Karlsruher Institut für Technologie

Karlsruhe Reports in Informatics 2018,2
Edited by Karlsruhe Institute of Technology,
Faculty of Informatics
ISSN 2190-4782

Takeuti's First-Order Theory of Ordinals Revisited

Peter H. Schmitt

2018

KIT – University of the State of Baden-Wuerttemberg and National Research Center of the Helmholtz Association

Fakultät für Informatik

Takeuti's First-Order Theory of Ordinals Revisited

Peter H. Schmitt

Karlsruhe Institute of Technology (KIT), Dept. of Informatics
Am Fasanengarten 5, 76131 Karlsruhe, Germany

Abstract. These notes contain technical details that could not be fitted into the paper submitted to IJCAR 2018. The conference submission analyses the relationship between Takeuti's theory of ordinals published in [6] and my theory in the paper [5].

1 Introduction

In [5] a theory Th^0_{ord} of ordinals was proposed. The theory was implemented in the KeY program verification system and was used to mechanically derive a large body of the result on ordinal arithmetic known from the pertinent textbooks. A detailed report of is effort is available in the technical report [4] . The paper also contained an account of a small case study proving the termination of Goodstein sequences. Since termination of Goodstein sequences cannot be proved in Peano Arithmetic this shows that Th^0_{ord} is strictly stronger than Peano Arithmetic.

A previous paper by Gaisi Takeuti [6] that also presented a theory Th_{Tak} of ordinals was quoted in [5]. We remarked that Th_{Tak} is more geared towards the construction of an inner model of full Zermelo-Fraenkel (ZF) set theory and less suitable for implementation than our theory Th^0_{ord}. In these notes we clarify the relation between Th^0_{ord} and Th_{Tak}. We consider a variation Th_{ord} of Th^0_{ord} that only differs in a stronger form of the replacement axiom scheme. The main result of the paper is that a definitional extention of Th_{ord} is equivalent to Th^-_{Tak}, which is Th_{Tak} without the cardinality axiom, axiom number 22 in Figure 23.

What do these notes supply that the conference paper does not:

1. With every axiom of Th_{ord} and its numerous definitional extension the name of the *taclet* is given that formalizes it in the KeY system.
   This helps to find the proof scripts for each derived lemma. The name of the taclet is part of the name of the .key file containing the proof script.
2. Sections 7 and 4

2 A Theory of Ordinals

We consulted the books [1,7] and also the books [2,3] in German on ordinal arithmetic in axiomatic set theory.

2.1 The Core Theory

We start out with a very simple core of the theory Th_{ord} with the vocabulary shown in Figure 1. It is more minimalistic than in [5] in that the supremum

	mathematical notation	Dynamic Logic
predicate	$n < m \ : \ (Ord, Ord)$	$olt(n, m)$
functions	$n + 1 \ : Ord \to Ord$	$oadd(n, o\_1)$
	$0 \ : \ Ord$	o_0
	$\omega \ : \ Ord$	$omega$

Fig. 1. The vocabulary of the Core Theory

operator is not in it. It will be introduced in definitional extensions further down the road. Also the theory in [5] used only a special case of the replacement axioms scheme.

1. $\forall x, y, z(x < y \wedge y < z \to x < z)$ transitivity
   taclet: `olt_transAxiom, olt_trans, olt_transAut`
2. $\forall x(\neg x < x)$ strict order
   taclet `olt_irref_Axiom, olt_irref`
3. $\forall x, y(x < y \vee x \doteq y \vee y < x)$ total order
   taclet: `olt_total_Axiom`
4. $\forall x(0 \leq x)$ 0 is smallest element
   taclet: `oleq_zeroAxiom, olt_OMin, oleq_zero`
5. $0 < \omega \wedge \neg\exists x(\omega \doteq x + 1)$ ω is a limit ordinal
   taclet: `omegaDef1`
6. $\forall y(0 < y \wedge \forall x(x < \omega \to x + 1 < y) - > \omega \leq y)$ ω is the least limit ordinal
   taclet: `omegaDefLeastInf`
7. $\forall x(x < x + 1) \wedge \forall x, y(x < y \to x + 1 \leq y)$ $x + 1$ is successor function
   taclets: `oSucc, oLeastSucc`
8. $\forall x(\forall y(y < x \to \phi(y)) \to \phi) \to \forall x\phi$ transfinite induction scheme
   taclets: `oIndBasic`
9. $\forall x, y, z(\phi(x, y) \wedge \phi(x, z) \to y = z) \to$ replacement axiom scheme
   $\forall a \exists b \forall y(\exists x(\phi(x, y) \wedge x < a) \to y < b)$
   taclet: `oReplacementScheme`
10. $\forall x, y(x \leq y \leftrightarrow x < y \vee x = y)$ Def. of $\leq$
    taclet `oleq_Def, oleq_replace`
11. $\forall x(lim(x) \leftrightarrow x \neq 0 \wedge \neg\exists y(x = y + 1))$ Def. of limit ordinal
    taclet: `olimDef`

Fig. 2. The axioms of the Core Theory

In this text we use the mathematical notation throughout. Figure 1 also gives the corresponding notation in Dynamic Logic that would be used in writing taclets.

The intended meaning of the symbols in Figure 1 is fixed by the core axioms in Figure 2. Definition of the auxilliary predicates $\leq$ and lim have already been included here in items 10 and 11 to facilitate the formalisation of the core axioms.

Axioms 1 to 4 state that $<$ is a strict linear ordering with least element 0. In Axiom 4 $\leq$ is of course defined by $x \leq y \leftrightarrow (x < y \vee x = y)$. Axiom 9 which is taken from [6] requires some explanation. If the premiss of the axiom $\forall x, y, z(\phi(x, y) \wedge \phi(x, z) \rightarrow y = z)$ is true we may unambiguously define a unary function f by $f(x) = y \Leftrightarrow \phi(x, y)$. The righthand side of the implication in Axiom 9 may thus be rewritten as $\forall a \exists x \forall z(z < a) \rightarrow f(z) < x)$ which is an instance of the replacement axiom of Zermelo-Fraenkel set theory.

Also a remark on our quite pragmatic notation used in the formulation of axiom schemata is in order here. The formula ϕ occuring in axiom scheme 8 may contain x as a free variable or not. The reader can convince himself that in case x does not occur free in ϕ the axioms reduces to a tautology. $\phi(y)$ is to stand for the formula arising from ϕ by replacing every free occurence of x by y, assuming that his does not lead to a clash with bound occurences of y in ϕ. There might be other free variables $\bar{x} = \langle x_1, \ldots, x_k \rangle$ in ϕ besides x. These are implicitly universally quantified. If you want to see this explicitely you have to put $\forall \bar{x}$ in front of $\forall x$ with scope extending over the whole formula.

The same remarks apply to the axiom scheme 9: $\phi(x, y)$ signals that we are interested in the free variables x, y in ϕ. There is no commitment involved that x or y actually occur free in ϕ. The reader is invited it convince himself that in case x or y does not occur freely the formula is either tautological or an easy consequence of $z < z + 1$. As mentined in the preseding paragraph $\phi(x, z)$ arises from ϕ by replacing every free occurence of y with z, as always assuming that this can be done without clashes. If ϕ contains further free variables $\bar{x}$ other than x and y these are implicitly universally quantified.

These remarks apply to all axioms or lemma schemes, in particular to those in Figure 3.

Figure 3 shows some important and usefull consequences of the core axioms. Lemma 13 is a frequently used variant of the induction scheme. To proof $\forall x \phi$ it suffices to proof three inductive steps.

1. The initial case, $\phi(0)$,
2. the successor inductive step,
   if $\phi(x)$ holds then also $\phi(x + 1)$ is true, and
3. the limit inductive step,
   for any limit number x such that $\phi(y)$ is true for all ordinals y less than x also $\phi(x)$ is true.

Lemma 15 is the special instance of our replacement scheme 15 with $\phi = y \doteq t$ where t is a term that will typically contain the variable λ. This is the version of the replacement scheme used in Th^0_{ord} in [5]

12. $lim(\lambda) \leftrightarrow \lambda \neq 0 \wedge \forall ov(ov < \lambda \rightarrow (ov + 1) < \lambda)$ equivalent Def. of limit numbers
taclets: `olimDefEquiv, olimDefAdd,notLim1,notLim2`
13. $\phi(o_0)\wedge$
$\forall x(\phi(x) \rightarrow \phi(x + 1))\wedge$
$\forall x(lim(x) \wedge \forall y(y < x \rightarrow \phi(y)) \rightarrow \phi(x)$
$\rightarrow$
$\forall x\phi(x)$ variant of induction scheme
taclet: `oInd`
14. $\exists x\phi \rightarrow \exists x(\phi \wedge \forall y(y < x \rightarrow \neg\phi(y)))$ least number principle
taclet: `least_number_principle`
15. $\forall a\exists b\forall\lambda(\lambda < a \rightarrow t < b)$ special case of replacement scheme
taclet: `oSpecialReplacment`

Fig. 3. Basic Lemmas of the Core Theory

3 First Definitional Extensions

16. $0 + 1 = 1$ Def. of constant 1
taclets: `one_Def, oadd01`
17. $\forall x(\neg x < 0)$ taclet: `olt_zero`
18. $0 < 1$ taclet `olt_01`
19. $0 \neq 1$ taclet: `oDiff01`
20. $\forall x(0 < x \rightarrow 1 \leq x)$ taclet `olt_discret`
21. $\forall x(x < 1 \rightarrow x = 0)$ taclet: `olt_one`
22. $0 < \omega$ taclet `omegaZero`
23. $1 < \omega$ taclet: `omegaOne`

Fig. 4. Definitional Extension for constant 1

$x + 1$ is a unary function, which we could have named - if we wanted to - also by $s(x)$. As a first and simple definitional extension we find it useful to also have the constant 1 available. This is defined in item 16 in Figure 4. Axiom 17 is an easy consequence of the inductive definition of addition to be considered later. But, we did not want to wait that long. Figure 4 then lists some easy lemmas about constant 1. We want these lemmas to be applied atomatically by the prover

Before we move on to more substantial extentions and lemmas we look at a few simple and useful lemmas in Figure 5 and 6. Sometimes more than one taclet is associated with a mathematical statement. In these cases the taclets take different forms to facilitate automatic proof search. There is a proof, of course, for every taclet. No further comments on these lemmas are needed.

24. $\forall x(x+1 \neq 0)$ taclet `o0notSuccQ, o0notSucc`
25. $x+1 \doteq y+1 \rightarrow x \doteq y$ taclet `OSuccInjective`
26. $x < y \rightarrow x+1 < y+1$ taclet `oltPlusOne`
27. $x \leq y \rightarrow x+1 \leq y+1$ taclet `oleqPlusOne`

Fig. 5. Definitional Extension for immediate successor

28. $x < y+1 \rightarrow (x < y \vee x \doteq y)$ taclet `olessPlusOne`
29. $x \leq y \wedge y \leq z \rightarrow x \leq z$ taclet `oleq_trans, oleq_transAut`
30. $x \leq y \wedge y < z \rightarrow x < z$
taclet `oltleq_trans, oltleq_transAut, oleqolt_transQ`
31. $x < y \wedge y \leq z \rightarrow x < z$ taclet `oleqolt_trans, oleqolt_transAut`
32. $x < y \rightarrow \neg y < x$ taclet `irrByolt`
33. $x \leq y \rightarrow \neg y < x$ taclet `irrByoltleq`
34. $x < y \rightarrow \neg y \leq x$ taclet `olt2oleq`
35. $x \leq y \wedge y \leq x \rightarrow x \doteq y$ taclet `oleq_antisym`

Fig. 6. Lemmas on transitivity and related topics

Figure 7 exhibits in Line 36 the explicit definition of the binary maximum operator and some easy consequences in the remaining lines. We found lemmas 47 - 50 particularly useful in the successor case of inductive proofs.

Let's move on to Figure 8. Axiom 51 defines the supremums operator $sup_{\lambda<t_0} t_1(\lambda)$, i.e., the least ordinal that is greater to or equal to all ordinals in the set $\{t_1(\lambda) \mid \lambda < t_0\}$. Here the term t will typically contain the variable λ, while λ is not allowed to occur in α.

While it is obvious that adding the binary maximum operator is a definitional extension an argument is needed that this is also true for adding the supremum operator. We assume that the reader is in sofar familiar with the concept of definitional extension that he knows that the following lemma is a sufficent condition.

Lemma 1. *Let $\mathcal{M}$ be a model of the core theory. Then we can define an expansion $\mathcal{M}_1$ that interprets for any two terms t_0, t_1 such that the variable λ does not occur in t_0 the function $sup_{\lambda<t_0}(t_1)$ such that $\mathcal{M}_1$ satisfies the axiom scheme 51.*

That $\mathcal{M}_1$ is an expansion of $\mathcal{M}$ means that all syntax apart from the sup operator is interpreted in $\mathcal{M}_1$ in the same way as it is interpreted in $\mathcal{M}$, and also that the universes of $\mathcal{M}_1$ and $\mathcal{M}$ are the same.

Proof. A complete proof would proceed by induction on the number of occurences of the *sup* operator in t_0 or t_1. We concentrate on the case where t_0 or t_1 do not contain the *sup* operator trusting that the reader can fill in the routine details for the general case.

For ease of notation we further assume that t_0 is a ground term, i.e. contains no variables, and that t_1 only contains the variable λ. Otherwise we would have

36. $\forall x, y(omax(x,y) \doteq (\text{if } x \leq y \text{ then } y \text{ else } x))$	Def. of binary maximum taclet: `omaxDef`
37. $z < omax(x,y) \leftrightarrow (z < x \vee z < y)$	taclet `omaxLess`
38. $omax(x,y) < z \leftrightarrow (x < z \wedge y < z)$	taclet `omaxGreater`
39. $z \leq omax(x,y) \leftrightarrow (z \leq x \vee z \leq y)$	taclet `omaxLeq`
40. $omax(x,y) \leq z \leftrightarrow (x \leq z \wedge y \leq z)$	taclet `omaxGeq`
41. $omax(0,x) \doteq x$	taclet `omax0Left`
42. $omax(x,0) \doteq x$	taclet `omax0Right`
43. $x \leq omax(x,y)$	taclet `omaxLeft`
44. $y \leq omax(x,y)$	taclet `omaxRight`
45. $(x < y \wedge y \doteq z) \rightarrow x < z$	taclet `WRolteq`
46. $omax(x,y) \doteq omax(y,x)$	taclet `omaxSymQ`
47. $x < y \rightarrow omax(x+1,y) \doteq omax(x,y)$	taclet `omaxPlusOnR`
48. $x < y \rightarrow omax(y,x+1) \doteq omax(x,y)$	taclet `omaxPlusOnL`
49. $omax(x,y+1) \leq omax(x,y)+1$	taclet `omaxPlusOneQR`
50. $omax(x+1,y) \leq omax(x,y)+1$	taclet `omaxPlusOneQL`

Fig. 7. Definitional Extensions for *omax*

to start with an arbitrary instantiation $\bar{c}$ of the extra variables in t_0 and t_1 which would only clog notation. Note, that the assumption that λ does not occur free in t_0 is crucial here.
This said, let a_0 denote the interpretation of t_0 in $\mathcal{M}$, in symbols $a_0 = t_0^{\mathcal{M}}$. From Lemma 15 we obtain

$$\mathcal{M} \models \exists b \forall x(x < a_0 \rightarrow t_1 < b) \tag{1}$$

By the least number principle we get a smallest ordinal b_0 such that $\mathcal{M} \models \forall x(x < a_0 \rightarrow t_1 < b_0)$. We now set

$$sup^{\mathcal{M}_0}_{\lambda < t_0}(t_1) = b_0$$

It is easy to check that with this stipulation $\mathcal{M}_1$ satisfies the axiom scheme 51. □

Besides the definition of *sup* Figure 8 lists properties of *sup*, simple ones and crucial ones. Equation 52 is true regardless of t. Lemma 54 could be rephrased as: x is the least ordinal that is greater or equal than all ordinals that are strictly less than x. This is only true if x is a limit ordinal. In the successor case we have $sup_{\lambda < x+1}\lambda \doteq x$. Lemma 56 is usefull in proving statements involving the *sup* operator via induction. Lemma 57 helps to show that two suprema are equal especially in the case when equality between t_1 and t_2 is not obvious.

We may look at a term t that contains λ as a sequence t_λ. We say sequence $t_{\lambda<\alpha_1}$ is confinal in $s_{\lambda<\alpha_2}$ if for every $x < \alpha_1$ there is $y < \alpha_2$ with $t[x/\lambda] \leq s[y/\lambda]$. If two sequences are mutually confinal in one another than they share the same supremum. This is Lemma 58 in Figure 8. Note, that we get equality of two suprema with different bounds α_1 and α_2. Lemmata 59 and 60 give simple and

51. $\forall x(x < t_0 \to t_1(x) \leq sup_{\lambda<t_0}(t_1(\lambda))) \quad \wedge$ Def. of supremum
$\forall y(\forall x(x < t_0 \to t_1(x) \leq y) \to sup_{\lambda<t_0}(t_1) \leq y))$
taclet: `osupDef`
52. $sup_{\lambda<0}t \doteq 0$ taclet `osup0`
53. $sup_{\lambda<1}t \doteq t[0]$ taclet `osup1`
54. $lim(x) \to sup_{\lambda<x}\lambda \doteq x$ taclet `oselfSup`
55. $sup_{\lambda<x+1}\lambda \doteq x$ taclet `oselfSupSuc`
56. $sup_{\lambda<x+1}t \doteq omax(sup_{\lambda<x}t, t[x])$ taclet `osupSucc`
57. $\forall\lambda(t_1 \doteq t_2) \to sup_{\lambda<x}t_1 \doteq sup_{\lambda<x}t_2$ taclet `osupEqualTerms`
58. $\forall x(x < z_1 \to \exists y(y < z_2 \wedge t_1[x] \leq t_2[y])) \wedge \forall y(y < z_2 \to \exists x(x < z_1 \wedge t_2[y] \leq t_1[x]))$
$\to sup_{\lambda<z_1}t_1 \doteq sup_{\lambda<z_2}t_2$
taclet `osupMutualCofinal`
59. $\forall\lambda(t_1 \leq t_2) \to sup_{\lambda<b}\ t_1 \leq sup_{\lambda<b}\ t_2$ taclet: `osupLocalLess`
60. $b_1 \leq b_2 \to sup_{\lambda<b_1}t \leq sup_{\lambda<b_2}t$ taclet: `osupShorter`
61. $sup_{\lambda<\omega}\lambda = \omega$ taclet: `enum:osupOmega`

Fig. 8. Definitional Extensions for *sup*

62. $\forall x, y, z(x \leq y \wedge y \leq z \to x \leq z)$ taclets. `oleq_trans, oleq_transAut`
63. $\forall x, y, z(x \leq y \wedge y < z \to x < z)$ taclets. `oltleq_trans, oltleq_transAut`
64. $\forall x, y, z(x < y \wedge y \leq z \to x < z)$ taclets: `oleqolt_trans, oleqolt_transAut`
65. $\forall x, y, z(z < (max(x, y) \leftrightarrow (z < x \vee z < y)))$ taclet: `omaxLess`
66. $\forall x, y, z(max(x, y) < z \leftrightarrow (x < z \wedge y < z))$ taclet: `omaxGreater`
67. $\forall x, y(max(x, y) \doteq max(y, x))$ taclet: `omaxSymQ`

Fig. 9. Derivable taclets on $\leq$ and *max*

and useful criteria for one supremum being less than another in specific cases. Lemma 61 is a sometimes useful special case of 54.

Figure 9 shows a set of derivable lemmas using $\leq$ and max. Though the KeY prover is rather strong in finding suitable instantiations of universal quantifiers it is completely at a loss to find useful instantiations of the three quantifiers involved in the transitivity axioms. Two taclets `olt_trans` and `olt_transAut` picking up suitable instantiations from the open goals are among the taclets not reproduced here. The lemmas shown in Figure 9 concern first the variations of transitivity where $\leq$ occurs once or twice instead of $<$. Secondly, lemmas involving the maximum function are shown.

We would like to view the natural numbers as a subtype of the ordinals. Since KeY offers only rudimentary support for subtypes we had to find another way to use natural numbers as ordinals. We introcude an injection $onat : \mathbb{Z} \to Ord$. Definition and some derivable consequences are presented in Figure 10. Note, that for negative integers $onat$ is not specified. To keep formulas short and readable our notation does not contain information on the type of a variable symbol. We trust that the reader can infer the type for the context. If $onta(n)$ occurs then

47. $onat(0) \doteq 0$ taclet: `onatZeroDef`
48. $0 \leq n \to onat(n+1) \doteq onat(n) + 1$ taclet: `onatSuccDef`
49. $onat(1) \doteq 1$ taclet: `onatOne`
50. $onat(2) \doteq (0+1) + 1$ taclet: `onatTwo`
51. $onat(3) \doteq onat(2) + 1$ taclet: `onatThree`
52. $onat(4) \doteq onat(3) + 1$ taclet: `onatFour`
53. $onat(5) \doteq onat(4) + 1$ taclet: `onatFive`
54. $onat(6) \doteq onat(5) + 1$ taclet: `onatSix`
55. $onat(7) \doteq onat(6) + 1$ taclet: `onatSeven`
56. $onat(8) \doteq onat(7) + 1$ taclet: `onatEight`
57. $onat(9) \doteq onat(8) + 1$ taclet: `onatNine`
58. $(0 \leq n \wedge 0 \leq m) \to onat(n+m) \doteq onat(n) + onat(m)$ taclet: `onatoadd`
59. $(0 \leq n \wedge 0 \leq m \wedge onat(n) \doteq onat(m)) \to n \doteq m$ taclet: `onatInj`
60. $(0 \leq n \wedge 0 \leq m) \to (onat(n) < onat(m) \leftrightarrow n < m)$ taclet: `onatolt, onatoltAut`
61. $0 \leq n \to onat(n) < \omega$ taclet: `onatLessOmega`

Fig. 10. Definition of and lemmas for the injection *onat*

n must be of type integer. Also the same constants 0, 1, ... are used for both integers and ordinals as well as + for integer and ordinal addition.

4 Ordinal Arithmetic

Ordinal arithmetic plays no role in the analysis of Takeuti's theory of ordinals. When we started this work it was not clear whether ordinal arithmetic might at some point be necessary or at least convenient. Anyhow, ordinal arithmetic could also be used in other project.

62. $x + 0 \doteq x$ taclet: `oadd_Def0Right`
63. $x + (y+1) \doteq (x+y) + 1$ taclet: `oadd_DefSucc`
64. $lim(y) \to x + y \doteq sup_{\lambda<y}(x+\lambda)$ taclet: `oadd_DefLim`
65. $x * 0 \doteq 0$ taclet: `otimes_Def0Right`
66. $x * (y+1) \doteq x * y + x$ taclet: `otimes_DefSucc`
67. $lim(y) \to x * y \doteq sup_{\lambda<y}(x * \lambda)$ taclet: `otimes_DefLim, otimes_DefLimQ`
68. $x^0 \doteq 1$ taclet: `oexp_Def0Right`
69. $x^{y+1} \doteq x^y * x$ taclet: `oexp_DefSucc`
70. $(lim(y) \wedge 0 < x) \to x^y \doteq sup_{\lambda<y} x^\lambda$ taclet: `oexp_DefLim`
71. $lim(y) \to 0^y \doteq 0$ taclet: `oexp_DefLim0`

Fig. 11. Definition of ordinal arithmetic operations

72. $y \neq 0 \rightarrow x < x + y$ taclet: `oaddStrictMonotone`
73. $x \leq x + y$ taclet: `oaddMonotone`
74. $y \leq x + y$ taclet: `oaddLeftMonotone`
75. $x + y \doteq 0 \rightarrow (x \doteq 0 \wedge y \doteq 0)$ taclet: `zerosum`
76. $x < y \rightarrow z + x < z + x$ taclet: `oltAddLessLeft`
77. $x \leq y \rightarrow z + x \leq z + x$ taclet: `oleqAddLessLeft`
78. $x \leq y \rightarrow x + z \leq y + z$ taclet: `oleqAddLessRight, oleqAddLessRightQ`
79. $(x < y \wedge u < w) \rightarrow x + u < y + w$ taclet: `oadd2olt`
80. $(x \leq y \wedge u \leq w) \rightarrow x + u \leq y + w$ taclet: `oadd2oleq`
81. $max(z + x, z + y) \doteq z + max(x, y)$ taclet: `omaxAddL`
82. $max(x + z, y + z) \doteq max(x.y) + z$ taclet: `omaxAddR`

Fig. 12. Lemmas on addition and order

We start out with the definition of the three arithmetic operations in Figure 11.

72. $lim(y) \to lim(x + y)$ taclet: `olimAddolim`
73. $lim(x) \to \omega \leq x$ taclet: `omegaLeastLim1, omegaLeastLim2`
74. $(lim(x) \wedge x \leq \omega) \to x \doteq \omega$ taclet: `omegaLeastLim3`
75. $lim(x) \to 0 < x$ taclet: `limitZero`
76. $lim(x) \to 1 < x$ taclet: `limitOne`
77. $z + x \doteq z + y \to x \doteq y$ taclet: `oaddRightInjective`
78. $(lim(y) \wedge x < y) \to (x + 1) < y$ taclet: `olimDedekind`
79. $x < \omega \wedge y < \omega) \to (x + y) < \omega$ taclet: `oaddLessOmega, oaddLessOmegaAxiom`
80. $0 + x \doteq x$ taclet: `oadd0Left`
81. $x < \omega \to x + \omega \doteq \omega$ taclet: `oaddLeftomega`
82. $(x < \omega \wedge \omega \leq y) \to x + y \doteq y$ taclet: `oaddLeftAbsorb`
83. $\omega \leq x \to \exists y, n(lim(y) \wedge n < \omega \wedge x \doteq y + n)$ taclet: `repLimPlusNat`
84. $x \leq y \to \exists z(x + z \doteq y)$ taclet: `ordDiff`
85. $x + 1 \doteq y + 1 \to x \doteq y$ taclet: `oAddOneInj`
86. $x + y < x + z \to y < z$ taclet: `oAdd0ltPreserv`
87. $b \neq 0 \to sup_{\lambda<b}(x + y) = x + sup_{\lambda<b}y$ if λ not free in x taclet: `osupAddStaticTerm`
88. $x + (y + z) \doteq (x + y) + z$ taclet: `oaddAssoc`
89. $y < \omega \to 1 + y \doteq y + 1$ taclet: `oaddFiniteComOn`
90. $(x < \omega \wedge y < \omega) \to x + y \doteq y + x$ taclet: `oaddFiniteCom`

Fig. 13. Lemmas on addition

91. $x * 1 \doteq x$ taclet: `otimesOneRight`
92. $1 * x \doteq x$ taclet: `otimesOneLeft`
93. $0 * x \doteq 0$ taclet: `otimesZeroLeft`
94. $(0 < z \wedge x < y) \rightarrow z * x < z * y$ taclet: `otimesMonotone, otimesMonotoneQ`
95. $x \leq y \rightarrow z * x \leq z * y$ taclet: `otimesWeakMonotoneQ`
96. $z * x < z * y \rightarrow (0 < z \wedge x < y)$ taclet: `otimesMonotoneRev`
97. $(0 < z \wedge z * x \doteq z * y) \rightarrow x \doteq y$ taclet: `otimesLeftInjective`
98. $x \leq y \rightarrow x * z \leq y * z$ taclet: `otimesLeftMonotone`
99. $0 \neq x \rightarrow y \leq x * y$ taclet: `otimesRightMonotoneQ`
100. $x * y \doteq 0 \rightarrow (x \doteq 0 \vee y \doteq 0)$ taclet: `otimesZero`
101. $x * y \doteq 1 \rightarrow (x \doteq 1 \wedge y \doteq 1)$ taclet: `otimesOne`
102. $(x < \omega \wedge y < \omega) \rightarrow x * y < \omega$ taclet: `otimesFiniteAxiom, otimesFinite`
103. $(x \neq 0 \wedge x < \omega) \rightarrow x * \omega \doteq \omega$ taclet: `otimesNomega, otimesNomegaQ`
104. $max(z * x, z * y) \doteq z * max(x, y)$ taclet: `omaxTimesL`
105. $max(x * z, y * z) \doteq max(x, y) * z$ taclet: `omaxTimesR`
106. $sup_{\lambda < b} x * y \doteq x * sup_{\lambda < b} y$ provided λ is not free in x. taclet: `osupTimesStaticTerm`
107. $x * (y + z) \doteq x * y + x * z$ taclet: `odistributive, odistributiveQ`
108. $(x < \omega \wedge y < \omega \wedge z < \omega) \rightarrow (x + y) * z \doteq x * z + y * z$ taclet: `odistributiveFinite`
109. $x * (y * z) \doteq (x * y) * z$ taclet: `otimesAssoc, otimesAssocQ`
110. $(x < \omega \wedge y < \omega) \rightarrow x * y \doteq y * x$ taclet: `otimesFiniteCom`
111. $(x < \omega \wedge y < \omega \wedge \omega * x < \omega * y) \rightarrow x < y$ taclet: `oltomegatimes`
112. $(x_1 < \omega \wedge x_2 < \omega \wedge y_1 < \omega \wedge y_2 < \omega \wedge$
$\omega * x_1 + y_1 < \omega * x_2 + y_2) \rightarrow$
$\omega * x_1 < \omega * x_2 \vee (\omega * x_1 \doteq \omega * x_2 \wedge y_1 < y_2)$
taclet: `oltlexicographic`
113. $(0 \leq n_1 \wedge 0 \leq n_2 \wedge 0 \leq m_1 \wedge 0 \leq m_2 \wedge$
$\omega * onat(n_1) + onat(m_1) < \omega * onat(n_2) + onat(m_2) \rightarrow$
$n_1 < n_2 \vee (n_1 \doteq n_2 \wedge m_1 < m_2)$
taclet: `oltlexicographicInt`
114. $(1 < x \wedge 1 < y) \rightarrow (x + y) \leq x * y$ taclet: `oleqAddTimes`
115. $(0 < x \wedge lim(y)) \rightarrow lim(x * y)$ taclet: `olimtimes1, olimtimes1Q`
116. $(0 < y \wedge lim(x)) \rightarrow lim(x * y)$ taclet: `olimtimes2, olimtimes2Q`
117. $(x \neq 0 \wedge y < \omega) \rightarrow (x + y) * \omega \doteq x * \omega$ taclet: `Klaua26c1a`
118. $(x \neq 0 \wedge y < \omega \wedge lim(z)) \rightarrow (x + y) * z \doteq x * z$ taclet: `Klaua26c1`
119. $(x < \omega \wedge lim(z)) \rightarrow ((x \doteq 0 \wedge x * z \doteq 0) \vee (x \neq 0 \wedge x * z \doteq z))$ taclet: `otimesNlimit`

Fig. 14. Lemmas on multiplication

120. $x^1 \doteq x$ taclet: `oexpOne`

Fig. 15. Lemmas on exponentiation

120. $y \neq 0 \to \exists z(y * z \leq x \wedge x < y * (z+1))$ taclet: `oleastMultiple`
121. $y \neq 0 \to \exists d, r(x \doteq y * d + r \wedge r < y)$ taclet: `odivQ`
122. $(y \neq 0 \wedge r_1 < y \wedge r_2 < y \wedge y * d_1 + r_1 \doteq y * d_2 + r_2) \to (d_1 \doteq d_2 \wedge r_1 \doteq r_2)$
taclet: `odivUnique`
123. $lim(y) \to \exists x(y \doteq \omega * x)$ taclet: `odivLim`

Fig. 16. Lemmas on decomposition

5 Well-ordering Pairs of Ordinals

62. $(v_1, v_2) \ll (v_3, v_4) \leftrightarrow$ Def. of $\ll$
$max(v_1, v_2) < max(v_3, v_4) \vee$
$max(v_1, v_2) = max(v_3, v_4) \wedge v_2 < v_4 \vee$
$max(v_1, v_2) = max(v_3, v_4) \wedge v_2 = v_4 \wedge v_1 < v_3$
taclets: `oltp_DefAxiom, oltp_Def`
63. $(0, 0) \ll (1, 0)$ taclet `oltpLess10`
64. $(0, 0) \ll (0, 1)$ taclet `oltpLess011`
65. $(1, 0) \ll (0, 1)$ taclet `oltpLess012`
66. $(0, x) \ll (0, 1) \to x \doteq 0$ taclet `oltpLess013`
67. $(x, y) \ll (1, 0) \to (x \doteq 0 \wedge y \doteq 0)$ taclet `oltpOne`
68. $(x, y) \ll (0, 1) \to ((x \doteq 0 \wedge y \doteq 0) \vee (x \doteq 1 \wedge y \doteq 0))$ taclet `oltpTwo`
69. $\forall v_1 \forall v_2 (\neg (v_1, v_2) \ll (v_1, v_2))$ taclets: `oltp_irr`
70. $\forall v_1 \forall v_2 \forall v_3 \forall v_4 \forall v_5 \forall v_5 ($
$(v_1, v_2) \ll (v_3, v_4) \wedge (v_3, v_4) \ll (v_5, v_6) \to (v_1, v_2) \ll (v_5, v_6))$
taclets: `oltp_transQ, oltp_trans`
71. $\forall v_1 \forall v_2 \forall v_3 \forall v_4 ((v_1, v_2) \ll (v_3, v_4) \vee (v_3, v_)) \ll (v_1, v_2) \vee (v_1 \doteq v_3 \wedge v_2 \doteq v_4))$
taclet: `oltp_totalAxiom`
72. $(t_1, t_2) \ll (t_3, t_2) \leftrightarrow t_1 < t_3$ taclet: `oltp_same2`
73. $(t_1, t_2) \ll (t_1, t_3) \leftrightarrow t_2 < t_3$ taclet: `oltp_same1`
74. $\forall v_1 \forall v_2 ((v_1, v_2) \ll (v_1 + 1, v_2)$ taclet: `oltp_addOneL`
75. $\forall v_1 \forall v_2 ((v_1, v_2) \ll (v_1, v_2 + 1)$ taclet: `oltp_addOneR`
76. $\exists v_1 \exists v_2 \phi(v_1, v_2) \to \exists v_1 \exists v_2 (\phi(v_1, v_2) \wedge \forall w_1 \forall w_2; (w_1, w_2) \ll (v_1, v_2) \to \neg \phi(w_1, w_2))$
taclet: `oltpLeastPair2, oltpLeastPair`
77. $\forall v_1, v_2 (\forall v_3, v_4; ((v_3, v_4) \ll (v_1, v_2) \to \phi(v_3, v_4)) \to \phi(v_1, v_2)) \quad \to \forall v_1 \forall v_2 \phi(v_1, v_2)$
taclet: `oltpInduction`

Fig. 17. Definition and fundamental consequences of $\ll$

One of the main goals of these notes is a closer look at the first-order theory of ordinals in [6] and its comparision with [5,4]. Among the axioms of this theory are axioms on coding and decoding of pairs of ordinals The basis of this coding is a well-ordering of pairs of ordinals, that we denote here by $\ll$. Item 62 in Figure 17 gives the definition of $\ll$: pairs are first odered by their maximum and pairs with the same are ordered lexicographically with the last entry as the most significant. This relation $\ll$ is irreflexive (Lemma 69), transitive (Lemma 70) and total (Lemma 71). Lemmas 74 and 75 states that increasing the first or second entry by one yields a greater pair, which in neither case needs to be an immediate successor as is shown by the examples $(2, 3) \ll (4, 1) \ll (2, 4)$ and $(3, 2) \ll (4, 1) \ll (4, 2)$. The topic of immediate succesors and limit pairs will be treated extensively in Figure 18 and the comments following it.

The most important fact about $\ll$ is that it is a well-ordering. The property of a well ordering, i.e. that there is not infinite decending chain $p_1 \gg p_2 \gg$

$\ldots \gg p_k \gg \ldots$ cannot be expressed in our first-order logic. The best we can do is to prove that induction with respect to $\ll$ works, Lemma 77. It turned out that induction is more cumbersome to prove than the equivalent least element principle. Thus we prove this property first, Lemma 76. To derive induction from the least pair principle is now a piece of cake.

78. $\forall v_1, v_2, w_1, w_2(succp(v_1, v_2, w_1, w_2) \leftrightarrow (v_1, v_2) \ll (w_1, w_2) \land \forall v_3, v_4((v_1, v_2) \ll (v_3, v_4) \rightarrow (w_1, w_2) = (v_3, v_4) \lor (w_1, w_2) \ll (v_3, v_4)))$ taclet: `succp_Def`
79. $\forall v_1, v_2, w_1, w_2(succp(v_1, v_2, w_1, w_2) \rightarrow \forall v_3, v_4((v_3, v_4) \ll (w_2, w_3) \rightarrow (v_1, v_2) = (v_3, v_4) \lor (v_3, v_4) \ll (w_1, w_2)$ taclet: `succpConseq`
80. $\forall v_1, v_2, v_3(succp(v_1, v_1, v_1 + 1, 0))$ taclets: `oltpSuccEq2`, `oltpSuccEq`
81. $\forall v_1, v_2(v_2 + 1 < v_1 \rightarrow succp(v_1, v_2, v_1, v_2 + 1)$ taclets: `oltpSuccMaxFirst`, `oltpSuccMaxFirstA`
82. $\forall v_1, v_2; (v_2 + 1 = v_1 \rightarrow succp(v_1, v_2, 0, v_2 + 1))$ taclets: `oltpSuccMaxFirst2`, `oltpSuccMaxFirst2A`
83. $\forall v_1, v_2(v_1 < v_2 \rightarrow succp(v_1, v_2, v_1 + 1, v_2))$ taclets: `oltpSuccMaxSecond`, `oltpSuccMaxSecond2`
84. $succp(v1, v1, w1, w2) \rightarrow w1 \doteq v1 + 1 \land w2 \doteq 0$ taclets: `succpConseqEqQ, succpConseqEq`
85. $(v2 + 1 < v1 \land succp(v1, v2, w1, w2) \rightarrow w1 \doteq v1 \land w2 \doteq v2 + 1$ taclets: `succpConseqLessRP, succpConseqLessRPQ`
86. $succp(v2 + 1, v2, w1, w2) \rightarrow w1 \doteq 0 \land w2 \doteq v2 + 1$ taclets: `succpConseqLessR, succpConseqLessRQ`
87. $(v1 < v2 \land succp(v1, v2, w1, w2)) \rightarrow w1 \doteq v1 + 1 \land w2 \doteq v2$ taclets: `succpConseqGreater, succpConseqGreaterQ`
88. $\forall v_1, v_2 \exists w_1, w_2(succp(v_1, v_2, w_1, w_2))$ taclets: `succpExists`
89. $succp(v_1, v_2, w_1, w_2) \rightarrow max(w_1, w_2) \leq max(v_1, v_2) + 1$ taclets: `succpOmax`

Fig. 18. Successor pairs in the well-ordering $\ll$

Now, that we know that $\ll$ is a well-ordering, or at least that the usual induction principle is true for this ordering, we can begin to investigate the notions of successor and limit pairs. When we say *successor* we always mean *immediate successor*. We introduce a new predicate $succp(x_1, x_2, y_1, y_2)$ for (y_1, y_2) to be the immediate successor of (x_1, x_2), and the predicate $limp(x_1, x_2)$ for (x_1, x_2) to be a limit pair. Their definitions in Lines 78 of Figure 18 and Line 90 of Figure 19 do not come as a surprise. We use $(w_1, w_2) = (v_3, v_4)$ as a shorthand for $w_1 = v_3 \land w_2 = v_4$. By definition (y_1, y_2) is a successor pairs of (x_1, x_2) if it is the least pair strictly greater then (x_1, x_2). This entails that (x_1, x_2) is the greatest pair strictly less than (y_1, y_2). This is stated in Line 79.

Conditions on when a pair is a succesors of (n_1, n_2) are given by the lemmas 80 to 83 and is summarized in the following table

successor of (n_1, n_2)	condition
$(n_1 + 1, 0)$	$n_1 = n_2$
$(n_1, n_2 + 1)$	$n_1 > n_2 + 1$
$(0, n_2 + 1)$	$n_1 = n_2 + 1$
$(n_1 + 1, n_2)$	$n_1 < n_2$

Thus

$$
\begin{array}{l}
(0,0) \ll \\
(1,0) \ll (0,1) \ll (1,1) \ll \\
(2,0) \ll (2,1) \ll (0,2) \ll (1,2) \ll (2,2) \ll \\
(3,0) \ll (3,1) \ll (3,2) \ll (0,3) \ll (1,3) \ll (2,3) \ll (3,3) \ll \\
\ldots \\
(\omega,0) \ll \ldots \ll (\omega,n) \ll \ldots (0,\omega) \ll \ldots (n,\omega) \ll \ldots (\omega,\omega) \ll
\end{array}
$$

Since we defined successor pairs by a predicate as opposed to a function we need to prove that also the reverse implications are true, i.e., for every case in the above summary table we need to verify that when a pair is the successor of (n_1, n_2) then the stated condition applies. This it the content of Lines 84 to 87 in Figure 18. From these the uniqueness of successors follows easily, i.e., from $succp(v_1, v_2, w_1, w_2)$ and $succp(v_1, v_2, w_3, w_4)$ we get $(w_1, w_2) = (w_3, w_4)$. We did not state this explicitly as a derived lemma. We did however find need to include the lemma that successors always exist, Line 88. Line 89 gives a useful restiction on the growth of the components of the successor pair.

90. $limp(t_1, t_2) \leftrightarrow \forall x, y((x, y) \ll (t_1, t_2) \to \exists u, w((x, y) \ll (u, w) \land (u, w) \ll (t_1, t_2)))$ taclet `enum:limp_Def`
91. $limp(x, y) \leftrightarrow \neg\exists u, w(succp(u, w, x, y))$ taclet `limp_DefAlt`
92. $limp(x_1, x_2) \land (y_1, y_2) \ll (x_1, x_2) \land succp(y_1, y_2, z_1, z_2) \to (z_1, z_2)) \ll (x_1, x_2)$ taclet `limp_SuccLess`
93. $limp(0, 0)$ taclet: `oltpLimZeroZero`
94. $succp(x, y, u, w) \to \neg limp(u, w)$ taclet: `limpSuccFalse`
95. $\forall v_1, v_2(lim(v_2) \land v_2 \leq v_1 \to limp(v_1, v_2))$ taclet: `oltpLimR`
96. $\forall v_1, v_2(lim(v_1) \land v_1 \leq v_2 \to limp(v_1, v_2 + 1))$ taclet: `oltpLimL`
97. $\forall v(lim(v) \to limp(0, v))$ taclet: `:oltpLimZeroR`
98. $\forall v(lim(v) \to limp(v, 0))$ taclet: `:oltpLimZeroL`
99. $\forall v_1, v_2(lim(v_1) \land lim(v_2) \to limp(v_1, v_2))$ taclet: `oltpLimLim`
100. $\forall v_1, v_2(limp(v_1, v_2) \to (v_1 = 0 \land v_2 = 0) \lor (v_1 = 0 \land lim(v_2)) \lor$
$(lim(v_1) \land v_2 = 0) \lor (lim(v_1) \land lim(v_2)) \lor$
$(v_2 \leq v_1 \land lim(v_2)) \lor (lim(v_1) \land \exists v_3(v_2 = v_3 + 1 \land v_1 < v_2)$
taclet: `limpConseq`

Fig. 19. Limit pairs in the well-ordering $\ll$

Lemmas 93 to 99 list conditions on the parts v_1 and v_2 that ensure that the pair (v_1, v_2), or $(v_1, v_2 + 1)$ in the case of lemma 96 is a limit pair. Lemma 100 states that the conditions from lemmas 93 to 99 exhaustively list all restrictions on v_1, v_2 for (v_1, v_2), respectively $(v_1, v_2 + 1)$, to be a limit pair. We remark that in contrast to the situation with ordinals where 0 was neither a successor nor a limit ordinal the pair $(0, 0)$ is a limit ordinal by the way we defined *limp*.

6 Coding Pairs of Ordinals

After the extensive preparations in Section 5 we can now take up defining a coding for pairs of ordinals. More precisely we introduce a function $encode : Ord\times \to Ord$ for encoding and $decode1 : Ord \to Ord$, $decode2 : Ord \to Ord$ for decoding.

It will be of great help to have another version of the induction principle from Line 77 in Figure 17 at our disposal. This new version, shown in Line 101 in figure 20, may be put in words as follow: If we can show

if	$\phi(v_1, v_2)$ is true then
	$\phi(w_1, w_2)$ is true for the successor pair (w_1, w_2) of (v_1, v_2)
and	
if	$\phi(w_1, w_2)$ is true for every pair (w_1, w_2) less than a limit pair (v_1, v_2)
	then $\phi(v_1, v_2)$ is true

then we have proved $\forall v_1, v_2 \phi(v_1, v_2)$.

The idea for the coding function *encode* is now quite simple: $encode(v_1, v_2)$ is the position of the pair (v_1, v_2) in the well-ordering $\ll$. This leads to

$$
\begin{array}{lll}
encode(0, 0) & = 0 & \\
encode(w_1, w_1) & = encode(v_1, v_2) + 1 & \text{if } succp(v_1, v_2, w_1, w_2) \\
encode(v_1, v_1) & = \text{the least ordinal greater than} & \\
 & \quad encode(w_1, w_2) \text{ for all } (w_1, w_2) \ll (v_1, v_2) & \text{if } limp(v_1.v_2)
\end{array}
$$

These definitions correspond to Lines 102 to 104 in Figure 20. As first consequences we list $encode(v_1, v_2) = n$ for $1 \leq n \leq 4$ in Line 105 of Figure 20. Compare this also to the diagram on page 14. For finite n, m we have in general

$$encode(n, m) = \{ \begin{matrix} n^2 & \text{if } m < n \\ m^2 + n + m & \text{if } n \leq m \end{matrix} .$$

This last equation has been proved using paper and pencil, is has not been verified with a theorem prover.

As a proper coding function *encode* should be injective. As a stepping stone we first prove (strict) monotony. as formulated in Line 107 of Figure 20 while injectivity follows in Line 109.

We remark that strict monotony easily implies weak monotony as formulated in Line 108.

It is a special, and very convenient, feature of the encoding function *encode* that it is surjective, i.e. every ordinal is the code of a pair of ordinals, Line

101. $\forall v_1, v_2(\ \phi(v_1, v_2) \to (\forall w_1, w_2(succp(v_1, v_2, w_1, w_2) \to \phi(w_1, w_2)))$
$\wedge$
$limp(v_1, v_2) \wedge \forall w_1, w_2((w_1, w_2) \ll (v_1, v_2) \to \phi(w_1, w_2)) \to \phi(v_1, v_2))$
$\to \quad \forall v_1, v_2 \phi(v_1, v_2)$
taclet: `oltpInd2`

102. $encode(0, 0)) \doteq 0$ taclet: `encodeZero`

103. $succp(v_1, v_2, w_1, w_2) \to encode(w_1, w_2) \doteq encode(v_1, v_2) + 1$ taclet: `encodeSucc`

104. $limp(v_1, v_2) \to (\forall w_1, w_2((w_1, w_2) \ll (v_1, v_2) \to encode(w_1, w_2) < encode(v_1, v_2))$
$\wedge$
$\forall x(\forall w_1, w_2((w_1, w_2) \ll (v_1, v_2) \to encode(w_1, w_2) < x)$
$\to encode(v_1, v_2) \leq x)$ *taclet* : `encodeLim`

105. $encode(1, 0) = 1 \wedge encode(0, 1) = 2 \ \wedge$
$encode(1, 1) = 3 \wedge encode(2, 0) = 4$
taclets: `encodeOne, encodeTwo, encodeThree, encodeFour`

106. $encode(x, y) \doteq 0 \to (x \doteq 0 \wedge y \doteq 0)$ taclets: `encodeZeroV`

107. $(v_1, v_2) \ll (w_1, w_2) \to encode(v_1, v_2) < encode(w_1, w_2)$ taclets: `encodeMonotone`

108. $((v_1, v_2) \ll (w_1, w_2) \vee (v_1, v_2) = (w_1, w_2)) \to encode(v_1, v_2) \leq encode(w_1, w_2)$
taclets: `encodeweakMonotone`

109. $encode(v_1, v_2) = encode(w_1, w_2) \to (v_1, v_2) = (w_1, w_2)$ taclets: `enum:encodeInj`

110. $\forall v_1, v_2(max(v_1, v_2) \leq encode(v_1, v_2))$ taclets: `encodeWeakIncreasing`

111. $\forall x, y(a + x \leq encode(a, x))$ taclet: `oaddEncode`

112. $\forall v_1, v_2, w_1, w_2; (encode(v_1, v_2) < encode(w_1, w_2) \to (v_1, v_2) \ll (w_1, w_2))$
taclet: `encodeoltpLess`

113. $\forall w \exists v_1, v_2(encode(v_1, v_2) = w)$ taclets: `encodeSurjective`

Fig. 20. Encoding pairs of ordinals

113. In the course of the proof of surjectivity the weak increasing property from Line 110 is used. If $max(v_1, v_2) > 1$ and $max(v_1, v_2) \neq \omega$ then even $max(v_1, v_2) < encode(v_1, v_2)$ is true. We did not need this fact, so it is not included in the list of proved lemmas.

114. $\forall w(encode(decode1(w), decode2(w)) = w)$ taclets: `decodeDef`

115. $\forall v_1, v_2(decode1(encode(v_1, v_2)) = v_1)$ taclets: `decode1Id`

116. $\forall v_1, v_2(decode2(encode(v_1, v_2)) = v_2)$ taclets: `decode2Id`

Fig. 21. Decoding for pairs of ordinals

Let us not turn to decoding. The definition of the two decoding functions is given in the one axiom in Line 114 of Figure 21. On the face of it this axiom looks just a some property that we want the decoding functions to satisfy. But, by what we have proved about the encoding function there is exactly one way to define *decode*1 and exactly one way to define *decode*2 such that the axiom holds true. Lines 115, 116 in Figure 21 present two lemmas derivable from the defining axiom.

7 The Bounded μ-Operator

Takeuti includes in his axioms in [6] a bounded μ-operator $\mu_{x<b}i(x)$ for terms t with the semantics

$$\begin{aligned}\mu_{x<b}t(x,\bar{a}) &= a \ \text{ if } t(a,\bar{a}) = 0 \text{ and } t(c,\bar{a}) \neq 0 \text{ for all } c < a \\ &= 0 \ \text{ if } t(c,\bar{a}) \neq 0 \text{ for all } c\end{aligned}$$

We introduce a definitional extension with the new variable binder $omu_{x<b}i(x)$. The axioms and some derivable lemmas are shown in Figure 22. We note that the term i may have other free variables besides x. Lemma 118 is just a simple test, while Lemmas 119 and 120 are the axioms in Takeuti's theory.

117. $(\exists y(y < b \wedge i(y) \doteq 0) \rightarrow \quad i(omu_{x<b}i(x)) \doteq 0 \wedge \forall z(z < b) \rightarrow i(z) \neq 0)))$
$\wedge$
$(\neg\exists y(y < b \wedge i(y) \doteq 0) \rightarrow omu_{x<b}i(x) \doteq 0)$
taclet: `omuDef`
118. $omu_{x<0}i(x) \doteq 0$ taclet: `omuZero`
119. $(i(c) \doteq 0 \wedge c < b) \rightarrow i(omu_{x<b}i(x)) \doteq 0 \wedge i(omu_{x<b}i(x)) < c$ taclet: `omuTak1`
120. $omu_{x<b}i(x) \doteq 0 \vee (i(omu_{x<b}i(x)) \doteq 0 \wedge omu_{x<b}i(x) < b)$ taclet: `omuTak2`

Fig. 22. The bounded μ-operator

We could with the same effort have introduced a bounded least-element operator that works on a formula instead of a term, i.e., $\mu_{x<b}\phi(x)$ is the least element $c < b$ such that $\phi(c)$ is true and 0 if there is no $c < b$ with $\phi(c)$. Since this operator plays no role in Takeuti's paper [6] we ignored this generalization. In total we have thus shown that Th_{Tak} is equivalent to a definitional extension of Th_{ord}.

8 Review of Takeuti's Theory

Figure 23 lists the axioms of Takeuti's theory of ordinals. Following the presentation in [6] we systematically omit explicit universal quantification, but we stick to the names of variables used in the previous sections. Instead of x' we use $x + 1$.

For each axiom in Figure 23 we list on the far right the axiom or lemma in a definitional extention of Th_{ord} that entails it. Thus there is a definitional extention of Th_{ord} that is at least as strong as Th_{Tak}.

In lines 12 and 13 a new binary function symbol $less : Ord \times Ord \rightarrow Ord$ is introduced. In [6] $\boldsymbol{<}$ is used instead of $less$. Since boldface $\boldsymbol{<}$ is hard to distinquish from regular $<$ we switched to the different notation $less$. This has no counterpart in our theory. If necessary $less(x, y)$ can be replaced by (if $x < y$ then 0 else 1). We also use more telling notation:

$$\begin{array}{ll} encode(x, y) & \text{for } j(x, y) \\ decode1(x) & \text{for } g^1(x) \\ decode2(x) & \text{for } g^2(x) \end{array}$$

For the formula on the righthand side of axiom 17 we used in the previous sections the relation $(v, w) \ll (x, y)$. It is a central part in the derivation of Takeuti's axioms from the axioms of Th_{ord} and its definitial extensions to show that $\ll$ is a well-ordering on pairs of ordinals.

1.	$x < y \lor x \doteq y \lor y < x$	axiom 3, Fig.2
2.	$\neg x < x$	axiom 2, Fig.2
3.	$x < y \land y < z \to x < z$	axiom 1, Fig.2
4.	$0 < \omega$	axiom 5, Fig.2
5.	$0 \leq x$	axiom 4, Fig.2
6.	$x < y \to x + 1 \leq y$	axiom 7, Fig.2
7.	$x < x + 1$	axiom 7, Fig.2
8.	$x < \omega \to x + 1 < \omega$	easy consequence of axiom 5, Fig.2
9.	$0 < y \land \forall x(x < y \to x + 1 < y) - > \omega \leq y)$	easy consequence of axiom 6, Fig.2
10.	$x \leq y \leftrightarrow max(x, y) \doteq y$	definition 36, Fig. 7
11.	$max(x, y) \doteq max(y, x)$	lemma 67, Fig. 9
12.	$x < y \leftrightarrow less(x, y) \doteq 0$	
13.	$less(x, y) \leq 1$	
14.	$encode(decode1(x), decode2(x)) = x$	lemma 114, Fig. 21
15.	$decode1(encode(x, y)) = x$	lemma 115, Fig. 21
16.	$decode2(encode(x, y)) = x$	lemma 116, Fig. 21
17.	$encode(v, w) < encode(x, y) \leftrightarrow max(v, w) < max(x, y) \lor$ $(max(v, w) \doteq max(x, y) \land w < y) \lor$ $(max(v, w) \doteq max(x, y) \land w \doteq y \land v < x)$	lemmas 107, 112, Fig. 20
18.	$(i(c) \doteq 0 \land c < b) \to i(\mu_{x<b} i(x)) \doteq 0 \land i(\mu_{x<b} i(x)) < c$	lemma 119 in Fig. 22
19.	$\mu_{x<b} i(x) \doteq 0 \lor (i(\mu_{x<b} i(x)) \doteq 0 \land \mu_{x<b} i(x) < b)$	lemma 120 in Fig. 22
20.	$\forall x(\forall y(y < x \to \phi(y)) \to \phi) \to \forall x \phi$	axiom 8 in Fig. 2.
21.	$\forall x, y, z(\phi(x, y) \land \phi(x, z) \to y = z) \to$ $\forall a \exists b \forall y(\exists x(\phi(x, y) \land x < a) \to y < b)$	axiom 9 in Fig. 2.
22.	$\exists u(\forall v, x, y, z(\phi(y, x, v) \land \phi(z, x, v) \to y = z)$ $\to \forall v \exists x(x < u \land \forall y(y < a \to \neg\phi(x.y.v))))$	not derivable from Th_{ord}

Fig. 23. Takeuti's Theory of Ordinals

Axioms 18 and 19 are Takeuti's definition of the bounded μ-operator. Lemmas 119 and 120 show that they follow from our definition of this operator in line

117 in Figure 22. Takeuti's axioms 20 and 21, the induction and replacement schemata, are literally the same as ours.

We also notice that Th_{Tak} is at least as strong as Th_{ord}: axioms 1 to 3 plus 5 in Figure 23 guarantee that $<$ is a strict total ordering with least element 0, axioms 4, 8, and 9 characterize ω as the least limit ordinal, axioms 6 and 7 stipulate the $x+1$ is the immediate succesor fo x. Finally the axiom schemes for transfinite induction and regularity are literally the same in both theories.

References

1. H. Bachmann. *Transfinite Zahlen*, volume 1 of *Ergebnisse der Mathematik und ihrer Grenzgebiete*. Springer Verlag, 2 edition, 1967.
2. D. Klaua. *Kardinal- und Ordinalzahlen, Teil 1*. Wissenschaftliche Taschenbücher: Mathematik,Physik. Vieweg Braunschweig, 1974.
3. D. Klaua. *Kardinal- und Ordinalzahlen, Teil 2*. Wissenschaftliche Taschenbücher: Mathematik,Physik. Vieweg Braunschweig, 1974.
4. P. H. Schmitt. A first-order theory of ordinals. Technical Report 6, Department of Informatics, Karlsruhe Institute of Technology, 2017.
5. P. H. Schmitt. A mechanizable first-order theory of ordinals. In R. A. Schmidt and C. Nalon, editors, *Automated Reasoning with Analytic Tableaux and Related Methods - 26th International Conference, TABLEAUX 2017, Brasília, Brazil, September 25-28, 2017, Proceedings*, volume 10501 of *Lecture Notes in Computer Science*, pages 331–346. Springer, 2017.
6. G. Takeuti. A formalization of the theory of ordinal numbers. *Journal of Symbolic Logic*, 30(295-317), 1965.
7. G. Takeuti and W. M.Zaring. *Introduction to Axiomatic Set Theory*, volume 1 of *Graduate Texts in Mathematics*. Springer Verlag, 1971.