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 was 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 identifying ordinals with wellorders, with all concepts and results proved to be invariant under order isomorphism. We also discuss several 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 non-strict 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 other words, Ord conveniently captures the notion of wellorder.

In higher-order logic (HOL, Section 2), the situation is quite different. 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 that provides such general-purpose theorems, as well as some more specialized results and applications. We take 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, to be the minimum ordinals on given underlying sets (Section 4).

The concepts we work with are more abstract than in set theory: Ordinal equality is replaced by a polymorphic relation $=_{\mathsf{o}}$ stating the existence of an order isomorphism, and membership is replaced by a polymorphic operator $<_{\mathsf{o}}$ stating the existence of a strict order embedding (with its non-strict counterpart $\leq_{\mathsf{o}}$ removing the requirement that the order embedding be strict). This abstract view takes more effort to maintain than the convenient concrete implementation from set theory, 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 $r_1 +_{\mathsf{c}} r_2$, 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, and then $+_{\mathsf{c}}$ is proved compatible with $=_{\mathsf{o}}$ and $\leq_{\mathsf{o}}$. Moreover, since the collection of all ordinals does not fit in one type, we must predict the size of the constructed objects and choose large enough support types for them.

The overcoming of those impediments allows us to validate 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.*

The truth of this thesis was not clear to us when we started the formalization, since we could not find any textbook or formalization 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 was incorporated in the *Archive of Formal Proofs* [16] 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 [17], and cardinal arithmetic was developed to support Isabelle's (co)datatype package [22] (Section 5).

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 [1].

Related Work. Ordinals, unlike cardinals, have been formalized in HOL before. Harrison [7] formalized ordinals in HOL88 and provesd theorems such as Zermelo, Zorn, and transfinite induction. Huffman [10] formalized countable ordinals in Isabelle/HOL, including arithmetics and the Veblen hierarchies; the countability assumption made it possible to fix a type of ordinals. Recently, Norrish and Huffman [13] independently redeveloped in HOL4 much of our theory of ordinals. But while Norrish and Huffman focus on establishing ordinals as quotients of wellorders under isomorphism and develop some deeper ordinal arithmetics including Cantor normal form, we see the ordinals mostly as a stepping stone toward the cardinals and focus on these.

Beyond HOL, Paulson and Grabczewski [14] 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 invest care in avoiding this axiom for part of the cardinal theory. If our case, the Hilbert 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 or no interest in developing ordinals and cardinals in a weaker setting than ZF. An exception is Taylor [21], 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 and Isabelle/HOL

By HOL we mean classical higher-order logic with Hilbert choice, the axiom of infinity, and rank-1 polymorphism. HOL is based on Church's simple type theory [4]. It is the logic of Gordon's system of the same name [5] 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. Our formalization is performed in Isabelle/HOL [12], an implementation enriched with Haskell-style type classes [6].

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 same notation is used for polymorphic types—e.g., α list denotes what would be more precisely written as $\forall\alpha.\ \alpha$ list. Type constructors may have any numeric arity. The type constructors $\alpha \to \beta$, $\alpha + \beta$, and $\alpha \times \beta$, for function space, sum, and product. 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 $c : \tau$. Definitions are introduced using the $\equiv$ symbol.

The following types and constants from the Isabelle library are heavily used in our formalization. $\mathsf{UNIV} : \alpha$ set is the universe set, i.e., 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 $\mathsf{Inr} : \beta \to \alpha + \beta$.

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

The type $(\alpha \times \alpha)$ set of binary relations on α is abbreviated to α rel. $\mathsf{Id} : \alpha$ rel is the identity relation. Given $r : \alpha$ rel, Field $r : \alpha$ 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 $A : \alpha$ set and $r : \alpha$ rel:

Reflexive	$\mathsf{refl\_on}\ A\ r \equiv r \subseteq A \times A \wedge \forall x \in A.\ (x, x) \in r$
Symmetric	$\mathsf{sym}\ r \equiv \forall a\ b.\ (a, b) \in r \to (b, a) \in r$
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. Also, well-orderedness is usually defined as partial-orderedness plus the existence of a smallest element for each nonempty subset in its field.

Note that refl_on $A\ r$ (thus also well_order_on $A\ r$) implies Field $r = A$. We abbreviate well_order_on (Field r) r to Well_order r and well_order_on UNIV r to well_order r.

3 Ordinals

This section give some highlights of our formalization of ordinals. 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. 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. Similarly, $\mathsf{underS}\ r\ a$ gives the elements strictly under a with respect to r. We call these *under-* and *strict-under*-intervals:

$$\mathsf{under} : \alpha\ \mathsf{rel} \to \alpha \to \alpha\ \mathsf{set} \qquad \mathsf{underS} : \alpha\ \mathsf{rel} \to \alpha \to \alpha\ \mathsf{set}$$
$$\mathsf{under}\ r\ a \equiv \{b \mid (b,a) \in r\} \qquad \mathsf{underS}\ r\ a \equiv \{b \mid (b,a) \in r \wedge b \neq a\}$$

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 already supported by Isabelle's library. We define slight variations of these notions tailored for wellorders.

Lemma 1 (Wellorder induction). *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 we do not prove a property, but define a function f on $\mathsf{Field}\ r$. It suffices that, for each $a \in \mathsf{Field}\ r$, we assume f already defined on $\mathsf{underS}\ r\ a$ and define $f\ a$. 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 supported by 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 (Wellorder recursion). *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:

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

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) $\mathsf{ofilter}\ r\ A\ \leftrightarrow\ A = \mathsf{Field}\ r\ \vee\ (\exists a \in \mathsf{Field}\ r.\ A = \mathsf{underS}\ r\ a)$.

3.2 Embedding and Isomorphism

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

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

We read $\mathsf{embed}\ r\ s\ f$ as "f embeds r into s"—this 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 way to define embedding, namely, stating that f is injective, order preserving, and maps $\mathsf{Field}\ r$ into a filter of s, is proved as a lemma (where $\mathsf{compat}\ r\ s\ f$ expresses order preservation of f, i.e.,$\forall a\ b.\ (a, b) \in r \to (f\ a, f\ b) \in s$).

Lemma 4. $\mathsf{embed}\ r\ s\ f\ \leftrightarrow\ \mathsf{compat}\ r\ s\ f\ \wedge\ \mathsf{inj\_on}\ f\ (\mathsf{Field}\ r)\ \wedge\ \mathsf{ofilter}\ s\ (f \bullet \mathsf{Field}\ r)$

Every embedding is either an (order) isomorphism, $\mathsf{iso}\ r\ s\ f$, or a strict embedding, $\mathsf{embedS}\ r\ s\ f$, depending on whether f is a bijection or not. These notions yield the following relations between wellorders:

$$\begin{array}{l} \leq_{\mathsf{o}},\ <_{\mathsf{o}},\ =_{\mathsf{o}} : (\alpha\ \mathsf{rel} \times \beta\ \mathsf{rel})\ \mathsf{set} \\ \leq_{\mathsf{o}}\ \equiv\ \{(r, s).\ \mathsf{Well\_order}\ r\ \wedge\ \mathsf{Well\_order}\ s\ \wedge\ \exists f.\ \mathsf{embed}\ r\ s\ f\} \\ <_{\mathsf{o}}\ \equiv\ \{(r, s).\ \mathsf{Well\_order}\ r\ \wedge\ \mathsf{Well\_order}\ s\ \wedge\ \exists f.\ \mathsf{embedS}\ r\ s\ f\} \\ =_{\mathsf{o}}\ \equiv\ \{(r, s).\ \mathsf{Well\_order}\ r\ \wedge\ \mathsf{Well\_order}\ s\ \wedge\ \exists f.\ \mathsf{iso}\ r\ s\ f\} \end{array}$$

We abbreviate $(r, s) \in \leq_{\mathsf{o}}$ by $r \leq_{\mathsf{o}} s$, and similarly for $<_{\mathsf{o}}$ and $=_{\mathsf{o}}$. Thus, $r \leq_{\mathsf{o}} s$ means that r is smaller than or equal to s, in that it can be embedded in s, and similarly $r <_{\mathsf{o}} s$ and $r =_{\mathsf{o}} s$ for strict embedding and isomorphism. These relations are well-behaved.

Theorem 1. *1.* $r =_{\mathsf{o}} r$
2. $r =_{\mathsf{o}} s \to s =_{\mathsf{o}} r$
3. $r =_{\mathsf{o}} s\ \wedge\ s =_{\mathsf{o}} t \to r =_{\mathsf{o}} t$
4. $r \leq_{\mathsf{o}} r$
5. $r \leq_{\mathsf{o}} s\ \wedge\ s \leq_{\mathsf{o}} t \to r \leq_{\mathsf{o}} t$
6. $\neg\, r <_{\mathsf{o}} r$
7. $r <_{\mathsf{o}} s\ \wedge\ s <_{\mathsf{o}} t \to r <_{\mathsf{o}} t$
8. $r \leq_{\mathsf{o}} s\ \leftrightarrow\ r <_{\mathsf{o}} s \vee r =_{\mathsf{o}} s$
9. $r =_{\mathsf{o}} s\ \leftrightarrow\ r \leq_{\mathsf{o}} s\ \wedge\ s \leq_{\mathsf{o}} r$

In particular, if we restrict the types of these relations from $(\alpha\ \mathsf{rel} \times \beta\ \mathsf{rel})\ \mathsf{set}$ to $(\alpha\ \mathsf{rel})\ \mathsf{rel}$ (taking $\beta = \alpha$), we obtain that $=_{\mathsf{o}}$ is an equivalence (1–3) and $\leq_{\mathsf{o}}$ is a preorder (4–5); moreover $<_{\mathsf{o}}$ is the strict version of $\leq_{\mathsf{o}}$ modulo the "equality" $=_{\mathsf{o}}$ (6–8). In fact, if think of $=_{\mathsf{o}}$ as the equality, $\leq_{\mathsf{o}}$ becomes a partial order (9), and correspondingly $<_{\mathsf{o}}$ 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 (consequently) 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 $\leq_{\mathsf{o}}$ to be total.

Theorem 2. $r \leq_{\mathsf{o}} s\ \vee\ r \leq_{\mathsf{o}} s$

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. By contrast, here we have to 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 in case that $r \leq_o s$, and is the inverse of an embedding of s into r otherwise. □

It remains to show that this total order is in effect a wellorder, or, equivalently its strict counterpart $<_o$ is well-founded (note the annotation restricting the type of $<_o$):

Theorem 3. $\mathsf{wf}\ (<_o : (\alpha\ \mathsf{rel})\ \mathsf{rel})$

Theorems 1, 2, and 3 yield directly 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

Holz et al. [9], like most textbooks, define operations on ordinals—sum, product, exponentiation—by transfinite recursion. On the other hand, these operations admit direct (nonrecursive) definitions, which we prefer since they are particularly suited to arbitrary wellorders. (In [9], these direct definitions are depicted as "visual" descriptions.)

We define the ordinal sum $+_o : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha + \beta)\ \mathsf{rel}$ by concatenating the two argument wellorders r and s such that elements of $\mathsf{Field}\ r$ come below those of $\mathsf{Field}\ s$.

$$r +_o s \equiv\ (\mathsf{Inl} \otimes \mathsf{Inl}) \bullet r \cup (\mathsf{Inr} \otimes \mathsf{Inr}) \bullet s \cup \{(\mathsf{Inl}\ x, \mathsf{Inr}\ y).\ x \in \mathsf{Field}\ r \wedge y \in \mathsf{Field}\ s\}$$

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

Ordinal multiplication $\times_o : \alpha\ \mathsf{rel} \to \beta\ \mathsf{rel} \to (\alpha \times \beta)\ \mathsf{rel}$ is defined as the anti-lexicographic ordering on the product type.

$$\begin{aligned} r \times_o s \equiv\ \{((x_1, y_1), (x_2, y_2)).\ & x_1, x_2 \in \mathsf{Field}\ r \wedge y_1, y_2 \in \mathsf{Field}\ s \wedge \\ & (y_1 \neq y_2 \wedge (y_1, y_2) \in s \vee y_1 = y_2 \wedge (x_1, x_2) \in r)\} \end{aligned}$$

For ordinal exponentiation, $r\ \hat{}_o\ s$, we consider functions of finite support from the field of s to the field of r. As all function in HOL are total, we represent the restricted domain $\mathsf{Field}\ (s : \beta\ \mathsf{rel})$ by considering only functions that are constant (equal to a particular unspecified value $\bot$) outside of the domain. We thus define the operator $\mathsf{Func} : \beta\ \mathsf{set} \to \alpha\ \mathsf{set} \to (\beta \to \alpha)\ \mathsf{set}$, where $\mathsf{Func}\ B\ A$ gives the set of all such functions with domain B and range A, namely, $\{f.\ f \bullet B \subseteq A \wedge (\forall x \notin B.\ f\ x = \bot)\}$. Additionally, finite support means that only finitely many elements of $\mathsf{Field}\ s$ are mapped to elements different from the minimal element 0_r of the wellorder r: $\mathsf{FinFunc}\ B\ A \equiv \mathsf{Func}\ B\ A\ \cap \{f.\ \mathsf{finite}\ \{x \in B.\ f\ x \neq 0_r\}\}$.

Now we are ready to define the exponentiation of r to s. Its underlying set consists of functions of finite support between $\mathsf{Field}\ s$ and $\mathsf{Field}\ r$. The order between two such functions f and g is defined as follows. Assuming $f \neq g$, thanks to the finite support there exists a maximum (with respect to s) $z \in \mathsf{Field}\ s$ such that $f\ z \neq g\ z$. If $(f\ z, g\ z) \in r$, we declare f smaller than g; otherwise (meaning $(g\ z, f\ z) \in r$ as r is a wellorder and hence total) we declare g smaller than f.

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

All these constructions yield wellorders. Moreover, they satisfy the following arithmetic properties, where 0 and 1 are the empty and singleton wellorder, respectively.

Theorem 4. $\mathsf{Well\_order}\ (r +_{\mathsf{o}} s)$; $\mathsf{Well\_order}\ (r \times_{\mathsf{o}} s)$; $\mathsf{Well\_order}\ (r\ {\hat{}}_{\mathsf{o}}\ s)$

Lemma 5 (Lemma 1.4.3 in [9]).

$$\begin{array}{ll}
0 +_{\mathsf{o}} r =_{\mathsf{o}} r =_{\mathsf{o}} r +_{\mathsf{o}} 0 & (r +_{\mathsf{o}} s) +_{\mathsf{o}} t =_{\mathsf{o}} r +_{\mathsf{o}} (s +_{\mathsf{o}} t) \\
s \leq_{\mathsf{o}} r +_{\mathsf{o}} s & r \leq_{\mathsf{o}} s \to r +_{\mathsf{o}} t \leq_{\mathsf{o}} s +_{\mathsf{o}} t \\
s <_{\mathsf{o}} t \to r +_{\mathsf{o}} s <_{\mathsf{o}} r +_{\mathsf{o}} t & \\
0 \times_{\mathsf{o}} r =_{\mathsf{o}} 0 =_{\mathsf{o}} r \times_{\mathsf{o}} 0 & 1 \times_{\mathsf{o}} r =_{\mathsf{o}} r =_{\mathsf{o}} r \times_{\mathsf{o}} 1 \\
(r \times_{\mathsf{o}} s) \times_{\mathsf{o}} t =_{\mathsf{o}} r \times_{\mathsf{o}} (s \times_{\mathsf{o}} t) & r \times_{\mathsf{o}} (s +_{\mathsf{o}} t) =_{\mathsf{o}} r \times_{\mathsf{o}} s +_{\mathsf{o}} r \times_{\mathsf{o}} t \\
r \leq_{\mathsf{o}} s \to r \times_{\mathsf{o}} t \leq_{\mathsf{o}} s \times_{\mathsf{o}} t & 0 <_{\mathsf{o}} r \wedge s <_{\mathsf{o}} t \to r \times_{\mathsf{o}} s <_{\mathsf{o}} r \times_{\mathsf{o}} t \\
0 <_{\mathsf{o}} r \to 0\ {\hat{}}_{\mathsf{o}}\ r =_{\mathsf{o}} 0 & 1\ {\hat{}}_{\mathsf{o}}\ r =_{\mathsf{o}} 1 \\
(r\ {\hat{}}_{\mathsf{o}}\ s)\ {\hat{}}_{\mathsf{o}}\ t =_{\mathsf{o}} r\ {\hat{}}_{\mathsf{o}}\ (s \times_{\mathsf{o}} t) & r\ {\hat{}}_{\mathsf{o}}\ s +_{\mathsf{o}} t =_{\mathsf{o}} r\ {\hat{}}_{\mathsf{o}}\ s \times_{\mathsf{o}} r\ {\hat{}}_{\mathsf{o}}\ t \\
r \leq_{\mathsf{o}} s \to r\ {\hat{}}_{\mathsf{o}}\ t \leq_{\mathsf{o}} s\ {\hat{}}_{\mathsf{o}}\ t & 1 <_{\mathsf{o}} r \wedge s <_{\mathsf{o}} t \to r\ {\hat{}}_{\mathsf{o}}\ s <_{\mathsf{o}} r\ {\hat{}}_{\mathsf{o}}\ t \\
1 <_{\mathsf{o}} r \to s \leq_{\mathsf{o}} r\ {\hat{}}_{\mathsf{o}}\ s &
\end{array}$$

A benefit 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. In our case, we went for direct definitions, and correspondingly aimed at direct proofs via the explicit indication of suitable isomorphisms or embeddings as in the definitions of $=_{\mathsf{o}}$, $\leq_{\mathsf{o}}$, $<_{\mathsf{o}}$. This approach works fine for the equations ($=_{\mathsf{o}}$ 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 $0 <_{\mathsf{o}} r \wedge s <_{\mathsf{o}} t \to r \times_{\mathsf{o}} s <_{\mathsf{o}} r \times_{\mathsf{o}} t$, we use the definition of $<_{\mathsf{o}}$ to obtain from $s <_{\mathsf{o}} t$ a strict embedding f of s into t; then the desired strict embedding of $r \times_{\mathsf{o}} s$ into $r \times_{\mathsf{o}} t$ is $\mathsf{id} \otimes f$.

By contrast, left-monotonicity properties such as $r \leq_{\mathsf{o}} s \to r \times_{\mathsf{o}} t \leq_{\mathsf{o}} s \times_{\mathsf{o}} t$ no longer follow that smoothly—it is not clear how to produce an embedding of $r \times_{\mathsf{o}} t$ into $s \times_{\mathsf{o}} t$ from one of r into s. To conveniently handle left-monotonicity, we introduced an alternative characterization of $\leq_{\mathsf{o}}$:

Lemma 6. $r \leq_{\mathsf{o}} s \leftrightarrow \mathsf{Well\_order}\ r \wedge \mathsf{Well\_order}\ s \wedge (\exists f.\ \forall a \in \mathsf{Field}\ r.\ f\ a \in \mathsf{Field}\ s \wedge f \bullet \mathsf{underS}\ r\ a \subseteq \mathsf{underS}\ s\ (f\ a))$

Thus, in order to show $r \leq_{\mathsf{o}} s$, it suffices to provide an order embedding, not necessarily a wellorder one, i.e., not necessarily an embedding of Field r *as a filter of* s. This is a dramatic simplification of the task of proving $r \leq_{\mathsf{o}} s$ (surprisingly not stated in textbooks). With it, we can readily prove left-monotonicity properties. E.g., to show $r \times_{\mathsf{o}} t \leq_{\mathsf{o}} s \times_{\mathsf{o}} t$ assuming an embedding f of r into s, we now take the luxury of defining a plain order embedding, the obvious candidate, $f \otimes \mathsf{id}$, doing the job.

Note that right-monotonicity holds for $<_{\mathsf{o}}$ (a fortiori for $\leq_{\mathsf{o}}$), while left-monotonicity only holds for $\leq_{\mathsf{o}}$. This is fortunate, since Lemma 6 is not adaptable to $<_{\mathsf{o}}$ either.

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 arithmetics.

4.1 Bootstrapping

We define cardinal orders (cardinals) on a set as special wellorders, namely, those 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" [9, p. 42].

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

Similarly to wellorders, we abbreviate $\mathsf{card\_order\_on}\ (\mathsf{Field}\ r)\ r$ by $\mathsf{Card\_order}\ r$ and $\mathsf{card\_order\_on}\ \mathsf{UNIV}\ r$ by $\mathsf{card\_order}\ r$. Note that, by definition, $\mathsf{card\_order\_on}\ A\ r$ implies $A = \mathsf{Field}\ r$, and therefore when we wish to omit A we can simply write $\mathsf{Card\_order}\ r$.

In general, cardinals are useful through their capability of measuring sets. In our setting, we first prove that for every set A (on any type), there exists a cardinal on it. This cardinal is not unique, but it is unique up to isomorphism.

Theorem 5. *1.* $\exists r.\ \mathsf{card\_order\_on}\ A\ r$
2. $\mathsf{card\_order\_on}\ A\ r \wedge \mathsf{card\_order\_on}\ A\ s\ \to\ r =_o s$

We are now ready to define the cardinality of a set $|\_| : \alpha\ \mathsf{set} \to \alpha\ \mathsf{rel}$:

$$|A|\ \equiv\ \mathsf{SOME}\ r.\ \mathsf{card\_order\_on}\ A\ r$$

Using Hilbert choice, we did pick one particular cardinal order on A (which is possible by Th. 5.1). The choice is irrelevant by Th. 5.2, and we can prove that the cardinality operator behaves as expected, in particular, it is monotonic.

Lemma 7. *1.* $\mathsf{card\_order\_on}\ A\ |A|$
2. $\mathsf{Field}\ |A| = A$
3. $A \subseteq B\ \to\ |A| \leq_o |B|$
4. $r \leq_o s\ \to\ |\mathsf{Field}\ r| \leq_o |\mathsf{Field}\ s|$

Cardinalities of sets were defined in an order-theoretic fashion, but we can now prove that they correspond to the more elementary comparisons in terms of functions.

Theorem 6. *1.* $|A| =_o |B|\ \leftrightarrow\ (\exists f.\ \mathsf{bij\_betw}\ f\ A\ B)$
2. $|A| \leq_o |B|\ \leftrightarrow\ (\exists f.\ \mathsf{inj\_on}\ f\ A\ \wedge\ f \bullet A \subseteq B)$
3. $A \neq \emptyset\ \to\ (|A| \leq_o |B|\ \leftrightarrow\ (\exists g.\ g \bullet B \subseteq A))$

Together with theorem 2 this allows to prove in HOL the aforementioned interesting order-free fact for the working mathematician.

Theorem 7. *For any two types σ and τ, one is embeddable in the the other, in that there exists either an injection from σ to τ or one 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), α set (powertype), α list (lists). In order to provide more generally usable results, we actually look at the homonymous set-based versions of these constructors, which take the form of polymorphic constants:[3]

$$\begin{array}{ll} + : \alpha\ \mathsf{set} \to \beta\ \mathsf{set} \to (\alpha+\beta)\ \mathsf{set}, & A + B \equiv \{\mathsf{Inl}\ a \mid a \in A\} \cup \{\mathsf{Inr}\ b \mid b \in B\} \\ \times : \alpha\ \mathsf{set} \to \beta\ \mathsf{set} \to (\alpha\times\beta)\ \mathsf{set}, & A \times B \equiv \{(a, b) \mid a \in A \wedge b \in B\} \\ \mathsf{Pow} : \alpha\ \mathsf{set} \to (\alpha\ \mathsf{set})\ \mathsf{set}, & \mathsf{Pow}\ A \equiv \{X \mid X \subseteq A\} \\ \mathsf{lists} : \alpha\ \mathsf{list} \to (\alpha\ \mathsf{list})\ \mathsf{set}, & \mathsf{lists}\ A \equiv \{as \mid \mathsf{set}\ as \subseteq A\} \end{array}$$

The cardinalities of these operators are compatible with isomorphism and embedding.

Lemma 8. *Let K be any of $+$, $\times$, Pow, lists, $n \in \{1, 2\}$ be its arity, and θ be either of $=_{\mathsf{o}}$, $\leq_{\mathsf{o}}$. If $\forall i \in \{1, \ldots, n\}.\ |A_i|\ \theta\ |B_i|$, then $|K\ A_1\ \ldots\ A_n)|\ \theta\ |K\ B_1\ \ldots\ B_n|$.*

In addition, we have the following ordering between cardinalities.

Lemma 9.

1. $|A| \leq_{\mathsf{o}} |A + B|$
2. $|A + B| \leq_{\mathsf{o}} |A \times B|$ [4]
3. $|A| <_{\mathsf{o}} |\mathsf{Pow}\ A|$
4. $|A| \leq_{\mathsf{o}} |\mathsf{lists}\ A|$

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

Lemma 10. *Assume* infinite A. *Then:*

1. $|A \times A| =_{\mathsf{o}} |A|$
2. $|A| =_{\mathsf{o}} |\mathsf{lists}\ A|$
3. $|A + B| =_{\mathsf{o}}$ *if* $A \leq_{\mathsf{o}} B$ *then* $|A|$ *else* $|B|$
4. $B \neq \emptyset \to |A \times B| =_{\mathsf{o}}$ *if* $A \leq_{\mathsf{o}} B$ *then* $|A|$ *else* $|B|$ ∎

Amongst those results, the property of products (Lemma 10.1) required significant formalization effort: its proof goes through the so-called bounded product construction—this is extensively discussed in [14], in the context of a formalization within Isabelle/ZF.

In Isabelle/HOL, $\times$ is an instance of the indexed sum (disjoint union) operator $\mathsf{SIG} : \alpha\ \mathsf{set} \to (\alpha \to \beta\ \mathsf{set}) \to (\alpha \times \beta)\ \mathsf{set}$, defined by $\mathsf{SIG}\ A\ B$ (written $\mathsf{SIG}_{a \in A}\ B\ a$) $\equiv \bigcup_{a \in A} \bigcup_{b \in B\ a} (a, b)$. The above properties for $\times$ carry over to SIG as well. The latter operator provides support for proving cardinality bounds of indexed unions:

Lemma 11.

1. $|\bigcup_{i \in I} A\ i| \leq_{\mathsf{o}} |\mathsf{SIG}_{i \in I}\ A\ i|$
2. $\mathsf{infinite}\ B \wedge |I| \leq_{\mathsf{o}} |B| \wedge (\forall i \in I.\ |A\ i| \leq_{\mathsf{o}} |B|) \to |\bigcup_{i \in I} A\ i| \leq_{\mathsf{o}} |B|$

[3] For a large class of type constructors (including the ones discussed here), set-based versions can be extracted uniformly [22]—see also Section 5.2.

[4] if both A and B have at least two elements

4.3 $\aleph_0$ and the Finite Cardinals

Our $\aleph_0$ is the existing constant natLeq : nat rel—the standard order on natural numbers. It behaves as expected for $\aleph_0$, 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 12. *1.* $\mathsf{infinite}\, A \leftrightarrow \mathsf{natLeq} \leq_o |A|$
2. $\mathsf{Card\_order}\ \mathsf{natLeq}$
3. $\mathsf{Card\_order}\ r \wedge \mathsf{infinite}\ (\mathsf{Field}\ r) \rightarrow r \leq_o \mathsf{natLeq}$
4. $\mathsf{ofilter}\ \mathsf{natLeq}\ A \leftrightarrow A = (\mathsf{UNIV} : \mathsf{nat\ set}) \vee (\exists n.\, A = \{0, \ldots, n\})$

We use these to define the finite cardinals as restrictions of natLeq: $\mathsf{natLeq\_on}\ n \equiv \mathsf{natLeq} \cap \{0, \ldots, n\} \times \{0, \ldots, n\}$. We prove that that these indeed behave like the finite cardinals (up to isomorphism):

Lemma 13. *1.* $\mathsf{card\_order}\ (\mathsf{natLeq\_on}\ n)$
2. $\mathsf{finite}\ A \leftrightarrow (\exists n.\, |A| =_o \mathsf{natLeq\_on}\ n)$
3. $\mathsf{finite}\ A \wedge |A| =_o |B| \rightarrow \mathsf{finite}\ B$

4.4 Some Cardinal Arithmetic

To define cardSuc r, the successor of a cardinal $r : \alpha$ rel, we first choose a type which we know is large enough to contain a cardinal greater than r, namely, α set. Then we define what it means to be a successor cardinal: to be a cardinal that is greater than r and, for now at least amongst all cardinals on the chosen type α set, to be $\leq_o$-minimal.

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

Thanks to the choice of the codomain type and Th. 3, we know such a cardinal exists.

Lemma 14. $\exists s.\ \mathsf{isCardSuc}\ r\ s$

This allows us to define $\mathsf{cardSuc} : \alpha\ \mathsf{rel} \rightarrow (\alpha\ \mathsf{set})\ \mathsf{rel}$ to assign any such cardinal to r and infer that it indeed satisfies its "defining" properties:

$$\mathsf{cardSuc}\ r \equiv \mathsf{SOME}\ s.\ \mathsf{isCardSuc}\ r\ s$$

Lemma 15. $\mathsf{isCardSuc}\ r\ (\mathsf{cardSuc}\ r)$

However, this is not yet good enough. We need to prove that cardSuc r is minimal not only amongst the cardinals on α set, but amongst all cardinals—this is achieved by a tedious process of making isomorphic copies. We obtain the desired characteristic properties of successor cardinals in full generality.

Theorem 8. *Assume* $\mathsf{Card\_order}\ (r : \alpha\ \mathsf{rel})$ *and* $\mathsf{Card\_order}\ (t : \beta\ \mathsf{rel})$. *Then:*

1. $r <_{\mathsf{o}} \mathsf{cardSuc}\ r$ 2. $r <_{\mathsf{o}} t \rightarrow \mathsf{cardSuc}\ r \leq_{\mathsf{o}} t$

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

Theorem 9. *Assume* Card_order r *and* Card_order s. *Then:*

1. $\mathsf{cardSuc}\ r =_{\mathsf{o}} \mathsf{cardSuc}\ s \leftrightarrow r =_{\mathsf{o}} s$ 2. $\mathsf{cardSuc}\ r <_{\mathsf{o}} \mathsf{cardSuc}\ s \leftrightarrow r <_{\mathsf{o}} s$

Thus, 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 at that point. But then we proved the characteristic property in full generality, and finally proved that the notion is compatible with $=_{\mathsf{o}}$ and $\leq_{\mathsf{o}}$. This route of introducing cardinality operators is certainly more bureaucratic than in set theory, but achieves the desired effect. We follow this route with all the standard cardinal operations, e.g, $+_{\mathsf{c}} : \alpha\ \mathsf{rel} \rightarrow \beta\ \mathsf{rel} \rightarrow (\alpha + \beta)\ \mathsf{rel}$, for which we prove the basic arithmetic properties.

Lemma 16 (Lemma 1.5.10 in [9]).

$(r +_{\mathsf{c}} s) +_{\mathsf{c}} t =_{\mathsf{o}} r +_{\mathsf{c}} (s +_{\mathsf{c}} t)$ $\quad r +_{\mathsf{c}} s =_{\mathsf{o}} s +_{\mathsf{c}} r$

$(r \times_{\mathsf{c}} s) \times_{\mathsf{c}} t =_{\mathsf{o}} r \times_{\mathsf{c}} (s \times_{\mathsf{c}} t)$ $\quad r \times_{\mathsf{c}} s =_{\mathsf{o}} s \times_{\mathsf{c}} r$

$r \times_{\mathsf{c}} 0 =_{\mathsf{o}} 0$ $\quad r \times_{\mathsf{c}} 1 =_{\mathsf{o}} r$

$r \times_{\mathsf{c}} (s +_{\mathsf{c}} t) =_{\mathsf{o}} r \times_{\mathsf{c}} s +_{\mathsf{c}} r \times_{\mathsf{c}} t$

$r\ \hat{}_{\mathsf{c}}\ (s +_{\mathsf{c}} t) =_{\mathsf{o}} r\ \hat{}_{\mathsf{c}}\ s \times_{\mathsf{c}} r\ \hat{}_{\mathsf{c}}\ t$ $\quad (r\ \hat{}_{\mathsf{c}}\ s)\ \hat{}_{\mathsf{c}}\ t =_{\mathsf{o}} r\ \hat{}_{\mathsf{c}}\ (s \times_{\mathsf{c}} t)$

$(r \times_{\mathsf{c}} s)\ \hat{}_{\mathsf{c}}\ t =_{\mathsf{o}} r\ \hat{}_{\mathsf{c}}\ t \times_{\mathsf{c}} s\ \hat{}_{\mathsf{c}}\ t$ $\quad \neg\, r =_{\mathsf{o}} 0 \rightarrow r\ \hat{}_{\mathsf{c}}\ 0 =_{\mathsf{o}} 1 \wedge 0\ \hat{}_{\mathsf{c}}\ r =_{\mathsf{o}} 0$

$r\ \hat{}_{\mathsf{c}}\ 1 =_{\mathsf{o}} r$ $\quad 1\ \hat{}_{\mathsf{c}}\ r =_{\mathsf{o}} 1$ $\quad r\ \hat{}_{\mathsf{c}}\ 2 =_{\mathsf{o}} r \times_{\mathsf{c}} r$

$r \leq_{\mathsf{o}} s \wedge t \leq_{\mathsf{o}} u \rightarrow r +_{\mathsf{c}} t \leq_{\mathsf{o}} s +_{\mathsf{c}} u$ $\quad r \leq_{\mathsf{o}} s \wedge t \leq_{\mathsf{o}} u \rightarrow r \times_{\mathsf{c}} t \leq_{\mathsf{o}} s \times_{\mathsf{c}} u$

$r \leq_{\mathsf{o}} s \wedge t \leq_{\mathsf{o}} u \wedge \neg =_{\mathsf{o}} 0 \rightarrow r\ \hat{}_{\mathsf{c}}\ t \leq_{\mathsf{o}} s\ \hat{}_{\mathsf{c}}\ u$

Another useful cardinal operation is the maximum of two cardinals, $\mathsf{cmax}\ r\ s$, which is well-defined by the totality of $\leq_{\mathsf{o}}$. Thanks to Lemma 10.1, for infinite cardinals it behaves like both sum and product:

Lemma 17. $\mathsf{infinite}\ (\mathsf{Field}\ r) \vee \mathsf{infinite}\ (\mathsf{Field}\ s) \rightarrow \mathsf{cmax}\ r\ s =_{\mathsf{o}} r +_{\mathsf{c}} s =_{\mathsf{o}} r \times_{\mathsf{c}} s$

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 called *regular*, written $\mathsf{regular}\ r$, if $\forall A.\ A \subseteq \mathsf{Field}\ r \wedge \mathsf{cofinal}\ A\ r \rightarrow |A| =_{\mathsf{o}} r$.

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

Lemma 18. 1. $\mathsf{regular}\ \mathsf{natLeq}$
2. $\mathsf{Card\_order}\ r \wedge \mathsf{infinite}\ (\mathsf{Field}\ r) \rightarrow \mathsf{regular}\ (\mathsf{cardSuc}\ r)$

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 boils down to inclusion in *one* of the sets in the chain.

Lemma 19. *Assume* $\mathsf{Card\_order}\ r$*,* $\mathsf{regular}\ r$*,* $\forall i\ j.\ (i,\ j) \in r\ \rightarrow\ A\ i \subseteq A\ j$*,* $|B| <_o r$*, 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 20. *Assume* $\mathsf{Card\_order}\ r$*,* $\mathsf{regular}\ r$*,* $|I| <_o r$*, and* $\forall i \in I.\ |A\ i| <_o r$*.*
Then $|\bigcup_{i \in I} A\ i| <_o r$*.*

5 Applications

Here we describe applications of our theory of cardinals in larger developments.

5.1 Syntax with Bindings

In his Ph.D. thesis [17–19], Popescu has formalized a general theory of syntax with bindings, parameterized over a binding signature with possibly infinitary operation symbols. For handling infinitary syntax, cardinal support was crucially needed. We illustrate the problem and solution on an example. Let index and var be types representing indexes and variables, respectively, and consider the following datatype of terms:

$$\texttt{datatype}\ \mathsf{term} = \mathsf{Var}\ \mathsf{var}\ |\ \mathsf{Lam}\ \mathsf{var}\ \mathsf{term}\ |\ \mathsf{Sum}\ (\mathsf{index} \rightarrow \mathsf{term})$$

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

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

To avoid capture, the Lam-clause for substitution performs a renaming of x into x', chosen to be fresh for y and t by the operator pickFresh (which takes a list of terms and returns a variable not free in any of them). But how can we be sure that such a choice exists, i.e., that we can define pickFresh? The standard solution of having the type var infinite does not suffice here—indeed, the Sum constructor introduces possibly infinite index-branching, and therefore $\mathsf{fvars}\ T$ may return an infinite set of variables, and in fact may return UNIV.

Fortunately, the rationale behind the standard solution generalizes smoothly to this infinitary situation. The key idea for finitely branching syntax is that no n-ary constructor breaks the finiteness of the set of free variables, since a finite union of finite sets is finite. As seen in Lemma 20, this generalizes to regular cardinals. So we can take var to have a regular cardinal greater than index, e.g., cardSuc |index|.

Lemma 21. $\mathsf{regular}\ |\mathsf{var}|\ \wedge\ |\mathsf{index}| <_o |\mathsf{var}|\ \rightarrow\ (\forall t.\ |\mathsf{fvars}\ t| <_o |\mathsf{var}|)$

Proof idea. Immediate by structural induction on t, using Lemma 20. □

Then $\mathsf{pickFresh}$ can be easily defined. After passing this milestone, a theory of substitution and free variables proceeds similarly to the finitary case [17]. Most current frameworks for syntax with bindings, including nominal logic [11, 15], assume finiteness of the syntactic objects—regular cardinals could provide a foundation for an infinitary generalization.

5.2 Bounded Functors and the (Co)datatype Package

Isabelle's new (co)datatype package is heavily based on both category theory and cardinal theory. It maintains a class of functors with extra 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 [22]. Here we focus on cardinality aspects omitted or very briefly mentioned in there.

BNFs are type constructors equipped with functorial (mapping) actions, natural transformations and a cardinality bound. For example, a unary BNF consists of: a type constructor $\alpha\ \mathsf{F}$; a constant $\mathsf{Fmap} : (\alpha \to \beta) \to \alpha\ \mathsf{F} \to \beta\ \mathsf{F}$; a constant $\mathsf{Fset} : \alpha\ \mathsf{F} \to \alpha\ \mathsf{set}$ (giving for each $x : \alpha\ F$ its set of "atoms") that is natural w.r.t F; 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 *internalization* of F to sets, i.e., the set-based version of F (as in the examples presented in Section 4.2), by $\mathsf{Fin}\ A = \{x \mid \mathsf{Fset}\ x \subseteq A\}$.

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 $\forall x \in \mathsf{Fin}\ A.\ s\ x \in A$—this condition qualifies s as a function between $\mathsf{Fin}\ A$ and A, written $s : \mathsf{Fin}\ A \to A$. We call A the *carrier* of $\mathscr{A}$ and s the *structural map* of A, the latter modeling the operations of the algebra. E.g., 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 (all having the same type T) by taking the product of the carrier sets and defining the structural map $s : \mathsf{Fin}\ (\prod_{i \in I} A_i) \to \prod_{i \in I} A_i$ by $s\ x = (s_i\ (\mathsf{Fmap}\ \mathsf{proj}_i\ x))_{i \in I}$. 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 and algebra $\mathsf{Min}(\mathscr{A})$, called the *minimal algebra* of $\mathscr{A}$ 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 a 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 any algebra $\mathscr{A}$, there surely exists a morphism h from $\mathscr{R}$ to $\mathscr{A}$: the corresponding projection. Then the restriction of h to $\mathsf{Min}(\mathscr{R})$ is the desired unique morphism from $\mathsf{Min}(\mathscr{R})$ to $\mathscr{A}$, and therefore $\mathsf{Min}(\mathscr{R})$ is our desired initial algebra.

This naive approach fails since we cannot possibly construct in HOL the product of all algebras (and even if we could, in a richer logic, it would not be an algebra itself due to its size). We use the boundedness of F to fix this flaw as follows. First note that, in the above context, it suffices to define h from $\mathscr{R}$ not to $\mathscr{A}$, but to $\mathsf{Min}(\mathscr{A})$. And therefore it would suffice to take $\mathscr{R}$ as the product of all *minimal* algebras, and, moreover, to only consider a complete collection of representatives (up to isomorphism). Hence, if we knew that all minimal algebras of all algebras had cardinality smaller than a given bound r_0, we could choose a type T_0 of cardinality r_0 and then define $\mathscr{R}$ as the product of all algebras on T_0: $\mathscr{R} = \prod \{\mathscr{A} \mid \mathscr{A} = (T_0, A : T_0\ \mathsf{set}, s : T_0\ \mathsf{F} \to T_0)\ \text{algebra}\}$. Then the naive construction would go through!

It remains to find a suitable r_0: as it turns out, $r_0 = \mathsf{cardSuc}\ \mathsf{Fbd}$ is such a cardinal.

Theorem 10. *For all algebras $\mathscr{A}$, let M be the carrier of $\mathsf{Min}(\mathscr{A})$. Then $|M| \leq_{\mathsf{o}} r_0$.*

Proof idea. The definition of $\mathsf{Min}(\mathscr{A})$ performs a construction of M "from above", as an intersection, yielding no cardinality information. We need to produce an alternative construction "from below", using the internal structure of F. Let $N = \bigcup_{i \in \mathsf{Field}\ r_0} N_i$, where each N_i is defined by wellorder recursion as follows: $N_i = \bigcup_{j.\ j \in \mathsf{underS}\ r_0\ i} s \bullet \mathsf{Fin}\ N_j$. To prove that N is a stable part of $\mathscr{A}$ (and hence that $M \subseteq N$), let $x \in \mathsf{Fin}\ N$. Then $\mathsf{Fset}\ x \subseteq N = \bigcup_{i \in \mathsf{Field}\ r_0} N_i$, and hence, since r_0 is regular by Lemma 18.2, we use Lemma 19 to obtain $i \in \mathsf{Field}\ r_0$ such that $\mathsf{Fset}\ x \subseteq N_i$, i.e., $x \in \mathsf{Fin}\ N_i$. Hence $s\ x \in N_{\mathsf{succ}\ r_0\ i} \subseteq N$, as desired. Conversely, $N \subseteq M$ follows by wellorder induction. We thus have $M = N$. Now, $|N| \leq_{\mathsf{o}} \mathsf{cardSuc}\ \mathsf{Fbd}$ follows by wellorder induction. □

As for the final coalgebra (codatatype) construction, we build from every BNF F a domain of infinitely branching trees, which we then quotient to F-bisimilarity [22, Section IV(F)]. The good behavior of this construction depends crucially on F being bounded by a polynomial functor:

Lemma 22. *The exist the cardinals k and l (not depending on the set A or on its type) such that $A \neq \emptyset \to |\mathsf{Fin}\ A| \leq_{\mathsf{o}} k \times_{\mathsf{c}} (|A|\ \ \hat{}_{\mathsf{c}}\ l)$*

Initially, we had maintained (a slight variation of) that property as another BNF axiom [22, Section IV], not realizing that it is redundant.[5] Removing it has simplified the package code substantially.

Another concept we extensively use in the package is cardinal arithmetic, mostly for showing that various constructions (composition, datatype, codatatype) on BNFs are themselves BNFs. All in all, our cardinal formalization was instrumental in the succinct and compositional package architecture.

6 More Details on the Formalization

Fig. 1 shows the main theory structure of our development, mapped to the (sub)section structure of the paper. To support the development presented here, we formalized many basic facts about wellorders and (order-)isomorphic transfer across bijection. When we started our development, Isabelle's library had extensive support for type-class-based

[5] Stefan Milius and Lutz Schröder suggested the elegant proof of Lemma 22 sketched in Appendix C.

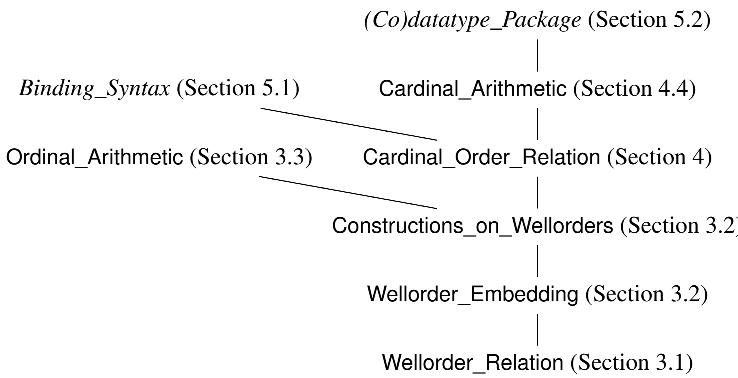

Fig. 1. Essential theory structure

orders. However, working with the wellorder type class was not an option, since we need several wellorders for the same type—e.g., the cardinal of a type is modeled as the minimum among all its wellorders. The overall development amounts to about 14000 lines of scripts (excluding the applications).

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. Anther limitation of HOL, faced quite often but seldom acknowledged by formalizers, is the inability of HOL to quantify over types except at the statements' top level. A notorious example comes from the formalizations of the FOL completeness theorem (e.g., Harrison [8]): a sentence is provable iff it is true in all models. The statement, more precisely, its right-to-left implication, is not expressible in HOL, since the right-hand side quantifies over all carrier types of all models. But one can prove an expressible stronger statement: Based on the language cardinality, one identifies and fixes a representative type so that satisfaction in all models on that type already ensures provability.

Our own formalization abounds in such cases of originally non-expressible statements, but which can be fixed by a proper choice of "representatives." One is the definition of the successor cardinal from Section 4.4: we cannot directly define $\mathsf{cardSuc}\ r$ requiring minimality with respect to all cardinals $>_{\mathsf{o}} r$ on all types, but we choose a type and then prove that the choice is irrelevant. Another is the claimed converse of Lemma 20, which, spelled with its explicit type quantifications, would look as follows:

$$\forall\alpha.\,\forall r:\alpha\ \mathsf{rel}.\ \mathsf{Card\_order}\ r\ \wedge\ (\forall\beta.\,\forall I:\beta\ \mathsf{set}.\ \forall A:\beta\to\alpha\ \mathsf{set}.|I|<_{\mathsf{o}} r\ \wedge\ (\forall i\in I.\ |A\ i|<_{\mathsf{o}} r) \\ \to\ |\bigcup\ i\in I.\ A\ i|<_{\mathsf{o}} r)\ \to\ \mathsf{regular}\ r$$

This is not expressible in HOL due to the inner universal quantification over the type β. But if we instantiate β to be a large enough index type, e.g., to α, we obtain a HOL-expressible statement. It would be interesting to identify a pattern for such statements not expressible in HOL, but with an expressible valid strengthening. However, the various solutions are apparently more or less ad hoc: for FOL completeness, an insight from the actual construction of the model (Löwenheim–Skolem); for the successor cardinal, invariance under isomorphism; for the alternative characterization of regularity, the properties of indexed union.

7 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)datatype theory. Moreover, Breitner employed it in formalizing free group theory [2].

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 ad hoc encodings.

An interesting question is whether the quotienting improvement of Norrish and Huffman would have helped with our cardinal theory. We believe the answer is "not significantly", since we would still be faced with the problem of changing the underlying type of cardinals to accommodate for larger and larger sizes. In HOL, there is no way to reason about arbitrary cardinals up to equality, so isomorphism still seems like the right compromise.

Acknowledgment. We thank Tobias Nipkow for making this work possible. 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. Blanchette, J.C., Popescu, A., Traytel, D.: Formal development associated with this paper. `http://www21.in.tum.de/~traytel/card_devel.tar.gz`
2. Breitner, J.: Free groups. In: Klein, G., Nipkow, T., Paulson, L. (eds.) Archive of Formal Proofs. `http://afp.sourceforge.net/entries/Free-Groups.shtml` (2011)
3. Chang, C.C., Keisler, H.J.: Model Theory. North-Holland (1973)
4. Church, A.: A formulation of the simple theory of types. J. Symb. Logic 5(2), 56–68 (1940)
5. Gordon, M.J.C., Melham, T.F. (eds.): Introduction to HOL: A Theorem Proving Environment for Higher Order Logic. Cambridge University Press (1993)
6. 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)
7. Harrison, J.: The HOL wellorder library. `http://www.cl.cam.ac.uk/~jrh13/papers/wellorder-library.html` (1992)
8. 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)
9. Holz, M., Steffens, K., Weitz, E.: Introduction to Cardinal Arithmetic. Birkhäuser Advanced Texts, Birkhäuser (1999)
10. Huffman, B.: Countable ordinals. In: Klein, G., Nipkow, T., Paulson, L. (eds.) Archive of Formal Proofs. `http://afp.sf.net/entries/Ordinal.shtml` (2005)

11. 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)
12. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL: A Proof Assistant for Higher-Order Logic, LNCS, vol. 2283. Springer (2002)
13. 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)
14. Paulson, L.C., Grabczewski, K.: Mechanizing set theory. J. Autom. Reasoning 17(3), 291–323 (1996)
15. Pitts, A.M.: Nominal logic, a first order theory of names and binding. Inf. Comput. 186(2), 165–193 (2003)
16. 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)
17. Popescu, A.: Contributions to the theory of syntax with bindings and to process algebra. Ph.D. thesis, University of Illinois at Urbana-Champaign (2010)
18. 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)
19. 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)
20. Sternagel, C.: Extending well-founded partial orders to total well-founded orders (February 2013), archived at `https://lists.cam.ac.uk/pipermail/cl-isabelle-users/2013-February/thread.html`
21. Taylor, P.: Intuitionistic sets and ordinals. J. Symb. Log. 61(3), 705–744 (1996)
22. 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)
23. 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 More on Finite Cardinals

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

Lemma 23. *Assume* $\mathsf{finite}\ A\ \wedge\ \mathsf{finite}\ B$*. Then:*

1. $|A| =_{\mathsf{o}} |B|\ \leftrightarrow\ \mathsf{card}\ A = \mathsf{card}\ B$
2. $|A| \leq_{\mathsf{o}} |B|\ \leftrightarrow\ \mathsf{card}\ A \leq \mathsf{card}\ B$

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 24. *1.* $\mathsf{Card\_order}\ r\ \to\ (\mathsf{finite}\ (\mathsf{cardSuc}\ r)\ \leftrightarrow\ \mathsf{finite}\ (\mathsf{Field}\ r))$
2. $\mathsf{cardSuc}\ (\mathsf{natLeq\_on}\ n) =_{\mathsf{o}} \mathsf{natLeq\_on}\ (\mathsf{Suc}\ n)$

B Case Study: An Order Extension Theorem

Recently, Christian Sternagel started a discussion on the Isabelle mailing list [20] 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. 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):
(a) $v_0 = \emptyset$
(b) $v_{i+1} = v_i$ extended with a maxim: the minimal element of p not in the field of v_i
(c) if i is a limit ordinal, $v_i = \bigcup_{j<i} v_j$
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 our wellorder recursor wo_rec. The recursor can cover the above definition, but is awkward to use in such situations which need to 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 only if $\mathsf{aboveS}\ r\ a \neq \emptyset$). 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 25. $a \in \mathsf{Field}\ r\ \wedge\ \mathsf{isLim}\ r\ a\ \to\ a = \mathsf{supr}\ r\ (\mathsf{underS}\ r\ a)$

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

Lemma 26 (Wellorder recursion with zero, successor and limit).
Assume $\mathsf{adm\_woL}_{\,r}\ L$ *and* $a \in \mathsf{Field}\ r$. *Then:*

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 26:

Lemma 27 (Wellorder induction with zero, successor and limit).
Assume the following hold:

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

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

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

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

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. w 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 [23], while transfinite recursion/induction is a speciality of logicians [3]. For this particular instance, the latter approach felt more intuitive and (its informal version) was easier to discover and formulate.

C Some Proof Ideas

Proof of Theorem 2.
We define together, by wellorder recursion on r, the functions $f : \alpha \to \beta$ and $g : \alpha \to \mathsf{bool}$. 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 $f\ a$ to be anything).

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 have 2 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 3.
Compared to the standard result about the class of ordinals, an extra difficulty arises from the use of embeddings (as opposed to plain inclusions) in the definition of $r <_{\mathsf{o}} s$. Direct reasoning about embeddings would be very tedious, since it would require reasoning about limit behavior of embedding composition (pretty much like limit constructions in category theory). Fortunately however, 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 minim with respect to $<_{\mathsf{o}}$. We pick $r_0 \in R$, and restrict attention to $R_0 = \{r \in R.\ r \leq_{\mathsf{o}} r_0\}$—if R_0 has a minim, 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: $r \leq_{\mathsf{o}} s \ \leftrightarrow\ H\ r \subseteq H\ s$; $\ r <_{\mathsf{o}} s \ \leftrightarrow\ H\ r \subset H\ s$; $\ r =_{\mathsf{o}} s \ \leftrightarrow\ H\ r = H\ s$. This effectively brings the problem to the familiar ground of filters on a fixed wellorder with inclusions between them, where the proof proceeds smoothly (and standardly). □

Proof of Lemma 6.
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 Th. 2). 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 $r \leq_{\mathsf{o}} s$. □

Proof of Lemma 22.
Let $k = |\mathsf{Fin}\ (\mathsf{Field}\ \mathsf{Fbd})|$ and $l = \mathsf{Fbd}$. We define $d : \mathsf{Fin}\ (\mathsf{Field}\ \mathsf{Fbd}) \times (\mathsf{Field}\ \mathsf{Fbd} \to A)$ by $d\ (y, f) = \mathsf{Fmap}\ f\ y$. It suffices to prove that d is surjective. To this end, let $x \in \mathsf{Fin}\ A$. Since $|\mathsf{Fset}\ x| \leq_{\mathsf{o}} \mathsf{Fbd}$, we obtain an injective function $g : \mathsf{Fset}\ x \to \mathsf{Field}\ \mathsf{Fbd}$. Let $y = \mathsf{Fmap}\ g\ x$. We choose $f : \mathsf{Field}\ \mathsf{Fbd} \to \mathsf{Fin}\ A$ be such that it is the 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, we have

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

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