Ordinal Computability

Peter Koepke, University of Bonn, Germany

Mathematical Logic Seminar

Stanford, April 14, 2006

Contents

- Standard computations over $\mathbb{N}$

- Infinite Time Turing Machines

- Ordinal machines

- Ordinal registers machines (ORMs)

- The Main Theorem

- A bounded truth predicate on the ordinals

- A recursion theorem

- Ordinal stacks

- The ordinal register program for the bounded truth predicate

Computations

Turing machines and register machines work on the structure $(\mathbb{N}, +1, 0) = (\omega, +1, 0)$:

- Turing cells can be indexed by $\mathbb{N}$ ("space")
- register contents are elements of $\mathbb{N}$ ("space")
- the length of a halting computation is an element of $\mathbb{N}$ ("time")

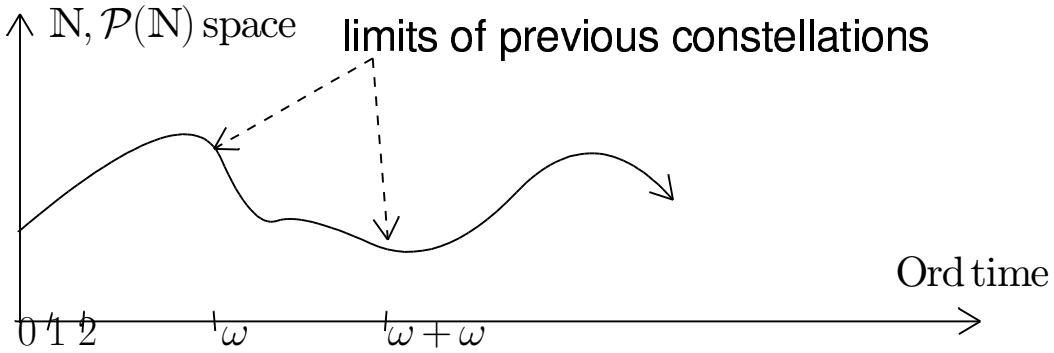
Can we replace $(\mathbb{N}, +1, 0)$ by the structure $(\mathrm{Ord}, +1, 0)$ of ordinals?

Can we replace integer time / integer space by ordinal time / ordinal space?

Infinite time Turing machines (ITTMs)

Joel D. Hamkins and Andy Lewis.
Infinite Time Turing Machines.
J. Symbolic Logic, 65(2):567-604, 2000.

Take standard Turing machines with "finite" space $\mathbb{N}$
and infinite time Ord.



Limits of ordinals

Let t be a limit ordinal and $(\alpha_i | i < t)$ be a sequence of ordinals:

- $\lim_{i<t} \alpha_i = \bigcup_{i<t} \alpha_i$ is the limit of (α_i)
- $\liminf_{i<t} \alpha_i = \lim_{i<t} \min \{\alpha_j \mid i \leqslant j < t\}$ is the inferior limit of (α_i)
- $\limsup_{i<t} \alpha_i = \min \{\lim_{i \leqslant j<t} \alpha_j | i < t\}$ is the superior limit of (α_i)

Define machine constellations at limit times
by inferior limits:

- head positions
- cell / register contents
- state ($\hat{=}$ number of command in program)

An ITTM can compute reals (= subsets of ω)

It can compute the standard halting problem

$\Delta^1_1 \subsetneq \{a \subseteq \omega | a$ is computable by an ITTM$\} \subsetneq \Delta^1_2$

Ordinal machines

space ω, time ω	space ω, time Ord	space Ord, time Ord
Turing machine	ITTM $\Delta^1_1 \subsetneq$ computable $\subsetneq \Delta^1_2$	Ordinal Turing machine OTM, computable $\leftrightarrow$ constructible
register machine	Infinite time register machine ITRM, computable $\leftrightarrow$ hyperarithmetic	Ordinal register machine ORM, computable $\leftrightarrow$ constructible

Various ordinal machines

space ω, time ω	space ω, time Ord	space Ord, time Ord
Turing machine	ITTM $\Delta^1_1 \subsetneq$ computable $\subsetneq \Delta^1_2$	Ordinal Turing machine OTM, computable $\leftrightarrow$ constructible[1)]
register machine	Infinite time register machine ITRM, computable $\leftrightarrow$ hyperarithmetic[2)]	Ordinal register machine ORM, computable $\leftrightarrow$ constructible[3)]

1) K., *Bull. Symb. Logic* 11 (2005)

2) K., to appear *Proc. CiE 2006*

3) R. Siders and K., submitted *Archive Math. Logic*

An Ordinal Register Machine (ORM)

has registers $R_0, R_1, \ldots$ which can hold *ordinal numbers*;

A register program consists of commands to increase or to reset a register;

The program may "jump" on condition of equality between two registers;

At limit times, register contents and program state are determined by $\liminf$'s.

An Ordinal Register Program

is a finite list $P = I_0, I_1, \ldots, I_{s-1}$ of *instructions* of the following types:

a) the *zero instruction* $Z(n)$ resets R_n to 0;

b) the *successor instruction* $S(n)$ increases R_n by 1;

c) the *transfer instruction* $T(m,n)$ replaces the contents of R_n by the contents of R_m;

d) the *jump instruction* $J(m,n,q)$:
if $R_m = R_n$, the ORM proceeds to the qth instruction of P,
if $R_m \neq R_n$, the ORM proceeds to the next instruction of P.

An (ORM) computation by P is a pair

$$I\colon \theta \to \omega,\ R\colon \theta \to ({}^{\omega}\mathrm{Ord})$$

such that:

- θ is an ordinal or $\theta = \mathrm{Ord}$; θ is the *length* of the computation;
- $I(0) = 0$; the machine starts in state 0;
- If $t < \theta$ and $I(t) \notin s = \{0, 1, \ldots, s-1\}$ then the machine *stops* with $\theta = t+1$;
- If $t < \theta$ and $I(t) \in s = \{0, 1, \ldots, s-1\}$ then the machine constellation $I(t+1) = I(t)+1$ and $R(t+1)$ is determined by $I_{I(t)}$;
- If $t < \theta$ is a limit ordinal let

$$\begin{aligned} \forall k \in \omega\, R_k(t) &= \liminf_{r<t} R_k(r); \\ I(t) &= \liminf_{r<t} I(r). \end{aligned}$$

About $I(t) = \liminf_{r<t} I(r)$

...

17:begin loop

...

21: begin subloop

...

29: end subloop

...

32:end loop

...

The computation by P with input $R(0)$

is determined recursively by the initial register contents $R(0)$ and the program P.

If the computation stops at $\theta = \beta + 1$ then $R(\beta)$ is the final register content which P has computed from $R(0)$.

Write $P\colon R(0) \mapsto R(\beta)(0)$.

Ordinal Computability

A function $F\colon \mathrm{Ord}^n \rightharpoonup \mathrm{Ord}$ is (ordinal) computable if there is a program P such that for $(\alpha_0, \ldots, \alpha_{n-1}) \in \mathrm{dom}(F)$ holds

$$P\colon (\alpha_0, \ldots, \alpha_{n-1}, 0, 0, \ldots) \mapsto F(\alpha_0, \ldots, \alpha_{n-1}).$$

A set $x \subseteq \mathrm{Ord}$ is computable (from ordinal parameters) if there is a program P and ordinals $\alpha_1, \ldots, \alpha_{n-1}$ such that

$$\forall \alpha \; P\colon (\alpha, \alpha_1, \ldots, \alpha_{n-1}) \mapsto \chi_x(\alpha).$$

Main Theorem.

$x \subseteq \mathrm{Ord}$ is computable iff $x \in L$, i.e., x is an element of Gödel's model of constructible sets.

Proof. ($\subseteq$) Let $x \subseteq \mathrm{Ord}$ be computable.
Take a program P and ordinals $\alpha_1, ..., \alpha_{n-1}$ such that

$$\forall \alpha \; P\colon (\alpha, \alpha_1, ..., \alpha_{n-1}, ...) \mapsto \chi_x(\alpha).$$

Computations by P are absolute between transitive models of set theory which have the same class of ordinals. Hence these computations can be carried out in the model L and χ_x , $x \in L$.

For the converse, one has to "compute L" by an ORM.

A bounded language

Let the language L_T be appropriate for first-order structures of the type

$$(\alpha, <, G, R)$$

where the Gödel pairing function G is viewed as a ternary *relation* on α and R is a unary relation on α:

- terms v_n and constants c_ξ for $\xi \in \mathrm{Ord}$; c_ξ will be interpreted as ξ;
- atomic formulas $t_1 \equiv t_2$, $t_1 < t_2$, $\dot{G}(t_1, t_2, t_3)$ and $\dot{R}(t_1)$;
- formulas $\neg\varphi$, $(\varphi \vee \psi)$, $\exists v_n < t\, \varphi$;
- assume an ordinal computable Gödelization such that for $\zeta < \xi$:

$$\varphi\frac{c_\zeta}{v_n} < (\exists v_n < c_\xi\, \varphi).$$

A bounded truth predicate on the ordinals

- Define the satisfaction relation $(\alpha, <, G, R) \vDash \varphi$ for sentences φ;
- $(\alpha, <, G, R) \vDash \varphi$ iff $(\varphi, <, G, R) \vDash \varphi$.

Define the *bounded truth predicate* $T \subseteq \mathrm{Ord}$ by

$$T(\alpha) \text{ iff } \alpha \text{ is a bounded } L_T\text{-sentence and } (\alpha, <, G, T \cap \alpha) \vDash \alpha.$$

In short

$$T(\alpha) \text{ iff } (\alpha, T \cap \alpha) \vDash \alpha.$$

T codes a model of set theory

For ordinals μ and α define "sections" of the truth predicate by

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

Set

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

Theorem. $\mathcal{S}=\{x\subseteq\mathrm{Ord}\,|\,x\in L\}$.

[Sketch for $\supseteq$: Show that $(\mathrm{Ord},\mathcal{S},<,=,\in,G)$ satisfies a natural theory of sets of ordinals; mathematics can be done in $(\mathrm{Ord},\mathcal{S},<,=,\in,G)$; define a version of Gödel's L inside $(\mathrm{Ord},\mathcal{S},<,=,\in,G)$; thus every constructible set of ordinals is an element of $\mathcal{S}$.]

A recursive definition of T

The characteristic function χ_T can be defined recursively

$$\chi_T(\alpha) = \begin{cases} 1 \text{ iff } \exists \nu < \alpha \; H(\alpha, \nu, \chi_T(\nu)) = 1 \\ 0 \text{ else} \end{cases}$$

with a computable function H:

$$\begin{aligned} H(\alpha,\nu,\chi)=1 \quad \text{iff} \quad & \alpha \text{ is an } L_T\text{-sentence and} \\ & \exists \xi,\zeta<\alpha\,(\alpha = c_\xi \equiv c_\zeta \wedge \xi=\zeta) \\ \text{or} \quad & \exists \xi,\zeta<\alpha\,(\alpha = c_\xi < c_\zeta \wedge \xi<\zeta) \\ \text{or} \quad & \exists \xi,\zeta,\eta<\alpha\,(\alpha = \dot{G}(c_\xi,c_\zeta,c_\eta) \wedge \eta = G(\xi,\zeta)) \\ \text{or} \quad & \exists \xi<\alpha\,(\alpha = \dot{R}(c_\xi) \wedge \nu=\xi \wedge \chi=1) \\ \text{or} \quad & \exists \varphi<\alpha\,(\alpha = \neg\varphi \wedge \nu=\varphi \wedge \chi=0) \\ \text{or} \quad & \exists \varphi,\psi<\alpha\,(\alpha = (\varphi\vee\psi) \wedge (\nu=\varphi \vee \nu=\psi) \wedge \chi=1) \\ \text{or} \quad & \exists n<\omega \exists \xi<\alpha \exists \varphi<\alpha\,(\alpha = \exists v_n < c_\xi\, \varphi \wedge \exists \zeta<\xi\; \nu = \varphi\tfrac{c_\zeta}{v_n} \wedge \chi=1). \end{aligned}$$

A recursion theorem

Let $H\colon \mathrm{Ord}^3 \to \mathrm{Ord}$ be ordinal computable and
define $F\colon \mathrm{Ord} \to \mathrm{Ord}$ recursively by

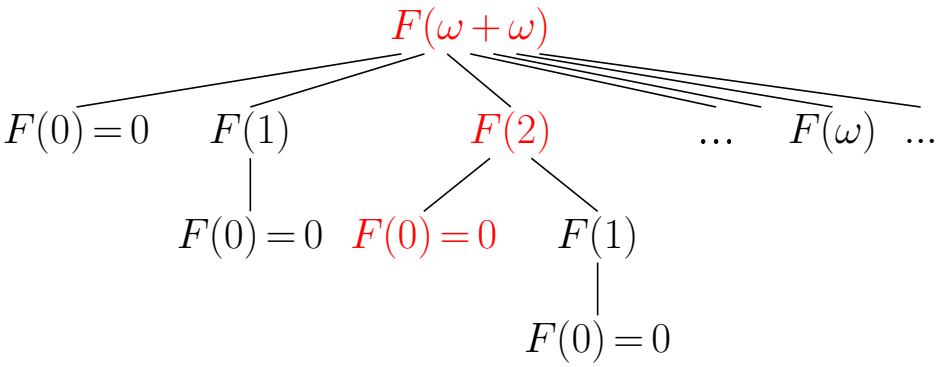
$$F(\alpha) = \begin{cases} 1 \text{ iff } \exists \nu < \alpha \; H(\alpha, \nu, F(\nu)) = 1 \\ 0 \text{ else} \end{cases}$$

Then F is ordinal computable.

The computation of F can be visualized along a well-founded tree:

Computing along a well-founded tree

$$F(\alpha) = \begin{cases} 1 \text{ iff } \exists \nu < \alpha \; H(\alpha, \nu, F(\nu)) = 1 \\ 0 \text{ else} \end{cases}$$



Stacks of ordinals

One can organize the traversal of the tree using descending stacks of ordinals.

Code $\alpha_0 > \alpha_1 > \ldots > \alpha_{n-2} \geqslant \alpha_{n-1}$ by

$$\alpha = \langle \alpha_0, \alpha_1, \ldots, \alpha_{n-2}, \alpha_{n-1} \rangle = 3^{\alpha_0} + 3^{\alpha_1} + \ldots + 3^{\alpha_{n-2}} + 3^{\alpha_{n-1}}.$$

The ordinals $\alpha_{n-1}, \alpha_{n-2}$ are ordinal computable by some programs `last`, `llast`. The functions return a special value `UNDEFINED` if the stack is too short.

The program

```
value:=2          %% set value undefined
MainLoop:
  nu:=last(stack)
  alpha:=llast(stack)
  if nu = alpha then
1:  do
    remove_last_element_of(stack)
    value:=0      %% set value equal to 0
    goto SubLoop
    end
  else
2:  do
    stack:=stack + 1   %% push 0 onto stack
    goto MainLoop
    end
SubLoop:
  nu:=last(stack)
  alpha:=llast(stack)
  if alpha = UNDEFINED then STOP
  else
    do
    if H(alpha,nu,value)=1 then
3:    do
      remove_last_element_of(stack)
      value:=1
      goto SubLoop
      end
    else
4:    do
      stack:=stack + (3**y)*2 %% push y+1
      value:=2     %% set value undefined
      goto MainLoop
      end
    end
```

The ordinal computation I, R by the program P has the following properties

a) If I, R is in state `MainLoop` at time s with `stack` contents $\langle \alpha_0, ..., \alpha_{n-1} \rangle$ where $n \geqslant 1$ then I, R will get into state `SubLoop` at a later time t with the same `stack` contents $\langle \alpha_0, ..., \alpha_{n-1} \rangle$ and the register `value` holding the value $F(\alpha_{n-1})$. Moreover in the interval $[s, t)$ the contents of `stack` will always be at least as big as $\langle \alpha_0, ..., \alpha_{n-1} \rangle$.

b) Let I, R be in state `MainLoop` at time s with `stack` contents $\alpha_0 > ... > \alpha_{n-1}$ where $n \geqslant 1$. Define $\bar{\alpha} =$ the minimal ordinal $\nu < \alpha_{n-1}$ such that $H(\alpha_{n-1}, \nu, F(\nu)) = 1$ if this exists and $\bar{\alpha} = \alpha_{n-1}$ else. Then there is a strictly increasing sequence $(t_i | i \leqslant \bar{\alpha})$ of times $t_i > t$ such that I, R is in state `MainLoop` at time t_i with `stack` contents $\langle \alpha_0, ..., \alpha_{n-1}, i \rangle$. Moreover in every time interval $[t_i, t_{i+1})$ the `stack` contents are $\geqslant \langle \alpha_0, ..., \alpha_{n-1}, i \rangle$.

c) If I, R is in state `MainLoop` with `stack` contents $\langle \alpha \rangle$ then it will later *stop* with `stack` contents $\langle \alpha \rangle$ and the register `value` holding the value $F(\alpha)$. Hence the function F is ordinal register computable.

Main Theorem.

$x \subseteq \mathrm{Ord}$ is computable iff $x \in L$, i.e., x is an element of Gödel's model of constructible sets.

Proof. ($\supseteq$) Let $x \subseteq \mathrm{Ord}$ be constructable.
By a previous theorem

$$x = X(\mu, \alpha) = \{\beta < \mu \,|\, T(G(\alpha, \beta))\}$$

for some $\mu, \alpha \in \mathrm{Ord}$. x is ordinal computable, since T is ordinal computable.

QED

Further aspects

- transfer notions along the correspondance
  ordinal computability - constructibility

- ordinalize other notions of computability