Computing a Model of Set Theory

Peter Koepke

Mathematisches Institut,
Rheinische Friedrich-Wilhelms-Universität Bonn,
Beringstraße 1,
53113 Bonn, Germany

Abstract. We define the notion of *ordinal computability* by generalizing standard Turing computability on tapes of length ω to computations on tapes of arbitrary ordinal length. The generalized Turing machine is able to compute a recursive bounded truth predicate on the ordinals. The class of sets of ordinals which can be read off the truth predicate satisfies a natural theory SO. SO is the theory of the sets of ordinals in a model of the Zermelo-Fraenkel axioms ZFC. Hence a set of ordinals is ordinal computable from ordinal parameters if and only if it is an element of Gödel's constructible universe L.

1 Introduction

A standard Turing computation may be visualized as a time-like sequence of elementary *read-write-move* operations carried out by one or more "heads" on "tapes". The sequence of actions is determined by the initial tape contents and by a finite Turing *program*. We may assume that Turing machines act on tapes whose cells are indexed by the set ω $(=\mathbb{N})$ of *natural numbers* $0, 1, \ldots$ and contain 0's or 1's.

		SPACE									
		0	1	2	3	4	5	6	7	...	...
	0	$\underline{1}$	0	0	1	1	1	0	0	0	0
	1	0	$\underline{0}$	0	1	1	1	0	0		
T	2	0	0	$\underline{0}$	1	1	1	0	0		
I	3	0	$\underline{0}$	1	1	1	1	0	0		
M	4	0	1	$\underline{1}$	1	1	1	0	0		
E	:										
	n	1	1	1	1	$\underline{0}$	1	1	1		
	$n+1$	1	1	1	1	1	$\underline{1}$	1	1		
	:										

A standard Turing *computation. Head positions are indicated by underlining.*

An obvious generalization from the perspective of transfinite ordinal theory is to extend Turing calculations to tapes whose cells are indexed by the class Ord of all *ordinal numbers*. At *limit* ordinals we define the tape contents, program states and head positions by appropriate limit operations which may be viewed as *inferior limits*.

		Ordinal Space															
		0	1	2	3	4	5	6	7	…	…	ω	…	α	…	κ	…
O	0	$\underline{1}$	1	0	1	0	0	1	1	…	…	1	…	1	0	0	0
r	1	0	$\underline{1}$	0	1	0	0	1	1			1					
d	2	0	0	$\underline{0}$	1	0	0	1	1			1					
i	3	0	$\underline{0}$	0	1	0	0	1	1			1					
n	4	0	0	$\underline{0}$	0	0	0	1	1			1					
a	:																
l	n	1	1	1	1	$\underline{0}$	1	0	1			1					
	$n+1$	1	1	1	1	1	$\underline{1}$	0	1			1					
T	⋮	⋮	⋮	⋮	⋮	⋮											
i	ω	0	0	1	0	0	0	1	1	…	…	$\underline{1}$					
m	$\omega+1$	$\underline{0}$	0	1	0	0	0	1	1			0					
e	:																
	θ	1	0	0	1	1	1	1	0	…	…	…	…	$\underline{0}$	…	…	…
⋮	⋮			⋮			⋮			⋮	⋮						
	⋮																

An ordinal computation.

This notion of *ordinal computability* obviously extends TURING computability. By the CHURCH-TURING thesis many operations on natural numbers are ordinal computable. The *ordinal arithmetical* operations (addition, multiplication, exponentiation) and other basic operations on ordinals are also ordinal computable.

Using GÖDEL's pairing function $G : \mathrm{Ord} \times \mathrm{Ord} \to \mathrm{Ord}$ one can view each ordinal α as a first-order sentence with constant symbols for ordinals $< \alpha$. One can then define a recursive truth predicate $T : \mathrm{Ord} \to \{0, 1\}$ by:

$$T(G(\alpha, \beta)) = 1 \text{ iff } (\alpha, <, G \cap \alpha^3, T \restriction \alpha) \models \alpha[\beta].$$

This recursion can be carried out by an ordinal TURING machine. For ordinals μ and α the function T codes the set

$$T(\mu, \alpha) = \{\beta < \mu | T(G(\alpha, \beta)) = 1\}.$$

The class

$$\mathcal{S} = \{T(\mu, \alpha) | \mu, \alpha \in \mathrm{Ord}\}$$

is the class of sets of ordinals of a transitive proper class model of set theory. Since the ordinal TURING computations can be carried out in the $\subseteq$-smallest such model, namely GÖDEL's model L of *constructible sets*, we obtain our main result characterizing ordinal computability:

Theorem 1. *A set $x \subseteq \mathrm{Ord}$ is ordinal computable from finitely many ordinal parameters if and only if $x \in L$.*

This theorem may be viewed as an analogue of the Church-Turing thesis: ordinal computability defines a natural and absolute class of sets, and it is stable with respect to technical variations in its definition. This work was inspired by the *infinite time* Turing machines introduced by Joel D. Hamkins, Jeff Kidder and Andy Lewis [3]. A more comprehensive technical account of ordinal computability, also indicating set theoretical applications is given in [5].

2 Ordinal Turing Machines

Ordinal Turing machines are defined in close analogy with standard Turing machines. At *successor ordinals* we use standard Turing steps. The behaviour at *limit ordinals* will be defined by simple limit operations.

Definition 2.

a) *A* ***command*** *is a 5-tuple $C=(s,c,c',m,s')$ where $s,s' \in \omega$ and $c,c',m \in \{0,1\}$; the natural number s is the* ***state*** *of the command C. The intention of the command C is that if the machine is in state s and reads the symbol c under its read-write head, then it writes the symbol c', moves the head left if $m=0$ or right if $m=1$, and goes into state s'. States correspond to the "line numbers" of some programming languages.*
b) *A* ***program*** *is a finite set P of commands satisfying the following structural conditions:*
   i. *If $(s,c,c',m,s') \in P$ then there is $(s,d,d',n,t') \in P$ with $c \neq d$; thus in state s the machine can react to reading a "0" as well as to reading a "1".*
   ii. *If $(s,c,c',m,s') \in P$ and $(s,c,c'',m',s'') \in P$ then $c'=c'', m=m', s'=s''$; this means that the course of the computation is completely determined by the sequence of program states and the initial cell contents.*
c) *For a program P let*

$$\mathrm{states}(P) = \{s | (s,c,c',m,s') \in P\}$$

be the set of program states.

Definition 3. *Let P be a program. A triple*

$$S : \theta \to \omega, H : \theta \to \mathrm{Ord}, T : \theta \to ({}^{\mathrm{Ord}}2)$$

is an ***ordinal computation*** *by P if the following hold:*

a) *θ is a successor ordinal or $\theta = \mathrm{Ord}$; θ is the* ***length*** *of the computation.*
b) *$S(0) = H(0) = 0$; the machine starts in state 0 with head position 0.*

c) *If $t < \theta$ and $S(t) \notin \mathrm{state}(P)$ then $\theta = t+1$; the machine* ***stops*** *if the machine state is not a program state of P.*

d) *If $t < \theta$ and $S(t) \in \mathrm{state}(P)$ then $t+1 < \theta$; choose the unique command $(s, c, c', m, s') \in P$ with $S(t) = s$ and $T(t)_{H(t)} = c$; this command is executed as follows:*

$$T(t+1)_\xi = \begin{cases} c', \text{ if } \xi = H(t); \\ T(t)_\xi \text{ , else;} \end{cases}$$
$$S(t+1) = s';$$
$$H(t+1) = \begin{cases} H(t)+1, \text{ if } m = 1; \\ H(t)-1, \text{ if } m = 0 \text{ and } H(t) \text{ is a successor ordinal;} \\ 0, \text{ else.} \end{cases}$$

e) *If $t < \theta$ is a limit ordinal, the machine constellation at t is determined by taking inferior limits:*

$$\forall \xi \in \mathrm{Ord}\, T(t)_\xi = \liminf_{r \to t} T(r)_\xi;$$
$$S(t) = \liminf_{r \to t} S(r);$$
$$H(t) = \liminf_{s \to t, S(s) = S(t)} H(s).$$

The computation is obviously recursively determined by the initial tape contents $T(0)$ and the program P. We call it the ***ordinal computation by P with input*** *$T(0)$. If the computation stops, $\theta = \beta + 1$ is a successor ordinal and $T(\beta)$ is the final tape content. In this case we say that P* ***computes*** *$T(\beta)$ from $T(0)$ and write $P : T(0) \mapsto T(\beta)$.*

This interpretation of programs yields associated notions of computability.

Definition 4. *A partial function $F : {}^{\mathrm{Ord}} 2 \rightharpoonup {}^{\mathrm{Ord}} 2$ is* ***ordinal computable*** *if there is a program P such that $P : T \mapsto F(T)$ for every $T \in \mathrm{dom}(F)$.*

By coding, the notion of ordinal computability can be extended to other domains. We can e.g. *code* an ordinal $\delta \in \mathrm{Ord}$ by the characteristic function $\chi_{\{\delta\}} : \mathrm{Ord} \to 2$, $\chi_{\{\delta\}}(\xi) = 1$ iff $\xi = \delta$, and define:

Definition 5. *A partial function $F : \mathrm{Ord} \rightharpoonup \mathrm{Ord}$ is* ***ordinal computable*** *if the function $\chi_{\{\delta\}} \mapsto \chi_{\{F(\delta)\}}$ is ordinal computable.*

We also consider computations involving finitely many ordinal *parameters*.

Definition 6. *A subset $x \subseteq \mathrm{Ord}$ is* ***ordinal computable from finitely many ordinal parameters*** *if there a finite subset $z \subseteq \mathrm{Ord}$ and a program P such that $P : \chi_z \mapsto \chi_x$.*

3 Ordinal Algorithms

The intended computations will deal with ordinals and sequences of ordinals. The simplest way of representing the ordinal $\alpha \in \mathrm{Ord}$ in an ordinal machine is by a tape whose content is the characteristic function of $\{\alpha\}$:

$$\chi_{\{\alpha\}} : \mathrm{Ord} \to 2,\ \chi_{\{\alpha\}}(\xi) = 1 \text{ iff } \xi = \alpha.$$

A basic task is to *find* or *identify* this ordinal α: initially the head is in position 0, it then moves to the right until it stops exactly at position α. This is achieved by the following program:

$$P = \{(0,0,0,1,0), (0,1,1,1,1), (1,0,0,0,2), (1,1,1,0,2)\}.$$

The program is in state 0 until it reads a 1, then it goes one cell to the right, one cell to the left, and stops because 2 is not a program state. Informally the algorithm may be written as

```
Find_Ordinal:
0   if head = 1 then STOP otherwise moveright, go to 0
```

It will be convenient to work with several tapes side-by-side instead of just one. One can simulate an n-tape machine on a 1-tape machine. The contents $(T^i_\xi | \xi \in \mathrm{Ord})$ of the i-th tape are successively written into the cells of tape T indexed by ordinals $2n\xi + 2i$:

$$T_{2n\xi+2i} = T^i_\xi.$$

The head position H^i on the i-th tape is simulated by writing 1's into an initial segment of length H^i of cells with indices of the form $2n\xi + 2i + 1$:

$$T_{2n\xi+2i+1} = \begin{cases} 1, \text{ if } \xi < H^i; \\ 0, \text{ else.} \end{cases}$$

So two tapes with contents $a_0a_1a_2a_3a_4\ldots$ and $b_0b_1b_2b_3b_4\ldots$ and head positions 3 and 1 respectively are coded as

$$T = a_0 1 b_0 1 a_1 1 b_1 0 a_2 1 b_2 0 a_3 0 b_3 0 a_4 0 b_4 0 \ldots\ldots$$

There are canonical but tedious translations from programs for n-tape machines into corresponding programs for 1-tape machines. One can assume that one or more of the machine tapes serve as standard TURING tapes on which ordinary TURING recursive functions are computed.

Basic operations on ordinals are ordinal computable. The GÖDEL pairing function for ordinals is defined recursively by

$$\begin{aligned} G(\alpha, \beta) = \{G(\alpha', \beta') | & \max(\alpha', \beta') < \max(\alpha, \beta) \text{ or} \\ & (\max(\alpha', \beta') = \max(\alpha, \beta) \text{ and } \alpha' < \alpha) \text{ or} \\ & (\max(\alpha', \beta') = \max(\alpha, \beta) \text{ and } \alpha' = \alpha \text{ and } \beta' < \beta)\}. \end{aligned}$$

We sketch an algorithm for computing $\gamma = G(\alpha, \beta)$ which can be implemented straightforwardly on a Turing machine with several tapes, each holding one of the variables.

```
Goedel_Pairing:
0  alpha':=0
1  beta':=0
2  eta:=0
3  flag:=0
3  gamma:=0
4  if alpha=alpha' and beta=beta' then print gamma, stop fi
5  if alpha'=eta and and beta'=eta and flag=0 then
      alpha'=0, flag:=1, go to 4 fi
6  if alpha'=eta and and beta'=eta and flag=1 then
      eta:=eta+1, alpha'=eta, beta'=0, gamma:=gamma+1, go to 4 fi
7  if beta'<eta and flag=0 then
      beta':=beta'+1, gamma:=gamma+1, go to 4 fi
8  if alpha'<eta and flag=1 then
      alpha':=alpha'+1, gamma:=gamma+1, go to 4 fi
```

Observe that at limit times this algorithm will always cycle to command 4. The inverse functions G_0 and G_1 satisfying

$$\forall \gamma \gamma = G(G_0(\gamma), G_1(\gamma))$$

are also ordinal computable. To compute $G_0(\gamma)$ compute $G(\alpha, \beta)$ for $\alpha, \beta < \gamma$ until you find α, β with $G(\alpha, \beta) = \gamma$; then set $G_0(\gamma) = \alpha$.

4 A Recursive Truth Predicate

The Gödel pairing function G allows to code a finite sequence $\alpha_0, \ldots, \alpha_{n-1}$ of ordinals as a single ordinal

$$\alpha = G(\ldots G(G(\alpha_0, \alpha_1), \alpha_2) \ldots).$$

The usual operations on finite sequences like concatenation, cutting at a certain length, substitution, etc. are ordinal computable using the Gödel functions G, G_0, G_1. We can thus code terms and formulas of a first-order language by single ordinals in an ordinal computable way.

We introduce a language L_T suitable for structures of the form

$$(\alpha, <, G \cap \alpha^3, f)$$

where $G \cap \alpha^3$ is viewed as a ternary relation and $f : \alpha \to \alpha$ is a unary function. The language has variables $v_n = G(0, n)$ for $n < \omega$ and constant symbols $c_\xi = G(1, \xi)$ for $\xi \in \mathrm{Ord}$; the symbol c_ξ will be interpreted as the ordinal ξ. Terms are defined recursively: variables and constant symbols are terms; if t is a Term then $G(2, t)$ is a term as well which stands for $f(t)$. Atomic formulas are of the forms

- $G(3, G(t_1, t_2))$ where t_1, t_2 are terms; this stands for the equality $t_1 = t_2$;
- $G(4, G(t_1, t_2))$ where t_1, t_2 are terms; this stands for the inequality $t_1 < t_2$;
- $G(5, G(G(t_1, t_2), t_3))$ where t_1, t_2, t_3 are terms; this stands for the relation $t_3 = G(t_1, t_2)$.

L_T-Formulas are defined recursively: atomic formulas are formulas; if φ and ψ are formulas then the following are formulas as well:

- $G(6, \varphi)$; this stands for the negation $\neg\varphi$;
- $G(7, G(\varphi, \psi))$; this stands for the conjunction $(\varphi \wedge \psi)$;
- $G(8, G(v_n, \varphi))$ where v_n is a variable; this stands for the existential quantification $\exists v_n \varphi$.

Then the satisfaction relation

$$(\alpha, <, G \cap \alpha^3, f) \vDash \varphi[b]$$

for φ an L_T-formula and b an assignment of values in α can be defined as usual. If the function f is ordinal computable then this property is ordinal computable, since the recursive Tarski truth definition can be carried out by an ordinal Turing machine.

Define the truth predicate $T : \mathrm{Ord} \to \{0, 1\}$ recursively by

$$T(\alpha) = 1 \text{ iff } (\alpha, <, G \cap \alpha^3, T \upharpoonright \alpha) \vDash G_0(\alpha)[G_1(\alpha)].$$

The assignments $\alpha \mapsto T(\alpha)$ can be enumerated successively by an ordinal Turing machine. Hence T is ordinal computable. We shall see shortly that T is a strong predicate which codes a model of set theory.

5 The Theory SO of Sets of Ordinals

It is well-known that a model of Zermelo-Fraenkel set theory with the axiom of choice (ZFC) is determined by its sets of ordinals [4], Theorem 13.28. We define a natural theory SO which axiomatizes the sets of ordinals in a model of ZFC. This theory was first defined and examined in [6].

The theory SO is two-sorted: *ordinals* are taken as atomic objects, the other sort corresponds to *sets of ordinals*. Let L_{SO} be the language

$$L_{\mathrm{SO}} := \{\mathrm{Ord}, \mathrm{SOrd}, <, =, \in, g\}$$

where Ord and SOrd are unary predicate symbols, $<$, $=$ and $\in$ are binary predicatesymbols and g is a two-place function. To simplify notation, we use lower case greek letters to range over elements of Ord and lower case roman letters to range over elements of SOrd.

1. Well-ordering axiom:
   $\forall \alpha, \beta, \gamma (\neg \alpha < \alpha \wedge (\alpha < \beta \wedge \beta < \gamma \to \alpha < \gamma) \wedge$
   $(\alpha < \beta \vee \alpha = \beta \vee \beta < \alpha)) \wedge$
   $\forall a (\exists \alpha (\alpha \in a) \to \exists \alpha (\alpha \in a \wedge \forall \beta (\beta < \alpha \to \neg \beta \in a)));$

2. Axiom of infinity (existence of a limit ordinal):
   $\exists\alpha(\exists\beta(\beta < \alpha) \wedge \forall\beta(\beta < \alpha \rightarrow \exists\gamma(\beta < \gamma \wedge \gamma < \alpha)))$;
3. Axiom of extensionality: $\forall a, b(\forall\alpha(\alpha \in a \leftrightarrow \alpha \in b) \rightarrow a = b)$;
4. Initial segment axiom: $\forall\alpha\exists a\forall\beta(\beta < \alpha \leftrightarrow \beta \in a)$;
5. Boundedness axiom: $\forall a\exists\alpha\forall\beta(\beta \in a \rightarrow \beta < \alpha)$;
6. Pairing axiom (Gödel Pairing Function):
   $\forall\alpha, \beta, \gamma(g(\beta, \gamma) \leq \alpha \leftrightarrow \forall\delta, \epsilon((\delta, \epsilon) <^* (\beta, \gamma) \rightarrow g(\delta, \epsilon) < \alpha))$.
   Here $(\alpha, \beta) <^* (\gamma, \delta)$ stands for
   $\exists\eta, \theta(\eta = \max(\alpha, \beta) \wedge \theta = \max(\gamma, \delta) \wedge (\eta < \theta \vee$
   $(\eta = \theta \wedge \alpha < \gamma) \vee (\eta = \theta \wedge \alpha = \gamma \wedge \beta < \delta)))$,
   where $\gamma = \max(\alpha, \beta)$ abbreviates $(\alpha > \beta \wedge \gamma = \alpha) \vee (\alpha \leq \beta \wedge \gamma = \beta)$;
7. g is onto: $\forall\alpha\exists\beta, \gamma(\alpha = g(\beta, \gamma))$;
8. Axiom schema of separation: For all L_{SO}-formulae $\phi(\alpha, P_1, \ldots, P_n)$ postulate:
   $\forall P_1, \ldots, P_n\forall a\exists b\forall\alpha(\alpha \in b \leftrightarrow \alpha \in a \wedge \phi(\alpha, P_1, \ldots, P_n))$;
9. Axiom schema of replacement: For all L_{SO}-formulae $\phi(\alpha, \beta, P_1, \ldots, P_n)$ postulate:
   $\forall P_1, \ldots, P_n(\forall\xi, \zeta_1, \zeta_2(\phi(\xi, \zeta_1, P_1, \ldots, P_n) \wedge \phi(\xi, \zeta_2, P_1, \ldots, P_n) \rightarrow \zeta_1 = \zeta_2) \rightarrow$
   $\forall a\exists b\forall\zeta(\zeta \in b \leftrightarrow \exists\xi \in a\phi(\xi, \zeta, P_1, \ldots, P_n)))$;
10. Powerset axiom:
    $\forall a\exists b(\forall z(\exists\alpha(\alpha \in z) \wedge \forall\alpha(\alpha \in z \rightarrow \alpha \in a) \rightarrow \exists^{=1}\xi\forall\beta(\beta \in z \leftrightarrow g(\beta, \xi) \in b)))$.

6 *T* Codes a Model of SO

The truth predicate T contains information about a large class of sets of ordinals.

Definition 7. *For ordinals μ and α define*

$$T(\mu, \alpha) = \{\beta < \mu | T(G(\alpha, \beta)) = 1\}.$$

Set

$$\mathcal{S} = \{T(\mu, \alpha) | \mu, \alpha \in \mathrm{Ord}\}.$$

Theorem 8. $(\mathrm{Ord}, \mathcal{S}, <, =, \in, G)$ *is a model of the theory* SO.

Proof. The axioms (1)-(7) are obvious. The proofs of axiom schemas (8) and (9) rest on a Levy-type reflection principle. For $\theta \in \mathrm{Ord}$ define

$$\mathcal{S}_\theta = \{T(\mu, \alpha) | \mu, \alpha \in \theta\}.$$

Then for any L_{SO}-formula $\varphi(v_0, \ldots, v_{n-1})$ and $\eta \in \mathrm{Ord}$ there is some limit ordinal $\theta > \eta$ such that

$$\forall\xi_0, \ldots, \xi_{n-1} \in \theta((\mathrm{Ord}, \mathcal{S}, <, =, \in, G) \models \varphi[\xi_0, \ldots, \xi_{n-1}] \text{ iff}$$

$$(\theta, \mathcal{S}_\theta, <, =, \in, G) \models \varphi[\xi_0, \ldots, \xi_{n-1}]).$$

Since all elements of $\mathcal{S}_\theta$ can be defined from the truth function T and ordinals $< \theta$, the right-hand side can be evaluated in the structure $(\theta, <, G \cap \theta^3, T)$ by an L_T-formula φ^*which can be recursively computed from φ. Hence

$$\forall \xi_0, \ldots, \xi_{n-1} \in \theta((\mathrm{Ord}, \mathcal{S}, <, =, \in, G) \models \varphi[\xi_0, \ldots, \xi_{n-1}] \text{ iff}$$

$$(\theta, <, G \cap \theta^3, T) \models \varphi^*[\xi_0, \ldots, \xi_{n-1}]).$$

So sets witnessing axioms (8) and (9) can be defined over $(\theta, <, G \cap \theta^3, T)$ and are thus elements of $\mathcal{S}$.

The powerset axiom can be shown by a similar reflection argument.

7 Ordinal Computability Corresponds to Constructibility

Kurt Gödel [2] defined the inner model L of *constructible sets* as the union of a hierarchy of levels L_α:

$$L = \bigcup_{\alpha \in \mathrm{Ord}} L_\alpha$$

where the hierarchy is defined by: $L_0 = \emptyset$, $L_\delta = \bigcup_{\alpha<\delta} L_\alpha$ for limit ordinals δ, and $L_{\alpha+1}$ =the set of all sets which are first-order definable in the structure $(L_\alpha, \in)$. The model L is the $\subseteq$-smallest inner model of set theory. The standard reference for the theory of the model L is the monograph [1].

The following main result provides a characterization of ordinal computability which does not depend on any specific machine model or coding of language:

Theorem 9. *A set x of ordinals is ordinal computable from a finite set of ordinal parameters if and only if it is an element of the constructible universe L.*

Proof. Let $x \subseteq \mathrm{Ord}$ be ordinal computable by the program P from the finite set $\{\alpha_0, \ldots, \alpha_{k-1}\}$ of ordinal parameters: $P : \chi_{\{\alpha_0, \ldots, \alpha_{k-1}\}} \mapsto \chi_x$. By the simple nature of the computation procedure the same computation can be carried out inside the inner model L:

$$(L, \in) \models P : \chi_{\{\alpha_0, \ldots, \alpha_{k-1}\}} \mapsto \chi_x.$$

Hence $\chi_X \in L$ and $x \in L$.

Conversely consider $x \in L$. Since $(\mathrm{Ord}, \mathcal{S}, <, =, \in, G)$ is a model of the theory SO there is an inner model M of set theory such that

$$\mathcal{S} = \{z \subseteq \mathrm{Ord} \,|\, z \in M\}.$$

Since L is the $\subseteq$-smallest inner model, $L \subseteq M$. Hence $x \in M$ and $x \in \mathcal{S}$. Let $x = T(\mu, \alpha)$. By the computability of the truth predicate, x is ordinal computable from the parameters μ and α.

References

1. Keith Devlin. *Constructibility.* Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1984.
2. Kurt Gödel. *The Consistency of the Continuum Hypothesis*, volume 3 of *Ann. of Math. Studies.* Princeton University Press, Princeton, 1940.
3. Joel David Hamkins and Andy Lewis. Infinite Time Turing Machines. *J. Symbolic Logic*, 65(2):567–604, 2000.
4. Thomas Jech. *Set Theory. The Third Millennium Edition.* Springer Monographs in Mathematics. Springer-Verlag, 2003.
5. Peter Koepke. Turing Computations on Ordinals. Submitted to the *Bulletin of Symbolic Logic*. Preprint math.LO/0502264 at the e-print archive arXiv.org.
6. Peter Koepke and Martin Koerwien. The Theory of Sets of Ordinals. Preprint math.LO/0502265 at the e-print archive arXiv.org.