A Mechanizable First-Order Theory of Ordinals

Peter H. Schmitt(✉)

Department of Informatics, Karlsruhe Institute of Technology (KIT),
Am Fasanengarten 5, 76131 Karlsruhe, Germany
pschmitt@ira.uka.de

Abstract. We present a first-order theory of ordinals without resorting to set theory. The theory is implemented in the KeY program verification system which is in turn used to prove termination of a Java program computing the Goodstein sequences.

1 Introduction

A number of automated reasoning systems have been put to the task to prove theorems about ordinal numbers. Here is a fair selection of pertinent papers: [4,5,9,15] for the Isabelle proof assistent, [6,8] for Coq, [3] for OTTER. All these efforts have in common that they start with a semantic definition of ordinals as sets with special properties. Of a different flavor are the papers [13,14], that present algorithms implemented in the ACL2 system for solving problems in ordinal arithmetic working on a normal form representation.

In this paper, we will present a first-order theory Th_{Ord} of ordinals. The models of Th_{Ord} are of the form $\mathcal{M} = (U, \dot{0}, \dot{\omega}, \dot{+}1, \dot{<})$ with universe U, constants $\dot{0}$, and $\dot{\omega}$, the unary successor function $\dot{+}1$ and the order relation $\dot{<}$. The logic itself contains the binary operator $sup_{x<n}m$ binding variable x. Its interpretation in $\mathcal{M}$ is the supremum of $\{m^{\mathcal{M}}(\alpha) \mid \alpha \in U \text{ and } \alpha \dot{<} n^{\mathcal{M}}\}$. Typically, the term m will contain the free variable x and $t^{\mathcal{M}}(\alpha)$ stands for the evaluation of m in $\mathcal{M}$ with x instantiated to the element $\alpha \in U$. The operator $sup_{x<n}m$ is the only construct in our axiomatization with a set theoretic flavor. This operator is, however, definable in the standard first-order logic; a proof of this result for variable-binders in general is available in Ulbrich's PhD thesis [22].

Already in 1965 Gaisi Takeuti presented in [20] a first-order theory $\mathcal{O}$ of ordinal numbers. His interest were in proof theoretic properties. The theory $\mathcal{O}$ allows to define an inner model of Zermelo-Fraenkel (ZF) set theory. Thus, a formula ϕ is derivable from $\mathcal{O}$ iff a canoncial translation of ϕ is derivable in ZF set theory. The vocabulary of $\mathcal{O}$ is much richer than ours: it contains e.g., already in its axiomatic basis coding and decoding functions for pairs of ordinals. As a consequence the theory $\mathcal{O}$ is not very well suited as the basis for automated reasoning on ordinals.

It is well-known that the Peano axioms, PA, for the natural numbers are incomplete. This did and does not cause much alarm since the examples of true statements not derivable in PA were consider too arcane to be of any pratical

© Springer International Publishing AG 2017
R.A. Schmidt and C. Nalon (Eds.): TABLEAUX 2017, LNAI 10501, pp. 331–346, 2017.
DOI: 10.1007/978-3-319-66902-1_20

relevance. This is beginning to change as these examples get more and more accessible. We present, based on [10], a Java program with less than 20 lines of code, computing the Goodstein sequences, such that termination of one of its two methods cannot be proved in PA. Still, there is no reason to be alarmed since termination can be proved in Th_{Ord}, which is a simple, plausible extension of PA as can be seen by scrutinizing the axioms in Fig. 1 below.

1.	$\forall x, y, z(x < y \land y < z \rightarrow x < z)$	transitivity
2.	$\forall x(\neg x < x)$	strict order
3.	$\forall x, y(x < y \lor x \doteq y \lor y < x)$	total order
4.	$\forall x(0 \leq x)$	0 is smallest element
5.	$0 < \omega \land \neg\exists x(\omega \doteq x + 1)$	ω is a limit ordinal
6.	$\forall y(0 < y \land \forall x(x < \omega \rightarrow x + 1 < y) - > \omega \leq y)$	ω is the least limit ordinal
7.	$\forall x(x < x + 1) \land \forall x, y(x < y \rightarrow x + 1 \leq y)$	successor function
8.	$\forall z(z < n \rightarrow m[z/x] \leq sup_{x<n} m)$	def of supremum, part 1
9.	$\forall u(\forall z(z < n \rightarrow m[z/x] \leq u) \rightarrow sup_{x<n} m \leq u)$	def of supremum, part 2
10.	$\forall x(\forall y(y < x \rightarrow \phi(y/x)) \rightarrow \phi) \rightarrow \forall x \phi$	transfinite induction scheme

Fig. 1. The axioms of the core theory

The first automatic termination proof for Goodstein sequences was recorded in [23]. In that paper a finite rewrite system is presented whose termination encodes the termination of these sequences and termination of the rewrite system is proved. A termination proof using the higher-order logic proof assistant Coq can be downloaded from [7] as part of the Coq project on ordinal notation [6]. Hereditary multisets, a variant of nested multisets, offer a convenient representation of ordinals below ϵ_0. A formalization of hereditary multisets in the Isabelle/HOL proof assistant is available from the Archive of Formal Proofs [5] also containing a termination proof for Goodstein sequences. A corresponding publication is pending [4]. The theory Th_{Ord} has been implemented in KeY, a first-order theorem proving and program verification system based on the sequent calculus [1]. This implementation has been put to use to obtain a machine assisted termination proof of the Java program mentioned above. This seems to be the first termination proof for Goodstein sequences using a first-order reasoning system.

2 A Theory of Ordinals

In the first subsection, we start out with a very simple core theory Th^0_{Ord}. This plays the same role here as Peano's theory for the arithmetic of natural number. In the next subsection, the final theory Th_{Ord} is obtained as a definitional extension of Th^0_{Ord}.

2.1 The Core Theory

The vocabulary Σ^0_{Ord} of the core theory contains the following symbols

predicate	$<$	binary
functions	$+1$	unary
	$0, \omega$	constants

Terms and formulas are defined as usual in first-order logic with the exception that we include a term building operator *sup*:

if n, m are terms and x is a variable that does not occur free in n, then

$$sup_{x<n} m$$

is a term.

Two operations, (1) start with 0 and (2) add one, suffice to generate all finite ordinals. We use $x + 1$ to denote the immediate successor of x to avoid the introduction of an additional symbol. In set theory, a third operation is added to generate all transfinite ordinals: (3) for any set X of ordinals, there is a least ordinal not less than all $\alpha \in X$. Here we avoid set theory and restrict this operation to sets X that can be obtained as instances of one term m, where furthermore only instantiations up to a bound n are allowed. This motivates the term constructor $sup_{x<n} m$. The term m will typically contain the variable x.

In most presentations of Peano arithmetic, the order relation $<$ is not part of the core theory, but it is later added as a definitional extension. It would have been extremely cumbersome to do so in our setting. So, we sacrificed on minimality and included $<$ from the outset.

The intended meaning of symbols in Σ^0_{Ord} is fixed by the axioms in Fig. 1.

We thus see that $<$ is to be interpreted as a strict, total, order relation. We use $x \leq y$ as a shorthand for $x \doteq y \vee x < y$. From the axioms, we see that 0 is the least element with respect to $<$ and ω is the first infinite ordinal. Without the two axioms 5 and 6 for ω, the natural numbers with strict order and successor would be a model of the remaining axioms and nothing would have been gained over Peano's theory. Axioms 8 and 9 establish the intended meaning of the supremum operator as explained in the introduction. We use the notation $m[z/x]$ to denote the term that arises from m by replacing x by z everywhere. Note that the way we defined *sup*, the formulas $sup_{x<0} m \doteq 0$ and $sup_{x<1} m \doteq m[0/x]$ can be derived.

The last and most powerful axiom is the axiom scheme of transfinite induction that is an extension the course-of-value induction scheme in finite arithmetic. Let ϕ be a formula that typically would contain x as a free variable and let $\phi(y)$ stand for the formula obtained from ϕ by replacing x by y (assuming of course that y does not occur in ϕ, neither free nor bound). If we can prove for every x the transfinite induction step $\forall y(y < x \rightarrow \phi(y)) \rightarrow \phi$, then we conclude that ϕ is true for all x.

One could ask whether the *sup* operator should have been omitted from the core vocabulary and added later as a definitional extension. The answer is *no!* Since we follow the usual set-up of first-order logic all functions are total. Consequently, inclusion of a function symbol in the vocabulary already implies an implicit existence axiom: the function values exist for all arguments. Adding *sup* is not a definitional extension since the associated existence claim could not be proved in the theory without *sup*.

An alternative would have been to include the following axiom scheme instead of the two parts of the definition of *sup*

$$\forall y(\exists z(\forall x(x < y \to t \leq z)))$$

A possible instantiation could be $\forall y(\exists z(\forall x(x < y \to \omega^x \leq z)))$. Then, it would have been possible to show, using transfinite induction, that adding the *sup* operator is a definitional extension of this version of the core theory. The adopted approach is more straightforward.

2.2 The Full Theory

The full vocabulary Σ_{Ord} of the theory Th_{Ord} is shown in Fig. 2.

predicates	$>, \leq$	binary
	lim	unary
functions	$0, 1$	constant
	max	binary
	$+, *, \hat{}$	binary

Fig. 2. The full vocabulary Σ_{Ord}

The axioms of Th_{Ord} are Th^0_{Ord} plus the definitions of the new symbols in Figs. 3 and 5.

$\forall x, y(x \leq y \leftrightarrow x \doteq y \vee x < y)$	(less or equal relation)
$\forall x(lim(x) \leftrightarrow 0 < x \wedge \neg\exists y(y + 1 \doteq x))$	(limit ordinal)
$\forall x, y(max(x, y) \doteq \text{if } x \leq y \text{ then } y \text{ else } x)$	(maximum operator)

Fig. 3. Definitional extension: axioms for auxiliary predicates

The three axioms in Fig. 3 define the auxiliary symbols $\leq$ and *max* plus the important concept of a limit ordinal. In contrast to other presentations, 0 is not a limit ordinal in our set-up.

Figure 4 shows a sample of lemmas that can already be derived at this point from the axioms considered so far. The lemmas are grouped together according to the syntactical symbols involved. This does not reflect the order in which the lemmas can or need to be proved.

Lemma 1 in Fig. 4 is the least number principle, a well known equivalent to the induction axiom scheme. It is instructive to figure out why this lemma is true even if x does not occur as a free variable in ϕ. Lemma 2 in Fig. 4 is the

1. $\exists x\phi \rightarrow \exists x(\phi \wedge \forall y(y < x \rightarrow \neg\phi[y/x]))$
2. $\phi[0/x] \wedge$
   $\forall x(\phi \rightarrow \phi[x+1/x]) \wedge$
   $\forall x(lim(x) \wedge \forall y(y < x \rightarrow \phi[y/x]) \rightarrow \phi)$
   $\rightarrow \forall x\phi$
3. $\forall x(lim(x) \rightarrow sup_{y<x}\ y \doteq x)$
4. $\forall x(sup_{z<x+1}\ t \doteq max(sup_{z<x}\ t, t[x/z]))$
5. $\forall x(\forall y(y < x \rightarrow t_1[y/z] \doteq t_2[y/z]) \rightarrow sup_{z<x}\ t_1 \doteq sup_{z<x}\ t_2)$
6. $\forall x_1, x_2(\ \forall x(x < x_1 \rightarrow \exists y(y < x_2 \wedge t_1[x/z] \leq t_2[y/z]))\wedge$
   $\forall y(y < x_2 \rightarrow \exists x(x < x_1 \wedge t_2[y/z] \leq t_1[x/z]))$
   $\leftrightarrow sup_{z<x_1}\ t_1 \doteq sup_{z<x_2}\ t_2)$
7. $lim(x) \leftrightarrow x \neq 0 \wedge \forall y(y < x \rightarrow (y+1) < x)$

Fig. 4. Some lemmas derivable from the axioms considered so far

$$\begin{array}{l}
\forall x(x+0 \doteq x)\\
\forall x, y(x+(y+1) \doteq (x+y)+1)\\
\forall x, y(lim(y) \rightarrow x + sup_{z<y}z \doteq sup_{z<y}(x+z))\\
\forall x(x * 0 \doteq 0)\\
\forall x, y(x*(y+1) \doteq (x*y)+x)\\
\forall x, y(lim(y) \rightarrow x*y \doteq sup_{z<y}(x*z))\\
\forall x(x^0 \doteq 1)\\
\forall x, y(x^{y+1}) \doteq (x^y)*x)\\
\forall x, y(lim(y) \wedge x \neq 0 \rightarrow x^y \doteq sup_{z<y}(x^z))\\
\forall y(lim(y) \rightarrow 0^y \doteq 0)
\end{array}$$

Fig. 5. Definitional extension: axioms for arithmetic operations

equivalent rephrasing of the transfinite induction scheme 10 in Fig. 1 that is most frequently employed: If a formula ϕ with free variable x is true for $x = 0$ (base case), if for all z we can prove that $\phi[z+1]$ follow from $\phi[z]$ (successor induction step), and if we can prove for every limit ordinal λ, when $\phi[z/x)$ is true for all $z < \lambda$ then $\phi[\lambda/x]$ follows (limit induction step), then we have proved $\forall x\phi$.

Lemma 3 could be rephrased as: if x is a limit ordinal then x is the least ordinal that is greater or equal than all ordinals that are strictly less than x. This is not true for successor ordinals x. In this case, we have $sup_{y<x+1}\ y \doteq x$ instead. Lemma 4 is useful in proving statements involving the *sup* operator via induction. Lemma 5 helps to show that two suprema are equal especially in the case when equality between t_1 and t_2 is not obvious.

For the purpose of this paragraph we look at a term t that contains z as a sequence t_z. We say that a sequence $t_{z<x_1}$ is confinal in $s_{z<x_2}$ if for every $x < x_1$ there is $y < x_2$ with $t[x/z] \leq s[y/z]$. If two sequences are mutually confinal in one another, then they share the same supremum. This is Lemma 6 in Fig. 4. Note, that we get equality of two suprema with different bounds x_1, and x_2. Lemma 7 gives an alternative definition of a limit ordinal.

The second installment of the axioms extending Th^0_{Ord} to Th_{Ord} is listed in Fig. 5. They give the usual recursive definitions of ordinal addition, multiplication and exponentiation.

2.3 Derived Lemmas

In this subsection, we will in Figs. 6, 7, 8 and 9 list and comment on a selection of lemmas derivable from Th_{Ord}.

1. $\forall x, y(y \neq 0 \rightarrow x < x + y)$
2. $\forall x, y(x \leq x + y)$
3. $\forall x, y(y \leq x + y)$
4. $\forall x, y, z(x < y \rightarrow z + x < z + y)$
5. $\forall x, y, z(x \leq y \rightarrow x + z \leq y + z)$
6. $\forall x, y, z(x + y < x + z \rightarrow y < z)$

Fig. 6. Lemmas involving addition and order relations

Figure 6 lists lemmas that are needed as intermediate stepping stones in the proofs of the properties of ordinal arithmetic.

Lemma 1 in Fig. 6 extends the axiom $\forall x(x < x+1)$ from Fig. 1 where we now add on the right side an arbitrary number greater than or equal to 1 instead of just 1. This is, of course, proved via transfinite induction. Lemma 2 is an easy variant of Lemma 1. Addition of ordinals is not commutative, so we cannot conclude from Lemma 1 that $\forall x, y(x \neq 0 \rightarrow y < x + y)$. Indeed, $y = \omega, x = 1$ is a counterexample. But the version for $\leq$ instead of $<$ is provable. This is Lemma 3. Lemma 4 is also proved using transfinite induction. We remark that $\forall x, y, z(x < y \rightarrow x + z < y + z)$ is not true, as can be seen by the instantiation 0 for x, 1 for y, and ω for z. But the relaxed version with $\leq$ instead of $<$ is derivable. This is Lemma 5. Lemma 6 is the reverse of Lemma 4.

Figure 7 gives lemmas on ordinal addition. Since ordinal addition is in general not commutative Lemma 1 in Fig. 7 may not be immediately obvious, but it can be easily proved using ordinal induction. Lemma 3 is a fact on ordinal addition that we have referred to already above. Lemma 4 is a useful lemma formalizing the intuition that the property of being a limit ordinal is determined by the right *end part* of the ordinal regardless of what comes before. Lemma 5 gives a first general representation theorem for ordinals. In [21, Theorem 8.13] it is proved using set comprehension. This is not available in our setting. Fortunately, it turned out that there is a much simpler proof using ordinal induction. Lemma 6 required the most complex proof so far. The basic idea, however, is quite simple. As a witness for z take b, the least ordinal such that $y \leq x + b$. It can easily be seen that such a number exists by the least number principle (Lemma 1 in Fig. 4). Then, a case distinction $b = 0, b \doteq b_0 + 1$ for some b_0, or $lim(b)$ leads to success. Lemma 7 is the wellknown associative law. Lemmas 8 and 9 correspond to the Peano axioms for the natural numbers, which say that 0 is not a successor

1. $\forall x(0 + x \doteq x)$
2. $\forall x, y(x + y \doteq 0 \leftrightarrow x \doteq 0 \wedge y \doteq 0)$
3. $\forall x(x < \omega \to x + \omega \doteq \omega)$
4. $\forall x \forall y(lim(y) \to lim(x + y))$
5. $\forall x(\omega \leq x \to \exists y, n(lim(y) \wedge n < \omega \wedge x \doteq y + n))$
6. $\forall x, y(x \leq y \to \exists z(x \doteq y + z))$
7. $\forall x, y, z(x + (y + z) \doteq (x + y) + z)$
8. $\neg \exists x(x + 1 \doteq 0)$
9. $\forall x, y((x + 1) \doteq (y + 1) \to x \doteq y)$
10. $\forall x, y, z((z + x) \doteq (z + y) \to x \doteq y)$
11. $sup_{z<x}\ (i + t) \doteq i + sup_{z<x}\ t$ if z does not occur in i and $x > 0$
12. $i + j \doteq j$ if $\omega \leq j$ and $i < \omega$

Fig. 7. Lemmas on addition

and the successor function is injective. Lemma 10 shows that addition on the right, with fixed left summand, is injective. Lemma 11 resisted for a while all my attempts to prove it. Since I could also not find it in [21], I was, at some point, even in doubt wether it is true at all. The inequality, $sup_{z<x}\ (i+t) \leq i+sup_{z<x}\ t$ is simple. For the reverse inequality a proof by contradiction turned out to be the right way of attack. So assume $sup_{z<x}\ (i+t) < i+sup_{z<x}\ t$ and try to find a contradiction. The key to the solution was the case distinction $sup_{z<x}\ (i+t) < i$ or $i \leq sup_{z<x}\ (i+t)$. Notice that in the first case, we arrive at the contradiction $i \leq i+t[0/z] < sup_{z<x}\ (i+t) < i$. Here also the assumption $x > 0$ comes in. In the second case, there is by Lemma 5 an ordinal k such that $i+k \doteq sup_{z<x}\ (i+t)$. By the proof-by-contradiction assumption, this yields $i+k < i+sup_{z<x}\ t$ and further by Lemma 7 in Fig. 6 $k < sup_{z<x}\ t$. By the definition of *sup* there, is $\lambda < x$ with $k \leq t[\lambda/z]$. This leads to the contradiction $i+k \leq i+t[\lambda/z] < sup_{z<x}\ (i+t)$. The *commuted* version of Lemma 11, i.e., $sup_{z<x}\ (t + i) \doteq (sup_{z<x}\ t) + i$, provided z does not occur in i and $x > 0$, is – as you would have expected –not true: $\omega \doteq sup_{z<\omega}\ (z + 1) \doteq (sup_{z<\omega}\ z) + 1 \doteq \omega + 1$. Lemma 12 shows a dramatic failure of commutativity for ordinal addition: A left finite ordinal summand is simply absorbed if the right summand is infinite. We found it helpful to split the proof of Lemma 12 in the cases $\omega \doteq j$ and $\omega < j$.

Figure 8 shows derivable properties of ordinal multiplication. Lemma 1 shows that the strict order relation is preserved by multiplication on the left, provided that the left multiplyer is not 0. Multiplication on the right only preserves $\leq$, as Lemma 4 shows. Lemma 2 is the reverse implication from Lemma 1. Lemma 3 states that multiplication on the right, with a fixed multiplicand on the left, is an injective function. Lemma 7 is crucial for the proof of distributivity (Lemma 8) and multiplicative associativity (Lemma 9).

After all the preparations the proof of multiplicative associativity is now straightforward. Let us for once give a detailed proof sketch in this exemplary case. We use ordinal induction (Lemma 2 in Fig. 4) on the variable k. The base case is trivial. The successor induction step is proved as follows:

1. $\forall x, y, z((0 < z \wedge x < y) \rightarrow z * x < z * y)$
2. $\forall x, y, z(z * x < z * y) \rightarrow (0 < z \wedge x < y))$
3. $\forall x, y, z(0 < z \wedge z * x \doteq z * y \rightarrow x \doteq y)$
4. $\forall x, y, z(x \leq y \rightarrow x * z \leq y * z)$
5. $\forall x, y(x \neq 0 \rightarrow y \leq x * y$
6. $\forall x(0 < x < \omega \rightarrow x * \omega \doteq \omega)$
7. $sup_{z<x}\ (i * t) \doteq i * sup_{z<x}\ t$
8. $\forall i, j, k(i * (j + k) \doteq i * j + i * k)$
9. $\forall i, j, k((i * j) * k \doteq i * (j * k)$
10. $\forall z, x((lim(z) \wedge 0 < x < \omega) \rightarrow x * z \doteq z)$
11. $\forall i, j(1 < i \wedge 1 < j \rightarrow i + j \leq i * j)$
12. $\forall i, z(0 < i \wedge lim(z) \rightarrow lim(i * z))$
13. $\forall i, z(0 < i \wedge lim(z) \rightarrow lim(z * i))$

Fig. 8. Lemmas on multiplication

$$\begin{aligned}
i * (j * (k + 1)) &\doteq i * (j * k + j) && \text{definition of } * \\
&\doteq i * (j * k) + i * j && \text{distributivity (Lemma 8)} \\
&\doteq (i * j) * k + i * j && \text{induction hypothesis} \\
&\doteq (i * j) * (k + 1) && \text{definition of } *
\end{aligned}$$

The induction step in the limit case is shown next. We us λ instead of k to signal that k is a limit ordinal:

$$\begin{aligned}
i * (j * \lambda) &\doteq i * sup_{x<\lambda} j * x && \text{definition of } * \\
&\doteq sup_{x<\lambda} i * (j * x) && \text{Lemma 7} \\
&\doteq sup_{x<\lambda} (i * j) * x && \text{induction hypothesis} \\
&\doteq (i * j) * \lambda && \text{definition of } *
\end{aligned}$$

Lemma 10 (we are still talking about Fig. 8) is a strengthening of Lemma 6: multiplicative absorbtion on the left of finite ordinals not only holds for ω, but for any limit ordinal. Lemma 11 states when addition of two ordinals is less than their product. The restrictions are necessary as can be seen by the following simple examples:

$$\begin{aligned}
j = 0 + j &\not\leq 0 * j = 0 \\
1 + j &\not\leq 1 * j = j \\
i = i + 0 &\not\leq i * 0 = 0 \\
i + 1 &\not\leq i * 1 = i
\end{aligned}$$

Finally, we turn to the lemmas on exponentiation in Figure 9. Note that the restriction on x in Lemma 1 is necessary since by definition $0^0 \doteq 1$. Also the restrictions in Lemma 2, which says that exponentiation is strictly increasing in the left argument, are necessary as can be seen by the following examples $2 \not< 2^0 \doteq 1, 2 \not< 2^1 \doteq 2, 0 \not< 0^2 \doteq 0$, and $1 \not< 1^2 \doteq 1$. It is only weakly increasing on the right, Lemma 3. The strict inequality is far from being true, as Lemma 4 shows. Exponentiation is also strictly monotone in the second argument, as Lemma 5 shows. The reverse implication is also true, as stated by

1. $\forall x(0 < x \to 0^x \doteq 0)$
2. $\forall x, y(1 < x \land 1 < y \to x < x^y)$
3. $\forall x, y(1 < x \to y \leq x^y)$
4. $\forall x, y(1 < x \land x < \omega \to x^\omega \doteq \omega)$
5. $\forall x, y_1, y_2(1 < x \land y_1 < y_2 \to x^{y_1} < x^{y_2})$
6. $\forall x, y_1, y_2(1 < x \land x^{y_1} < x^{y_2} \to y_1 < y_2)$
7. $\forall x_1, x_2, y(x_1 < x_2 \to x_1^y \leq x_2^y)$
8. $\forall x, y(0 < x \land lim(y) \to lim(y^x))$
9. $\forall x, y(1 < x \land lim(y) \to lim(x^y))$
10. $\forall x, y, z(x^{y+z} \doteq x^y * x^z)$
11. $\forall x, y, z((x^y)^z \doteq x^{y*z})$
12. $\forall b((0 < b \land \forall x(x < b \to 0 < j)) \to sup_{x<b}(i^j) = i^{sup_{x<b}(j)})$
    for all terms i, j such that x does not occur in i.

Fig. 9. Lemmas on exponentiation

Lemma 7. In the first argument exponention is only weakly monotone, as stated by Lemma 7. A counterexample to strict monotonicity is given by the instantiations $x_1 = 2, x_2 = 3$, and $y = \omega$. Lemmas 8 and 9 show how the property of being a limit ordinal is propagated by exponentiation. Lemmas 10 and 11 are laws of exponentiation familiar from finite arithmetic. Lemma 12 is in fact a lemma scheme. Note that in typical applications x is a free variable in j. It states an indispensable continuity property for exponentiation.

The theory Th_{Ord} has been implemented in the KeY system. Interactive proof for over 170 lemmas found in the set theory textbooks [2,11,12,21] have been obtained documenting the strength of Th_{Ord}.

3 Termination of Goodstein Sequences

The sequences under investigation were first introduce by Goodstein in [18]. In fact, Goodstein considered in his paper more general sequences involving a non-decreasing function $f : \mathbb{N} \to \mathbb{N}$ as a parameter. The Goodstein sequences considered here, these are the same as the ones considered by Kirby and Paris, are obtained by the choice $f(i) = i + 2$. Kirby and Paris showed in their highly acclaimed paper [10] that termination of Goodstein sequences cannot be proved in Peano arithmetic, stronger principles, like e.g., ordinal induction, are needed. We will use ordinal induction, as provided by the theory Th_{Ord} presented in Sect. 2, to prove termination of a Java program computing the Goodstein sequences in Subsect. 3.3.

3.1 Injecting Natural Numbers

The termination proof in Subsect. 3.3 will be done using the program verification system KeY. KeY employs a many-sorted first-order logic. For ease of presentation the theory Th_{Ord} was formulated in Sect. 2 as a one-sorted theory. In the implementation of Th_{Ord} within the KeY prover, Ord is used to name the sort of ordinals. Among the other sorts present in the KeY prover, there are mathematical integers int. It is essential for the intended proof to relate non-negative integers to the finite ordinals. To this end, we add a function $onat : int \to Ord$.

Figure 10 shows an axiomatisation of the function $onat : int \to Ord$ that maps the non-negative integers into corresponding ordinals less than ω. Obviously, $onat$ is a partial function. The KeY system deals with partial function by underspecification. That means that $onat$ is a total function, but the axioms do not include any commitment on the values for negative arguments.

Definitional Extension

1. $onat(0) \doteq 0$
2. $\forall n(0 \leq n \to onat(n+1) \doteq onat(n) + 1)$

Derived Lemmas

3. $onat(1) \doteq 1$
4. $\forall n, m(0 \leq n \wedge 0 \leq m \to onat(n+m) \doteq onat(n) + onat(m))$
5. $\forall n, m((0 \leq n \wedge 0 \leq m) \to (onat(n) \doteq onat(m) \to n \doteq m))$
6. $\forall n, m((0 \leq n \wedge 0 \leq m) \to (onat(n) < onat(m) \leftrightarrow n < m))$
7. $\forall n(0 \leq n \to onat(n) < \omega)$

Fig. 10. Positive integers as ordinals

Figure 10 also shows useful derived lemmas. We use in this figure and also later on overloaded syntax. Thus, whether 0 denotes an integer or an ordinal, whether + is ordinal addition or addition of non-negative integers can be found out by looking at the type information.

3.2 Definition of Goodstein Sequences

We start with an informal explanation. First, the auxiliary concept of a hereditary base-n notation is needed. This makes only sense for $n \geq 2$. The hereditary base-n notation for a natural number m is obtained from its ordinary base-n notation

$$m = m_k \cdot n^k + m_{k-1} \cdot n^{k-1} + \ldots m_1 \cdot n + m_0, \quad 0 \leq m_i < n, m_k \neq 0$$

by also writing the exponents $k, k-1, \ldots$ in base-n notation and again the thus arising exponents, and so on.

Example 1.

base-2	35	$= 2^5 + 2^1 + 2^0$
hereditary base-2	35	$= 2^{2^2+1} + 2 + 1$
base-3	100	$= 3^4 + 2 \cdot 3^2 + 3^0$
hereditary base-3	100	$= 3^{3+1} + 2 \cdot 3^2 + 1.$

The Goodstein sequence $G_k(m)$ with initial value m is computed as follows

$G_1(m) = m$

$G_2(m) =$ in the hereditary base-2 representation of m
replace every occurence of 2 by 3 and subtract 1

...

$G_k(m) =$ in the hereditary base-k representation of $G_{k-1}(m)$
replace every occurence of k by $k+1$ and subtract 1

Example 2. The Goodstein sequence for $m = 3$

$G_1(3)$	By definition		3
$G_2(3)$	Write 3 in her. base 2 notation	$2^1 + 1$	
	Replace 2 by 3 minus 1	$3^1 + 1 - 1$	3
$G_3(3)$	Write 3 in her. base 3 notation	3^1	
	Replace 3 by 4 minus 1	$4^1 - 1$	3
$G_4(3)$	Write 3 in her. base 4 notation	3	
	Replace 4 by 5 minus 1	$3 - 1$	2
$G_5(3)$	Write 2 in her. base 5 notation	2	
	Replace 5 by 6 minus 1	$2 - 1$	1
$G_6(3)$	Write 1 in her. base 6 notation	1	
	Replace 6 by 7 minus 1	$1 - 1$	0

Example 3. Initial part of the Goodstein sequence for $m = 4$

		4	
2^{2^1}	$3^{3^1} - 1$	26	ω^ω
$3^2 * 2 + 3^1 * 2 + 2$	$4^2 * 2 + 4^1 * 2 + 2 - 1$	41	$\omega^2 * 2 + \omega * 2 + 2$
$4^2 * 2 + 4^1 * 2 + 1$	$5^2 * 2 + 5^1 * 2 + 1 - 1$	60	$\omega^2 * 2 + \omega * 2 + 1$
$5^2 * 2 + 5^1 * 2$	$6^2 * 2 + 6^1 * 2 - 1$	83	$\omega^2 * 2 + \omega * 2$
$6^2 * 2 + 6^1 * 1 + 5$	$7^2 * 2 + 7^1 * 1 + 5 - 1$	109	$\omega^2 * 2 + \omega + 5$
$7^2 * 2 + 7^1 * 1 + 4$	$8^2 * 2 + 8^1 + 1 + 4 - 1$	139	$\omega^2 * 2 + \omega + 4$
$8^2 * 2 + 8^1 * 1 + 3$	$9^2 * 2 + 9^1 * 1 + 3 - 1$	173	$\omega^2 * 2 + \omega + 3$
$9^2 * 2 + 9^1 * 1 + 2$	$10^2 * 2 + 10^1 * 1 + 2 - 1$	211	$\omega^2 * 2 + \omega + 2$
$10^2 * 2 + 10^1 * 1 + 1$	$11^2 * 2 + 11^1 * 1 + 1 - 1$	253	$\omega^2 * 2 + \omega + 1$
$11^2 * 2 + 11^1 * 1$	$12^2 * 2 + 12^1 * 1 - 1$	299	$\omega^2 * 2 + \omega$
$12^2 * 2 + 11$	$13^2 * 2 + 10$	348	$\omega^2 * 2 + 11$
		1058	
$23^2 * 2$	$24^2 * 2 - 1$	1151	$\omega^2 * 2$
$24^2 + 24 * 23 + 23$	$25^2 + 25 * 23 + 23 - 1$	1222	$\omega^2 + \omega * 23 + 23$

Example 3 shows the Goodstein sequence with initial value 4 upto its 25-th term. The last column should be ignored on first reading. We will come back to it in the next subsection. Also $G_k(4)$ will eventually reach 0, but for k in the order of magnitude of $10^{121210700}$.

The following mathematical formalization of these informal explanations differ in detail greatly from those in the paper [10]. Intuitively the value of the function $\mathrm{oHNf}(n, m)$ is obtained by computing the hereditary base-n representation of m and replacing all occurences of n by $n + 1$. This is turned into the following recursive definition:

Definition 1 **(**$\mathrm{oHNf}(n.m)$**).** *For* $n \geq 2, m \geq 0$

$$\mathrm{oHNf}(n,m) = \begin{cases} m & \textit{if } m < n \\ (n+1)^{\mathrm{oHNf}(n,k)} * a + \mathrm{oHNf}(n,c) & \textit{if } m = n^k * a + c \textit{ with} \\ & 1 \leq k \wedge 0 < a < n \wedge c < n^k \end{cases}$$

This is a complete definition since we can easily prove:

$$\forall m, n(2 \leq n \wedge n \leq m \rightarrow \exists r, a, c\ (m = n^r * a + c \quad \wedge \\ 1 \leq r \wedge 0 < a \wedge a < n \wedge c < n^r \wedge 0 \leq c))$$

Definition 2 **(**$G_n(m)$**).** *For* $n \geq 2$ *and* $m \geq 0$

$G_1(m) = m$

$G_k(m) = \mathrm{oHNf}(k, G_{k-1}(m)) - 1$

3.3 Termination Proof

Close inspection showed that the original termination proof of Goodstein sequences in [10] is more complicated than necessary. We follow instead the idea of a short proof of the termination of general Goodstein sequences from [16,17]. Figure 11 shows the Java program to be verified. Since in the default setting the KeY system treats Java integers as mathematical integers this is what we need. Running this program, however, would yield wrong results as soon as `maxInt` is reached. Since exponention is not part of the Java language, the method `intPow` had to be implemented. Since this is a standard task, the code is not shown here.

We do not assume that every reader is familiar with program verification and will complement the code with explaining comments. Figure 11 contains the Java code plus annotations in the Java Modeling Language (JML). A lucid introduction to JML can be found in [1, Chap. 7]. The following comments should, we hope, be sufficient to provide the reader with a clear understanding of the central points. JML annotations needed to guide the system but not essential for the casual human reader have been omitted, i.e., replaced by ....

JML allows to add specifications enclosed between special comments `/*@...@*/` to a Java program. Formal verification then provides a mathematical proof that the code meets its JML specifications.

JML provides method contracts. These are placed immediately before the method code. The crucial method in Listing 11 is `GoodsteinSequence` in lines 5–14. Its method contract spans lines 2–4. The `requires` clause states a precondition that must be met to guarantee the postcondition. Here the precondition requires the initial value for the Goodstein sequence to be strictly positive. The postcondition is - in this case - hidden in the keyword `normal_behaviour`. This says that the method `GoodsteinSequence` terminates and no uncaught exceptions

```
1  public class Goodstein{
2    /*@ normal_behavior
3      @  requires startM > 0;
4      @*/
5   public void GoodsteinSequence(int startM){
6     int base = 2; int m = startM;
7       /*@ loop_invariant
8         @  m >= 0 & base >= 2;
9         @  ...
10        @  decreases \dl_oGS(base,m);
11        @*/
12       while   (m > 0) {
13           m = nextExpand(m,base);
14           m = m-1;
15           base = base+1;} }
16  /*@ normal_behavior
17    @ requires m >= 0 & oldBase >= 2;
18    @ ...
19    @ ensures \result ==  \dl_oHNf(oldBase,m);
20    @ ensures \dl_oGS(oldBase,m)==\dl_oGS(oldBase+1,\result);
21    @ measured_by m;  @*/
22   public int nextExpand(int m, int oldBase){
23      if (oldBase > m) { return m;}
24      else { int exp=0; int factor=1; int base=oldBase;
25       /*@ loop_invariant
26         @  factor == intPow(oldBase,exp) &   ... ;
27         @  decreases (m - factor);
28         @*/
29   while (m>=factor*oldBase){exp=exp+1; factor=factor*oldBase;};
30         int a = 1; int c = 0;
31  /*@ normal_behavior
32    @ ensures m == \dl_pow(oldBase, exp) * a + c &
33    @ 2<=oldBase&1<=exp&0<a&a<oldBase&c<\dl_pow(oldBase,exp)&c>=0;
34    @ ...
35    @*/
36      {a =  m/factor; c = m - factor*a;};
37   int r = intPow(oldBase+1,nextExpand(exp,oldBase))*a;
38       r = r + nextExpand(c,oldBase);
39     return r;
40     }}}
```

Fig. 11. Goodstein program for verification

are thrown. The code of the method consists of a simple `while` loop computing the Goodstein sequence and breaking out when 0 is reached. Now, a second type of JML contracts comes into play namely, loop contracts. The first part of the loop contract in lines 7–11 requires that the formula `m >= 0 & base >= 2` be true before entering the loop and after each iteration of the loop body. This is easy. The crucial part of the loop contract is the `decreases` clause that gives a quantity in some well-founded ordering that is strictly decreased with every loop iteration. Here in line 10 the function oGS(n, m) provides an ordinal for this purpose. The KeY prover knows about the function oGS, but it is by no means part of standard JML. The escape sequence `\dl_oGS` triggers the JML parser to pass oGS directly to the underlying logic. The same applies for the functions oHNf and *pow*. In line 19 the JML keyword $\backslash result$ for the first time. It denotes the return value of the method the annotation belong to, in this case the return value of `nextExpand`.

The definition of $\mathrm{oGS}(n, m) : int \times int \to Ord$ lies at the very heart of the termination argument. Informally, $\mathrm{oGS}(n, m)$ is computed by replacing in the hereditary base-n expansion of m every occurence of n by ω.

Definition 3. *For $n \geq 2$, $m \geq 0$*

$$\mathrm{oGS}(n,m) = \begin{cases} onat(m) & if\ m < n \\ (\omega)^{\mathrm{oGS}(n,k)} * onat(a) + \mathrm{oGS}(n,c) & if\ m = n^k * a + c\ with \\ & 1 \leq k \wedge 0 < a < n \wedge c < n^k \end{cases}$$

Examples of oGS for an initial segment of the Goodstein sequence with initial value 4 are displayed in the last column in Example 3. The next lemma lists the crucial properties of oGS, that have also been interactively verified with the KeY prover.

Lemma 1

1. $\forall n, m_1, m_2 (2 \leq n \wedge 0 \leq m_1 < m_2 \to \mathrm{oGS}(n, m_1) < \mathrm{oGS}(n, m_2))$
2. $\forall n, m (2 \leq n \wedge 0 \leq m \to \mathrm{oGS}(n, m) = \mathrm{oGS}(n + 1, \mathrm{oHNf}(n, m)))$

We conclude the subsection by revealing the plan to prove that oGS decreases. An arbitrary loop iteration starts with $\mathrm{oGS}(base, m)$. After normal termination of the loop body, the decreasing function evaluates to $\mathrm{oGS}(base+1, \backslash result-1)$, where $\backslash result$ is the return value of the call to method `nextExpand` in line 13. The method contract for `nextExpand` guarantees that $\mathrm{oGS}(base, m)$ and $\mathrm{oGS}(base + 1, \backslash result)$ are equal (line 20). By Lemma 1(1), *oSG* is strictly monotone in its second argument. Thus, $\mathrm{oGS}(base + 1, \backslash result - 1)$ is strictly smaller than $\mathrm{oGS}(base + 1, \backslash result)$ in the ordinal ordering. Bingo.

In this argument, we have made use of the method contract for `nextExpand`, but we also need to establish it. It turns out that for this we need to know that the return value is $\mathrm{oHNf}(oldBase, m)$, Line 19 and Lemma 1(2) will come into play at this point.

4 Concluding Remarks

What are the limits of Th_{Ord}? Let ϵ_0 be the first epsilon ordinal, i.e., the least ordinal ϵ with $\omega^\epsilon = \epsilon$. Let $\mathcal{M}_{\epsilon_0}$ be the structure with all ordinals less than ϵ_0 as universe and the standard interpretation of Σ_{Ord}. It can be easily checked that $\mathcal{M}_{\epsilon_0}$ is a model of Th_{Ord}. Closure under *sup* is the crucial part. This shows that $\exists x(\omega^x = x)$ cannot be derived in Th_{Ord}. It can furthermore be shown that for a model $\mathcal{M}$ of Th_{Ord} that does not contain nonstandard natural numbers $\mathcal{M}_{\epsilon_0}$ is a substructure of $\mathcal{M}$. In a way, Th_{Ord} is the analogon of Peano arithmetic for

$\mathcal{M}_{\epsilon_0}$. Precise formulations of these claims and complete proofs can be found in the technical report [19].

If we had only intended to present a machine assisted proof of the mathematical theorem that all Goodstein sequences terminate, this could already have been obtained from Lemma 1. We wanted – however – to make the point that there are simple Java programs whose termination cannot be proved in Peano arithmetic, but Th_{Ord} is strong enough to prove it.

Runable Java code, saved proofs and the version of the KeY system needed can be downloaded or "webstarted" from https://www.key-project.org/papers/ordinal-numbers/.

References

1. Ahrendt, W., Beckert, B., Bubel, R., Hähnle, R., Schmitt, P.H., Ulbrich, M. (eds.): Deductive Software Verification - The KeY Book - From Theory to Practice. LNCS, vol. 10001. Springer, Cham (2016). doi:10.1007/978-3-319-49812-6
2. Bachmann, H.: Transfinite Zahlen. Ergebnisse der Mathematik und ihrer Grenzgebiete, vol. 1, 2nd edn. Springer, Heidelberg (1967). doi:10.1007/978-3-642-88514-3
3. Belinfante, J.G.F.: On computer-assisted proofs in ordinal number theory. J. Autom. Reason. **22**(2), 341–378 (1999)
4. Blanchette, J.C., Fleury, M., Traytel, D.: Nested multisets, hereditary multisets, and syntactic ordinals in isabelle/hol (under submission)
5. Blanchette, J.C., Fleury, M., Traytel, D.: Formalization of nested multisets, hereditary multisets, and syntactic ordinals. Archive of Formal Proofs, November 2016. http://isa-afp.org/entries/Nested_Multisets_Ordinals.shtml. Formal proof development
6. Castéran, P., Contejean, E.: On ordinal notations. https://github.com/coq-contribs/cantor
7. Castéran, P., Contejean, E.: On ordinal notations. https://coq.inria.fr/V8.2pl1/contribs/Cantor.epsilon0.Goodstein.html
8. Grimm, J.: Implementation of three types of ordinals in Coq. Research report RR-8407, CRISAM - Inria Sophia Antipolis (2013)
9. Huffman, B.: Countable ordinals. Archive of Formal Proofs, November 2005. http://afp.sf.net/entries/Ordinal.shtml. Formal proof development
10. Kirby, L., Paris, J.: Accessible independence results for Peano arithmetic. Bull. Lond. Math. Soc. **14**(4), 285–293 (1982)
11. Klaua, D.: Kardinal- und Ordinalzahlen, Teil 1. Wissenschaftliche Taschenbücher: Mathematik, Physik. Vieweg Braunschweig (1974)
12. Klaua, D.: Kardinal- und Ordinalzahlen, Teil 2. Wissenschaftliche Taschenbücher: Mathematik, Physik. Vieweg Braunschweig (1974)
13. Manolios, P., Vroon, D.: Algorithms for ordinal arithmetic. In: Baader, F. (ed.) CADE 2003. LNCS (LNAI), vol. 2741, pp. 243–257. Springer, Heidelberg (2003). doi:10.1007/978-3-540-45085-6_19
14. Manolios, P., Vroon, D.: Ordinal arithmetic: algorithms and mechanization. J. Autom. Reason. **34**(4), 387–423 (2005)
15. 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, Heidelberg (2013). doi:10.1007/978-3-642-39634-2_12

16. Rathjen, M.: Goodstein revisited. ArXiv e-prints, May 2014
17. Rathjen, M.: Goodstein's theorem revisited. In: Kahle, R., Rathjen, M. (eds.) Gentzen's Centenary, pp. 229–242. Springer, Cham (2015). doi:10.1007/978-3-319-10103-3_9
18. Goodstein, R.L.: On the restricted ordinal theorem. JSL **9**, 33–41 (1944)
19. Schmitt, P.H.: A first-order theory of ordinals. Technical report 6, Department of Informatics, Karlsruhe Institute of Technology (2017)
20. Takeuti, G.: A formalization of the theory of ordinal numbers. J. Symb. Logic **30**, 295–317 (1965)
21. Takeuti, G., Zaring, W.M.: Introduction to Axiomatic Set Theory. Graduate Texts in Mathematics, vol. 1. Springer, New York (1971). doi:10.1007/978-1-4684-9915-5
22. Ulbrich, M.: Dynamic logic for an intermediate language: verification, interaction and refinement. Ph.D. thesis, Karlsruhe Institute of Technology, June 2013
23. Winkler, S., Zankl, H., Middeldorp, A.: Beyond Peano arithmetic–automatically proving termination of the goodstein sequence. In: van Raamsdonk, F. (ed.) 24th International Conference on Rewriting Techniques and Applications (RTA 2013). Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, vol. 21, pp. 335–351. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik (2013)