ELSEVIER

Theoretical Computer Science 284 (2002) 539–555

Theoretical Computer Science

www.elsevier.com/locate/tcs

Presentations of computably enumerable reals

Rodney G. Downey[a, *], Geoffrey L. LaForte[b]

[a] *Department of Mathematics, Victoria University of Wellington, P.O. Box 600, New Zealand*
[b] *Department of Computer Science, University of West Florida, 11000 University Parkway, Pensacola, FL 32514, USA*

Abstract

We study the relationship between a computably enumerable real and its presentations: ways of approximating the real by enumerating a prefix-free set of binary strings. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Computably enumerable; Presentation; Real; Kraft–Chaitin

1. Introduction

Much of modern computability theory is concerned with understanding the computational complexity of sets of *positive integers*, yet, even in the original paper of Turing [23], a central topic of interest is effectiveness considerations for *reals.* Of particular interest to computable analysis (e.g. [24, 17, 18, 12]), and to algorithmic information theory (e.g. [6, 2, 16, 14]), is the collection of *computably enumerable reals.*

As in [20], a real α is computably enumerable if we can effectively generate it from, say, below. That is, there is a computable sequence of rationals $\{q_i\colon i\in\mathbb{N}\}$ with $q_{i+1}\geqslant q_i$ converging to α. If we can effectively compute the radius of convergence, then the real is computable, in the sense that we can compute effectively the nth bit of its dyadic expansion. But many interesting computably enumerable reals such as Chaitin's halting probability, definitely do *not* have such effectively converging sequences.

It is a very natural question to ask, given such a computably enumerable real how can it be generated? That is what kinds of effective sequences can be used to "present" the real. For simplicity we consider only reals between 0 and 1. Two classical representations of reals are Cauchy sequences and Dedekind cuts. Let α be a real. Then $L(\alpha)=\{q\in\mathbb{Q}\colon q\leqslant\alpha\}$ is the natural Dedekind cut associated with α; this was investigated by Soare [20, 21]. It is clear that α is a computably enumerable real iff $L(\alpha)$ is a

* Corresponding author.
E-mail addresses: rod.downey@vuw.ac.nz (R.G. Downey), glaforte@uwf.edu (G.L. LaForte).

0304-3975/02/$ - see front matter
PII: S0304-3975(01)00110-4

computably enumerable set of rationals. Soare investigated how $L(\alpha)$ relates to A where $\alpha = \Sigma_{n\in A} 2^{-n}$ for $A \subseteq \mathbb{N}$. That there exist computably enumerable reals where such A cannot be computably enumerable had already been observed by C.G. Jockusch. We will call a real α such that $\alpha = \Sigma_{n\in A} 2^{-n}$ for a computably enumerable $A \subseteq \mathbb{N}$ *strongly* computably enumerable. The other way that one can think of reals is as limits of Cauchy sequences: that is, $\alpha = \lim_s \{q_s\colon q_s \in B\}$.

The situation was recently clarified by Calude et al. [4].

Theorem 1 (Calude et al. [4]). *The following are equivalent for a real* α*:*

(i) α *is computably enumerable.*
(ii) *The lower Dedekind cut of* α *is computably enumerable.*
(iii) *There is an infinite computably enumerable prefix-free set* $W \subset \Sigma^*$ *such that* $\alpha = \sum_{x\in W} 2^{-|x|}$.
(iv) *There is a computable prefix-free set* $W \subset \Sigma^*$ *such that* $\alpha = \sum_{x\in W} 2^{-|x|}$.
(v) *There is a computable function* $f(x, y)$ *of two variables such that*
  (a) *for all* k, s, *if* $f(k,s) = 1$ *and* $f(k, s+1) = 0$, *then there exists* $k' < k$ *such that* $f(k', s) = 0$ *and* $f(k', s+1) = 1$.
  (b) $\alpha = a_1 a_2 \ldots$, *where* $a_i = \lim_s f(i,s)$.
(vi) *There is a computable increasing sequence of rationals with limit* α.

Although the apparently stronger (iv) is not explicitly stated in [4], it follows from (iii), since there are always an infinite number of strings we can add at any particular stage in the enumeration. Hence we can rule out larger and larger subsets of $\{0,1\}^*$ thereby making the complement of W computably enumerable as well. We refer to the approximation in (v.) above as an *almost-c.e.* representation of α. We remark that we desire the language W to be prefix free because convergence is guaranteed by the Kraft–Chaitin inequality. Prefix free languages are necessary for a proper treatment of, say, randomness, as in [16] or [6]. Together with Hirschfeldt, we also examine structural properties of randomness under randomness preserving reductions in the later paper [7].

In [4], and the later paper [3], the authors investigated the relationship between possible Cauchy sequences converging to α and $L(\alpha)$. Those authors examined the cut definition of real, and its effective content. They defined a *representation* A of a (c.e.) real a as a computable increasing sequence of rationals q_i for $i \in A$ with limit a. They asked what types of degrees can A have. Already we have seen that A can be computable. Furthermore, if α is computable that is the best we can do. It is not difficult to prove the following.

Lemma 1 (Soare [20], Calude et al. [3]). *For a c.e. real* α, *if* A *is a representation of* α, $A \leqslant_T L(\alpha)$.

Here $\leqslant_T$ denotes Turing reducibility. Lemma 1 extends earlier work of Soare who examined, in particular, the relationship between $L(\alpha)$ and $\deg(B)$ for $\alpha = \Sigma_{n\in B} 2^{-n}$. In [20], Soare observed that $L(\alpha) \leqslant_T B$ and $B \leqslant_{tt} L(\alpha)$, where $\leqslant_{tt}$ denotes truth table reducibility. However, he also proved that there are strongly c.e. α, as above, with $L(\alpha) \nleqslant_{tt} B$.

Calude et al. asked what is the relationship between degrees of representations of α and the degree of α, or, equivalently, $L(\alpha)$ and what can be said about the sequences in these terms. Calude et al. proved the following very interesting generalization of Lemma 1.

Lemma 2 (Calude et al. [3]). *Suppose that A represents α. Then A is an infinite half of a splitting of $L(\alpha)$.*

The proof is easy: Clearly, if A represents α then A must be an infinite c.e. subset of $L(\alpha)$. The thing to note is that $L(\alpha)-A$ is also c.e.. Given rational q, if q occurs in $L(\alpha)$, we need only wait till either q occurs in A or some rational bigger than q does. In fact, Calude et al. proved similarly that if B is a c.e. subset of a *representation* A of α then B is a representation of α iff it is a half of a splitting of A. These simple observations lead one to speculate that to understand the representations of α then we need only understand the splittings. This intuition was borne out by the definite results of Calude et al. [3] and Downey [8], who showed the following.

First not every splitting of $L(\alpha)$ can be a representation because they may not be increasing, for instance. However, Downey [8], improved an earlier result of Calude et al. to show that this is the answer up to m-degree.

Theorem 2 (Downey [8]). **a** *is the m-degree of a c.e. splitting of $L(\alpha)$ iff* **a** *is the m-degree of a representation of α.*

In the present paper, we will examine the relationship between dyadic representations of a real α (i.e. $\alpha = \sum_{\sigma\in W} 2^{-|\sigma|}$) and α, coming from the Calude et al. characterization in Theorem 1. We are led to the following basic definition.

Definition 1. For any $W \subseteq \{0,1\}^*$, we say W is a *presentation* of a c.e. real α if W is a prefix-free c.e. set such that $\alpha = \sum_{\sigma\in W} 2^{-|\sigma|}$.

Now it is an immediate consequence of the theorems above that a c.e. noncomputable real has infinitely many different *representations*, one of each c.e. m-degree below that of $L(\alpha)$. The situation for *presentations* is surprisingly different, as we see in our first theorem.

Theorem 3. *There is a c.e. real α which is not computable, but such that if W presents α, then W is computable.*

We will prove Theorem 3 in Section 3. We remark that the proof itself is quite interesting and fairly complex. It involves the use of a $\mathbf{0}''$ or "infinite injury" priority argument, an argument of a type hitherto not found in computable analysis.

The remainder of the paper is devoted to trying to understand what sorts of reals are "nearly computable" in the sense that they only have computable presentations; and what can be said about the types of presentations that a real might have.

As an illustration, one might expect that a real of high complexity, measured by, say, Turing degree might not be able to have only computable presentations. This is not the case. We are able to show the following.

Theorem 4. *There is a c.e. real α with $\alpha' \equiv_T \emptyset''$ such that if W presents α, then W is computable.*

This is but one example of a general collection of theorems of this type. One could no doubt prove that every jump class has such a c.e. real. Now the manner of the proof of Theorem 3 is somewhat reminiscent of that of a lattice embedding result (1-3-1) into the computably enumerable degrees. This suggests that it is not the complexity of the real which is important but the "dynamic speed of formation". One notion capturing this idea is that of "prompt simplicity" introduced by Maass [15]. Roughly speaking (precise definitions are given in Section 4), a degree is prompt if infinitely often, it is demonstrated to be noncomputable "quickly". The degree $\mathbf{0}'$ is prompt in this sense. We prove the following.

Theorem 5. *Suppose α has promptly simple degree. Then there is a presentation, A, of α that is noncomputable.*

The exact classification of degrees containing only reals with noncomputable presentations seems difficult.

Our last section tries to examine the question: Suppose we have a noncomputable presentation of a real α. What other sorts of presentations can α have? As with several other questions arising from computable mathematics, the answer seems to lie in strong reducibilities. Specifically, we use *weak truth table* reducibility. Thus $A \leqslant_{wtt} B$ iff there exists a Turing procedure Γ and a computable function γ such that $\Gamma^B = A$ and for all x, the maximum element queried in the computation $\Gamma^B(x)$, is $\leqslant \gamma(x)$. Wtt reducibility has proven useful in other parts of computable mathematics, notably Calude and Nies [5] proved that Ω Chaitin's number is *wtt*-complete but not *tt*-complete, and Downey and Remmel [9], showed that the degrees of c.e. bases of a c.e. vector space V are precisely the *wtt*-degrees below $\deg_{wtt} V$.

In our setting, in Section 5, we prove the following.

Theorem 6. *Let α be a computably enumerable real, with $\alpha = .\chi_A$ for some set A. Suppose that B is any presentation of α. Then $B \leqslant_{wtt} A$ with use function the identity.*

Theorem 7. *If A is a presentation of a c.e. real α and $C \leqslant_{wtt} A$ is computably enumerable, then there is a presentation B of α with $B \equiv_{wtt} C$.*

Note that if α is strongly c.e., so that $\alpha = .\chi_A$ for some c.e. set A, then $L(\alpha) \equiv_{tt} A$ and hence the degrees of presentations of α are exactly the c.e. *wtt* degrees below that of A. Because of this, it is possible to completely classify the degrees of presentations of such reals. Because there are sets, such as K, the halting problem, where for all $C \leqslant_T K$, $C \leqslant_{wtt} K$, we see that there are reals with presentations of each possible c.e. degree. So we can have the two extremes. A c.e. real can be only computably presentable, and at the other extreme, a c.e. real can have presentations of each c.e. degree.

We remark that there are a number of questions left open by this paper. First, we have seen that wtt-reducibility seems the correct one to use for presentations. The *wtt*-degrees of c.e. presentations of a real clearly form an uppersemilattice. What can be said about this semilattice? For instance, does it always have a top element? Is any Σ_3^0 semilattice realizable as the structure? Another question is to look at this material for reals that are the limits of computable, but not necessarily increasing, sequences of rationals. Ho [10] has proven that these are exactly the $\mathbf{0}'$-presentable reals, but what can be said about their possible presentations and representations? The smallest real closed field containing the c.e. reals turns out to be the set of weakly computable reals; these are d.c.e. reals—those with sequences obtained from differences of c.e. sets. That this set forms a field is not obvious, but was established by Weihrauch and Zheng [25]. What more can be said about such reals?

Finally, there are other reducibilities better tailored to study c.e. reals and their *relative randomness* such as Solovay's *domination reducibility.* What can be said about reals analysed under such reducibilities? Some results in this context can be found in [4, 7].

2. Notational niceties

In the following sections, we generally use notation that is standard from descriptive complexity theory and computability theory. In particular, when we construct c.e. sets to be presentations of reals with various computational properties, we generally follow the terminology of Soare [22]. An important abbreviation that deserves special note is the following: We fix in advance an enumeration of all c.e. sets $\langle x \in W_e \rangle$ as the output of some suitably-universal Turing machine such that exactly one pair $\langle e, x \rangle$ with $x \in W_e$ is listed at each stage s. We can then use "[s]" to relativize entire expressions involving computable dynamic processes with the meaning that the state of each such process is evaluated at stage s. This saves a considerable amount of notational clutter, with the cost of a small period of adjustment for the reader new to this convention.

3. A noncomputable real with only computable presentations

We restate Theorem 3 below.

Theorem 8. *There is a c.e. real* α *which is not computable, but such that if* W *presents* α, *then* W *is computable.*

Proof. We will construct an increasing computable sequence of rationals converging to α by defining a computable function $\alpha(x,y)$ such that

(i) for all k,s, if $\alpha(k,s)=1$ and $\alpha(k,s+1)=0$, then there exists $k'<k$ such that $\alpha(k',s)=0$ and $\alpha(k',s+1)=1$.

(ii) $\alpha=0.a_1a_2,\ldots,$ where $a_i=\lim_s \alpha(i,s)$.

We write $\alpha[s]$ for the approximation to α at stage s, that is, $\alpha[s]=0.b_1b_2\ldots$ where $b_i=\alpha(i,s)$.

The real α must satisfy two seemingly incompatible properties: α must itself be non-computable, yet every presentation of α must be computable. We fix in advance some computable one-to-one enumeration of the set $\{0,1\}^*$. In this way, we can naturally identify finite binary strings with natural numbers. We can therefore treat c.e. sets of binary inputs to machines as sets of natural numbers, and hence the descriptions involved in our construction can be a little simpler.

Notice that if α were computable, there would be some n indexing a program ϕ_n which halted on input x exactly when the xth digit of α turned out to be 1. Hence, a natural way to ensure the noncomputability of α is to satisfy the infinite sequence of requirements:

$$P_n\colon\ \exists x(x\in W_n \text{ if and only if } \alpha(x)=0).$$

We write W_n for the domain of the nth partial computable function ϕ_n, and $\alpha(x)$ for the xth digit of α. By the convention mentioned above, we can overlook the fact that W_n is a set of strings, not one of the numbers. To ensure the computability of all of α's presentations, we must also satisfy another infinite sequence of requirements:

$$N_e\colon\ \sum_{x\in W_e} 2^{-|x|}=\alpha \rightarrow W_e \text{ is computable.}$$

The basic strategy for satisfying P_n is simple: we choose some large number x, and set the xth digit of α equal to 1 by setting $\alpha(x,s)=1$. We then wait until x enters $W_n[s']$ at some $s'>s$, and then set $\alpha(x-1,s')=1$, but $\alpha(x,s')=0$.

The strategy for satisfying the N_e requirements is, in some sense, even simpler, whenever we see that $\sum_{x\in W_e}2^{-|x|}$ gets closer in value to our current approximation to α, $\alpha[s]$, we promise to never again change α enough to allow a string of length greater than the least difference between $\sum_{x\in W_e}2^{-|x|}$ and $\alpha[s]$ to enter W_e. Below, it will become clear that this involves slowing down the convergence of our approximation to α in order to slow down the convergence of $\sum_{x\in W_e}2^{-|x|}$ so much that W_e becomes computable. Notice that, almost-paradoxically, this *speeds up* the convergence of W_e when it is viewed as an infinite sequence of 0s and 1s indexed by a computable listing of all finite strings. Of course, speeding up the convergence of W_e enough makes W_e computable.

It should be clear that, as usual, the two kinds of requirements are in direct conflict with each other, since the strategy for P_n wishes to change $\alpha(x)$ at some noncomputable stage of the construction, whereas N_e wishes to guarantee at some computable stage that $\alpha(x)$ will never change again. The algorithm for specifying α must therefore be a priority construction, one which we organize using the tree-of-strategies architecture due to Lachlan.

Because the P-type strategies only require finite action, the main problem is to coordinate lower priority P-type strategies with higher priority N-type strategies. First we must describe in more detail the strategy for satisfying N_e. We define a length-of-agreement function

$$l(e)[s] = \begin{cases} s & \text{if } \alpha[s] = \sum_{x \in W_{e,s}} 2^{-|x|}, \\ \min\left\{ n\colon \alpha[s] - \sum_{x \in W_{e,s}} 2^{-|x|} > 2^{-n} \right\} & \text{otherwise.} \end{cases}$$

If $s > 0$, we let $m(e)[s] = \max\{l(e)[t]\colon t < s\}$, and define a stage to be e-expansionary if $l(e)[s] > m(e)[s]$.

Notice that we may assume that $\alpha[s] \geqslant \sum_{x \in W_{e,s}} 2^{-|x|}$, since otherwise we can create a permanent disagreement between α and the real presented by W_e by simply refusing to change α by more than $\frac{1}{2}(\sum_{x \in W_{e,s}} 2^{-|x|} - \alpha[s])$. Hence, we can satisfy N_e by refusing to change $\alpha(x,s)$ on any $x < m(e)[s]$.

Now, suppose we have reached a stage s such that some lower-priority requirement P_n *does* need to change α on some $x < m(e)[s]$. Our approach is to gradually change α in tiny steps that add up to a change on x. First we change α on $m(e)[s]$, increasing α by $1/2^{m(e)[s]}$. This forces $(\alpha - \sum_{x \in W_e} 2^{-|x|} \geqslant 2^{-m(e)})[s]$. There are two possibilities: either W_e never changes enough to converge again to within $2^{-m(e)[s]}$ of α, or eventually it does. In the first case, we can forever ignore requirement N_e, and we do so by starting over again at some point very far out in α and working to satisfy all the requirements there, viewing the part of α below $m(e)[s]$ as dead except insofar as it plays a role in permanently satisfying N_e. On the other hand, if W_e does change enough at some stage $t > s$ to approach $\alpha[s+1]$, then it can only do so by enumerating some string with length greater than $m(e)[s] - 1$. For if W_e had some shorter string added to it at stage t, we would have

$$\sum_{x \in W_{e,t}} 2^{-|x|} \geqslant \sum_{x \in W_{e,s}} 2^{-|x|} + 2^{-m(e,s)+1} > \alpha[s] + 2^{-m(e)[s]} = \alpha[t].$$

As pointed out above, once the approximation derived from W_e is greater than $\alpha[t]$ at some stage t, the requirement N_e can be satisfied forever with only a single large restraint. But then, once again $\alpha[s] - \sum_{x \in W_{e,s}} 2^{-|x|} < 2^{-m(e)[s]}$, and we can again increase the approximation to α by $2_{-m(e)[s]}$ and wait for W_e to respond again by enumerating some string longer than $m(e)[s] - 1$. Once it does, we have actually changed $\alpha(m(e))[s] - 1$, and continuing the process will enable us to change α on smaller and smaller numbers while forcing the enumeration into W_e to always be on very long strings as required.

Of course, while waiting for all the appropriate changes to take place in W_e, we must prevent lower-priority strategies from acting, even though they may be compatible with the final outcome that W_e is a presentation of α. To arrange this, we equip each strategy γ with a *counter* $c(\gamma)$ to indicate how many enumerations must occur before these lower-priority strategies can legitimately take action. At the beginning of our attempt to change α on some small number x, we set $c(\gamma)$ equal to the total number of changes on longer strings which we need in order to change α's value on x. Each time $\sum_{x\in W_{e,s}} 2^{-|x|}$ gets close enough to the current approximation for us to enumerate a new string, we do so and decrement the counter. Once the counter reaches 0, we allow the lower-priority strategies to act before increasing the restraint m associated to γ again. This gives these strategies the opportunity to take action at some later step in the process, in the matter familiar from other infinite-injury priority constructions.

Construction: We use the tree $\mathscr{T} = {}^{<\omega}2$ as our tree of strategies, assigning P_e and N_e to each string of length e. Let

$$g(e) = \begin{cases} 0 & \text{if } \alpha = \sum_{x\in W_e} 2^{-|x|}, \\ 1 & \text{if } \alpha \neq \sum_{x\in W_e} 2^{-|x|}. \end{cases}$$

g is the *true path* through $\mathscr{T}$, and encodes for each set W_e, whether or not W_e is a representation of α. At each stage s, we will have an approximation $g[s]$ with the property that $\liminf_s g[s] = g$. For any string $\sigma \in \mathscr{T}$, a stage s is a σ-stage if either $s = 0$ or $s > 0$ and $\sigma \subseteq g[s]$. Each string σ has several functionals and parameters used in its strategies for satisfying the requirements assigned to it. A strategy associated to a string is *initialized* by setting all of its associated parameters and functionals to diverge, except for its counter function c, which we set to 0.

To coordinate the actions taken during the computation correctly, we must specify when one of our strategies has higher priority than another. If σ and τ are two strings in $\mathscr{T}$, let $\rho = \sigma \cap \tau$; then we define $\sigma <_L \tau$ if and only if $\rho^\frown\langle 0\rangle \subseteq \sigma$ and $\rho^\frown\langle 1\rangle \subseteq \tau$. We can then define $\sigma < \tau$ if and only if $\sigma <_L \tau$ or $\sigma \subset \tau$. In this case, the strategy assigned to σ has higher priority than that assigned to τ.

The construction proceeds in stages.

Stage 0: All strategies are initialized and $g[0] = \lambda$, the empty string. For every x, $\alpha(x)[0] = 0$.

Stage $s > 0$: We define $g[s]$ of length less than or equal to s by recursion and let the strategies associated to each $\sigma \subseteq g[s]$ act as follows in order of increasing length of σ. Any parameter or functional value not explicitly set, or caused to diverge, at stage s gets the same value assigned to it at the end of stage s as it had at stage $s+1$.

Positive requirements: If $x(\sigma)\uparrow[s-1]$, then set $x(\sigma)[s]$ equal to 3 plus the least number greater than any yet mentioned in the construction. Let $\alpha(x(\sigma))[s] = 1$, and initialize all strategies associated to any $\tau > \sigma$ and the negative strategy associated to σ. Notice that in this case $\alpha(x(\sigma)-1)[s]=0$. If $x(\sigma)\downarrow[s-1]$, then let $x(\sigma)[s]=x(\sigma)[s-1]$. If $(x(\sigma)\in W_e)[s]$, then initialize all strategies associated to any $\tau>\sigma$ and the negative strategy associated to σ. Let $\gamma\subset\sigma$ be the longest initial segment of σ such that $\gamma^\frown\langle 0\rangle\subseteq\sigma$ and $r(\gamma)>x(\sigma))[s]$. Notice that $c(\gamma)[s-1]=0$, since otherwise $\gamma^\frown\langle 0\rangle$ cannot act. Let $(c(\gamma)=2^{r(\gamma)-x(\sigma)})[s]$. This will be a new counter that indicates how many γ-expansionary stages there must be before $\gamma^\frown\langle 0\rangle$ can act again. If there does not exist any such γ, then we set $a(x(\sigma))[s]=0$ and $a(x(\sigma)-1)[s]=1$.

Negative requirements: If N_e has already been permanently satisfied by $\sum_{x\in W_e}2^{-|x|}$ growing above $\alpha[s]$ since σ's negative strategy was last initialized, then we let $\sigma^\frown\langle 1\rangle$ act at stage s. Otherwise, we first test whether or not $(\sum_{x\in W_e}2^{-|x|}>\alpha)[s]$. If so, we let k be the least number greater than $-\log_2(\sum_{x\in W_e}2^{-|x|}-\alpha)[s])$, such that $\alpha(k)[s]=0$ and $\alpha(k-1)[s]=0$. We initialize all τ such that $\sigma<\tau$, and declare N_e to be permanently satisfied. Notice that after this stage, all lower-priority positive strategies will start over with witnesses larger than k, and so will never be able to increase α above $\sum_{x\in W_e}2^{-|x|}$.

Otherwise, suppose $\sum_{x\in W_e}2^{-|x|}$ is still less than or equal to α at s. For each string σ of length e, let $l(\sigma)[s]=l(e)[s]$ and let

$$m(\sigma)[s]=\max\{l(\sigma)[t]\colon\ t<s \text{ and } t \text{ is a } \sigma\text{-stage}\}.$$

A σ-stage s is *σ-expansionary* if $l(\sigma)[s]>m(\sigma)[s]$. If s is not σ-expansionary, then we let $\sigma^\frown\langle 1\rangle$ act at stage s, and we initialize all strategies associated to any τ such that $\sigma<_L\tau$. If s is σ-expansionary, we initialize all strategies associated to $\tau\geqslant\sigma^\frown\langle 1\rangle$. There are two main cases:

1. If $c(\sigma)[s-1]>0$, then let $c(\sigma)[s]=c(\sigma)[s-1]-1$. Let $\gamma\subset\sigma$ be the longest initial segment of σ such that $\gamma^\frown\langle 0\rangle\subseteq\sigma$ and $r(\gamma)>r(\sigma))[s]$. Notice that $c(\gamma)[s-1]=0$, since otherwise $\gamma^\frown\langle 0\rangle$ cannot act. Let $c(\gamma)=2^{r(\gamma)-r(\sigma)})[s]$. Again, this is a counter indicating how many γ-expansionary stages there must be before $\gamma^\frown 0$ can act again. If there does not exist any such γ, then we add $2^{-r(\sigma)}[s]$ to α, resetting the appropriate values of $\alpha(y)[s]$ for $y\leqslant r(\sigma)[s]$. (If $y\leqslant r(\sigma)$ is greatest such that for all z with $y\leqslant z\leqslant r(\sigma)[s]$, $\alpha(z)[s-1]=1$, we let $\alpha(y-1)[s]=1$ and for all such z, $\alpha(z)[s]=0$. If no such y exists, that is, $\alpha(r(\sigma))[s-1]=0$, then we simply let $a(r(\sigma))[s]=1$.)
2. If $c(\sigma)[s-1]=0$, then let $(r(\sigma)=l(\sigma))[s]$ and let $\sigma^\frown\langle 0\rangle$ act.

This completes the construction.

Verification: The key fact about our construction is the following.

Lemma 3. *Suppose s is a σ-expansionary stage such that σ's negative strategy is never initialized after s and $c(\sigma)[s]>0$. Then if t is the next σ-expansionary stage after s,*

$$2^{-r(\sigma)[s]}\leqslant\alpha[t]-\alpha[s-1]\leqslant\alpha[t]-\alpha[s]\leqslant 2^{-r(\sigma)[s]+1}.$$

Proof. This follows by induction. By construction, the only thing that can prevent $\alpha[s]=\alpha[s-1]+2^{-r(\sigma)}[s]$ is the existence of some longest $\gamma\subset\sigma$ such that $\gamma^\frown\langle 0\rangle\subseteq\sigma$ and $r(\gamma)>r(\sigma))[s]$. Since $c(\gamma)[s-1]=0$, we have $c(\gamma)=2^{r(\gamma)-r(\sigma)})[s]$. By construction, this is a counter indicating how many γ-expansionary stages there must be before $\gamma^\frown\langle 0\rangle$ can act again—that is, before the next σ-stage, which must be $\leqslant t$. By induction, then

$$\alpha[t] \geqslant \alpha[s-1] + \sum_{1}^{2^{r(\gamma)-r(\sigma)}} 2^{-r(\gamma)} = a[s-1] + 2^{r(\gamma)-r(\sigma)}2^{-r(\gamma)} = a[s-1] + 2^{-r(\sigma)}.$$

Because σ is never intialized before t, the only increase in α due to some strategy assigned to a string compatible with $\sigma^\frown\langle 0\rangle$ is just the addition of $2^{-r(\sigma)[s]}$. On the other hand, notice that only positive strategies assigned to strings τ with $\sigma^\frown\langle 1\rangle \leqslant \tau$ can act at any s' between stages s and t, and all these strategies were initialized at s and hence have witnesses $x(\tau)[s']$ greater than $r(\sigma)[s]+2$. Since any of these at most wishes to add $2^{-x(\tau)[s']-1}$ to a, the total increase due to σ itself and all these τ by stage t is at most

$$2^{-r(\sigma)[s]} + 2^{-r(\sigma)[s]-2}\left(\sum_{j=0}^{\infty} 2^{-1}\right) = 2^{-r(\sigma)[s]} + 2^{-r(\sigma)[s]-1} < 2^{-r(\sigma)[s]+1}$$

as required. $\square$

We can now show by induction that all requirements are satisfied. Assume $\sigma \subset g$ and we have reached a stage such that $g[s]$ is never to the left of σ again, and furthermore, that no strategy assigned to $\tau \subset \sigma$ ever initializes either of σ's strategies again. Let $e = |\sigma|$.

First we show that P_e is satisfied and that σ's positive strategy initializes lower-priority strategies at most two more times. Clearly, at the first σ stage after which σ is never initialized, a witness $x = x(\sigma)$ is chosen permanently and all lower-priority strategies are initialized. This action sets $\alpha(x)[s] = 1$ and $\alpha(x-1)[s] = 0$. Since all lower-priority strategies are initialized at this point, no value of α on any number less than or equal to x can ever change after s unless the positive strategy for σ itself changes $a(x)$ or $a(x-1)$. Hence, if $x \notin W_e$, P_e is satisfied. Else x enters W_e at some stage after s. At the next σ-stage after this point, the strategy for σ initializes all lower priority strategies and attempts to change $\alpha(x-1)$ to 1 and $\alpha(x)$ to 0 by incrementing $\alpha[s']$ by 2^{-x}. Notice that any $\tau <_L \sigma$ would have initialized σ at any τ-stage—hence any such τ must have a restraint lower than x. Therefore, if there is no $\tau \subset \sigma$ with $\tau^\frown\langle 0\rangle \subseteq \sigma$ and $r(\tau)[s] > x$, $\alpha(x)[s'] = 0$. Otherwise, the longest such τ gets its counter $c(\tau)[s']$ set to $2^{r(\tau)[s]-x}$). By Lemma 3, and the action assigned to negative strategies, there cannot be another $\tau^\frown\langle 0\rangle$-stage until α has been incremented to at least

$$\alpha[s'] + \sum_{j=1}^{2^{r(\tau)[s']-x}} 2^{-r(\tau)[s']} = \alpha[s'] + 2^{-x}.$$

Since no other strategies can change a below x, $\alpha(x)$ is set permanently to 0 after this point. Thus, P_e is satisfied. Clearly, this strategy initializes all lower priority strategies at most two more times after s.

Now we can show N_e is satisfied, since we know the σ-strategy for N_e is only initialized at most twice after s. Since $r(\sigma)[t]$ is a computable function increasing monotonically with the stage t, this is straightforward. After any stage t, only strategies assigned to $\tau > \sigma$ can attempt to change α below $r(\sigma)[t]-1$. Hence, if some string of length less than $r(\sigma)[t]$ were added to W_e at $t' > t$, this would increase $\sum_{x\in W_e} 2^{-|x|}[t']$ by at least $2^{-r(\sigma)+1}$, in which case by Lemma 3,

$$\alpha[t] < \alpha[s] + 2^{-r(\sigma)+1} \leqslant \left(\sum_{x\in W_e} 2^{-|x|}\right)[t'].$$

By the construction, all lower priority strategies below σ would be initialized at the next σ-stage, and N_e would be permanently satisfied.

This completes the verification. □

It turns out that using a standard technique, we can improve our result to show that a real with only computable presentations can even have high Turing degree.

Theorem 9. *There is a c.e. real α with $\alpha' \equiv_T \emptyset''$ such that if W presents α, then W is computable.*

Sketch. This has the same relationship to Theorem 3 as the existence of a high minimal pair does to the mere existence of a minimal pair. (See [22, 13.2].) We use the same tree to control the enumeration of A, but assign strategies for the negative requirement N_e, as above, to all strings of length $2e+1$. We have a different sequence of positive requirements. Let $g : \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be any one-to-one computable function such that for every $x_0,\ y_0,\ x_1,\ y_1,\ |g(x_0,y_0)-g(x_1,y_1)|>3$. (Multiplying the usual pairing function by 4 will obviously work.) We will satisfy the infinite sequence of positive requirements

$$P_e\colon\ \exists y_0(e\in\emptyset'' \Leftrightarrow \forall y > y_0\, \alpha(g(e,y)) = 1).$$

If P_e is satisfied, then we can use the jump of α to search for the least such y_0 for each e; clearly, then, $e\in\emptyset''$ if and only if $\alpha(y_0+1)$. Since $\alpha\leqslant_T\emptyset'$, it is of course immediate that $\alpha'\leqslant_T\emptyset''$. We assign P_e to each string in $\mathscr{T}$ of length $2e$.

Because $\emptyset''$ is c.e. relative to $\emptyset'$, it has a Σ^0_2 definition—in other words, there is a computable function $c(x,y,z)$ such that

$$x\in\emptyset'' \quad \text{if and only if} \quad \exists y\forall z\, c(x,y,z)=1.$$

The idea for ensuring $\emptyset''\leqslant_T\alpha'$ is that after σ with P_e stops being initialized by γ below it at some stage s_0, the strategy will start trying to change α beyond s_0. Originally, we set $\alpha(g(y,e))=1$. If, later we find that for all $y_0\leqslant y$ there is a z with $c(y_0,z,e)=0$, then we must change $\alpha(g(y,e))=0$. We do this as before, by changing $\alpha(g(y,e)-1)$ to 1, adding strings that have to wait to get past the restraints due to higher-priority strategies. To allow all positive strategies to get a chance to be satisfied, we must prevent higher priority positive requirements from enumerating at stages where they get an infinitary outcome. (In other words, once a strategy has successfully changed α on the argument that was its goal, it will have to wait until the next stage before trying to change α again.)

N_e is satisfied almost as before—we set restraints which prevent lower-priority strategies from changing our real directly on small values. However, strategies for higher-priority positive requirements must in general take infinite action. We are therefore prevented from protecting the strategy for N_e from them by merely assuming that the negative strategy starts acting after all higher-priority positive requirements are satisfied. Instead, we must guess at the infinite behaviour of each positive strategy assigned to some P_a. Letting f be the true path, this means we set $f(2a)[s]=0$ if the strategy associated to $f[s]\restriction_{2a}$ has just finished changing α because some $c(y_0,z,a)=0$, and set $f(2a)[s]=1$ otherwise. In this case, if the strategy for N_e is assigned to a string extending $f[s]\restriction_{2a+1}$, its length-of-agreement function must assume that $\alpha(g(y,a)-1)=1$ and $\alpha(g(y,a))=0$ for every y such that $g(y,a)$ is below any restraint that the strategy for N_e might wish to set. In other words, if $\sum_{x\in W_e} 2^{-|x|}$ does not agree with these values, then the length-of-agreement function is reduced accordingly, even if these are not the current values. If in fact $f(2a)=0$, then eventually these values will change and the real presented by W_e will agree with α enough to allow the restraint to grow. Furthermore, since the strategy for P_a will already have changed α as much as it ever will below this new restraint, the requirement N_e will be satisfied as before.

For every e, P_e is satisfied, since we can eventually change any value we want by changing α repeatedly on small values and waiting for all necessary agreements for negative strategies to be restored. So, if $e\in\emptyset''$, there is some y_0 such that every z has $c(y_0,z,e)=1\neq 0$, and so, for all $y\geqslant y_0$, $\alpha(g(y,e))=1$. If $e\notin\emptyset''$, then for every y, there is a z with $c(y,z,e)=0$. So, of course, for every y beyond the stage where the strategy is last initialized, we put $\alpha(g(y,e))=0$. This shows every presentation of α is computable, and $\emptyset''\leqslant_T \alpha'$, as required.

We leave the details to the enterprising reader. $\square$

4. Promptly simple presentations of reals

A coinfinite computably enumerable set D is promptly simple if there is a computable function p such that for every infinite c.e. set W, there exists a stage $s>0$ and number x such that $x\in(W[s]-W[s-1])\cap D[p(s)]$. It is clear that no generality is lost by also requiring that for every s, $p(s)>s$: the intuitive meaning is therefore that p enables D to eventually guess correctly about some immediate change in W. This notion was introduced by Maass in [15], and general technical methods for working with promptly simple sets were developed in [1]. The discussion in [22, XIII] is a useful one. We now show that if a c.e. real a has promptly simple degree, it has a noncomputable presentation.

Constructions involving promptly simple sets are simplified by the use of the following technical result due to Ambos-Spies et al. [1]:

Theorem 10 (Slowdown lemma). *Let $U_e[s]$ be a computable sequence of finite sets such that for all e $U_e[s] \subseteq U_e[s+1]$ and $U_e = \bigcup_{s=0}^{\infty} U_e[s]$. Then there exists a computable function g such that for all e,*
(i) *$W_{g(e)} = U_e$, and*
(ii) *if $x \in U_e[s] - U_e[s-1]$, then $x \notin W_{g(e)}[s]$.*

Condition (ii) on $W_{g(e)}$ means that every element enumerated into U_e appears strictly later in $W_{g(e)}$.

Theorem 11. *Suppose α has promptly simple degree. Then there is a presentation, A, of α that is noncomputable.*

Proof. Let $\alpha \equiv_T D$, where D is promptly simple, and suppose α is given to us with an almost-c.e. approximation—that is, there is a computable function $\alpha(i,s)$ such that
(i) for all i, $\alpha(i) = \lim_{s\to\infty} \alpha(i,s)$,
(ii) for all i and s, if $\alpha(i,s) = 1$ and $\alpha(i,s+1) = 0$, then there exists some $j < i$ such that $\alpha(j,s) = 0$ and $\alpha(j,s+1) = 1$.

In particular, let $D = \Gamma(\alpha)$. We must give an algorithm to enumerate a prefix-free set of strings A so that $\sum_{x\in A} 2^{-|x|} = \alpha$, and for every program index e,

$$P_e\colon\ W_e \text{ infinite} \Rightarrow \bar{A} \neq W_e.$$

This ensures A is noncomputable, since $\bar{A}$ is not c.e.

The following result shows that we can always assume we have some strings of the proper length available to add to our set. It is worth noting that this result does not enable us to choose the particular strings we intend to add, however: this is one feature that makes working with presentations of a c.e. real different from working with the Dedekind-cut type representations.

Theorem 12 (Chaitin–Kraft [6]). *Given any computable sequence $\langle s_i, n_i\rangle$ of elements of $s_i \in \{0.1\}^* \times \mathbb{N}$, such that $\sum_{i=0}^{\infty} 2^{-n_i} < 1$, there exists a one-to-one computable enumeration $\langle x_i\rangle$ of elements of $\{0,1\}^*$ and a Turing machine M such that*
(1) *$\{x_i\colon i \in \mathbb{N}\}$ is prefix-free,*
(2) *for all $i \in \mathbb{N}$, $|x_i| = n_i$,*
(3) *for all $i \in \mathbb{N}$, $M(x_i) = s_i$, and*
(4) *for all $x \in \{0,1\}^* - \{x_i\colon\ i \in \mathbb{N}\}$, $M(x)\uparrow$.*

In fact, we only need (1) and (2).

We will build A in stages, enumerating at most one binary string at each stage, in such a way that there are infinitely many stages s such that $(\sum_{x\in A} 2^{-|x|} = \alpha)[s]$.

In order to diagonalize against all c.e. sets, we need to add strings to A at various stages which have relatively short lengths. The strategy for satisfying P_e will involve a finite sequence of attempts to enumerate a short string from W_e into A, at least one of which will work if W_e is infinite. Very briefly the idea is the following: Because α must compute the promptly simple set D, we can use the function p giving the prompt simplicity of D to search for a stage at which α must change on some relatively small value, enabling us to enumerate a string from W_e into A. The key fact is that the search can be computably bounded by p and is guaranteed to eventually succeed by the prompt simplicity which p witnesses for D.

Construction: At stage 0, $A[0]=\emptyset$.

We now specify the actions at stage $s>0$: Notice that by Chaitin–Kraft, since $\alpha\leqslant 1$, we can enumerate a string of any length we wish into A at a given stage and simultaneously preserve prefix-freeness as long as we guarantee that $(\sum_{x\in A} 2^{-|x|}\leqslant\alpha)[s]$. At any stage s where we are not in the process of making some attempt on a requirement P_e, we simply add enough strings of the proper length into A to ensure $(\sum_{x\in A} 2^{-|x|}=\alpha)[s]$.

First, suppose there is some least $e<s$ for which P_e has an active witness $x=x(e,i)[s]$ such that $D(x)[s]=\Gamma(\alpha;x)[s]=0$ with the use of y and $\bar{A}[s]$ and $W_e[s]$ are equal on all strings of length less than or equal to y. Then we enumerate x into the auxiliary set U_e. Using the appropriate function g giving an index for a $W_{g(e)}$ which slows down the enumeration into U_e, we let $t>s$ be least such that $x\in W_{g(e)}[t]$. We now freeze all action for our construction until stage $p(t)$ is reached, and then check to see whether $x\in D[p(t)]$. If so, then $\alpha[p(t)]$ must have increased below 2^{-y}, so that we may add a new string of length y into A at stage $p(t)$ and satisfy P_e permanently. Otherwise we release A, enumerate sufficient strings of the proper length to restore $(\sum_{x\in A} 2^{-|x|}=\alpha)[p(t)]$, and declare attempt i on P_e to have ended in failure.

Finally, let $j<s$ be least such that P_j is unsatisfied and there is no active witness defined for requirement P_j at s. Choose a new witness $x(j,k)[s]$, where k is the least number for which all previous attempts at satisfying P_j have ended in failure.

This ends the construction.

Verification: We just need to show that every requirement P_e is eventually satisfied and we only freeze A finitely often for the associated strategy.

Since D is coinfinite, if W_e is infinite and all our attempts at satisfying P_e were to end in failure, then the set $U_e=W_{g(e)}$ would be infinite. But then, since D is promptly simple, this means there would be some x in $(W_{g(e)}[t]-W_{g(e)}[t-1])\cap D[p(t)]$. By definition of $W_{g(e)}=U_e$, this means D must have changed value on x between the stage $s<t$ at which $\bar{A}[s]$ appeared to equal $W_e[s]$ and $p(t)$. But then some element from $W_e[s]$ is enumerated into A at stage $p(t)$ by construction, satisfying the requirement. Since each P_e can be satisfied after a finite number of attempts, it is a straightforward induction to show that each associated strategy only freezes A finitely often. Thus all requirements can be satisfied, and

$$\lim_{s\to\infty}\sum_{x\in A[s]} 2^{-|x|}=\alpha$$

as required. $\square$

5. Weak-truth-table reducibility and presentations of reals

We start by pointing out the following:

Theorem 13. *Let α be a computably enumerable real, with $\alpha = .\chi_A$ for some set A. Suppose that B is any presentation of α. Then $B \leqslant_{wtt} A$ with use function the identity.*

Proof. We use the enumeration of B to approximate α at stage s, defining $\alpha[s] = \sum_{\sigma\in B[s]} 2^{-|\sigma|}$. Let n be any natural number, and let $t(n)$ be the least stage such that for all $i \leqslant n$, $\alpha(i)[t(n)] = A(i)$. Notice this only checks α up to n itself. Clearly, if τ is any string of length less than n, and $\tau \in B - B[t(n)]$, then

$$\sum_{\sigma\in B} 2^{-|\sigma|} \geqslant 2^{-|\tau|} + \sum_{\sigma\in B[t(n)]} 2^{-|\sigma|}.$$

This implies that there exists some $i \leqslant n$ such that $A(i) = 1$ and $\alpha(i)[t(n)] = 0$. This is a contradiction. $\square$

Theorem 14. *If A is a presentation of a c.e. real α and $C \leqslant_{wtt} A$ is computably enumerable, then there is a presentation B of α with $B \equiv_{wtt} C$.*

Proof. Suppose $\Gamma(X)$ is a computable functional with a computable use function γ such that $\Gamma(A) = C$. We can assume γ is monotonically increasing. Let $\langle n, m\rangle : \mathbb{N}\times\mathbb{N}\to\mathbb{N}$ be a computable one-to-one function such that for all n, m, $\max\{n, m\} < \langle n, m\rangle$. (Adding 1 to the usual pairing function $\frac{1}{2}((n+m)^2 + 3n + n)$ gives such a function.) Notice that, since A presents α, using the Chaitin–Kraft theorem we can enumerate strings of any length we wish into $B[s]$ at as long as we ensure

$$\sum_{\sigma\in B[s]} 2^{-|\sigma|} \leqslant \sum_{\sigma\in A[s]} 2^{-|\sigma|}.$$

We fix enumerations of Γ, C and A so that at each stage s, exactly one element enters C, exactly one element enters A, and for every $x < s$, $(\Gamma(A; x) = C(x))[s]$. We may assume A is infinite, since there is nothing to prove if A is computable. We construct B in stages, using the function $\langle n, m\rangle$ as follows.

At stage 0, let $B[0] = \emptyset$.

At stage $s+1$, we first find the unique number n_s and string σ_s that enter C and A, respectively at stage $s+1$. If $|\sigma_s| < \gamma(n_s)$, then we enumerate $2^{\langle|\sigma_s|, n_s\rangle - |\sigma_s|}$ strings of length $\langle|\sigma_s|, n_s\rangle$ into $B[s+1]$. If $|\sigma_s| \geqslant \gamma(n_s)$, then we enumerate $2^{\langle|\sigma_s|, |\sigma|+s\rangle - |\sigma_s|}$ strings of length $\langle|\sigma_s|, |\sigma_s| + s\rangle$ into $B[s]$.

This ends the construction of B.

Notice that either of the actions taken at stage $s+1$ merely serves to ensure that

$$\sum_{\sigma\in B[s+1]} 2^{-|\sigma|} = \sum_{\sigma\in A[s+1]} 2^{-|\sigma|},$$

hence, we always have enough strings available to keep B prefix-free.

Suppose $n \in \mathbb{N}$. Let $s(n)$ be least so that $B[s(n)]$ agrees with B on all strings less than or equal to length $\langle \gamma(n), n \rangle$. Now, suppose there exists $t > s(n)$ such that $n \in C[t] - C[t-1]$. In this case, because for every s and $x < s$, $C(x)[s] = \Gamma(A; x)[s]$, there must be some σ with $|\sigma| < \gamma(n)$ which enters A at t. By construction, then, since $\sigma = \sigma_t$ and $n = n_t$, we have $2^{\langle |\sigma_t|, n_t \rangle - |\sigma_t|} > 1$ strings of length $\langle |\sigma_t|, n_t \rangle$ entering B at stage $t > s(n)$, which is a contradiction. Hence we can compute $C(n)$ from $B(n)$ with a use bounded by the number of strings of length less than or equal to $\langle \gamma(n), n \rangle$, which is a computable function. This gives $C \leqslant_{wtt} B$.

Next, consider any binary string τ. Using the computability of $\langle i, n \rangle$ and the fact that $\max\{i, n\} < \langle i, n \rangle$ we can ask whether there exist i and n such that $|\tau| = \langle i, n \rangle$. If not, then $\tau \notin B$. In this case, let $t(n) = 0$. Otherwise, suppose $|\tau| = \langle i, n \rangle$, If, $i \geqslant \gamma(n)$ then τ can only enter B at stage s if $s = n - i$. If, on the other hand, $i < \gamma(n)$. Then if τ enters B at stage $s+1$, this can only be because $|\tau| = \langle |\sigma_s|, n_s \rangle$. We enumerate $2^{\langle |\sigma_s|, n_s \rangle - |\sigma_s|}$ strings of length $\langle |\sigma_s|, n_s \rangle$ into $B[s+1]$. In either case, if we let $t(n)$ be the least number greater than $n - i$ so that $C[t(n)] \upharpoonright_{n+1} = C \upharpoonright_{n+1}$, we have $B(\tau) = B(\tau)[t(n)]$. Since n is computable from $|\tau|$, $B \leqslant_{wtt} C$, as required. □

References

[1] K. Ambos-Spies, C.G. Jockusch, R.A. Shore, R.I. Soare, An algebraic decomposition of the recursively enumerable degrees and the coincidence of several degree classes with the promptly simple degrees, Trans. Amer. Math. Soc. 281 (1984) 109–128.

[2] C. Calude, Information Theory and Randomness, an Algorithmic Perspective, Springer, Berlin, 1994.

[3] C. Calude, R. Coles, P. Hertling, B. Khoussainov, Degree-theoretic aspects of computably enumerable reals, in: Cooper, Truss (Eds.), Models and Computability, Cambridge University Press, Cambridge, 1999.

[4] C. Calude, P. Hertling, B. Khoussainov, Y. Wang, Recursively enumerable reals and Chaitin's Ω number, STACS '98, Springer Lecture Notes in Computer Science, Vol. 1373, 1998, Springer, Berlin, pp. 596–606.

[5] C. Calude, A. Nies, Chaitin's Ω numbers and strong reducibilities, J. Universal Comput. Sci. 1997.

[6] G. Chaitin, A theory of program size formally identical to information theory, J. ACM 22 (1975) 329–340.

[7] D. Hirschfeldt, R. Downey, G. Laforte, Randomness and Reducibility, in preparation.

[8] R. Downey, Computability, definability, and algebraic structures, Proc. 7th Asian Logic Conf., Taiwan, to appear.

[9] R.G. Downey, J.B. Remmel, Classification of degree classes associated with r.e. subspaces, Ann. Pure Appl. Logic 42 (1989) 105–125.

[10] Chun-Kuen Ho, Relatively recursive reals and real functions, Theoret. Comput. Sci. 219 (1999) 99–120.

[11] Ker-I Ko, On the continued fraction representation of computable real numbers, Theoret. Comput. Sci. A 47 (1986) 299–313.

[12] Ker-I Ko, Complexity of Real Functions, Birkhauser, Berlin, 1991.

[13] Ker-I Ko, H. Friedman, On the computational complexity of real functions, Theoret. Comput. Sci. 20 (1982) 323–352.

[14] Ming Li, P. Vitanyi, Kolmogorov Complexity and its Applications.

[15] W. Maass, Recursively enumerable generic sets, J. Symbolic Logic 47 (1982) 809–823.

[16] P. Martin-Löf, The definition of random sequences, Inform. Control 9 (1966) 602–619.

[17] M. Pour-El, From axiomatics to intrinsic characterization, some open problems in computable analysis, Theoret. Comput. Sci. 219 (1999) 319–329.

[18] M. Pour-El, I. Richards, Computability in Analysis and Physics, Springer, Berlin, 1989.
[19] H. Rice, Recursive real numbers, Proc. Amer. Math. Soc. 5 (1954) 784–791.
[20] R. Soare, Recursion theory and Dedekind cuts, Trans. Amer. Math. Soc. 140 (1969) 271–294.
[21] R. Soare, Cohesive sets and recursively enumerable Dedekind cuts, Pacific J. Math. 31 (1) (1969) 215–231.
[22] R. Soare, Recursively Enumerable Sets and Degrees, Springer, Berlin, 1987.
[23] A. Turing, On computable numbers with an application to the Entscheidungsproblem, Proc. Amer. Math. Soc. 43 (1937) 544–546.
[24] K. Weihrauch, Computability, Springer, Berlin, 1987.
[25] K. Weihrauch, X. Zheng, Arithmetical hierarchy of real numbers, MFCS'99, Sklarska Poreba, Poland, September 1999, pp. 23–33.