CDMTCS
Research
Report
Series

On a Theorem of Solovay

Cristian S. Calude and
Richard Coles
Department of Computer Science
University of Auckland
Auckland, New Zealand

CDMTCS-094
February 1999

Centre for Discrete Mathematics and
Theoretical Computer Science

On a Theorem of Solovay

Cristian S. Calude and Richard Coles
Department of Computer Science
University of Auckland
Private Bag 92019, Auckland
New Zealand
Emails: {cristian,coles}@cs.auckland.ac.nz

Abstract

We present a proof of the following result due to Solovay: There exists a noncomputable Δ_2^0 real x such that $H(x_n) \leqslant H(n) + O(1)$.

1 Introduction

Solovay [10] introduced the domination relation which plays an important role in defining the so-called Ω–like reals. The class of Ω–like reals coincides with the class of Chaitin Ω reals (halting probabilities of universal self-delimiting Turing machines, see for example [5, 6, 7, 2, 9]) (cf. [4]) and the class of c.e. random reals (cf. Slaman [11]). Solovay proved that if x and y are two c.e. reals and y dominates x, then $H(x_n) \leqslant H(y_n) + O(1)$. In Calude and Coles [3] we prove that the converse implication is false, namely there are c.e. reals x and y such that $H(x_n) \leqslant H(y_n) + O(1)$ and y does not dominate x. We do this by constructing a noncomputable c.e. real x such that $H(x_n) \leqslant H(n) + O(1)$. The proof is based upon techniques of Solovay [10] where he proved the following theorem:

Theorem 1. *There is a noncomputable Δ_2^0 real x such that*

$$H(x_n) \leqslant H(n) + O(1).$$

In this report we present a write up of Solovay's proof of Theorem 1. The original handwritten proof can be found in Solovay [10]. In Section 2 we introduce our notation and basic concepts. Sections 3, 4 and 5 deal with constructing the Δ_2^0 real mentioned above.

2 Preliminaries

Suppose $a, b \in \{0,1\}^*$, the set of binary strings. The concatenation of a, b is denoted by $a^\frown b$. Let $|a|$ denote the length of a. For $j \geqslant 0$, we write $a(j) = k$ iff the jth bit of a is k. We let $\prec$ denote the quasi-lexicographical ordering of finite binary strings. We write $string(n)$ to denote the nth string with respect to $\prec$. By $\min_\prec$ we denote the minimum operation taken according to $\prec$.

We fix a computable bijective function $\langle \cdot, \cdot \rangle$ from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$. We write $\log n$ to denote $\log_2 n$. For a function $f : \{0,1\}^* \to \mathbb{N}$, we define $O(f) = \{g : \{0,1\}^* \to \mathbb{N} \mid \exists c \in \mathbb{N} \forall a \in \{0,1\}^* (g(a) \leqslant c \cdot f(a))\}$. Suppose $h_0, h_1 : \{0,1\}^* \to \mathbb{N}$. We write $O(1)$ for $O(f)$ when f is the constant function $f(a) = 1$ for all $a \in \{0,1\}^*$. We write $h_0 \leqslant h_1 + O(f)$ if there is a function $g \in O(f)$ such that $h_0(a) \leqslant h_1(a) + g(a)$ for all $a \in \{0,1\}^*$.

We will look at real numbers in the interval $[0,1]$ through their binary expansions, i.e., in terms of functions $n \mapsto x_n$ (from $\mathbb{N}$ into $\{0,1\}$). We write $(x_n)_n$ for the sequence $n \mapsto x_n$. A real is computable if the function $n \mapsto x_n$ is computable. A real $x = (x_n)_n$ is computable enumerable (c.e.) if it is the limit of a computable, increasing, converging sequence of rationals. Equivalently, $\alpha = 0.\chi_A$ is a c.e. real if A has a computable approximation $\{A[s]\}_{s \geqslant 0}$ such that whenever $i \in A[s]$ and $i \notin A[s+1]$, then there is some $j < i$ such that $j \notin A[s]$ and $j \in A[s+1]$; χ_A is the characteristic function of A.

Let ϕ_e be a standard list of all partial computable functions from $\mathbb{N}$ into $\mathbb{N}$. In case $\phi_e(x)$ halts (and produces y) we write $\phi_e(x)\downarrow$ ($\phi_e(x)\downarrow = y$); otherwise, $\phi_e(x)\uparrow$. By $\phi_e(x)[t]$ we denote the time relativised version of $\phi_e(x)$, i.e., $\phi_e(x)[t] = \phi_e(x)$ in case $\phi_e(x)$ halts in time t. The binary predicate $\phi_e(x)[t]\downarrow$ is primitive recursive. We will adopt the following *convention*: if $\phi_e(x)[t]\downarrow$, then $\phi_e(x) \leqslant t$.[1] By $\mathrm{dom}\,\phi_e$ we denote the domain of the partial function ϕ_e.

A self-delimiting computer is a partial computable function C from $\{0,1\}^* \times \{0,1\}^*$ with values in $\{0,1\}^*$ such that for every $y \in \{0,1\}^*$ the set $\{x \mid C(x,y)\downarrow\}$ is prefix-free. Here C stands for the interpreter, x for the program, and y for the input data.

The Invariance Theorem ([5, 7, 2, 10]) states the existence of a universal computer U with the property that for every computer C there is a constant d (depending upon U and C) such that if $C(x,y) = z$, then $U(x', y) = z$, for some program x' with the length $|x'| \leqslant |x| + d$. Note that U does not need more than a primitive recursive extra time to simulate C, i.e., there is a primitive recursive function h such that if $C(x,y)[t]\downarrow = z$, then $U(x',y)[h(t)]\downarrow = z$. In what follows we will fix a universal computer U. Let a^* be the quasi-lexicographical least p such that $U(p, \lambda) = a$.

[1] Sometimes we will be concerned with constructions occuring over ω-many stages and thus often append $[t]$ to parameters to denote the value of the parameter at the end of stage t.

Define the following program-size complexities ([6]):

$$H(a) = \min\{|p| \mid U(p, \lambda) = a\},\ H(a, b) = H(\langle a, b\rangle),$$

$$H(a/b) = \min\{|p| \mid U(p, b^*) = a\},\ \tilde{H}(a/b) = \min\{|p| \mid U(p, b) = a\}.$$

Note that $\tilde{H}(a/b^*) = H(a/b)$.

In expressions relating to strings, such as $U(p, \lambda) = n$, we are identifying n with the binary string 1^n of length n. We also write $H(n)$ for $H(1^n)$. In fact notice that $|H(n) - H(string(n))| = O(1)$. For integers n, we write n^* for $\min_{\prec}\{p \mid U(p, \lambda) = n\}$.

We continue by defining some useful functions. For all $j, n, t \in \mathbb{N}$, define

$$H(n)[t] = \min\{|p| \mid U(p, \lambda)[t] = n \,\&\, |p| \leqslant t\},$$

if such a p exists, $H(n)[t] = 0$, otherwise. Further let,

$$\alpha(n)[t] = \min\{H(j)[t] \mid j \geqslant n\}, \alpha(n) = \min\{H(j) \mid j \geqslant n\}.$$

It is seen that $H(n)[t]$ and $\alpha(n)[t]$ are primitive recursive functions (reason: for the computation we only need the set $\{p \mid U(p, \lambda)[t], |p| \leqslant t\}$), decreasing in t, and $H(n) = \lim_t H(n)[t]$, $\alpha(n) = \lim_t \alpha(n)[t]$.

Assume that D is an oracle and consider the relativised computation U^D. Then the relativised program-size complexities are defined in the obvious way, for example,

$$H^D(a) = \min\{|p| | U^D(p, \lambda) = a\}.$$

If instead of a self-delimiting universal computer we work with a universal partial computable function V, then the induced complexities will be denoted by $K(a), K(a/b), \tilde{K}(a/b)$. We now summarise the known results relating the complexity of initial segments to the computability of a real. Let $x = (x_n)_n$ be a binary sequence.

Theorem 2 (Loveland [8]). *x is computable iff $\tilde{K}(x_n/n) = O(1)$.*

Corollary 3. *x is computable iff $\tilde{H}(x_n/n) = O(1)$.*

Theorem 4 (Chaitin [6]). *x is computable iff $K(x_n) \leqslant K(n) + O(1)$.*

Theorem 5 (Chaitin [6]). *If $H(x_n) \leqslant H(n) + O(1)$, then $x \in \Delta^0_2$.*

Proof. Start by noting that

$$H(x_n) = H(n, x_n) + O(1) = H(x_n/n) + H(n) + O(1).$$

Now if x satisfies the hypothesis of the theorem, then we have

$$H(x_n/n) + H(n) + O(1) \leqslant H(n) + O(1).$$

Therefore $H(x_n/n) = O(1)$, and so $\tilde{H}(x_n/n^*) = O(1)$.

Relativising to the oracle $D = \{p \mid U(p,\lambda)\downarrow\}$, we have $\tilde{H}^D(n^*/n) = O(1)$, since the mapping $n \mapsto n^*$ is Turing reducible to D. Consequently,

$$\tilde{H}^D(x_n/n) \leqslant \tilde{H}^D(x_n/n^*) + \tilde{H}^D(n^*/n) = O(1),$$

therefore $\tilde{H}^D(x_n/n) \leqslant O(1)$. So by the relativised version of Corollary 3 we see that x is Δ^0_2. □

3 Solovay-Universal Functions

We start by introducing the notion of a Solovay-universal function.

Definition 6. A partial computable function f from $\{0,1\}^*$ into $\{0,1\}^*$ is *Solovay-universal* if it satisfies the following three conditions:

(1) $\mathrm{dom}(f) = \mathrm{dom}(U,\lambda)$,

(2) $a \in \mathrm{dom}(U) \implies |f(a)| = |U(a,\lambda)|$,

(3) for all $m,n \in \mathbb{N}$, $m < n \implies f(m^*) \preccurlyeq f(n^*)$.

We will later prove that Solovay-universal functions do exist. The following proposition reveals the motivation for the definition.

Proposition 7. *Suppose f is a Solovay-universal function. Then the real $x_n = f(n^*)$ satisfies $H(x_n) \leqslant H(n) + O(1)$.*

Proof. We have

$$\begin{aligned} H(x_n) &\leqslant H(x_n/n) + H(n) + O(1) \\ &= \tilde{H}(x_n/n^*) + H(n) + O(1). \end{aligned}$$

Since f is a partial computable function, $\tilde{H}(x_n/n^*) = O(1)$ and therefore $H(x_n) \leqslant H(n) + O(1)$ as required. □

Hence, if we can construct a noncomputable real $x = \lim_n x_n$, where $x_n = f(n^*)$ for some Solovay-universal f, then we have proved Theorem 1.

We now show the existence of a Solovay-universal functions. First we need some definitions. Recall

$$H(n)[t] = \min\{|p| | U(p,\lambda)[t] = n \,\&\, |p| \leqslant t\}.$$

Let $(t_i)_i$ be a computable increasing sequence of natural numbers. Define the total computable function σ as follows:

$$\sigma(i) = \max\{j \leqslant i \mid H(j)[t_i] = H(j)[t_i + 1]\}.$$

Notice that the graph of $\sigma(i)$ is primitive recursive and that $\sigma(i) \leqslant i$.

Let $(t_i)_i$ be a computable sequence of times such that $i < t_i < t_{i+1}$ and a c.e. set $\{p_i\}$ of programs such that $U(p_i, \lambda)[t_i] = i$. For $i \in \mathbb{N}$ define

$$A_i = \{p \mid U(p, \lambda)[t_i]\downarrow \& |U(p, \lambda)| \leqslant i\}.$$

Notice that for every $i \in \mathbb{N}$, A_i is computable, $A_i \subset A_{i+1}$. For every $n \in \mathbb{N}$, $n^* \in A_{i+1} \setminus A_i$ for some i.

The following statement is immediate from the definitions of $H(n)[t]$, σ, and A_i.

Proposition 8. *If $n^* \in A_{i+1} \setminus A_i$ then $n^* \in A_j$ for all $j \geqslant i+1$, and consequently $H(n)[t] = |n^*|$ and $\sigma(i') \geqslant n$ for all $i' \geqslant i+1$ and $t \geqslant t_{i+1}$.*

Now suppose $(z[i])_{i\in\mathbb{N}}$ is a computable sequence of binary strings such that $|z[i]| \geqslant i$, and

$$\forall i(j < \sigma(i) \implies z[i](j) = z[i+1](j)). \qquad (\dagger)$$

Recall that $z[i](j)$ is the jth bit of $z[i]$. The existence of such a sequence of strings will be proven in Theorem 14.

We now define a partial function F as follows: If $p \in A_{i+1} \setminus A_i$, then let $F(p)$ be the initial segment of $z[i+1]$ of length $|U(p, \lambda)|$.

Proposition 9. *F is Solovay-universal.*

Proof. Since $\{z[i]\}_{i\in\mathbb{N}}$ is a computable sequence of binary strings and each A_i is computable, then F is a partial computable function. If $p \in \operatorname{dom} U$, then $p \in A_{i+1}$ for some minimal i and hence $\operatorname{dom} F = \operatorname{dom} U$. By definition of F, $p \in \operatorname{dom} U$ implies that $|F(p)| = |U(p, \lambda)|$. It remains to show that condition (3) of Definition 6 is satisfied.

Suppose $n^* \in A_{i_0+1} \setminus A_{i_0}$. Then $H(n)[t] = |n^*|$ and $\sigma(i) \geqslant n$ for all $i \geqslant i_0 + 1$ and for all $t \geqslant t_{i_0+1}$ by Proposition 8. Then by ($\dagger$), $z[k](j) = z[i_0+1](j)$ for all $k \geqslant i_0 + 1$ and $j \leqslant \sigma(i_0+1)$. Thus for $j < n$, $\lim_i z[i](j) = z[i_0+1](j)$. Define $z(j) = \lim_i z[i](j)$. Now $F(n^*)$ is defined to be the initial segment of z of length n and hence condition (3) of Definition 6 is satisfied. Therefore F is a Solovay-universal. □

4 Two Useful Results

Before proving Theorem 1 we need two useful theorems, namely Theorems 10 and 11 below. Let B be a total increasing computable function which is not primitive recursive, but has a primitive recursive graph. For example take B to be Ackermann's diagonal function. Then in fact, B dominates any primitive recursive function, that is, for every primitive recursive function g, $B(n) > g(n)$, for almost all natural n; see [1].

Theorem 10. *There is a computable sequence of times* $(t_n)_n$*, such that* $t_n < t_{n+1}$ *for all* $n \in \mathbb{N}$*, and for all* $n > 0$*,*

(1) there is a natural number s_n *such that* $B(t_{n-1}) \leqslant s_n < B(s_n) = t_n$*,*

(2) for all programs p *with* $|p| \leqslant B(t_{n-1})$*,* $U(p, \lambda)\downarrow$ *implies that either* $U(p, \lambda)[s_n]\downarrow$ *or* $U(p, \lambda)[B(s_n)]\uparrow$*,*

(3) the predicate $t_n = j$ *is primitive recursive, but the function* $n \mapsto t_n$ *is not primitive recursive.*

Proof. Let $t_0 = 0$ and assume that $s_1, \dots, s_{n-1}, t_1, \dots, t_{n-1}$ have been defined. Let Q be the set of all programs of length less than or equal to $B(t_{n-1})$. Let $r_1 = t_{n-1}$ and define the following three sets: $X_1 = \{p \in Q \mid U(p, \lambda)[r_1]\downarrow\}, Y_1 = Q \setminus X_1, Z_1 = \{p \in Y_1 \mid U(p, \lambda)[B(r_1)]\downarrow\}$.

If $Z_1 = \emptyset$, then we let $s_n = r_1$ and $t_n = B(s_n)$. Otherwise we have $Z_1 \neq \emptyset$, in which case set $r_2 = B(r_1)$ and define $X_2 = X_1 \cup Z_1, Y_2 = Q \setminus X_2, Z_2 = \{p \in Y_2 \mid U(p, \lambda)[B(r_2)]\downarrow\}$.

If $Z_2 = \emptyset$, then let $s_n = r_2$ and $t_n = B(s_n)$. Otherwise continue this process until reaching a step i with $Z_i = \emptyset$ and hence let $s_n = r_i$ and $t_n = B(s_n)$. Such an i must be reached since $X_1 \subset X_2 \subset \ldots \subseteq Q \supseteq Y_1 \supset Y_2 \supset \ldots$, Q is finite and $Y_i \supset Z_i$.

It follows that (1) and (2) are satisfied. Part (3) follows from properties of Ackermann's diagonal function B. □

We use the sequence of times $(t_n)_n$ constructed above in the following theorem, which plays a crucial role in the priority argument in Section 5. In particular, the function σ we work with is defined with respect to this sequence of times.

Theorem 11. *There is a computable sequence of times* $(t_i)_{i \geqslant 0}$*,* $t_i < t_{i+1}$ *and a c.e. set* $\{p_i\}_{i \geqslant 0}$ *of programs such that* $U(p_i, \lambda)[t_i] = i$*, for which the following condition holds true: if* $g : \mathbb{N} \to \mathbb{N}$ *is a total computable function, then for infinitely many natural numbers* i*,* $g(\sigma(i)) < i$*.*

Proof. Take the sequence of times $(t_i)_i$ constructed in Theorem 10. Suppose for a contradiction that $g(\sigma(n)) \geqslant n$ for almost all n. We may assume that g is increasing, and in fact that it dominates all primitive recursive functions.

Let $G(n) = \min\{m \mid g(m+1) \geqslant n\}$. Then our assumption on g implies that $G(n) \leqslant \sigma(n)$ for almost all n. Note that $G(n) \leqslant t_n$ under this assumption.

The intuition for obtaining the contradiction is as follows. The function $\sigma(i)$, roughly speaking, is the largest j such that the approximation to $H(j)$ does not change between time steps t_i and t_{i+1}. Therefore to show $G(n) > \sigma(n)$ we would like to construct short programs (of length $\leqslant B(t_n)$) for $j \in [G(n), n]$ which halt by time t_{n+1}. We will be able to do this if

(i) n has a short program that converges in approximately time t_n,

(ii) $\sum_{G(n)\leqslant j\leqslant n} 2^{-H(j)[t_n]}$ is small,

because then we can use the Kraft-Chaitin Theorem to construct new codes (programs) for all $j \in [G(n), n]$. The next two lemmata perform this task.

Lemma 12. *There is a primitive recursive function h such that for almost all k, there is a number n_k such that $\alpha(G(n_k))[t_{n_k}] \geqslant k$, and a program p_{n_k} such that $U(p_{n_k}, \lambda)[h(t_{n_k})]\downarrow = n_k$, and $|p_{n_k}| = O(\log k)$.*

Proof. Fix $k \in \mathbb{N}$. If n is sufficiently large then $\alpha(G(n)) \geqslant k$ because g is unbounded, and so $\alpha(G(n))[t_n] \geqslant k$. Let

$$n_k = \min\{m \mid m > k \text{ and } \alpha(G(m))[t_m] \geqslant k\}.$$

For large n,

$$G(n) = \min\{m \mid m \leqslant \sigma(n) \text{ and } g(m+1) \geqslant n\},$$

so G is primitive recursive. Further, the predicate, $\alpha(n)[t] \geqslant k$ is also primitive recursive as it can be expressed by the formula

$$(\forall n \leqslant j \leqslant t, |p| \leqslant t(U(p,\lambda)[t] = j \implies |p| \geqslant k)) \text{ or } (n > t).$$

So the function $k \mapsto n_k$ has a primitive recursive graph: $U(p,\lambda)[t_m] = j$ iff $U(p,\lambda)[s] = j$, for some $s \leqslant t_m$. Let $h_1(k,s)$ be the primitive recursive function evaluating the running time of the predicate $n_k = s$.

Now consider the self-delimiting computer $C(1^{\log k}0, \lambda) = n_k$. Use the Invariance Theorem to show that for every natural k there is a program, p_{n_k} say, such that $|p_{n_k}| = O(\log k)$ such that $U(p_{n_k}, \lambda) = n_k$. The time necessary to compute p_{n_k} from k is the sum between the time to get n_k from k (a primitive recursive function in n_k, $h_1(k, n_k)$) and the time to simulate C on U, which is primitive recursive in the running time of C, i.e., $h(h_1(k, n_k))$. □

Lemma 13. *For sufficiently large k,*

$$\sum_{G(n_k)\leqslant j\leqslant n_k} 2^{-H(j)[t_{n_k}]} \leqslant 2^{-k/2}.$$

Proof. Again, let k be sufficiently large so that $\alpha(G(n_k)) \geqslant k$. Suppose for a contradiction that

$$\sum_{G(n_k)\leqslant j\leqslant n_k} 2^{-H(j)[t_{n_k}]} > 2^{-k/2}.$$

The number $\sum_{G(n_k)\leqslant j\leqslant n_k} 2^{-H(j)[t_{n_k}]}$ can be computed in a primitive recursive way and is less than 1. Computing its most significant $k/2$ digits we get a $j \in [G(n_k), n_k]$ and a program p_j such that $|p_j| \leqslant k/2 + O(1)$, $U(p_j, \lambda) = j$, and $U(p_j, \lambda)[z]\downarrow$ for some z. Consequently, $H(j) \leqslant k/2 + O(1)$, so

$$\alpha(G(n_k)) = \min\{H(j) \mid j \geqslant G(n_k)\} \leqslant k/2 + O(1).$$

For sufficiently large k, this contradicts $\alpha(G(n_k)) \geqslant k$, hence the required inequality has been demonstrated. □

We now continue with the proof of Theorem 11. Consider the following self-delimiting computer:

(1) input $string(k)^\frown y$,

(2) compute n_k using the procedure in Lemma 12,

(3) compute t_{n_k} and $G(n_k)$,

(4) now use the Kraft-Chaitin Theorem[2] to construct a prefix-free set $E = \{e_j \mid j \in [G(n_k), n_k]\}$ such that $|e_j| \leqslant H(j)[t_{n_k}] - k/2$, which is possible by Lemma 13,

(5) output the code e_y of y if $y \in [G(n_k), n_k]$.

The time of this procedure is $h_0(t_{n_k})$ for some primitive recursive function h_0, and so $t_{n_k+1} > B(t_{n_k}) \geqslant h_0(t_{n_k})$. Hence for $j \in [G(n_k), n_k]$,

$$H(j)[t_{n_k+1}] \leqslant H(j)[t_{n_k}] - k/2 + O(\log k).$$

The $O(\log k)$ term is derived from the length of the program in Lemma 12 for computing n_k from k. Hence for sufficiently large k, $\sigma(n_k) \leqslant G(n_k)$, contradicting our assumption that $G(n) < \sigma(n)$ for almost all n. $\square$

5 The Priority Argument

Having established Theorem 11, we are now in a position to complete the proof of Theorem 1.

Theorem 14. *There is a noncomputable real x satisfying*

$$H(x_n) \leqslant H(n) + O(1).$$

Proof. We construct a computable sequence of binary strings $(z[i])_{i\in\mathbb{N}}$ such that $z = \lim_i z[i]$ exists. Furthermore we construct $(z[i])_{i\in\mathbb{N}}$ to satisfy

$$\forall s(j < \sigma(s+1) \implies z[s](j) = z[s+1](j)). \qquad (\dagger)$$

Then as in Section 3 we define $F(p)$ to be the initial segment of $z[i+1]$ of length $|U(p,\lambda)|$ if $p \in A_{i+1} \setminus A_i$. Hence F is Solovay-universal.

Consequently, defining $x = \lim_n x_n$, where $x_n = F(n^*)$ provides the necessary x satisfying, for all $n \in \mathbb{N}$,

$$H(x_n) \leqslant H(n) + O(1).$$

[2] Given a recursive list of "requirements" $\langle n_i, s_i\rangle$ ($i \geq 0, s_i \in \Sigma^*, n_i \geqslant 0$) such that $\sum_i 2^{-n_i} \leq 1$, we can effectively construct a self-delimiting computer C and a recursive one-to-one enumeration $x_0, x_1, x_2, \ldots$ of words x_i of length n_i such that $C(x_i,\lambda) = s_i$ for all i and $C(x,\lambda) = \infty$ if $x \notin \{x_i \mid i \geqslant 0\}$; see [2].

Notice that $x = z$.

We construct z to meet the following requirements for all $e \in \mathbb{N}$:

$\mathrm{R}_e : \exists m\, (\phi_e(m) \neq z(m))$.

The classical diagonalisation strategy for this requirement is to choose some m at stage s with $z(m)[s] = 0$ and wait for $\phi_e(m)\downarrow[t] = 0$. When this occurs we set $z(m) = 1$. If $\phi_e(m)\uparrow$ or $\phi_e(m)\downarrow\neq 0$ then $z(m) = 0 \neq \phi_e(m)$. If $\phi_e(m)\downarrow= 0$ at some stage t then of course $\phi_e(m) = 0 \neq 1 = z(m)$.

We modify this strategy so that we construct z to meet (†). We first address the issue of meeting (†) for one R_e requirement.

To satisfy (†) we must ensure $z(j)[s] = z(j)[s + 1]$ for all $j < \sigma(s + 1)$. In Proposition 8 we saw that if $n^* \in A_{i+1} \setminus A_i$ then $n^* \in A_j$ for all $j \geqslant t_{i+1}$ and consequently $H(n)[t] = |n^*|$ and $\sigma(i') \geqslant n$ for all $i' \geqslant i + 1$ and $t \geqslant t_{i+1}$. This means that when $n^* \in A_{i+1} \setminus A_i$, then we may no longer change our approximation to z on the initial segment of length n. That is $z(j)[s] = z(j)$ for all $j < n$ and $s \geqslant i + 1$. However, H is not computable and so we cannot know when an element in A_i is n^* for some n.

Now suppose we assign $\sigma(m)$ to be a witness for R_e. We do not know if $\phi_e(\sigma(m))$ converges, and if so, at which stage it converges. If $\phi_e(\sigma(m)) = 0$ and converges at stage t and yet $n^* \in A_s$ for some $s < t$, then we cannot define $z(\sigma(m))[t] = 1$ to diagonalise against $\phi_e(\sigma(m))$ without compromising condition (†). The solution is to have witnesses that can be used before condition (†) prevents it. The existence of such witnesses is provided by Theorem 11. If ϕ_e is a total computable function then let $g : \mathbb{N} \mapsto \mathbb{N}$ be the total computable function such that $g(i)$ is the number of steps it takes for $\phi_e(i)$ to converge. Then Theorem 11 tells us that there are infinitely many $\sigma(i)$ such that $\phi_e(\sigma(i))$ converges in less than i steps.

Thus at stage $s+1$ of the construction if R_e is currently not satisfied and $\phi_e(\sigma(s+1))\downarrow= 0$ then we define $z(\sigma(s+1)) = 1$ and $z[s+1](j) = z[s](j)$ for all $j < \sigma(s + 1)$, thus satisfying (†). If ϕ_e is total then Theorem 11 provides many such $\sigma(s + 1)$ as discussed above.

We now consider how to meet (†) in the presence of more than one requirement. The problem is that σ is not a 1-1 function. That is $\sigma(s_1)$ may equal $\sigma(s_2)$ for $s_1 \neq s_2$. Therefore higher priority requirements are allowed to change the value of $z(\sigma(i))[s+1]$ even if it has been defined to be something different by a lower priority requirement. The priority ordering ensures that for every j, $z[s + 1](j) \neq z[s](j)$ for only finitely many s. Hence z will be Δ^0_2.

For example, suppose R_1 is of higher priority than R_2. At stage $s_0 + 1$ suppose $\phi_2(\sigma(s_0+1))[s_0+1]\downarrow= 0$ and we defined $z[s_0+1](\sigma(s_0+1)) = 1$. Then at some later stage, s_1 say, $\phi_1(\sigma(s_1 + 1))\downarrow= 1$ where $\sigma(s_1 + 1) = \sigma(s_0 + 1)$. Then R_1 wants to define $z[s_1 + 1](\sigma(s_1 + 1)) = 0$. This can be done and still meet (†), but results in a Δ^0_2 real z being constructed. R_1 must be allowed to take $\sigma(s_1 + 1)$ as a witness otherwise we run the risk of more and more lower priority requirements from depriving R_1 of ever having a usable witness.

The Construction.

Stage 0 : $z[0] = \lambda$, the empty string.
Stage s + 1 :
We say that R_e requires attention at stage $s+1$ if

(i) $e < s+1$,

(ii) $\phi_e(\sigma(s+1))$ converges in at most $s+1$ steps,

(iii) $\sigma(s+1)$ is not a witness for any $\mathrm{R}_{e'}$ for $e' < e$,

(iv) R_e has no witness at stage $s+1$ and $\sigma(s+1)$ was not previously a witness of e.

We define $z[s+1]$ to be a string of length $s+1$ as follows.

- If no R_e requires attention at stage $s+1$ then let $z[s+1]$ be the extension of $z[s]$ such that $z[s+1](s) = 0$.
- Otherwise let e_0 be the least e such that R_e requires attention at stage $s+1$. We say R_{e_0} receives attention at stage $s+1$.

  Define $z[s+1](\sigma(s+1)) \neq \phi_{e_0}(\sigma(s+1))$, and make $\sigma(s+1)$ a witness for R_{e_0}. Define $z[s+1](n) = z[s](n)$ for all $n < \sigma(s+1)$. Define $z[s+1](n) = 0$ for all n such that $\sigma(n) < n < s+1$.

  Cancel all witnesses assigned to $\mathrm{R}_{e'}$ for $e' > e$.

The Verification.
It is clear that $(z[i])_{i\geqslant 0}$ is a computable sequence and that $z[i](k) = z[i+1](k)$ if $k < \sigma(i+1)$, thus satisfying (†). We now define F as in Section 3, that is if $y \in A_{i+1} \setminus A_i$, then $F(y)$ is the initial segment of $z[i+1]$ of length $|U(y,\lambda)|$. We let $x_n = F(n^*)$ and $x = \lim_n x_n$. It follows that F is Solovay-universal and so x satisfies $H(x_n) \leqslant H(n) + O(1)$.
It remains to show that x is a noncomputable real.

Proposition 15. *For all* $e \in \mathbb{N}$, R_e *has at most a finite number of witnesses.*

Proof. An easy induction on e shows that R_e only requires attention finitely often. □

Proposition 16. *If* ϕ_e *is total then* R_e *has a final witness.*

Proof. Suppose R_e does not get a final witness and all $\mathrm{R}_{e'}$ for $e' < e$ have a final witness by stage s_0 and never require attention after stage s_0. Then it is clear that clause (i) holds for all $s \geqslant s_0$. Furthermore we may choose $s_1 \geqslant s_0$ so that (iii) holds for R_e for all $s \geqslant s_1$ since $\lim_s \sigma(s) = \infty$. Let $g(n)$ be the number of steps it takes for $\phi_e(n)$ to converge. Then clause (ii) holds infinitely often since by Theorem 11 there are infinitely many i such that $g(\sigma(i)) < i$. Thus clause (iv) must fail, otherwise R_e would receive attention at some stage $s_2 \geqslant s_1$. But then the witness that R_e has that contradicts (iv) is permanent. □

Proposition 17. *If m is the final witness of* R_e, *then* $x(m) = z(m) \neq \phi_e(m)$.

Proof. By construction, if m is the final witness for R_e then $x(m) \neq \phi_e(m)$. □

This completes the proof of Theorem 14. □

References

[1] C. Calude. *Theories of Computational Complexity*, North-Holland, Amsterdam, 1988.

[2] C. Calude. *Information and Randomness. An Algorithmic Perspective,* Springer-Verlag, Berlin, 1994.

[3] C. S. Calude, R. J. Coles. Program-size complexity of initial segments and domination relation reducibility, in J. Karhumäki, H. A. Maurer, G. Păun, G. Rozenberg (eds.). *Jewels Are Forever*, Springer-Verlag, Berlin, 1999. (in press)

[4] C. S. Calude, P. Hertling, B. Khoussainov, Y. Wang. Recursively enumerable reals and Chaitin Ω numbers, in M. Morvan, C. Meinel, D. Krob (eds.). *Proceedings of STACS'98, Paris,1998*, Springer-Verlag, Berlin, 1998, 596-606.

[5] G. J. Chaitin. Algorithmic information theory, *IBM J. Res. Develop.* 21(1977), 350-359, 496.

[6] G. J. Chaitin. Information-theoretic characterizations of recursive infinite strings, *Theoretical Computer Science,* 2 (1976), 45–48.

[7] G. J. Chaitin. *The Limits of Mathematics*, Springer-Verlag, Singapore, 1997.

[8] D. W. Loveland. A variant of the Kolmogorov concept of complexity, *Information and Control,* 15 (1969), 510–526.

[9] G. Rozenberg, A. Salomaa. *Cornerstones of Undecidability*, Prentice Hall, Englewood Cliffs, 1994.

[10] R. M. Solovay. *Draft of a paper (or series of papers) on Chaitin's work ... done for the most part during the period of Sept. to Dec. 1974*, unpublished manuscript, IBM Thomas J. Watson Research Center, Yorktown Heights, New York, May 1975, 215 pp.

[11] T. A. Slaman. *Random Implies Ω-Like,* manuscript, 14 December 1998, 2 pp.