A| \text{ else } |B|)$
(4) $B \neq \emptyset \rightarrow |A \times B| =_o (\text{if } A \leq_o B \text{ then } |A| \text{ else } |B|)$

The formalization of property (1) required a significant effort. Its proof relies on the so-called bounded product construction, which is extensively discussed by Paulson and Grabczewski [16] in the context of Isabelle/ZF.

In Isabelle/HOL, the cartesian product is a special case of the indexed sum (or disjoint union) operator:

$$\Sigma : \alpha\ \mathsf{set} \rightarrow (\alpha \rightarrow \beta\ \mathsf{set}) \rightarrow (\alpha \times \beta)\ \mathsf{set}$$
$$\Sigma\ A\ f \equiv \bigcup_{a \in A} \bigcup_{b \in f\ a} \{(a,b)\}$$

We write $\sum_{a \in A} f\ a$ for $\Sigma\ A\ f$. The properties of $\times$ given above carry over to Σ. In addition, Σ can be used to prove cardinality bounds of indexed unions:

Lemma 18. (1) $|\bigcup_{a \in A} f\ a| \leq_o |\sum_{a \in A} f\ a|$;
(2) $\mathsf{infinite}\ B \wedge |A| \leq_o |B| \wedge (\forall a \in A.\ |f\ a| \leq_o |B|) \rightarrow |\bigcup_{a \in A} f\ a| \leq_o |B|$.

4.3 $\aleph_0$ and the Finite Cardinals

Our $\aleph_0$ is the standard order $\leq : \mathsf{nat\ rel}$ on natural numbers, which we denote by natLeq. It behaves as expected of $\aleph_0$; in particular, it is $\leq_o$-minimal among infinite cardinals. Proper filters of natLeq are precisely the finite sets of the first consecutive numbers.

Lemma 19. (1) $\mathsf{infinite}\ A \leftrightarrow \mathsf{natLeq} \leq_o |A|$; (2) $\mathsf{Card\_order\ natLeq}$;
(3) $\mathsf{Card\_order}\ r \wedge \mathsf{infinite}\ (\mathsf{Field}\ r) \rightarrow r \leq_o \mathsf{natLeq}$;
(4) $\mathsf{ofilter\ natLeq}\ A \leftrightarrow A = (\mathsf{UNIV} : \mathsf{nat\ set}) \vee (\exists n.\ A = \{0,\ldots,n\})$.

The finite cardinals are obtained as restrictions of natLeq: $\mathsf{natLeq\_on}\ n \equiv \mathsf{natLeq} \cap \{0,\ldots,n\} \times \{0,\ldots,n\}$. These behave like the finite cardinals (up to isomorphism):

Lemma 20. (1) $\mathsf{card\_order}\ (\mathsf{natLeq\_on}\ n)$; (2) $\mathsf{finite}\ A \wedge |A| =_o |B| \rightarrow \mathsf{finite}\ B$;
(3) $\mathsf{finite}\ A \leftrightarrow (\exists n.\ |A| =_o \mathsf{natLeq\_on}\ n)$.

For finite cardinalities, we prove backward compatibility with the preexisting cardinality operator $\mathsf{card} : \alpha\ \mathsf{set} \rightarrow \mathsf{nat}$, which maps infinite sets to 0:

Lemma 21. *Assuming* $\mathsf{finite}\ A \wedge \mathsf{finite}\ B$:

(1) $|A| =_o |B| \longleftrightarrow \mathsf{card}\ A = \mathsf{card}\ B$ (2) $|A| \leq_o |B| \longleftrightarrow \mathsf{card}\ A \leq \mathsf{card}\ B$

The card operator has extensive library support in Isabelle. It is still the preferred cardinality operator for finite sets, since it refers to numbers with order and equality rather than the more bureaucratic order embeddings and isomorphisms.

cardSuc preserves finiteness and behaves as expected for finite cardinals:

Lemma 22. (1) $\mathsf{Card\_order}\ r \rightarrow (\mathsf{finite}\ (\mathsf{cardSuc}\ r) \longleftrightarrow \mathsf{finite}\ (\mathsf{Field}\ r))$;
(2) $\mathsf{cardSuc}\ (\mathsf{natLeq\_on}\ n) =_o \mathsf{natLeq\_on}\ (\mathsf{Suc}\ n)$.

4.4 Cardinal Arithmetic

To define cardSuc r, the successor of a cardinal $r : \alpha\ \mathsf{rel}$, we first choose a type that is certainly large enough to contain a cardinal greater than r, namely, $\alpha\ \mathsf{set}$. The successor cardinal is then defined as a cardinal that is greater than r and that is $\leq_o$-minimal among all cardinals on the chosen type $\alpha\ \mathsf{set}$:

$$\begin{aligned}&\mathsf{isCardSuc} : \alpha\ \mathsf{rel} \rightarrow (\alpha\ \mathsf{set})\ \mathsf{rel} \rightarrow \mathsf{bool}\\ &\mathsf{isCardSuc}\ r\ s \equiv \mathsf{Card\_order}\ s \wedge r <_o s \wedge\\ &\qquad\qquad (\forall t : (\alpha\ \mathsf{set})\ \mathsf{rel}.\ \mathsf{Card\_order}\ t \wedge r <_o t \rightarrow s \leq_o t)\end{aligned}$$

The choice of the second argument's type, together with Theorem 7, ensures that such a cardinal exists:

Lemma 23. $\exists s.\ \mathsf{isCardSuc}\ r\ s$.

This allows us to define the function $\mathsf{cardSuc} : \alpha\ \mathsf{rel} \rightarrow (\alpha\ \mathsf{set})\ \mathsf{rel}$ that yields an arbitrary successor cardinal of its argument r: $\mathsf{cardSuc}\ r \equiv \varepsilon s.\ \mathsf{isCardSuc}\ r\ s$. The chosen cardinal is really a successor cardinal:

Lemma 24. $\mathsf{isCardSuc}\ r\ (\mathsf{cardSuc}\ r)$

To obtain the desired characteristic properties of successor cardinals in full generality, we must prove that cardSuc r is minimal not only among the cardinals on $\alpha\ \mathsf{set}$ but among all cardinals. This is achieved by a tedious process of making isomorphic copies.

Theorem 25. *Assuming* $\mathsf{Card\_order}\ (r : \alpha\ \mathsf{rel})$ *and* $\mathsf{Card\_order}\ (s : \beta\ \mathsf{rel})$:

(1) $r <_o \mathsf{cardSuc}\ r$ (2) $r <_o s \rightarrow \mathsf{cardSuc}\ r \leq_o s$

Finally, we prove that cardSuc is compatible with isomorphism and is monotonic.

Theorem 26. *Assuming* $\mathsf{Card\_order}\ r$ *and* $\mathsf{Card\_order}\ s$:

(1) $\mathsf{cardSuc}\ r =_o \mathsf{cardSuc}\ s \longleftrightarrow r =_o s$ (2) $\mathsf{cardSuc}\ r <_o \mathsf{cardSuc}\ s \longleftrightarrow r <_o s$

In summary, we first introduced the successor in a type-specific manner, asserting minimality within a chosen type, since HOL would not allow us to proceed more generally. Then we proved the characteristic property in full generality, and finally we showed that the notion is compatible with $=_o$ and $\leq_o$.

This approach is certainly more bureaucratic than the traditional set theoretic constructions, but it achieves the desired effect. The same technique is used to introduce all the standard cardinal operations (e.g, $+_c : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha+\beta)\ \mathsf{rel}$), for which we prove the basic arithmetic properties.

Lemma 27 (Lemma 1.5.10 in Holz et al. [10]). *The following properties hold:*

(1) $(r +_c s) +_c t =_o r +_c (s +_c t)$
(2) $r +_c s =_o s +_c r$
(3) $(r \times_c s) \times_c t =_o r \times_c (s \times_c t)$
(4) $r \times_c s =_o s \times_c r$
(5) $r \times_c 0 =_o 0$
(6) $r \times_c 1 =_o r$
(7) $r \times_c (s +_c t) =_o r \times_c s +_c r \times_c t$
(8) $r \wedge_c (s +_c t) =_o r \wedge_c s \times_c r \wedge_c t$
(9) $(r \wedge_c s) \wedge_c t =_o r \wedge_c (s \times_c t)$
(10) $(r \times_c s) \wedge_c t =_o r \wedge_c t \times_c s \wedge_c t$
(11) $\neg\, r =_o 0 \to r \wedge_c 0 =_o 1 \wedge 0 \wedge_c r =_o 0$
(12) $r \wedge_c 1 =_o r$
(13) $1 \wedge_c r =_o 1$
(14) $r \wedge_c 2 =_o r \times_c r$
(15) $r \leq_o s \wedge t \leq_o u \to r +_c t \leq_o s +_c u$
(16) $r \leq_o s \wedge t \leq_o u \wedge \neg\, t =_o 0 \to r \wedge_c t \leq_o s \wedge_c u$
(17) $r \leq_o s \wedge t \leq_o u \to r \times_c t \leq_o s \times_c u$

Another useful cardinal operation is the maximum of two cardinals, $\mathsf{cmax}\ r\ s$, which is well-defined by the totality of $\leq_o$. Thanks to Lemma 17(1), it behaves like both sum and product for infinite cardinals:

Lemma 28. $(\mathsf{infinite}\ (\mathsf{Field}\ r) \wedge \mathsf{Field}\ s \neq \emptyset) \vee (\mathsf{infinite}\ (\mathsf{Field}\ s) \wedge \mathsf{Field}\ r \neq \emptyset) \to \mathsf{cmax}\ r\ s =_o r +_c s =_o r \times_c s.$

4.5 Regular Cardinals

A set $A : \alpha\ \mathsf{set}$ is *cofinal* for $r : \alpha\ \mathsf{rel}$, written $\mathsf{cofinal}\ A\ r$, if $\forall a \in \mathsf{Field}\ r.\ \exists b \in A.\ a \neq b \wedge (a,b) \in r$; and r is *regular*, written $\mathsf{regular}\ r$, if $\forall A.\ A \subseteq \mathsf{Field}\ r \wedge \mathsf{cofinal}\ A\ r \to |A| =_o r$.

Regularity is a generalization of the property of natLeq of not being "coverable" by smaller cardinals—indeed, no finite set A of numbers fulfills $\forall m.\ \exists n \in A.\ m < n$. The infinite successor cardinals are further examples of regular cardinals.

Lemma 29. (1) $\mathsf{regular}\ \mathsf{natLeq}$;
(2) $\mathsf{Card\_order}\ r \wedge \mathsf{infinite}\ (\mathsf{Field}\ r) \to \mathsf{regular}\ (\mathsf{cardSuc}\ r)$.

A property of regular cardinals useful in applications is the following: Inclusion of a set of smaller cardinality in a union of a chain indexed by the cardinal behaves similarly to membership, in that it amounts to inclusion in one of the sets in the chain.

Lemma 30. *Assume* $\mathsf{Card\_order}\ r$, $\mathsf{regular}\ r$, $\forall i\ j.\ (i,j) \in r \to A\ i \subseteq A\ j$, $|B| <_o r$, *and* $B \subseteq \bigcup_{i \in \mathsf{Field}\ r} A\ i$. *Then* $\exists i \in \mathsf{Field}\ r.\ B \subseteq A\ i$.

Finally, regular cardinals are stable under unions. They cannot be covered by a union of sets of smaller cardinality indexed by a set of smaller cardinality.

Lemma 31. *Assuming* $\mathsf{Card\_order}\ r$, $\mathsf{regular}\ r$, $|I| <_o r$, *and* $\forall i \in I.\ |A\ i| <_o r$, *we have* $|\bigcup_{i \in I} A\ i| <_o r$.

We also proved the converse: The above property is not only necessary but also sufficient for regularity.

5 Discussion of the Formalization

Figure 1 shows the main theory structure of our development. The overall development amounts to about 14 000 lines of scripts, excluding the applications. We also formalized many basic facts about wellorders and (order) isomorphic transfer across bijection. When we started, Isabelle's library had extensive support for orders based on type classes [7]. However, working with the wellorder type class was not an option, since we need several wellorders for the same type—for example, the cardinal of a type is defined as the minimum of all its wellorders.

Reasoning about the modified version of equality and order ($=_o$, $\leq_o$, and $<_o$) was probably the most tedious aspect of the formalization effort. The standard Isabelle proof methods (*auto*, *blast*, etc.) are optimized for reasoning about actual equality and order. Some of the convenience could be recovered via an appropriate setup; for example, declaring these relations as transitive enables calculational reasoning in Isar [1].

For the initial version of the formalization, developed in 2009, Isabelle's Sledgehammer tool for deploying external automatic theorem provers [15] did not help much. The proofs required a careful combination of facts on orders and isomorphic transfer, and Sledgehammer was not as powerful as it is today. In contrast, cardinal arithmetic was developed later and largely automated in this way.

Throughout the paper, we have illustrated our effort to adapt the theory of cardinals to the HOL types, doing without a canonical class of ordinals ordered by membership. Another limitation of HOL is its inability to quantify over types except universally and at the statements' top level. A notorious example comes from the formalizations of the FOL completeness theorem (e.g., Harrison [9]): A sentence is provable if and only if it is true in all models. The 'if' direction is not expressible in HOL, because the right-hand side quantifies over all carrier types of all models, which amounts to an existential type quantification at the top of the formula. But one can express and prove a stronger statement: Based on the language cardinality, one fixes a witness type so that satisfaction in all models on that type already ensures provability. Our formalization abounds in such apparently inexpressible statements. One example is the definition of the successor cardinal from Section 4.4. Another is the claimed converse of Lemma 31. Each time, we had to select a suitable witness type in an ad hoc fashion.

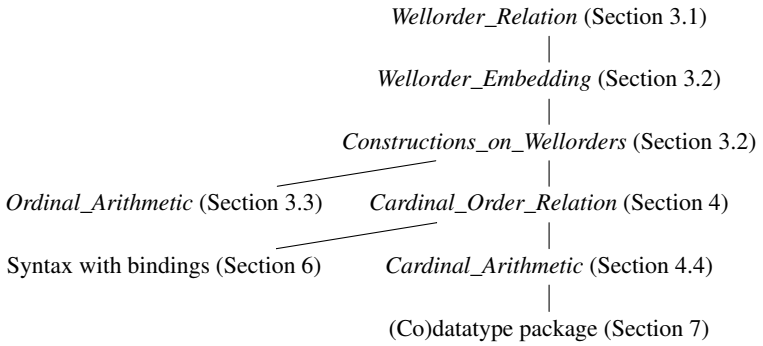

Fig. 1. Theory structure

6 Application: Syntax with Bindings

Popescu has formalized a general theory of syntax with bindings, parameterized over a binding signature with possibly infinitary operation symbols [19–21]. Cardinals were crucially needed for supporting infinitary syntax.

We illustrate the problem and solution on an example. Let index and var be types representing indices and variables, and consider the freely generated type of terms

datatype term = Var var | Lam var term | Sum (index → term)

Thus, a term is either (an injection of) a variable, a λ-abstraction, or an indexed sum of a family of terms. We define the standard operators of free variables fvars : term → var set and capture-avoiding substitution _[_/_] : term → term → var → term:

$$\begin{array}{lll} \mathsf{fvars}\ (\mathsf{Var}\ x) & = \{x\} & (\mathsf{Var}\ x)[s/y] = (\text{if } x = y \text{ then } s \text{ else } \mathsf{Var}\ x) \\ \mathsf{fvars}\ (\mathsf{Lam}\ x\ t) & = \mathsf{fvars}\ t - \{x\} & (\mathsf{Lam}\ x\ t)[s/y] = (\text{let } x' = \mathsf{pickFresh}\ [\mathsf{Var}\ y, s] \\ \mathsf{fvars}\ (\mathsf{Sum}\ f) & = \bigcup_{i \in I} \mathsf{fvars}\ (f\ i) & \qquad\qquad \text{in } \mathsf{Lam}\ x'\ (t[x'/x][s/y])) \\ & & (\mathsf{Sum}\ f)[s/y] = \mathsf{Sum}\ (\lambda i.\ (f\ i)[s/y]) \end{array}$$

To avoid capture, the Lam case of substitution renames x to x', which is chosen by pickFresh to be fresh with respect to y and s. But how can we be sure that such a choice exists? The standard solution of making the type var infinite does not suffice here: The Sum constructor introduces possibly infinite index-branching, and therefore fvars T may return an infinite set of variables, potentially even UNIV.

Fortunately, the idea behind the standard solution can be generalized to the infinitary situation. Finitely branching syntax relies on the observation that no n-ary constructor violates the finiteness of the set of free variables, since a finite union of finite sets is finite. Lemma 31 generalizes this notion for regular cardinals. Hence, we simply need to define var so that it has a regular cardinal greater than index: var = cardSuc |index|.

Lemma 32. regular |var| ∧ |index| $<_o$ |var| → (∀t. |fvars t| $<_o$ |var|).

Proof idea. Immediate by structural induction on t, using Lemma 31. □

After passing this milestone, a theory of substitution and free variables proceeds similarly to the finitary case [19]. Most current frameworks for syntax with bindings, including nominal logic [12, 17], assume finiteness of the syntactic objects. Regular cardinals provide a foundation for an infinitary generalization.

7 Application: Bounded Functors and the (Co)datatype Package

Isabelle's new (co)datatype package draws on both category theory and cardinal theory. It maintains a class of functors with additional structure, called *bounded natural functors* (BNFs), for which it constructs initial algebras (datatypes) and final coalgebras (codatatypes). The category theory underlying the package is described in Traytel et al. [24]; here, we focus on the cardinality aspects.

BNFs are type constructors equipped with functorial actions, n natural transformations, and a cardinality bound. A unary BNF consists of a type constructor α F,

a constant $\mathsf{Fmap} : (\alpha \to \beta) \to \alpha\ \mathsf{F} \to \beta\ \mathsf{F}$, a constant $\mathsf{Fset} : \alpha\ \mathsf{F} \to \alpha\ \mathsf{set}$ that is natural with respect to F, and a cardinal Fbd such that $\forall x.\ |\mathsf{Fset}\ x| \leq_{\mathsf{o}} \mathsf{Fbd}$. We define $\mathsf{Fin} : \alpha\ \mathsf{set} \to (\alpha\ F)\ \mathsf{set}$, the set-based version of F, by $\mathsf{Fin}\ A = \{x \mid \mathsf{Fset}\ x \subseteq A\}$—this is a common generalization of the specific set-based operators from Section 4.2.

An algebra for F is a triple $\mathscr{A} = (T, A : T\ \mathsf{set}, s : T\ \mathsf{F} \to T)$ (where T is a type) such that $\forall x \in \mathsf{Fin}\ A.\ s\ x \in A$. The condition ensures that s is a function from $\mathsf{Fin}\ A$ to A, and we allow ourselves to write $s : \mathsf{Fin}\ A \to A$. The set A is the *carrier* of $\mathscr{A}$, and s is the *structural map* of A. The structural map models the operations of the algebra. For example, if $\alpha\ \mathsf{F} = \mathsf{unit} + \alpha \times \alpha$, an algebra $\mathscr{A}$ consists of a set $A : T\ \mathsf{set}$ with a constant and a binary operation on it, encoded as $s : \mathsf{unit} + \alpha \times \alpha \to \alpha$.

This notion accommodates standard algebraic constructions. One forms the *product* $\prod_{i \in I} \mathscr{A}_i$ of a family of algebras (of type T) by taking the product of the carrier sets and defining the structural map $s : \mathsf{Fin}\ (\prod_{i \in I} A_i) \to \prod_{i \in I} A_i$ as $s\ x = (s_i\ (\mathsf{Fmap}\ \mathsf{proj}_i\ x))_{i \in I}$. A *stable part* of $\mathscr{A}$ is any set $A' \subseteq A$ such that $\forall x \in \mathsf{Fin}\ A'.\ s\ x \in A'$. Since the intersection of stable parts is a stable part, we can define an algebra $\mathsf{Min}(\mathscr{A})$, the *minimal algebra* of $\mathscr{A}$, by taking its carrier to be the intersection of all stable parts and its structural map to be (the restriction of) s. This corresponds to the notion of subalgebra generated by $\emptyset$. A *morphism* between two algebras $\mathscr{A}$ and $\mathscr{A}'$ is a function $h : A \to A'$ that commutes with the structural maps, in that $\forall x \in \mathsf{Fin}\ A.\ h\ (s\ x) = s'\ (\mathsf{Fmap}\ h\ x)$.

Building the initial algebra of F (an algebra such that for any algebra $\mathscr{A}$, there exists precisely one morphism between it and $\mathscr{A}$) can be naively attempted as follows: First we take $\mathscr{R} = \prod \{\mathscr{A} \mid \mathscr{A}\ \text{algebra}\}$, the product of all algebras. Given an algebra $\mathscr{A}$, there must exist a morphism h from $\mathscr{R}$ to $\mathscr{A}$—the corresponding projection. The restriction of h to $\mathsf{Min}(\mathscr{R})$ is then the desired unique morphism from $\mathsf{Min}(\mathscr{R})$ to $\mathscr{A}$, and $\mathsf{Min}(\mathscr{R})$ is our desired initial algebra.

This naive approach fails since we cannot construct the product of all algebras in HOL—and even if we could, it would not be an algebra itself due to its size. Fortunately, it suffices to define the morphism h from $\mathscr{R}$ not to $\mathscr{A}$ but to $\mathsf{Min}(\mathscr{A})$. Hence, we can take $\mathscr{R}$ as the product of all *minimal* algebras and consider only a complete collection of representatives (up to isomorphism). This is where the bound on F comes into play. If we know that all minimal algebras of all algebras had cardinality smaller than a given bound r_0, we can choose a type T_0 of cardinality r_0 and define $\mathscr{R}$ as the product of all algebras on T_0: $\mathscr{R} = \prod \{\mathscr{A} \mid \mathscr{A} = (T_0, A : T_0\ \mathsf{set}, s : T_0\ \mathsf{F} \to T_0)\ \text{algebra}\}$. A suitable cardinal bound is $r_0 = 2\ \hat{}_{\mathsf{c}}\ k$, where $k = \mathsf{cardSuc}\ (\mathsf{Fbd} +_{\mathsf{c}} |\mathsf{Fin}\ (\mathsf{Field}\ \mathsf{Fbd})|)$. To prove this, we first establish the following consequence of the BNF boundedness property:[3]

Lemma 33. $|A| \geq_{\mathsf{o}} 2 \to |\mathsf{Fin}\ A| \leq_{\mathsf{o}} |A|\ \hat{}_{\mathsf{c}}\ k$.

Theorem 34. *For all algebras $\mathscr{A}$, let M be the carrier of* $\mathsf{Min}(\mathscr{A})$. *Then* $|M| \leq_{\mathsf{o}} 2\ \hat{}_{\mathsf{c}}\ k$.

Proof idea. The definition of $\mathsf{Min}(\mathscr{A})$ performs a construction of M "from above," as an intersection, yielding no cardinality information. We must produce an alternative construction "from below," exploiting the internal structure of F. Let $N = \bigcup_{i \in \mathsf{Field}\ k} N_i$,

[3] Initially, we had maintained a slight variation of this property as an additional BNF requirement [24, Section IV], not realizing that it is redundant. Removing it has simplified the package code substantially.

where each N_i is defined by wellorder recursion as follows: $N_i = \bigcup_{j \in \mathsf{underS}\, k\, i} s \bullet \mathsf{Fin}\, N_j$. We first prove that N is a stable part of $\mathscr{A}$, and hence $M \subseteq N$. Let $x \in \mathsf{Fin}\, N$. Then $\mathsf{Fset}\, x \subseteq N = \bigcup_{i \in \mathsf{Field}\, k} N_i$, and since k is regular by Lemma 29(2), we use Lemma 30 to obtain $i \in \mathsf{Field}\, k$ such that $\mathsf{Fset}\, x \subseteq N_i$ (i.e., $x \in \mathsf{Fin}\, N_i$). Hence, $s\, x \in N_{\mathsf{succ}\, k\, i} \subseteq N$, as desired. Conversely, $N \subseteq M$ follows by wellorder induction. Thus, we have $M = N$. The inequality $|N| \leq_{\mathsf{o}} 2\, \hat{}_{\mathsf{c}}\, k$ follows by wellorder induction, using Lemma 33 and cardinal arithmetic to keep the passage from N_i to $\mathsf{Fin}\, N_i$ bounded. Knowing $|N_i| \leq_{\mathsf{o}} 2\, \hat{}_{\mathsf{c}}\, k$, we obtain $|\mathsf{Fin}\, N_i| \leq_{\mathsf{o}} |N_i|\, \hat{}_{\mathsf{c}}\, k \leq_{\mathsf{o}} (2\, \hat{}_{\mathsf{c}}\, k)\, \hat{}_{\mathsf{c}}\, k =_{\mathsf{o}} 2\, \hat{}_{\mathsf{c}}\, (k \times_{\mathsf{c}} k) =_{\mathsf{o}} 2\, \hat{}_{\mathsf{c}}\, k$. □

Cardinal arithmetic is also used throughout the package for showing that the various constructions on BNFs (composition, initial algebra, and final coalgebra) yield BNFs.

8 Conclusion

We have formalized in Isabelle/HOL a theory of cardinals, proceeding locally and abstractly, up to wellorder isomorphism. The theory has been applied to reason about infinitary objects arising in syntax with bindings and (co)datatypes.

We hope our experiment will be repeated by the other HOL provers,where a theory of cardinals seems as useful as in any other general-purpose framework for mathematics. Indeed, the theory provides working mathematicians with the needed injections and bijections (e.g., between lists over an infinite type, or the square of an infinite type, and the type itself) without requiring them to perform awkward encodings.

An open question is whether the quotient construction of Norrish and Huffman (briefly discussed in the introduction) would have helped the cardinal formalization. With their approach, we would still need to change the underlying type of cardinals to accommodate for increasingly large sizes. HOL offers no way to reason about arbitrary cardinals up to equality, so isomorphism appears to be the right compromise.

Acknowledgment. Tobias Nipkow made this work possible. Stefan Milius and Lutz Schröder shared their elegant proof of Lemma 33 with us. Blanchette is supported by the Deutsche Forschungsgemeinschaft (DFG) project Hardening the Hammer (grant Ni 491/14-1). Popescu is supported by the DFG project Security Type Systems and Deduction (grant Ni 491/13-2) as part of the program Reliably Secure Software Systems (RS^3, priority program 1496). Traytel is supported by the DFG program Program and Model Analysis (PUMA, doctorate program 1480). The authors are listed alphabetically regardless of individual contributions or seniority.

References

1. Bauer, G., Wenzel, M.: Calculational reasoning revisited (an Isabelle/Isar experience). In: TPHOLs 2001. pp. 75–90 (2001)
2. Blanchette, J.C., Popescu, A., Traytel, D.: Formal development associated with this paper. `http://www21.in.tum.de/~traytel/card_devel.tar.gz`
3. Breitner, J.: Free groups. In: Klein, G., Nipkow, T., Paulson, L. (eds.) Archive of Formal Proofs. `http://afp.sf.net/entries/Free-Groups.shtml` (2011)

4. Chang, C.C., Keisler, H.J.: Model Theory. North-Holland (1973)
5. Church, A.: A formulation of the simple theory of types. J. Symb. Logic 5(2), 56–68 (1940)
6. Gordon, M.J.C., Melham, T.F. (eds.): Introduction to HOL: A Theorem Proving Environment for Higher Order Logic. Cambridge University Press (1993)
7. Haftmann, F., Wenzel, M.: Constructive type classes in Isabelle. In: Altenkirch, T., McBride, C. (eds.) TYPES 2006. LNCS, vol. 4502, pp. 160–174. Springer (2007)
8. Harrison, J.: The HOL wellorder library. `http://www.cl.cam.ac.uk/~jrh13/papers/wellorder-library.html` (1992)
9. Harrison, J.: Formalizing basic first order model theory. In: Grundy, J., Newey, M.C. (eds.) TPHOLs '98. LNCS, vol. 1479, pp. 153–170. Springer (1998)
10. Holz, M., Steffens, K., Weitz, E.: Introduction to Cardinal Arithmetic. Birkhäuser Advanced Texts, Birkhäuser (1999)
11. Huffman, B.: Countable ordinals. In: Klein, G., Nipkow, T., Paulson, L. (eds.) Archive of Formal Proofs. `http://afp.sf.net/entries/Ordinal.shtml` (2005)
12. Huffman, B., Urban, C.: Proof pearl: A new foundation for Nominal Isabelle. In: Kaufmann, M., Paulson, L.C. (eds.) ITP 2010. LNCS, vol. 6172, pp. 35–50. Springer (2010)
13. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL: A Proof Assistant for Higher-Order Logic, LNCS, vol. 2283. Springer (2002)
14. Norrish, M., Huffman, B.: Ordinals in HOL: Transfinite arithmetic up to (and beyond) ω_1. In: Blazy, S., Paulin-Mohring, C., Pichardie, D. (eds.) ITP 2013. LNCS, vol. 7998, pp. 133–146. Springer (2013)
15. Paulson, L.C., Blanchette, J.C.: Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers. In: IWIL 2010 (2010)
16. Paulson, L.C., Grabczewski, K.: Mechanizing set theory. J. Autom. Reasoning 17(3), 291–323 (1996)
17. Pitts, A.M.: Nominal logic, a first order theory of names and binding. Inf. Comput. 186(2), 165–193 (2003)
18. Popescu, A.: Ordinals and cardinals in HOL. In: Klein, G., Nipkow, T., Paulson, L. (eds.) Archive of Formal Proofs. `http://afp.sf.net/entries/Ordinals_and_Cardinals.shtml` (2009)
19. Popescu, A.: Contributions to the theory of syntax with bindings and to process algebra. Ph.D. thesis, University of Illinois at Urbana-Champaign (2010)
20. Popescu, A., Gunter, E.L.: Recursion principles for syntax with bindings and substitution. In: Chakravarty, M.M.T., Hu, Z., Danvy, O. (eds.) ICFP '11. pp. 346–358. ACM (2011)
21. Popescu, A., Gunter, E.L., Osborn, C.J.: Strong normalization of System F by HOAS on top of FOAS. In: LICS 2010. pp. 31–40. IEEE (2010)
22. Sternagel, C.: Extending well-founded partial orders to total well-founded orders (2013), archived at `https://lists.cam.ac.uk/pipermail/cl-isabelle-users/2013-February/thread.html`
23. Taylor, P.: Intuitionistic sets and ordinals. J. Symb. Log. 61(3), 705–744 (1996)
24. Traytel, D., Popescu, A., Blanchette, J.C.: Foundational, compositional (co)datatypes for higher-order logic: Category theory applied to theorem proving. In: LICS 2012, pp. 596–605. IEEE (2012)
25. Wisbauer, R.: Foundations of Module and Ring Theory: A Handbook for Study and Research, Algebra, Logic and Applications, vol. 3. Gordon and Breach (1991)

A Proof Ideas

Proof of Theorem 6. We define $f : \alpha \to \beta$ and $g : \alpha \to \mathsf{bool}$ together by wellorder recursion on r. Assume $a \in \mathsf{Field}\ r$ and f and g have already been defined on $\mathsf{underS}\ r\ a$. Let $A = \mathsf{Field}\ s - (f \bullet \mathsf{underS}\ r\ a)$. If $A \neq \emptyset$, we define $f\ a$ to be the s-minimum of A and $g\ a$ to be True; otherwise, we define $g\ a$ to be False and let $f\ a$ take any value.

We first prove by well-founded induction that for all $a \in \mathsf{Field}\ r$, if $\mathsf{False} \notin g \bullet \mathsf{underS}\ r\ a$, then $\mathsf{bij\_betw}\ f\ (\mathsf{underS}\ r\ b)\ (\mathsf{underS}\ s\ (f\ b))$ for all $b \in \mathsf{underS}\ r\ a$. Then we distinguish two cases: If $\mathsf{False} \notin (g \bullet \mathsf{Field}\ r)$, then f establishes an embedding of r in s; otherwise, the inverse of f establishes an embedding in the opposite direction. □

Proof of Theorem 7. Compared with the standard result about the class of ordinals, an additional difficulty arises from the use of embeddings (as opposed to plain inclusions) in the definition of $r <_o s$. Direct reasoning about embeddings would be tedious, since it would require reasoning about limit behavior of embedding composition (pretty much like limit constructions in category theory). Fortunately, this is not necessary, as we can reduce embeddings to inclusions as follows: Let R be a nonempty set of wellorders on α. We need to prove that it has a minimum with respect to $<_o$. We pick $r_0 \in R$, and restrict our attention to $R_0 = \{r \in R.\ r \leq_o r_0\}$: If R_0 has a minimum, then so does R. Next, we show that there exists a surjection H between R_o and the filters of r_0 such that the following hold: $r \leq_o s \leftrightarrow H\ r \subseteq H\ s$; $r <_o s \leftrightarrow H\ r \subset H\ s$; $r =_o s \leftrightarrow H\ r = H\ s$. This effectively brings the problem to the familiar ground of filters on a fixed wellorder with inclusions between them, where the proof proceeds in a standard fashion. □

Proof of Lemma 10. Assume $r : \alpha\ \mathsf{rel}$ and $s : \beta\ \mathsf{rel}$: for the nontrivial implication, we assume the existence of an $f : \alpha \to \beta$ as above. We define a function $g : \alpha \to \beta$ by wellorder recursion on r in the "tightest" possible way, each time choosing the s-smallest element not taken so far (similarly to the definition of f in the proof of Theorem 6). By wellorder induction and the properties of f, we prove that g is always below f. This ensures that g is a (wellorder) embedding of r into s, which proves $r \leq_o s$. □

Proof of Lemma 33. This proof is due to Stefan Milius and Lutz Schröder. Let $k_A = |\mathsf{Fin}\ (\mathsf{Field}\ \mathsf{Fbd})| \times_c |A| \hat{}_c\ \mathsf{Fbd}$. Since $k_A \leq_o |A| \hat{}_c\ k$, it suffices to show that $|\mathsf{Fin}\ A| \leq_o k_A$. To this end, we define $d : \mathsf{Fin}\ (\mathsf{Field}\ \mathsf{Fbd}) \times (\mathsf{Field}\ \mathsf{Fbd} \to A) \to \mathsf{Fin}\ A$ by $d\ (y, f) = \mathsf{Fmap}\ f\ y$. It suffices to prove that d is surjective. Let $x \in \mathsf{Fin}\ A$. Since $|\mathsf{Fset}\ x| \leq_o \mathsf{Fbd}$ by the boundedness of F, we obtain an injective function $g : \mathsf{Fset}\ x \to \mathsf{Field}\ \mathsf{Fbd}$. Let $y = \mathsf{Fmap}\ g\ x$. We choose $f : \mathsf{Field}\ \mathsf{Fbd} \to \mathsf{Fin}\ A$ to be a left inverse of g on $g \bullet \mathsf{Fset}\ x$—this choice is possible since g is injective and $\mathsf{Fset}\ x \subseteq A$. By the functoriality of Fmap,

$$d\ (y, f) = \mathsf{Fmap}\ f\ y = \mathsf{Fmap}\ f\ (\mathsf{Fmap}\ g\ x) = \mathsf{Fmap}\ (f \circ g)\ x = \mathsf{Fmap}\ \mathsf{id}\ x = x$$

and hence y and f witness the surjectivity of d. □

B Case Study: An Order Extension Theorem

Recently, Christian Sternagel started a discussion on the Isabelle mailing list [22] concerning the desire to prove the following theorem: Every well-founded relation p can be extended to a wellorder w. (This was needed in a larger development of a framework for termination proofs.)

There were several proof idea proposals, including the following one involving transfinite recursion. The relation p can be traversed by recurring over a sufficiently large cardinal k, producing larger and larger relations $(v_i)_{i<k}$, as follows (in standard ordinal notation):

(1) $v_0 = \emptyset$
(2) $v_{i+1} = v_i$ extended with a maximum: the minimal element of p not in v_i's field
(3) $v_i = \bigcup_{j<i} v_j$ if i is a limit ordinal

Then w can be taken to be $\bigcup_{j<k} v_j$.

An alternative proposal was based on Zorn's lemma, which is the proof Sternagel eventually formalized. A main reason for not preferring transfinite recursion was apparently the difficulty of using the wellorder recursor wo_rec. The recursor can cover the above definition, but is awkward to use in situations that must distinguish between the successor and the limit case. To address this, we formalized support for successor and limit ordinals, including a customized recursor.

Given r and $a \in \mathsf{Field}\ r$, $\mathsf{aboveS}\ r\ a$ is the set of elements strictly r-above a, $\{b \mid b \neq a \wedge (a,b) \in r\}$. The successor of an element, $\mathsf{succ}\ r\ a$, is the r-minimum of $\mathsf{aboveS}\ r\ a$ (well-defined if $\mathsf{aboveS}\ r\ a \neq \emptyset$). The element a is a *limit element* if it is not a proper successor: $\mathsf{isLim}\ r\ a \equiv \neg\ (\exists b.\ \mathsf{aboveS}\ r\ b \neq \emptyset \wedge \mathsf{succ}\ b = a)$. The characteristic property of limit elements is that they are the suprema of their *strict*-under-intervals:

Lemma 35. $a \in \mathsf{Field}\ r \wedge \mathsf{isLim}\ r\ a \rightarrow a = \mathsf{supr}\ r\ (\mathsf{underS}\ r\ a)$.

The corresponding recursor, $\mathsf{wo\_recZSL}_r : \beta \rightarrow (\alpha \rightarrow \beta \rightarrow \beta) \rightarrow ((\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \beta$, is a modification of wo_rec that distinguishes three cases.

Lemma 36. *Assuming* $\mathsf{adm\_woL}_r\ L$ *and* $a \in \mathsf{Field}\ r$:

(1) $\mathsf{wo\_recZSL}_r\ Z\ S\ L\ 0_r = Z$
(2) $\mathsf{aboveS}\ r\ a \neq \emptyset \rightarrow \mathsf{wo\_recZSL}_r\ Z\ S\ L\ (\mathsf{succ}\ r\ a) = S\ a\ (\mathsf{wo\_recZSL}_r\ Z\ S\ L\ a)$
(3) $\mathsf{isLim}\ r\ a \wedge a \neq 0_r \rightarrow \mathsf{wo\_recZSL}_r\ Z\ S\ L\ a = L\ (\mathsf{wo\_recZSL}_r\ Z\ S\ L)\ a$

This recursor is less bureaucratic than wo_rec since the 0 and successor cases are "statically" known to be admissible—only the limit case needs to be checked, via the admissibility predicate $\mathsf{adm\_woL}_r$, a variation of $\mathsf{adm\_wo}_r$ restricted to limit elements:

$$\mathsf{adm\_woL}_r\ H \equiv \forall f\ g\ a.\ \mathsf{isLim}\ r\ a \wedge (\forall a' \in \mathsf{underS}\ r\ a.\ f\ a' = g\ a') \rightarrow H\ f\ a = H\ g\ a$$

The following proof principle complements the recursion principle of Lemma 36:

Lemma 37. *Assume the following conditions hold:*

(1) $P\,0_r$
(2) $\forall a.\ \mathsf{aboveS}\ r\ a \neq \emptyset \wedge P\ a \rightarrow P\ (\mathsf{succ}\ r\ a)$
(3) $\forall a \in \mathsf{Field}\ r.\ \mathsf{isLim}\ r\ a \wedge a \neq 0_r \wedge (\forall a' \in \mathsf{underS}\ r\ a.\ P\ a') \rightarrow P\ a$

Then $\forall a \in \mathsf{Field}\ r.\ P\ a$.

Now we can faithfully formalize the above three-case definition. Let $k = \mathsf{cmax}\ \mathsf{natLeq}\ |\mathsf{Field}\ p|$ and let T be its type. We define $v : T \rightarrow \alpha\ \mathsf{rel}$ by $v = \mathsf{wo\_recZSL}\ k\ Z\ S\ L$, where

- $Z \equiv \emptyset$
- $S \equiv \lambda a\ r.\ \mathsf{let}\ A = \mathsf{Field}\ p \setminus \mathsf{Field}\ r\ \mathsf{and}\ a = (\varepsilon a.\ \mathsf{minimal}\ p\ A\ a)$
  $\mathsf{in}\ \begin{cases} r \cup \{(a,b) \mid a \in \mathsf{Field}\ r \cup \{b\}\} & \text{if } A \neq \emptyset \\ r & \text{otherwise} \end{cases}$
- $L \equiv \lambda R\ a.\ \bigcup \{R\ b \mid b \in \mathsf{underS}\ k\ a\}$

Next, we consider the following predicates, the second intended as a chain invariant:

$$\mathsf{incl\_on}\ A\ p\ r \equiv \forall a \in A.\ \forall b \in A.\ (a,b) \in p \rightarrow (a,b) \in r$$
$$\mathsf{invar}\ p\ r \equiv \mathsf{Wellorder}\ r \wedge \mathsf{ofilter}\ p\ (\mathsf{Field}\ r) \wedge \mathsf{incl\_on}\ (\mathsf{Field}\ r)\ p\ r$$

Thus, $\mathsf{incl\_on}\ A\ p\ r$ says that on A relation p is included in relation r. We show, by wellorder recursion, that for all $i \in \mathsf{Field}\ k$, $\mathsf{invar}\ p\ (v\ i)$ holds, and that for all i, j, if $(i, j) \in k$ and $i \neq j$, the inclusion $v\ i \subseteq v\ j$ is a wellorder embedding.

Finally, we prove that $w = \bigcup_{i \in \mathsf{Field}\ k} v\ i$ is the desired wellorder extension. It is a wellorder as a union of a wellorder-embedding chain, and $p \subseteq w$ holds because, due to the size of k, $v\ (\mathsf{succ}\ k\ i) = v\ i$ (and hence $p \subseteq v\ i$) for some $i \in \mathsf{Field}\ k$.

Interestingly, the same invariant invar works for the Zorn-based approach, yielding a slightly more compact proof. In the literature, Zorn seems to be generally preferred by algebraists [25], while transfinite recursion/induction is a specialty of logicians [4]. For this particular instance, the latter approach felt more intuitive and (its informal version) was easier to discover and formulate.