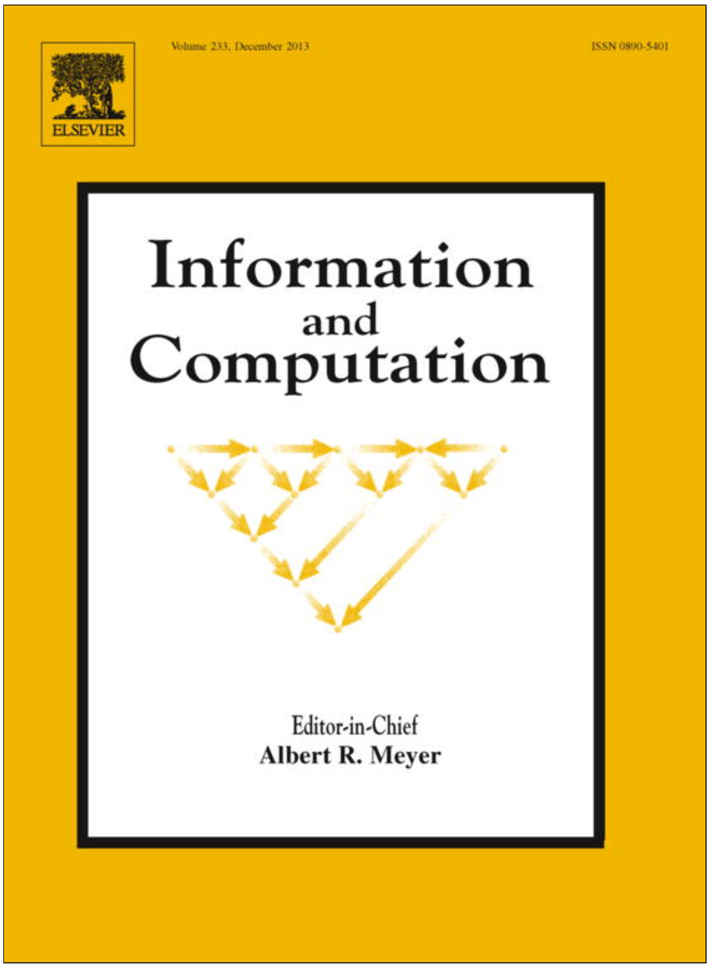
Volume 233, December 2013
ISSN 0890-5401
ELSEVIER
Information and Computation
Editor-in-Chief
Albert R. Meyer

Information and Computation 233 (2013) 41–59

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Universal computably enumerable sets and initial segment prefix-free complexity

George Barmpalias [1]

State Key Lab of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China

ARTICLE INFO

Article history:
Received 11 October 2011
Received in revised form 16 April 2013
Available online 10 December 2013

Keywords:
Universal sets
Computably enumerable
Kolmogorov complexity
Initial segment complexity

ABSTRACT

We show that there are Turing complete computably enumerable sets of arbitrarily low nontrivial initial segment prefix-free complexity. In particular, given any computably enumerable set A with nontrivial prefix-free initial segment complexity, there exists a Turing complete computably enumerable set B with complexity strictly less than the complexity of A. On the other hand it is known that sets with trivial initial segment prefix-free complexity are not Turing complete.

Moreover we give a generalization of this result for any finite collection of computably enumerable sets A_i, $i < k$ with nontrivial initial segment prefix-free complexity. An application of this gives a negative answer to a question from a monograph by Downey and Hirschfeldt (also raised in an article by Merkle and Stephan) which asked for minimal pairs in the structure of the c.e. reals ordered by their initial segment prefix-free complexity.

Further consequences concern various notions of degrees of randomness. For example, the Solovay degrees and the K-degrees of computably enumerable reals and computably enumerable sets are not elementarily equivalent. Also, the degrees of randomness of c.e. reals based on plain and prefix-free complexity are not elementarily equivalent; the same holds for the degrees of c.e. sets.

1. Introduction

The interplay between the information that can be coded into an infinite binary sequence and its initial segment complexity has been the subject of a lot of research in the last ten years. A rather influential result from [20] that spawned a renewed interest in this area was that sequences with very easily describable initial segments cannot compute the halting problem. Moreover the method that was used to establish it, often referred to as the decanter method, was novel and inspired much of the deeper work in this area. We show that although a universal computably enumerable set does not have trivial initial segment complexity, it can have arbitrarily low nontrivial initial segment complexity. Moreover our method is dual to the decanter method and in this sense the present paper can be seen as a missing companion to [20].

We start with a brief overview of Kolmogorov complexity in Section 1.1 and measures of relative randomness in Section 1.2 with a special attention to the topics around our results. In Section 1.3 we discuss the class of sequences with trivial initial segment complexity along with the motivation of our results, which are presented in Section 1.4. A number of applications are given in Section 1.5 and Section 1.6 discusses connections of the present work with research on other

E-mail address: barmpalias@gmail.com.
URL: http://www.barmpalias.net.

[1] Barmpalias was supported by a research fund for international young scientists No. 611501-10168 and an *International Young Scientist Fellowship* number 2010-Y2GB03 from the Chinese Academy of Sciences. Partial support was also obtained by the *Grand project: Network Algorithms and Digital Information* of the Institute of Software, Chinese Academy of Sciences. Special thanks go to the referees who carefully read a previous draft and suggested many improvements.

0890-5401/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ic.2013.12.001

reducibilities that are related to Kolmogorov complexity. In Section 2 we introduce the main technical tools that are required for the proofs of our results and Sections 3 and 4 contain the proofs of the two main results respectively.

1.1. Kolmogorov complexity and randomness

A standard measure of the complexity of a finite string was introduced by Kolmogorov in [22]. The basic idea behind this approach is that simple strings have short descriptions relative to their length while complex or random strings are hard to describe concisely. Kolmogorov formalized this idea using the theory of computation. In this context, Turing machines play the role of our idealized computing devices, and we assume that there are Turing machines capable of simulating any mechanical process which proceeds in a precisely defined and algorithmic manner. Programs can be identified with binary strings. A string τ is said to be a description of a string σ with respect to a Turing machine M if this machine halts when given program τ and outputs σ. Then the Kolmogorov complexity of σ with respect to M (denoted by $C_M(\sigma)$) is the length of its shortest description with respect to M. It can be shown that there exists an *optimal* prefix-free machine V, i.e. a machine which gives optimal complexity for all strings, up to a certain constant number of bits. This means that for each Turing machine M there exists a constant c such that $C_V(\sigma) < C_M(\sigma) + c$ for all finite strings σ.

When we come to consider randomness for infinite strings, it becomes important to consider machines whose domain satisfies a certain condition; the machine M is called *prefix-free* if it has prefix-free domain (which means that no program for which the machine halts and gives output is an initial segment of another). The complexity of a string σ with respect to a prefix-free machine M is denoted by $K_M(\sigma)$. As with the case of plain Turing machines, there exists an *optimal* prefix-free machine U. This means that for each prefix-free machine M there exists a constant c such that $K_U(\sigma) < K_M(\sigma) + c$ for all finite strings σ.

According to the above discussion, both in the case of plain or prefix-free Turing machines the choice of the underlying optimal machine does not change the complexity distribution significantly. Hence the theories of plain and prefix-free complexity can be developed without loss of generality, based on fixed underlying optimal plain and prefix-free machines V, U. We let $C = C_V$ and $K = K_U$.

In order to define randomness for infinite sequences, we consider the complexity of all finite initial segments. A finite string σ is said to be c*-incompressible* if $K(\sigma) \geqslant |\sigma| - c$. Levin [23] and Chaitin [14] defined an infinite binary sequence X to be random if there exists some constant c such that all of its initial segments are c-incompressible. By identifying subsets of $\mathbb{N}$ with their characteristic sequence we can also talk about randomness of sets of numbers. This definition of randomness of infinite sequences is independent of the choice of underlying optimal prefix-free machine, and coincides with other definitions of randomness like the definition given by Martin-Löf in [24]. The coincidence of the randomness notions resulting from various different approaches may be seen as evidence of a robust and natural theory.

1.2. Measures of relative randomness

Once a solid definition of initial segment complexity and randomness is in place, it is often desirable to have a way to compare two infinite binary sequences in this respect. One of the early measures of relative initial segment complexity was developed by Solovay in [30] especially for the computably enumerable (c.e.) reals. These are binary expansions of the real numbers in the unit interval which are limits of increasing computable sequences of rationals. The Solovay reducibility gave a formal way to compare c.e. reals with respect to the difficulty of getting good approximations to them. Solovay showed in [30] that the induced degree structure has a complete element which contains exactly the random c.e. reals. The Solovay degrees of c.e. reals where further studied in [19,18] (see [16, Section 9.5] for an overview).

Downey, Hirschfeldt and LaForte [17] introduced and studied a number of other measures of relative initial segment complexity that are not restricted to the c.e. reals. Most of them are extensions of the Solovay measure of relative complexity. For example, they defined $A \leqslant_K B$ if $\exists c\ \forall n\ (K(A\restriction_n) \leqslant K(B\restriction_n) + c)$; in other words, if the prefix-free complexity of each initial segment of A is bounded by the prefix-free complexity of the corresponding initial segment of B, modulo a constant. This reducibility, already implicit in [30], is a proper extension of the Solovay reducibility on the c.e. reals and was further studied in [31,26,27] with a special attention to random sequences and in [15,25,13], [6, Section 5] with more focus on local properties. A lot of these results refer to the degree structure that is induced by $\leqslant_K$, the K-degrees. A version of $\leqslant_K$ for plain Kolmogorov complexity was also defined in [17], which induces the structure of the C-degrees. In particular, $A \leqslant_C B$ if $\exists d\ \forall n\ (C(A\restriction_n) \leqslant C(B\restriction_n) + d)$.

1.3. Trivial initial segment complexity and Turing degrees

A string σ that has prefix-free complexity as low as the prefix-free complexity of the sequence of 0s of the same length may be regarded as trivial. Indeed, if we consider the prefix-free complexity of a string as a measure of the information that is coded in the string, in this case there is no information coded in the bits of the sequence. The infinite sequences whose initial segments have trivial prefix-free complexity are known as the K-trivial sequences. Formally, X is K-trivial if $\exists c\ \forall n\ (K(A\restriction_n) \leqslant K(n) + c)$, where we may identify $K(n)$ with $K(0^n)$. Surprisingly, there are noncomputable K-trivial sequences and this was already proved in [30]. Note that the K-trivial sequences are the contents of the least element in the K-degrees that were discussed in Section 1.2.

An interesting question that motivated a lot of later research was the following.

How much information can be encoded in an infinite binary sequence with very simple initial segments? (1.1)

In particular, is it possible to encode a Turing complete problem into a K-trivial sequence. A particularly simple construction of a noncomputable K-trivial c.e. set that was presented in [20] made this possibility plausible. However in the same paper it was shown that this is not the case. In particular, if an oracle A computes the halting problem then for each constant c there are initial segments σ of A such that $K(\sigma) > K(|\sigma|) + c$. The proof of this result was quite novel, and along with its extensions it became known as *the decanter method*. Hirschfeldt and Nies extended this method in [28] and showed that the amount of information that can be coded into K-trivial sequences is in fact quite limited. Quite interestingly, they also showed that K-triviality is downward closed with respect to Turing reductions. We refer to [16, Section 11.4] and [29, Section 5] for detailed presentations of the decanter method.

1.4. Motivation and results

In this paper we revisit question (1.1) by examining the possibility of coding considerable information in an infinite sequence with initial segments of very low but not necessarily trivial prefix-free complexity. We initially focus in the special case of c.e. sets, where Turing completeness provides a notion of maximality of information that can be coded. Hence we may ask the following question.

How low can the initial segment prefix-free complexity of
a Turing complete computably enumerable set be? (1.2)

How can we qualify the notion of 'low initial segment complexity' in question (1.2)? Note that modulo an additive constant, $K(n)$ is a lower bound on the complexity of the first n bits of any infinite sequence. Since the K-trivial sequences are ruled out by the result in [20], we turn our attention to sequences whose initial segment prefix-free complexity may deviate from the lower bound $K(n)$ but is still quite low. One way we could try to make this lowness condition precise is to look among sequences A such that $K(A\upharpoonright_n) - K(n)$ is bounded from above by a very slow growing function g, as it is shown in (1.3).

$$\exists c\, \forall n \quad \big(K(A\upharpoonright_n) \leqslant K(n) + g(n) + c\big). \tag{1.3}$$

The notion of 'slow growing' may be quantified through the arithmetical hierarchy of complexity. For example there are Δ^0_3 unbounded nondecreasing functions that are dominated by all Δ^0_2 functions with the same properties. In this sense, as the rate of growth of a function is reduced (but remains nontrivial) the arithmetical complexity of it increases. Let us first consider nondecreasing functions g. In [12,7] it was shown that if g is nondecreasing, unbounded and Δ^0_2 then there is a large uncountable collection of oracles A that satisfy (1.3). Hence a class that includes functions with these properties is not sufficiently restrictive for our purpose and we need to look in higher complexity classes. On the other hand in [15,7] it was shown that there are nondecreasing unbounded functions g in Δ^0_3 such that any set A that satisfies (1.3) is K-trivial. Moreover allowing functions that may decrease occasionally introduces similar problems. For example, it was shown in [13, Section 5] that there is a Δ^0_2 function g such that $\lim_n g(n) = \infty$ and any c.e. set which satisfies (1.3) is K-trivial. Hence condition (1.3) in combination with standard ways to quantify the rate of growth of the function g is not a fruitful way to formalize the notion of 'low nontrivial initial segment complexity'.

Another approach is to compare the initial segment complexity of a c.e. set with the complexity of other sets. Although this would not give us an absolute notion of low nontrivial complexity, an answer of the type 'lower than the complexity of any sequence with nontrivial complexity' to the question (1.2) would be definitive. The existence of minimal K-degrees is an open problem, but since this question refers to c.e. sets, such a positive answer is still not possible. Indeed, it was shown in [13] that there is a Δ^0_2 set B which is not K-trivial but every c.e. set with $A \leqslant_K B$ is K-trivial. In other words the initial segment complexity of B does not bound the complexity of any c.e. set with nontrivial initial segment complexity. This shows that the comparison needs to involve the complexities of c.e. sets and not arbitrary sequences. In this sense, the best possible answer to question (1.2) would be the existence of Turing complete c.e. sets with initial segment complexity strictly lower than the complexity of any given c.e. set that is not K-trivial. Our first result establishes exactly this.

Theorem 1.1. *Let A be a computably enumerable set which is not K-trivial. Then there exists a computably enumerable set B such that $B \equiv_T \emptyset'$ and $B <_K A$.*

The proof of Theorem 1.1 involves a very sparse coding of complete information, which produces a sequence with very simple initial segments, in the sense of the prefix-free complexity. A crucial part of the argument is the exploitation of the fact that the given set is c.e. and has nontrivial initial segment prefix-free complexity. In this sense Theorem 1.1 is dual to the main result of [20] that K-trivial sets are incomplete. More generally, the decanter method that was developed in [20] is a tool for exploiting the lack of complexity of a set in order to deduce additional properties. The method used in the proof of Theorem 1.1 is a tool for exploiting the complexity of a sequence (in combination with an effective approximation to it) in order to absorb the complexity of a coding procedure. In this sense the two methods are dual.

It is instructive to compare Theorem 1.1 with condition (1.3). If we wish to express our result in these terms we can set $g(n) = K(A \upharpoonright_n) - K(n)$. We note that $g(n)$ will be occasionally decreasing. In fact, it is well known that for every c.e. set A the $\liminf(K(A \upharpoonright_n) - K(n))$ is finite. In other words, c.e. sets are infinitely often K-trivial (see [13, Section 2] for a proof and a general discussion about infinitely often K-trivial sets). This observation gives some idea about the challenges of implementing the coding that is required in Theorem 1.1 as well as the qualification of the idea of 'low initial segment complexity' for c.e. sets.

Our second result is a generalization of Theorem 1.1 to any finite collection of c.e. sets with nontrivial initial segment prefix-free complexity. We state it and prove it for the special case of two c.e. sets since the more general version may be obtained trivially and effectively by an iterated application.

Theorem 1.2. *Let A, D be computably enumerable sets which are not K-trivial. Then there exists a computably enumerable set B such that $B <_K A$, $B <_K D$ and $B \equiv_T \emptyset'$.*

This extension has several applications that are discussed in Section 1.5, including the solution to an open question from [16, Section 11.12]. Moreover its proof goes considerably beyond a routine adaptation of the special case established in Theorem 1.1. As we elaborate in Section 4.2 the main obstacle is the lack of uniformity in the complexities of the given c.e. sets. This can be better understood if we recall that K-trivial sets are infinitely often K-trivial. In particular, as we discuss in Section 1.5, Theorem 1.2 shows that if two c.e. sets A, D are not K-trivial their initial segment complexity must rise simultaneously on some lengths. Hence despite the potential lack of uniformity in the oscillations of the complexity of two c.e. sets, there must be some uniformity on a local level i.e. places where the complexities $K(A \upharpoonright_n)$, $K(D \upharpoonright_n)$ deviate from $K(n)$ simultaneously.

Finally, we would like to mention another approach that has been used in the recent work by Ian Herbert with regard to reals of low initial segment complexity. Let K^A denote the prefix-free complexity with respect to oracle A. Herbert studied the class of reals A such that $K(n) \leqslant K^A(n) + f(n) + c$ for all n, where c is a constant and f is a slow growing function. This class is also a proper extension of the K-trivial reals.

1.5. Applications

The first application concerns various local structures of the K-degrees. The existence of minimal pairs of K-degrees was established in [15], where two Δ^0_4 sets forming a minimal pair in this structure were constructed. In [25] a minimal pair of Σ^0_2 sets was presented and in [13, Section 3] it was shown that there is a Σ^0_2 set that forms a minimal pair with all Σ^0_1 sets in the K-degrees. Theorem 1.2 implies that there are no minimal pairs in the structure of the K-degrees of c.e. sets. In particular, there is no pair of Σ^0_1 sets that form a minimal pair of K-degrees. This complements the existence results for minimal pairs in the K-degrees.

Downey and Hirschfeldt [16, Section 11.12] as well as Merkle and Stephan [25] asked if there is a pair of c.e. reals that form a minimal pair in the K-degrees. This question is particularly interesting since $\leqslant_K$ is often introduced as a generalization of the Solovay reducibility, which is the standard measure of relative randomness on the class of c.e. reals. We show that Theorem 1.2 answers this question in the negative. We need the following fact.

Lemma 1.3. *If A is a c.e. real such that $\emptyset <_K A$ then there exists a c.e. set B with $\emptyset <_K B \leqslant_K A$.*

Proof. Since A is a c.e. real, it has a computable approximation (A_s) according to which if $A(n)[s] = 1$ and $A(n)[s+1] = 0$ then there is some $i < n$ such that $A(i)[s] = 0$ and $A(i)[s+1] = 1$. A canonical encoding of the approximation (A_s) into a c.e. set B can be achieved based on the fact that for each n the value of $A(n)[s]$ can only change at most 2^n times during the stages s. The first bit of B encodes the oscillations to $A(0)$, the next 2 bits encode $A(1)$, the next 2^2 bits encode $A(2)$ and so on. In particular if $A(k)$ is encoded in the bits $(m, m + 2^k]$ of B, upon each change in $A(k)[s]$ during the stages s we enumerate into B the largest element of $(m, m + 2^k]$ that is not yet in B. In this way we have $A \equiv_T B$ and $B \leqslant_T A$ through a Turing reduction that uses at most n bits of A in the computation of n bits of B. Since K-triviality is a degree-theoretic property we have $\emptyset <_K B$ and by the basic properties of $\leqslant_K$ on the c.e. reals we also have $B \leqslant_K A$. □

By Theorem 1.2 and Lemma 1.3 we get the desired result about minimal pairs.

Corollary 1.4. *There are no minimal pairs in the K-degrees of c.e. reals.*

The separation of Solovay reducibility from $\leqslant_K$ on the c.e. reals was already achieved in [17], where a pair of c.e. reals A, B was constructed such that $A \leqslant_K B$ but A is not Solovay reducible to B. However these examples are artificial since they were obtained via diagonalization. A more natural separation would be obtained by an elementary difference in the corresponding degree structures of c.e. reals. This is provided by the existence of minimal pairs which occurs in the Solovay degrees of c.e. reals by [17] but not in the K-degrees of c.e. reals by Corollary 1.4. The same holds for c.e. sets according to Theorem 1.2.

Corollary 1.5. *The structures of the Solovay degrees and the K-degrees of computably enumerable reals are not elementarily equivalent. Moreover the same holds for the Solovay degrees and the K-degrees of computably enumerable sets.*

Merkle and Stephan showed in [25] that there exist two c.e. sets that from a minimal pair with respect to $\leqslant_C$. Hence Corollary 1.4 also provides an elementary difference between the C-degrees and the K-degrees of c.e. reals and c.e. sets.

Corollary 1.6. *The structures of the C-degrees and the K-degrees of c.e. reals are not elementarily equivalent. Moreover the same holds for the corresponding structures of c.e. sets.*

A final application of Theorem 1.2 concerns the following question.

$$\text{Is there a pair of sequences } X, Y \text{ which are not } K\text{-trivial but} \\ \min\{K(X\upharpoonright_n), K(Y\upharpoonright_n)\} - K(n) \text{ has a constant upper bound?} \tag{1.4}$$

Theorem 1.2 in combination with Lemma 1.3 answers (1.4) in the negative in the case where X, Y are required to be computably enumerable reals.

Corollary 1.7. *Suppose that A_i, $i < k$ is a finite collection of computably enumerable reals and none of them is K-trivial. Then for all c there exist n such that $K(A_i\upharpoonright_n) > K(n) + c$ for all $i < k$.*

We do not know the answer of (1.4) in general.

We conclude this section with a brief discussion on the topic of the initial segment complexity of c.e. sets. It would be interesting to locate elementary differences between the K-degrees of c.e. reals and the K-degrees of c.e. sets. This was done in [3] for the Solovay degrees by showing that there are no maximal elements in the Solovay degrees of c.e. sets. This line of research on the c.e. sets with respect to reducibilities that are sensitive to initial segment complexity measures was extended in [1]. The quest for elementary differences between K-degrees of c.e. reals and the K-degrees of c.e. sets lead to more general questions regarding the c.e. sets in the K-degrees and the C-degrees which were articulated in a research proposal that was presented (along with several related results) in [10]. An interesting product of this project is the following result from [8].

$$\textit{There is a maximum in the } K\textit{-degrees and the } C\textit{-degrees of c.e. sets.} \tag{1.5}$$

In other words, there are c.e. sets with maximum initial segment complexity. For the case of the plain complexity, a c.e. set A has maximum initial segment complexity if and only if the halting problem is reducible to it via a Turing oracle computation where the oracle use is bounded by a linear function. Moreover, it turned out that the above condition is equivalent to $\forall n,\ C(A\upharpoonright_n) \geqslant \log n - c$ which is a well known property that was studied in [2]. It follows from (1.5) and [3] (see [10] for more details) that the existence of a maximum degree is an elementary difference between the K-degrees of c.e. sets and the K-degrees of c.e. reals.

1.6. Related work on weak reducibilities

A method for exploiting the power of an oracle to achieve better compression of programs (along with a computable approximation to it) has been used in the study of another reducibility that is related to randomness and is called $\leqslant_{LK}$. We say that $X \leqslant_{LK} Y$ if $\exists c\ \forall\sigma\ (K^Y(\sigma) \leqslant K^X(\sigma) + c)$. In other words $X \leqslant_{LK} Y$ formalizes the notion that Y can achieve an overall compression of the strings that is at least as good as the compression achieved by X. Moreover by [21] it coincides with $X \leqslant_{LR} Y$ which denotes the relation that every random sequence relative to Y is also random relative to X. The degree structure that is induced by $X \leqslant_{LK} Y$ has a least element that turns out to contain exactly the K-trivial sequences. In [11] an argument was used that exploits the compression power of nontrivial c.e. sets in the study of the structure of c.e. sets under $\leqslant_{LK}$. A similar argument was used in [5] in order to show that every Δ^0_2 set with nontrivial compression power has uncountably many predecessors with respect to $\leqslant_{LK}$. In [4] this approach was further developed in order to exhibit elementary differences between various local structures of the LK degrees and the Turing degrees. We note that the arguments in these references work explicitly with $\leqslant_{LR}$ but can alternatively be implemented with the equivalent $\leqslant_{LK}$.

However there are some differences between $\leqslant_K$ and $\leqslant_{LK}$, the most important being that in $\leqslant_{LK}$ we usually work with oracle computations while in $\leqslant_K$ we only work with descriptions. It is quite remarkable that the triviality notion with respect to $\leqslant_K$ coincides with the triviality notion with respect to $\leqslant_{LK}$. As soon as we consider sequences of non-zero K-degrees or LK-degrees, the study of the two structures becomes less uniform. A comparison of the arguments about the non-existence of minimal pairs of K-degrees in this paper with the corresponding arguments in [4] that refer to the LK degrees shows that they follow a similar structure, yet various aspects need to be addressed individually. We discuss the high level view of these arguments in Section 5.

2. Preliminaries

The main tool in the proof of these theorems is a method of coding information into a set B that is constructed, while keeping its initial segment complexity below the complexity of a given c.e. set A that is not K-trivial. It is a method for exploiting the fact that a given set has a computable enumeration and nontrivial initial segment complexity, for the purpose of coding. In particular, it allows to meet the conflicting requirements $\emptyset' \leqslant_T B$ and $B \leqslant_K A$.

2.1. Prefix-free machines

For $B \leqslant_K A$ we need to build a prefix-free machine that witnesses the relation of the two complexities. Let U be the optimal prefix-free machine which underlies the prefix-free complexity K. Hence $K = K_U$. This machine is optimal in the sense that given any other prefix-free oracle machine M there is a constant c such that $K(\sigma) \leqslant K_M(\sigma) + c$ for all strings σ. The *weight* of a prefix-free set S of strings, denoted $\mathtt{wgt}(S)$, is defined to be the sum $\sum_{\sigma \in S} 2^{-|\sigma|}$. The *weight* of a prefix-free machine M is defined to be the weight of its domain and is denoted $\mathtt{wgt}(M)$. Without loss of generality we assume that $\mathtt{wgt}(U) < 2^{-2}$.

Prefix-free machines are most often built in terms of *request sets*. A request set L is a set of tuples $\langle \rho, \ell \rangle$ where ρ is a string and ℓ is a positive integer. A 'request' $\langle \rho, \ell \rangle$ represents the intention of describing ρ with a string of length ℓ. We define the *weight of the request* $\langle \rho, \ell \rangle$ to be $2^{-\ell}$. We say that L is a *bounded request set* if the sum of the weights of the requests in L is less than 1. This sum is the *weight of the request set* L and is denoted by $\mathtt{wgt}(L)$. The Kraft–Chaitin theorem (see e.g. [16, Section 2.6]) says that for every bounded request set L which is c.e., there exists a prefix-free machine M such that for each $\langle \rho, \ell \rangle \in L$ there exists a string τ of length ℓ such that $M(\tau) = \rho$. We freely use this method of construction without explicit reference to the Kraft–Chaitin theorem. A real number $0 \leqslant r < 1$ is called computably enumerable (c.e.) if it is the limit of a nondecreasing computable sequence of rational numbers. The binary strings are ordered first by length and then lexicographically.

2.2. Constructions in computability theory

This brief discussion is relevant to the constructions of Sections 3 and 4 and is likely to be handy to a reader who is not expert in such arguments. Constructions in computability theory typically take place in stages and involve various parameters. Given a parameter, we use the suffix '$[s]$' to denote the value of a parameter at the end of stage s. In the particular case of some sets A, B, D, $\emptyset'$ that are enumerated in the course of a construction, we simplify this notation by making 's' a subscript, thus obtaining A_s, B_s, D_s, $\emptyset'_s$ respectively. Parameters may have different values at different stages. Some parameters are defined in terms of the *given objects*, for example a fixed universal Turing machine (which is not in our control) or a given set that is mentioned in the hypothesis of the theorem that we want to prove. In the case of the construction of Section 3, the set A and the universal machine U (along with the Kolmogorov function $K = K_U$) are such parameters. We call these *parameters of the first type*. Some parameters are defined in terms of the objects that we construct, like a machine or a set. In the case of the construction of Section 3, machines M, N_i and the set B are such parameters. We call these *parameters of the second type*. Most constructions in computability theory are 'recursive', in the sense that each stage of the construction is defined in terms of the values of the parameters at the previous stages. Usually, we only need to refer to the values of the parameters at the present stage or the previous stage. The general rule is that at each stage of the construction we refer to the values that the parameters of first type have at this very stage, while we refer to the values that the parameters of second type have at (the end of) the previous stage. We follow this standard convention since the values of the parameters of second type at stage s are only determined at the end of stage s. This rule of thumb is helpful in understanding the formal description of the constructions of Sections 3 and 4.

2.3. Coding

The coding of $\emptyset'$ into B will be implemented through a system of movable markers $m_n, n \in \mathbb{N}$, where m_n represents position in the characteristic sequence of B in which we code the information of whether $n \in \emptyset'$. Hence we may call m_n the B-code of the possible event that consists of the enumeration of n into $\emptyset'$. The movement of the markers as well as the computable enumeration of B will take place in the stages of the enumeration of $\emptyset'$. In particular the value of m_n at stage s is denoted by $m_n[s]$. It is possible that $m_n[s]$ is undefined (in symbols, $m_n[s]\uparrow$) for some $n, s \in \mathbb{N}$. The movement of the markers satisfies the following standard properties:

(i) *Monotonicity on stages*: if $m_n[s]\downarrow, m_n[s+1]\downarrow$ then $m_n[s] \leqslant m_n[s+1]$;
(ii) *Monotonicity on indices*: if $m_n[s]\downarrow, m_{n+1}[s]\downarrow$ then $m_n[s] < m_{n+1}[s]$;
(iii) *Consistency*: if $m_n[s]\downarrow, m_n[t]\downarrow$, $m_n[s] \neq m_n[t]$ and $s < t$, then $m_n[s] \in B$;
(iv) *Convergence*: $\forall n\ \exists t, k\ \forall s\ (s > t \Rightarrow m_n[s]\downarrow = k)$;
(v) *Coding:* $\forall n\ (n \in \emptyset' \iff m_n \in B)$ where $m_n = \lim_s m_n[s]$.

Given a system of markers (m_n) with the above properties, we can compute $\emptyset'$ given B as follows. In order to decide if $n \in \emptyset'$, by clause (iii) we may use B in order to find a stage s such that either $n \in \emptyset'_s$ or $m_n[s] \notin B$. In the latter case we know by (v) that $n \notin \emptyset'$.

The essence of our method lies on the specific rules that determine the movement of the markers m_i. Intuitively, in order to maintain $B \leqslant_K A$ the markers are forced to move many times. Their convergence is a consequence of the failure to construct a machine demonstrating that A is K-trivial. Section 3 contains the formal argument.

It turns out that this type of sparse coding may be 'permitted' by any finite number of given c.e. sets that are not K-trivial. In particular, with some additional effort we can do the same coding into B while keeping its initial segment complexity below any two given c.e. sets A, C that are not K-trivial. Section 4 is devoted to the proof of this generalized result.

3. Proof of Theorem 1.1

Let A be a computably enumerable set which is not K-trivial. For the proof of Theorem 1.1 it suffices to construct a computably enumerable set B such that $B \equiv_T \emptyset'$ and $B \leqslant_K A$. This follows from the fact that the c.e. K-degrees are downward dense, i.e. for each c.e. set X such that $\emptyset <_K X$ there exists a c.e. set Y such that $\emptyset <_K Y <_K X$; see [6, Section 5].

3.1. Parameters and formal requirements of the construction

In order to make B Turing complete we will use a system of markers (m_i) as we discussed in Section 2.3. In order to establish $B \leqslant_K A$ it suffices to construct a prefix-free machine M such that

$$K_M(B \upharpoonright_n) \leqslant K(A \upharpoonright_n) \quad \text{for all } n \tag{3.1}$$

where K_M denotes the prefix-free complexity relative to machine M. Recall that K denotes the prefix-free complexity relative to a fixed universal prefix-free machine U such that $\mathtt{wgt}(U) < 2^{-2}$.

For each marker m_i we enumerate a prefix-free machine N_i during the construction. The purpose of N_i is to achieve $\forall n\ (K_{N_i}(A \upharpoonright_n) \leqslant K(n) + c_i)$ for some constant c_i. Since A is not K-trivial, this will ultimately fail. However this failure will help demonstrate that m_i converges: if m_i moves at stage $s+1$ (and all m_j, $j < i$ remain stable), the construction refreshes N_i so that $K_{N_i}(A \upharpoonright_n)[s+1] \leqslant K(n)[s+1] + c_i$ holds for the least n such that $K_{N_i}(A \upharpoonright_n)[s] > K(n)[s] + c_i$. The value of c_i may increase during the construction. This happens each time some m_j, $j < i$ moves. Such an event is often described as an 'injury' of m_i. In particular, if at some stage s marker m_k moves while m_j, $j < k$ remain constant this causes m_i, $i > k$ to be injured, which has the following consequences:

- for each $i > k$, markers m_i become undefined and N_i is reset;
- the values c_i, $i > k$ increase by 1.

To 'reset' machine N_i means to discard all of its computations thus starting to build a new machine. Each marker will only be injured finitely many times. We let $c_i[s]$ denote the value of c_i at stage s. At each stage s let $t_i[s]$ be defined as follows:

$$t_i[s] \text{ is the least number } t \text{ such that } {}_{N_i}(A_{s+1} \upharpoonright_t)[s] > K(t)[s+1] + c_i[s].$$

Each marker m_i has the incentive to move at some stage $s+1$ if it observes a set of descriptions of sufficient weight of segments of A_{s+1} that are longer than its current position. This weight is determined by the number (a sort of a 'threshold')

$$q_i[s] = 2^{-K(t_i[s])[s+1] - c_i[s]}. \tag{3.2}$$

The marker m_i requires attention at stage $s+1$ if $m_i[s]$ is defined, $m_i[s] \notin B_s$ and one of the following occurs:

(a) $i \in \emptyset'[s+1]$;
(b) $\sum_{m_i[s] < j \leqslant s} 2^{-K(A \upharpoonright_j)[s+1]} \geqslant q_i[s]$.

Note that if $m_i[s] \in B_s$ then we must have $i \in \emptyset'_s$. Hence in this case we do not have any direct reason to move m_i even if (b) holds, because there will not be any latter stage where we need to enumerate $m_i[s]$ into B (it is already in it). Of course some m_j with $j < i$ may move at a latter stage, in which case we will need to move m_i too, but this amounts to a typical finite injury aspect of the construction. Alternatively it is clear that we could have set up the construction so that the condition $m_i[s] \notin B_s$ is not present in the above definition of m_i 'requiring attention'.

For each $i \in \mathbb{N}$ we set $c_i[0] = i + 3$. At each stage $s+1$ the machines N_i will be adjusted according to changes of $K(n)$ for $n < t_i[s]$. This is done by running the subroutine (3.3) of the construction in Section 3.3. A *large* number at stage $s+1$ is one that is larger than any number that has been the value of any parameter in the construction up to stage s. Note that an enumeration of a number n into B only changes the segments $B \upharpoonright_i$ for $i > n$, since $B \upharpoonright_i$ consists of the first i bits of B, and the last of these is $B(i-1)$. This is why in the construction below, if we enumerate $m_n[s]$ into B, we only need to 'refresh' the descriptions of $B \upharpoonright_k$ for $k > m_n[s]$.

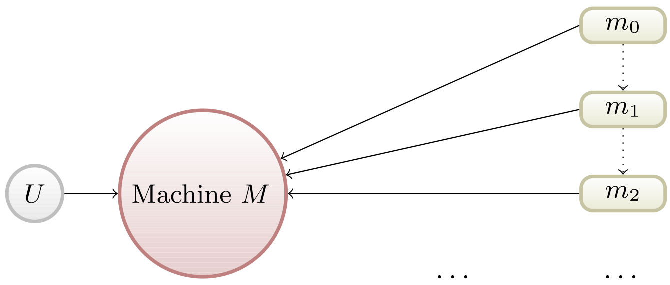

Fig. 1. Machine U adds computations in M, while the activity of the markers causes additional ('copies' of the previous) computations to be added in M.

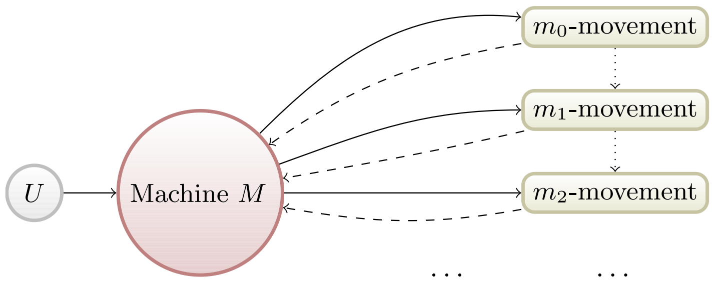

Fig. 2. Bounding the weight of M requires several movements of the markers. In this scheme, increase of the domain of M provokes the movement of the markers, which in turn induces additional enumerations of computations into M.

3.2. *Intuitive explanation of the dynamics in the construction*

Before we give the formal construction and verification, we present some intuitive and informal comments that illustrate the ideas behind the argument. The discussion consists of three parts: the description of the main conflict (the coding increases the size of M), the simplistic solution to the conflict (which unfortunately causes the coding procedure to diverge) and the final solution that makes all requirements satisfied. The arguments that we present informally here (especially the third part of the discussion) correspond to the formal part of the proof in Section 3.4.

3.2.1. *The main conflict: bounding M versus coding*

We use the family of (movable) markers (m_i) in order to ensure that $\emptyset' \leqslant_T B$ as we elaborated in Section 2.3. On the other hand, we continuously enumerate computations in the machine M according to (3.1). At each stage these computations ensure that the initial segment complexity of B (up to a certain length) with respect to M is not greater than the initial segment complexity of A. In this way, certain descriptions in the domain of U (which defines the Kolmogorov function $K = K_U$) induce the enumeration of M-descriptions of the same length.

The primary conflict in this argument is that the coding will cause certain numbers to be enumerated into B, and these changes of the approximation to B will increase the weight of the domain of M. This happens because for every change of the approximation to $B \upharpoonright_n$ we need to enumerate an additional description (corresponding to the new value of $B \upharpoonright_n$), possibly of the same length $K(A \upharpoonright_n)$ (if the approximation to A has remained the same). This standard conflict is depicted in Fig. 1 and will be present throughout the argument. Here the solid arrows indicate that enumeration of the codes into B cause the enumeration of additional weight in the domain of machine M. The dashed arrows between the markers indicate the finite injury effect that occurs amongst them, which was already indicated in Section 2.3.

A typical situation which illustrates this conflict is the following. At some stage s_0 we enumerate an M-description of $B \upharpoonright_{m_i}[s_0]$ of length $K(A \upharpoonright_{m_i})[s_0]$ according to (3.1). Let $k = m_i[s_0]$ and assume that $K(A \upharpoonright_k)[s_0] = K(A \upharpoonright_k)$. At some latter stage s_1, the number $i - 1$ enters $\emptyset'$ and we are forced to enumerate $m_{i-1}[s_1] < k$ into B. Subsequently, $i - 2$ enters $\emptyset'$ at some stage s_2 and provokes the enumeration of $m_{i-2}[s_2] < k$ into B. And so on, until $m_0[s_i] < k$ is enumerated into B at some stage s_i. During this 'cascade' of enumerations, the construction will be enumerating descriptions of the current approximation to $B \upharpoonright_k$. Hence the construction will enumerate at least k descriptions of the same length $K(A \upharpoonright_k)$. If $K(A \upharpoonright_k) = 3$ and $k = 2^4$ then clearly it is not possible to ensure that the weight of M is bounded.

3.2.2. *A step to the solution: additional movement of the markers*

We deal with the problem of bounding the weight of M by moving the markers (m_i), even before their index (or a smaller index) is enumerated in $\emptyset'$. Of course, this movement will obey the rules that we set out in Section 2.3. We will show that by setting appropriate movement rules for the markers, we can argue that the weight of M is bounded. Fig. 2 illustrates the dynamics of this construction (which is determined below). The features of the crude construction of Section 3.2.1 continue to apply here: computations in U provoke the enumeration of computations in M and the activity of the markers trigger the enumeration of additional M-descriptions (while 'injuring' the markers with larger index). Note that the additional movement of the markers that we enforce in the current form of the construction (see below) induce additional enumerations of computations in M. Fig. 2 also features arrows from M to the markers: these illustrate that the enumeration of M-computations sometimes triggers the movement of the markers. In the following we explain exactly how this construction works and why it ensures that the weight of M is bounded.

We describe a rule for moving the markers m_i which guarantees that the weight of M is bounded. Let $w_i[s] = \sum_{n>i} 2^{-K(n)[s]}$. The rule is that marker m_i will move at stage s if $w_i[s] > 2^{-r_i[s]-i}$, where $r_i[s]$ is the number of times that it has moved prior to stage s. Of course we also obey the movement rules that were set out in Section 2.3 (i.e. it also moves if i is newly enumerated in $\emptyset'$ or if some m_j with $j < i$ moves at stage s). In a standard fashion, we will only enumerate an M-description for some $B\restriction_n$ if all markers that occupy positions $< n$ 'appear to be stable', namely they have not moved since the last stage.

We can argue that in this case the weight of M is bounded is as follows. Every M-description (of an approximation to a segment of B) corresponds to a U-description (of an approximation to a segment of A), where U is the universal prefix-free machine. Indeed, every M-description τ (describing an initial segment of the current approximation to B) is issued according to a certain U-description σ (describing an initial segment of the same length of the current approximation to A). In this case we say that σ is *used* by τ. If σ has already been used by τ and it is later used by a different string τ', then we say that σ has been *reused*. As illustrated in the above 'cascade' example, a U-description may be *used* by several M-descriptions. In other words, the correspondence between the domains of U and M is not necessarily one-to-one. However every M-description always corresponds to a U-description of equal length. We will use the weight of U in order to bound the weight of M as follows. Let S_0 denote the strings in U that are *used* by at least one description in M during the construction. Clearly S_0 is a subset of the domain of U, so $\texttt{wgt}(S_0) < 2^{-k}$. Also let S_1 be the set of U-descriptions that are used by at least two M-descriptions. More generally, let S_k be the set of U-descriptions that are used by at least $k+1$ descriptions in M. Then $S_{k+1} \subseteq S_k$, so this family of sets can be illustrated as in Fig. 4. Note that if a string σ in S_k enters S_{k+1} then there is a unique marker m_i that 'causes' this change. Indeed, σ is *used* a one more time, which means that the approximation to the segment $B\restriction_n$ (where n is the length of the segment of the current approximation to A that σ describes) changes, due to the enumeration of (the current value of) a marker into B. Let m_i be this marker (if there are more than one markers with this property, we choose the one with the least index). We say that the entry of σ into S_{k+1} is due to the movement of m_i.

According to the correspondence between the domains of U and M that we discussed above, we can use

$$\texttt{wgt}(M) \leqslant \sum_k \texttt{wgt}(S_k)$$

to bound the weight of M. Note that each description in S_k is counted $k+1$ times in this sum as it belongs to all S_i, $i \leqslant k$. So it suffices to show that $\texttt{wgt}(S_k) < 2^{-k}$ for each k. Since $\texttt{wgt}(S_0) < 2^{-2}$ we also have $\texttt{wgt}(S_1) < 2^{-2}$. Let $k > 1$. Every entry of a string into S_k must have occurred due to the movement of a marker m_x. Moreover, it must have followed the entry of the string into S_{k-1}, which in turn must have occurred due to the movement of a marker m_y with $y > x$. Inductively, every string that enters S_k must be one of the strings that was previously enumerated in S_1 due to the movement of a marker m_z with $z \geqslant k-1$. Let S_k^z be the set of U-descriptions in S_k that enter S_1 due to the movement of marker m_z. Then $\texttt{wgt}(S_k) \leqslant \sum_{z \geqslant k-1} \texttt{wgt}(S_k^z)$ for each $k > 1$. Hence it remains to show that $\texttt{wgt}(S_1^z) < 2^{-z-2}$ for each z. This follows by the way we defined the movement of each m_z. The ith time it moves it is responsible for new M-descriptions of weight at most 2^{-z-i-3}. So overall $\texttt{wgt}(S_k^z)$ is bounded by $\sum_i 2^{-z-i-3} = 2^{-z-i-2}$.[2]

3.2.3. *Ensuring that the markers eventually halt*

The construction of Section 3.2.1 is based on the rule 'we move a marker m_i at stage s if the weight of the M descriptions of B that we will be called to re-describe if m_i is enumerated in B, is *large*'. In this case we interpreted 'large' as 'more than $2^{-r_i[s]-i}$ where $r_i[s]$ is the number of times that m_i has moved by stage s. We refer to $2^{-r_i[s]-i}$ as the *threshold for the movement of* m_i. Although this rule allows us to argue that the weight of M is bounded (which was the main conflict that was described in Section 3.2.1), it is not hard to see that it causes some markers to move indefinitely. Clearly this is not in line with the requirements that we set out in Section 2.3 (which are sufficient for deducing that $\emptyset' \leqslant_T B$), so we need to tune the movement rules for the markers in order to ensure that all requirements are satisfied. This adjustment will take into account the so-far-unused hypothesis that A is not K-trivial.

The idea here is to tie the movement of each marker m_i with the computations enumerated in an auxiliary machine N_i (constructed by us) which attempts to show that A is K-trivial. The enumerations into N_i will take place at stages where m_i moves and will keep the weight of N_i bounded. We need to define the threshold q_i for the movement of m_i in such a way that indefinite movement of m_i implies that N_i succeeds its purpose, i.e. $\forall n,\ K_{N_i}(A\restriction_n) \leqslant K(n) + c_i$ for some constant c_i. The threshold q_i is defined in such a way that the enumerations of computations into M, U, N_i are connected quantitatively. This is essential as the bounds on the weight of N_i, M are eventually reduced to a bound on the weight of U. The formal definitions of the parameters were given in the beginning of Section 3 and the formal construction is given in Section 3.3.

[2] There is a more direct way to argue that the weight of M is bounded, by assigning the additional M-descriptions that are issued to the individual markers that caused the relevant changes to the approximation to B. However this argument does not apply to the full construction. The argument we presented here will be used largely intact in the proof of Theorems 1.1 and 1.2.

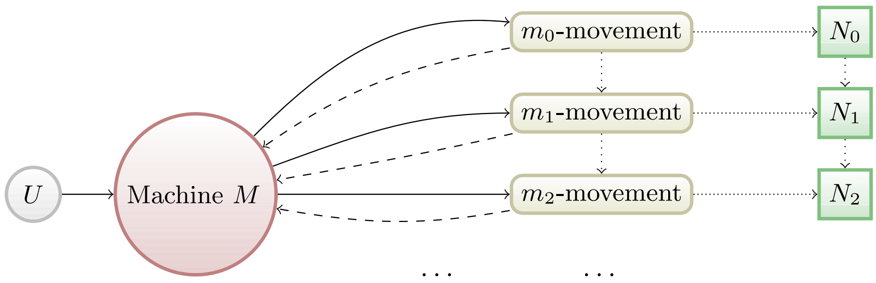

Fig. 3. Auxiliary machines N_i are fuelled by the cycles between the enumeration of M-computations and movement of the markers. They ensure that if A is not K-trivial these cycles have to stop, which implies that the markers eventually halt.

Fig. 3 illustrates the dynamics of this refined argument. The features that were discussed in Sections 3.2.1 and 3.2.2 continue to be present here. In addition, the cycle between the growth of M and the movement of m_i fuels the growth of an auxiliary machine N_i In particular, the movement of m_i not only adds to the weight of M but also triggers the enumeration of additional computations into N_i. The growth of N_i threatens to show that A is K-trivial, so it cannot continue indefinitely (and the same holds for the movement of m_i). Moreover, enumeration into N_i causes the 'injury' of N_j for all $j > i$. This means that in such cases we initialize N_j, deleting all of its computations and start with a new copy of it. This does not cause any problem to the verification of the argument, which is done inductively.

Let us conclude this informal discussion with a summary of the mechanics that is illustrated in Fig. 3 and the way it relates to the formal definitions of the parameters N_i, q_i, m_i, t_i, c_i. Every time m_i moves, it enumerates N_i descriptions (threatening to show that A is K-trivial, if these movements happen indefinitely). But to keep the weight of N_i bounded, we need to count it against the weight of the universal machine (via the parameter c_i). This is why the condition for movement is the inequality (b), which is based on the threshold q_i which in turn is defined in terms of t_i. Actually this is only one of the two reasons, the other being the use of (b) in bounding the weight of M (see below). The moment that the sum of descriptions of initial segments of A (for larger lengths than m_i) *hits* q_i, we may move m_i and add weight q_i to N_i. This is because the opponent (the universal machine U) showed us weight q_i (or even more) in descriptions of certain lengths. The next time that we enumerate in N_i, we justify the increase in the weight of N_i with *different* descriptions of U (indeed, descriptions that describe strings of different lengths, because each time we move m_i to large values).

This is just one side of the picture. The other side is the dynamics regarding the enumeration of machine M. Here, intuitively, the more we move the markers, the more we can save on the weight of M (and the more we add to the machines N_i) as we illustrated in Section 3.2.2. So it is a rather delicate balance that makes the construction work. This is crystallized by the inequality (b). Choosing the suitable threshold q_i for triggering movement of m_i is a crucial part of the argument, as it provides a quantitative connection between the movement of marker m_i with the enumeration of additional computations in M and in N_i. In the verification of the construction, the fact that machines N_i have bounded weight (as long as they are not 'injured') will be immediate. Then an argument along the lines of the argument of Section 3.2.2 shows that the weight of M is bounded. Finally, the convergence of the markers m_i follows inductively, using N_i and the hypothesis that A is not K-trivial.

3.3. Construction of B, M, N_i

At stage 0 place m_0 on 1. At stage $s+1$ run subroutine (3.3).

$$\begin{aligned}&\text{For each } i \leqslant s \text{ and each } k < t_i[s],\ \text{if } K(k)[s+1] < K(k)[s] \text{ then}\\ &\text{enumerate an } N_i\text{-description of } A_{s+1} \restriction_k \text{ of length } K(k)[s+1] + c_i[s].\end{aligned} \tag{3.3}$$

Let z be the least number $< s$ such that $K_M(B_s \restriction_z)[s] > K(A \restriction_z)[s+1]$. If none of the currently defined markers requires attention, let n be the least number such that $m_n[s]$ is undefined, and

- if $n < z$ place m_n on the least *large* number;
- if $z \leqslant n$ enumerate an M-description of $B_s \restriction_z$ of length $K(A \restriction_z)[s+1]$;
- end this stage.

Otherwise let n be the least number such that m_n requires attention, put $m_n[s]$ into B, let $m_n[s+1]$ be a *large* number and for each $k < s$ such that $k > m_n[s]$ and $K_M(B \restriction_k)[s] \leqslant K(A \restriction_k)[s+1]$ enumerate an M-description of $B_{s+1} \restriction_k$ of length $K(A \restriction_k)[s+1]$. Moreover for each $j > n$ declare $m_j[s+1]$ undefined, reset N_j and set $c_j[s+1] = c_j[s] + 1$. If clause (b) of Section 3.1 applies,

$$\text{enumerate an } N_n\text{-description of } A_{s+1} \restriction_{t_n[s]} \text{ of length } K\big(t_n[s]\big)[s+1] + c_n[s]. \tag{3.4}$$

End this stage.

3.4. Verification

Before we start with the main part of the verification, we make two preliminary observations that follow directly from the construction. The first one concerns the relationship between the values of parameters t_i and m_i during the stages of the construction. When m_i is first defined at some stage s it takes a *large* value so $t_i[s] < m_i[s]$. Moreover t_i can only increase when N_i computations are enumerated on strings of length t_i, which happens only when m_i moves. Also if $A\restriction_{t_i}$ changes, by the definition of t_i (since A is c.e. and N_i is built by us) it follows that t_i decreases as soon as m_i moves. Hence by induction we have (3.5).

$$\text{For all } i, s,\ \text{ if } m_i[s] \text{ is defined then } t_i[s] < m_i[s]. \tag{3.5}$$

The second observation is a conditional monotonicity on the values of t_i during the stages. If $K(k)$ decreases at some stage $s+1$ for some $k < t_i[s]$, subroutine (3.3) will ensure that $K_{N_i}(A\restriction_k)[s+1] \leqslant K(k)[s+1] + c_i[s]$. Hence t_i may only decrease at $s+1$ if $A_{s+1}\restriction_{t_i[s]} \neq A_s\restriction_{t_i[s]}$.

$$\text{If } A_s\restriction_{t_i[s]} = A_{s+1}\restriction_{t_i[s]} \text{ then } t_i[s] \leqslant t_i[s+1]. \tag{3.6}$$

We are now ready to proceed with the first step of the verification, which is to show that for each i there is a machine N_i as prescribed in the construction. Recall that the construction may reset N_i. This means that for each i we have many versions of N_i. A new version of N_i is placed when the latest one is reset. In that case all the previous versions of N_i are no more relevant in the rest of the construction (in particular, they do not change anymore). When we refer to N_i we refer to an arbitrary version of it and the interval of stages from its introduction until (if ever) it is reset (before its introduction it is empty and after it is reset it remains constant).

Lemma 3.1. *For each i the weight of the requests in N_i is bounded.*

Proof. We consider an arbitrary version of N_i and it suffices to prove the lemma for the interval of stages $[b_0, b_1)$ where b_0 is the stage where it was introduced and b_1 is the stage when it was reset (so b_1 may be ∞). By the construction, all $m_j[s]$, $j < i$ and $c_i[s]$ remain stable during the stages s in $[b_0, b_1)$. So in the following we may refer to $c_i[s]$ by c_i.

A request is enumerated into N_i either by subroutine (3.3) or due to the movement of a marker m_i. We will bound each part of the N_i requests separately and then add the bounds. First, we consider the requests that are enumerated by subroutine (3.3). Each such request is associated with a unique pair (k, s) such that $K(k)[s+1] < K(k)[s]$. Moreover such a request has weight $K(k)[s+1] + c_i$. It follows that the total weight of these requests is bounded by $2^{-c_i} \cdot \mathtt{wgt}(U)$, which is at most 2^{-2}.

The only other way that an enumeration into N_i may be requested is when a marker m_i requires attention at some stage $s+1$. Recall from Section 3.1 the conditions that need to be met in order for m_i to require attention at stage $s+1$, and in particular clause (b). It follows that in this case the marker moves to a *large* value and the weight of the request is $q_i[s] \leqslant \sum_{m_i[s] < j \leqslant s} 2^{-K(A\restriction_j)[s+1]}$. Let (s_j) be the sequence of stages in $[b_0, b_1)$ where m_i moves. Then the weight of the requests that are enumerated in N_i in this way (via the movement of m_i) is bounded by

$$\sum_j \left(\sum_{m_i[s_j] < j \leqslant m_i[s_{j+1}]} 2^{-K(A\restriction_j)[s_j]} \right) \leqslant \mathtt{wgt}(U).$$

Hence the weight of the requests that are enumerated in N_i in the latter manner (i.e. via the movement of m_i) is bounded by 2^{-2}. Since we established the same bound for the weight of the requests that are enumerated in N_i via the first manner (i.e. via (3.3)) it follows that $\mathtt{wgt}(N_i) \leqslant 2^{-2} + 2^{-2} = 2^{-1}$. $\square$

The following lemma is essential in showing that $\emptyset' \leqslant_T B$. The proof of it, uses the fact that each N_i is a prefix-free machine, which was established in Lemma 3.1.

Lemma 3.2. *For each i, marker m_i is defined, injured only finitely many times and reaches a limit.*

Proof. We argue by induction on i. In order to conclude the induction step and the proof of this lemma, it suffices to show that m_{i+1} will reach a limit. By the induction hypothesis, m_{i+1} stops being injured after stage some s_0. Hence c_{i+1} reaches a limit at s_0. Since A is not K-trivial there is some least n such that $K_{N_{i+1}}(A\restriction_n) > K(n) + c_{i+1}$. Let $s_1 > s_0$ be a stage where the approximations to $A\restriction_n$ and $K(j)$, $j \leqslant n$ have settled. If marker m_{i+1} moved after stage s_1 the construction would enumerate an N_{i+1}-description of $A\restriction_n$ of length $K(n) + c_{i+1}[s_1]$ which contradicts the choice of n. Hence m_{i+1} reaches a limit by stage s_1 and this concludes the induction step and the proof. $\square$

We define (S_i) exactly as in the discussion of Section 3.2.2. Let S_0 be the set of strings in the domain of U that are used at least one time. More generally for each $k \geqslant 0$ we let S_k be the set of descriptions in the domain of U which are used

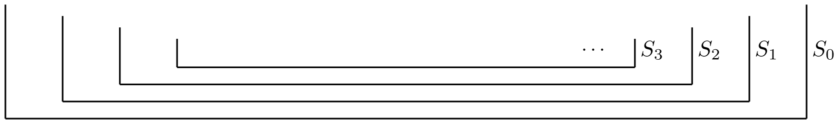

Fig. 4. Infinite nested decanter model.

at least $k+1$ times. Note that $S_{i+1} \subseteq S_i$ for each i. According to the correspondence between the domains of U and M, a string σ in the domain of U that is used k times incurs weight $k \cdot 2^{-|\sigma|}$ to the domain of M. Hence (3.7).

$$\mathtt{wgt}(M) \leqslant \sum_k \mathtt{wgt}(S_k). \tag{3.7}$$

Note that in the above sum each description in S_k is counted $k+1$ times, since it is also a member of S_j for $j<k$. A U-description σ is called *active* at stage s if $U(\sigma)[s] \subseteq A_s$. By the construction, all descriptions that enter S_1 at some stage s are active at that stage. More generally, at any given stage s, only strings that are active at stage s may move from S_k to S_{k+1} at stage s.

The sets S_k may be visualized as the nested containers of the infinite decanter model of Fig. 4. As the figure indicates, descriptions that are currently in container S_k may enter container S_{k+1} while they continue to be members of S_k. In particular, once a description enters a container it will remain in that container indefinitely. If at some stage a marker m_i moves (while m_j, $j<i$ remain stable), some strings of S_k enter S_{k+1} for various $k \in \mathbb{N}$. Indeed, when m_i moves it enumerates its former value into B. This action changes the approximation to B, which in turn causes some descriptions to be used an additional time. By the definition of the sets (S_k), this means that some of the strings in some containers enter the next container. In this case we may say that these strings were reused by m_i (since they were used an additional time). In order to calculate a suitable upper bound for each $\mathtt{wgt}(S_k)$ we need Lemma 3.3.

Lemma 3.3. *If during the interval of stages $[s,r]$ a marker m_n is not injured and $n \notin \emptyset'_r$ then the weight of the strings that m_n reuses during this interval which remain active at stage r is at most $2^{-c_n[s]}$.*

Proof. Note that by the assumption, parameter c_n remains constant throughout the interval $[s,r]$. Suppose that m_n moves at stage $x+1 \in [s,r]$, after requiring attention. Then since the markers always move to *large* values it follows that m_n did not move at stage x. Recall the definition of when m_n requires attention, which was given in Section 3.1 (the clauses (a) and (b)). Since m_n did not move at stage x, it follows that it did not require attention at that stage and by clause (b) of Section 3.1 we get

$$\sum_{j>m_n[x]} 2^{-K(A\restriction_j)[x]} < q_n[x]. \tag{3.8}$$

Note that when m_n moves at stage $x+1$, the weight of the U-descriptions that it reuses is at most $\sum_{j>m_n[x]} 2^{-K(A\restriction_j)[x]}$ (and not $\sum_{j>m_n[x]} 2^{-K(A\restriction_j)[x+1]}$). This happens because the construction first moves marker m_n and then enumerates additional computations in M. In other words, the descriptions that m_n reuses at $x+1$ correspond to M-computations that occurred in the previous stages, not the M-computations that may occur by the end of stage $x+1$. Hence by (3.8), the weight of the U-descriptions that are reused by m_n at stage $x+1$ are bounded by $q_n[x]$, which is $2^{-K(t_n)[x]-c_n[x]}$.

Now let us consider the overall effect of the movement of m_n in the interval of stages $[s,r]$. If at least one of the descriptions in U that m_n reused at some stage $x+1 \in [s,r]$ continues to be active at stage r, then $A_{x+1}\restriction_{m_n[x]+1} = A_r\restriction_{m_n[x]+1}$. By (3.5), under the same assumptions this implies

$$A_{x+1}\restriction_{t_n[x]+1} = A_r\restriction_{t_n[x]+1}.$$

By (3.4) of the construction (i.e. the enumeration of a computation in N_i upon the movement of a marker), since at stage $x+1$ the marker m_n moved, we have $t_n[x+1] = t_n[x]+1$. Hence by (3.6) we get that

$$t_n[y] \geqslant t_n[x+1] > t_n[x] \quad \text{for all } y \in [x+1,r].$$

The above observation along with the bound that we established in the previous paragraph on the weight of the U-descriptions that are reused by m_n at a stage in $[s,r]$, imply the following fact.

> During the stages in $[s,r]$ the weight of the descriptions in U that m_n reuses and remain active at stage r, is each time bounded by $2^{-K(t_n)-c_n}$, where t_n is larger and larger and c_n remains equal to $c_n[s]$

(while the Kolmogorov function follows its usual approximation). Formally, if y_j, $j<k$ are the stages in $[s,r]$ where marker m_n moves, we have $t_n[y_j] < t_n[y_{j+1}]$ and the weight of U-descriptions that m_n reuses at stage y_j and remain active at stage

r is at most $2^{-K(t_n)[y_j]-c_n[s]}$. So the total weight of the U-descriptions that m_n uses during the stages in $[s,r]$ and which remain active at stage r is less than

$$\sum_i 2^{-K(i)-c_n[s]}.$$

Since the above sum is bounded by $2^{-c_n[s]}$, this concludes the proof. □

Lemma 3.4. *The weight of the requests that are enumerated in M is finite.*

Proof. According to the correspondence between the domains of U and M that we discussed, we can use (3.7) to bound the weight of M. Note that each description in S_k is counted $k+1$ times in this sum as it belongs to all S_i, $i \leqslant k$. Since only strings in the domain of U are used, $\mathtt{wgt}(S_0) < 2^{-2}$. So it suffices to show that

$$\mathtt{wgt}(S_k) < 2^{-k-1} \quad \text{for each } k > 0. \tag{3.9}$$

Since $S_1 \subseteq S_0$, condition (3.9) holds for $k=1$. Let $k>1$. Every entry of a string into S_k is due to a marker m_x which reused it when it was already in S_{k-1}. Since $k>1$, this string entered S_{k-1} due to another marker m_y with $y > x \geqslant 0$. Inductively, that string entered S_1 due to a marker m_z with $z \geqslant k-1$. Fix z, and let S_k^z contain the strings in S_k that entered S_1 due to marker m_z. Then $S_k = \bigcup_{z \geqslant k-1} S_k^z$ and $S_{k+1}^z \subseteq S_k^z$ for each $k>1$. Hence

$$\mathtt{wgt}(S_k) \leqslant \sum_{z \geqslant k-1} \mathtt{wgt}\big(S_k^z\big) \quad \text{for each } k > 1.$$

So in order to prove (3.9) for $k>1$ it suffices to show that

$$\mathtt{wgt}\big(S_k^z\big) < 2^{-z-2} \quad \text{for each } z \geqslant 0. \tag{3.10}$$

Let (s_i) be the increasing sequence of stages where m_z is injured. Note that at this point we do not assume that (s_j) is a finite sequence. We may count the weight of S_k^z by counting the weight of the bunches of descriptions that moved to S_1^z and then moved to S_2^z (necessarily by some m_j with $j<z$). This is justified because every description that enters S_k^z must have passed from S_2^z first.

Since the movement of a marker m_i injures all m_j, $j>i$, the only stages were strings move from S_1^z to S_2^z are the stages (s_i). Moreover since only active strings move from S_1^z to S_2^z at stage s_i, according to Lemma 3.3 (applied to the intervals $[s_i+1, s_{i+1}-1]$) their weight is bounded by $2^{-c_z[s_i-1]}$. So the weight of the strings that enter S_2^z from S_1^z is bounded above by $\sum_j 2^{-c_z[s_j-1]}$. Since $c_z[s_{j+1}-1] = c_z[s_j] > c_z[s_j-1]$ for all j, this weight is bounded by $\sum_j 2^{-c_z[0]-j} = 2^{-c_z[0]+1}$. Since $c_z[0] = z+3$ this bound becomes 2^{-z-2}, which establishes (3.10). □

We conclude with the proof that (3.1) is met.

Lemma 3.5. *For each i there is an M-description of $B\restriction_i$ of length $\leqslant K(A\restriction_i)$.*

Proof. We argue by induction on i. Suppose that the lemma holds for $i \in \mathbb{N}$. Then by Lemma 3.2, there is some stage s_0 at which the approximations to $A\restriction_{i+1}$, $B\restriction_{i+1}$, $K(A\restriction_{i+1})$, $K_M(B\restriction_i)$ and m_i have settled and $K_M(B\restriction_i)[s_0] \leqslant K(A\restriction_i)[s_0]$. If $K_M(B\restriction_{i+1})[s_0] > K(A\restriction_{i+1})[s_0]$ the construction at stage s_0+1 will enumerate an M-computation that describes $B\restriction_{i+1}$ with a string of length $K(A\restriction_{i+1})$. □

By Lemma 3.2 and the construction we get that the movement of the markers satisfies properties (i)–(v) of Section 2. Hence $\emptyset' \leqslant_T B$. We conclude the proof of Theorem 1.1 by observing that (3.1) is met. By Lemma 3.5 the construction enumerates the required requests in M which ask for a description of $B\restriction_i$ with a string of length at most $K(A\restriction_i)$, for each i. On the other hand Lemma 3.4 establishes that this request set corresponds to a prefix-free machine, via the Kraft–Chaitin lemma. Hence (3.1) is met, which concludes the verification of the construction and the proof of Theorem 1.1.

4. Proof of Theorem 1.2

Let A, D be two computably enumerable sets which are not K-trivial. For the proof of Theorem 1.2 it suffices to construct a computably enumerable set B such that $B \leqslant_K A$, $B \leqslant_K D$ and $B \equiv_T \emptyset'$. This follows from the downward density of the c.e. K-degrees as we discussed in Section 3. The coding of $\emptyset'$ into B will be done via the markers (m_i) and the relations $B \leqslant_K A$, $B \leqslant_K D$ will be achieved with the construction of two prefix-free machines M_a, M_d respectively such that

$$K_{M_a}(B\restriction_n) \leqslant K(A\restriction_n) \quad \text{and} \quad K_{M_d}(B\restriction_n) \leqslant K(D\restriction_n) \quad \text{for all } n. \tag{4.1}$$

4.1. Merging two constructions

The basic plan of the construction of M_a, M_d is to merge a construction for M_a of the type that was given in Section 3 with a construction for M_d of the same type. Note that we will have a single set of markers m_i but their movement will be stimulated by both requirements in (4.1). We will use the same set of constants c_i for both A and D, since their values only depend on the movement of the markers on B. However for each i we have N_i^a, N_i^d instead of N_i. At each stage s we let $t_i^a[s]$ be the least number x such that $K_{N_i^a}(A_{s+1}\upharpoonright_x)[s] > K(x)[s+1]+c_i[s]$ and we let $t_i^d[s]$ be the least number y such that $K_{N_i^d}(D_{s+1}\upharpoonright_y)[s] > K(y)[s+1]+c_i[s]$. For each $i\in\mathbb{N}$ we set $c_i[0]=i+4$. The universal machine U and the notion of injury of a marker remains the same. In particular, if at some stage s marker m_k moves while m_j, $j<k$ remain constant this causes m_i, $i>k$ to be injured. This means that m_i, $i>k$ become undefined and the values of c_i, $i>k$ increase by 1.

At each stage $s+1$ the machines N_i^a, N_i^d will be adjusted according to changes of $K(n)$ for $n<t_i[s]$. This is done by running subroutine (4.2) (which is analogous to (3.3) of the argument in Section 3). We define

$$q_i^a[s]=2^{-K(t_i^a)[s]-c_i[s]} \quad\text{and}\quad q_i^d[s]=2^{-K(t_i^d)[s]-c_i[s]}.$$

The thresholds $q_i^a[s]$, $q_i^d[s]$ play a similar role as $q_i[s]$ in the argument of Section 3. However since $K(A\upharpoonright_n)$ and $K(D\upharpoonright_n)$ may differ for various n, the definition of a marker requiring attention will be modified, as we elaborate in Section 4.2.

4.2. Lack of uniformity and solution

The main issue that we have to deal with when we merge two constructions of the type used in Section 3 which depend on different c.e. sets A, D is that the thresholds q_i^a, q_i^d that correspond to some marker m_i may have different values. Hence the marker may be motivated to move by M_a but not by M_d. This lack of uniformity has an impact in the calculations of the weight of the machines N_i^a, N_i^d, which in turn affects a verification along the lines of Section 3.4.

The solution to this obstacle is to use the additional parameters p_i^a, p_i^d which record the weight of the M_a or M_d descriptions respectively that were reissued when only M_d or M_a respectively motivated the movement of marker m_i. For example, at some stage $s+1$ we may have $\sum_{m_i[s]<j\leqslant s}2^{-K(A\upharpoonright_j)[s]}\geqslant q_i^a[s]$ but this may not hold for D in place of A and $q_i^d[s]$ in place of $q_i^a[s]$. This means that at this stage M_a requires the movement of m_i but M_d does not. At such a stage we will move m_i for the sake of M_a, also enumerating an N_i^a-description of $t_i^a[s]$ of length $K(t_i^a)[s]+c_i[s]$. However an enumeration of an N_i^d-description of $t_i^d[s]$ of length $K(t_i^d)[s]+c_i[s]$ is not justified and will not take place. Instead, we will store the value $\sum_{m_i[s]<j\leqslant s}2^{-K(D\upharpoonright_j)[s]}$ into $p_i^d[s+1]$, which is the weight of the M_d descriptions we need to reuse due to the movement of m_i at stage $s+1$. At the next stage the threshold in the condition for the movement of m_i for the sake of D will be $q_i^d[s+1]-p_i^d[s+1]$. As long as m_i moves for the sake of M_a the value of p_i^d will keep on increasing, recording the weight of the M_d descriptions that we need to pay due to the M_a-motivated movements of m_i. When m_i moves for the sake of M_d, the value of p_i^d will drop to 0 and the enumeration into N_i^d will be justified. The same holds symmetrically for A with q_i^a and p_i^a. With this amendment a combined construction can be verified along the lines of the argument of Section 3.4.

According to the above motivation, we say that the marker m_i requires attention at stage $s+1$ if $m_i[s]$ is defined, $m_i[s]\notin B_s$ and one of the following occurs:

(a) $i\in\emptyset'_{s+1}$;
(b) $\sum_{m_i[s]<j\leqslant s}2^{-K(A\upharpoonright_j)[s]}\geqslant q_i^a[s]-p_i^a[s]$;
(c) $\sum_{m_i[s]<j\leqslant s}2^{-K(D\upharpoonright_j)[s]}\geqslant q_i^d[s]-p_i^d[s]$.

The condition $m_i[s]\notin B_s$ in the above definition can be justified as the same condition was justified in the construction of Section 3.3 (see the discussion in the end of Section 3.1). The definition of a *large* number is as in the argument of Section 3. Recall that the parameters q_i^a, q_i^d are defined in terms of the given sets A, D (and the universal machine U) while the parameters p_i^a, p_i^d are defined dynamically within the construction. We define $p_i^a[0]=p_i^d[0]=0$.

4.3. Construction of B, M_a, M_d, N_i^a, N_i^d

At stage 0 place m_0 on 1. At stage $s+1$ run subroutine (4.2) for $(X,x)\in\{(A,a),(D,d)\}$.

$$\begin{array}{l}\text{For each } i\leqslant s,\ \text{if } K(k)[s+1]<K(k)[s] \text{ for some } k<t_i^x[s] \text{ then enumerate}\\ \text{an } N_i^x\text{-description of } X_{s+1}\upharpoonright_k \text{ of length } K(k)[s+1]+c_i[s].\end{array} \tag{4.2}$$

Let z_a, z_d be the least numbers $\leqslant s$ such that

$$K_{M_a}(B\upharpoonright_{z_a})[s]>K(A\upharpoonright_{z_a})[s+1] \quad\text{and}\quad K_{M_d}(B\upharpoonright_{z_d})[s]>K(D\upharpoonright_{z_d})[s+1].$$

If none of the currently defined markers requires attention, let n be the largest number such that $m_n[s]$ is undefined, and

- if $n < z_a$ and $n < z_d$, place m_n on the least *large* number;
- otherwise enumerate an M_a-description of $B_s \restriction_{z_a}$ of length $K(A \restriction_{z_a})[s+1]$ and an M_d-description of $B_s \restriction_{z_d}$ of length $K(D \restriction_{z_d})[s+1]$;
- end this stage.

Otherwise let n be the least number such that m_n requires attention, put $m_n[s]$ into B, let $m_n[s+1]$ be a *large* number and for each $k < s$ with $k > m_n[s]$

- if $K_{M_a}(B \restriction_k)[s] \leqslant K(A \restriction_k)[s+1]$ enumerate an M_a description of $B_{s+1} \restriction_k$ of length $K(A \restriction_k)[s+1]$;
- if $K_{M_d}(B \restriction_k)[s] \leqslant K(D \restriction_k)[s+1]$ enumerate an M_d-description of $B_{s+1} \restriction_k$ of length $K(D \restriction_k)[s+1]$.

Moreover for each $j > n$

- declare $m_j[s+1]$ undefined and reset machines N_j^a, N_j^d;
- set $c_j[s+1] = c_i[s] + 1$ and $p_j^a[s+1] = p_j^d[s+1] = 0$.

Finally for $(X, x) \in \{(A, a), (D, d)\}$ consider the action

(∗) enumerate an N_n^x-description of $X \restriction_{t_n^x[s]} [s+1]$ of length $K(t_n^x[s])[s+1] + c_n[s]$.

and do the following, according to whether clauses (b), (c) of Section 4.2 hold:

- If (b), (c) hold, for $(X, x) \in \{(A, a), (D, d)\}$ execute (∗) and set $p_n^x[s+1] = 0$;
- otherwise, if (b) holds, execute (∗) for $(X, x) = (A, a)$ and set $p_n^a[s+1] = 0$, $p_n^d[s+1] = p_n^d[s] + \sum_{m_n[s]<j\leqslant s} 2^{-K(D\restriction_j)[s+1]}$;
- otherwise, if (a) holds execute (∗) for $(X, x) = (D, d)$ and set $p_n^d[s+1] = 0$, $p_n^a[s+1] = p_n^a[s] + \sum_{m_n[s]<j\leqslant s} 2^{-K(A\restriction_j)[s+1]}$.

End this stage.

4.4. *Verification*

As in the verification of Section 3 we have (4.3).

$$\text{For all } i, s, \text{ if } m_i[s] \text{ is defined then } t_i^a[s] < m_i[s] \text{ and } t_i^d[s] < m_i[s]. \tag{4.3}$$

Moreover the justification of (3.6) also applies to (4.4).

$$\begin{aligned} &\text{If } A_s \restriction_{t_i^a[s]} = A_{s+1} \restriction_{t_i^a[s]} \text{ then } t_i^a[s] \leqslant t_i^a[s+1], \\ &\text{If } D_s \restriction_{t_i^d[s]} = D_{s+1} \restriction_{t_i^d[s]} \text{ then } t_i^d[s] \leqslant t_i^d[s+1]. \end{aligned} \tag{4.4}$$

Next, we show that for each i there are machines N_i^a, N_i^d as prescribed in the construction. The proof of this fact is slightly more involved than the corresponding fact in the argument of Section 3 due to the amendment that was discussed in Section 4.2.

Lemma 4.1. *For each i the weights of the requests in N_i^a and N_i^d are bounded.*

Proof. Let $(X, x) \in \{(A, a), (D, d)\}$. As in Section 3, each machine N_i^x is valid only as long as m_i is not injured. In this way we have many copies of N_i^x and it suffices to argue about a fixed version of it (which is relevant only in an interval of stages where m_i is not injured).

A request is enumerated into N_i^x either by subroutine (4.2) or due to the movement of a marker m_i. We will bound each part of the N_i requests separately and then add the two bounds. First, we consider the requests that are enumerated by subroutine (4.2). Each such request is associated with a unique pair (k, s) such that $K(k)[s+1] < K(k)[s]$. Moreover such a request has weight $K(k)[s+1] + c_i$. It follows that the total weight of these requests is bounded by $2^{-c_i} \cdot \mathtt{wgt}(U)$, which is at most 2^{-2}.

Let (s_j) be the sequence of stages where m_i moves, inside an interval J of stages s where m_i is not injured and $i \notin \emptyset'_s$. Moreover let $I_j = (m_i[s_j], m_i[s_{j+1}]]$ be the interval that marker m_i crosses when it moves at stage s_j. For each j let

$$x_j = \sum_{n\in I_j} 2^{-K(X\restriction_n)[s_j-1]}.$$

If $x = a$ let (e_x) be clause (b) of Section 4.2 and if $x = d$ let (e_x) be clause (c) of Section 4.2. Let (k_j^x) be the monotone sequence of those numbers k such that at stage s_k marker m_i moves due to clause (e_x). According to the construction and the way we increase p_i^x, the weight of the N_i^x descriptions that is enumerated at stage $s_{k_j^x}$ is bounded by the sum of x_v for all $v \in [k_j^x, k_{j+1}^x)$. Let us explain this later fact in more detail. In-between the stages $s_{k_j^x}$ and $s_{k_{j+1}^x}$ the weight of the descriptions that are enumerated in N_i^x is bounded by the increase in p_i^x, which in turn corresponds to a limited part of x_j. At stage $s_{k_{j+1}^x}$ the parameter p_i^x is set to 0 and the overall weight of N_i^x descriptions that were issued since stage $s_{k_j^x}$ is bounded by x_j.

In this way the different weights of descriptions that are enumerated in N_i^x at the key stages $s_{k_j^x}$ correspond to disjoint parts of the domain of the universal machine U, of larger or equal weight. It follows that the total weight that is enumerated in N_i^x due to movements of m_i during the construction is bounded by 2^{-2}. If we combine this with the weight that is added by applications of (4.2) we get $\mathtt{wgt}(N_i^x) < 2^{-1}$. □

The following fact is crucial in showing that $\emptyset' \leqslant_T B$. Its proof uses the fact that each (version of) N_i^a, N_i^d is a prefix-free machine, which was established in Lemma 4.1. It is instructive to compare this proof with the proof of the analogous Lemma 3.5 of Section 3, and identify the way that the non-uniformity (i.e. the fact that we have to deal with two given sets A, D, and construct two corresponding machines M_a and M_d) is dealt with.

Lemma 4.2. *For each i, marker m_i is defined, injured only finitely many times and reaches a limit.*

Proof. We argue by induction. Suppose that the lemma holds for $i \in \mathbb{N}$. Then there is some stage s_0 at which marker m_i has stopped moving. In order to conclude the induction step and the proof of this lemma, it suffices to show that m_{i+1} will reach a limit. By the induction hypothesis, m_{i+1} stops being injured after stage s_0. Hence c_{i+1} reaches a limit at s_0. Since A is not K-trivial, there is some least n_a such that $K_{N_{i+1}^a}(A \upharpoonright_{n_a}) > K(n_a) + c_{i+1}$. Similarly, since D is not K-trivial, there is some least n_d such that $K_{N_{i+1}^d}(A \upharpoonright_{n_d}) > K(n_d) + c_{i+1}$.

Let $s_1 > s_0$ is a stage where

- the approximations to $A \upharpoonright_{n_a}$ and $K(j)$, $j \leqslant n_a$ have settled;
- the approximations to $D \upharpoonright_{n_d}$ and $K(j)$, $j \leqslant n_d$ have settled.

Then the approximations to t_{i+1}^a, q_{i+1}^a and t_{i+1}^d, q_{i+1}^d also reach a limit by stage s_1. In particular, the limit of t_{i+1}^a is n_a and the limit of t_{i+1}^d is n_d.

If marker m_k moved after stage s_1, this would be either due to clause (b) or due to clause (c) of Section 4.2. In the first case the construction would enumerate an N_{i+1}^a-description of $A \upharpoonright_{n_a}$ of length $K(n_a) + c_{i+1}[s_0]$ and in the second case an N_{i+1}^d-description of $D \upharpoonright_{n_d}$ of length $K(n_d) + c_{i+1}[s_0]$. The first action would contradict the choice of n_a and the second action would contradict the choice of n_d. Hence m_k reaches a limit by stage s_1 and this concludes the induction step. □

As in the argument of Section 3, there is a many-one correspondence between the domain of M_a and the domain of the universal machine U. We say that a U-description is *A-used* if it corresponds to a string in the domain of M_a. Moreover it is *A-used n* times if it corresponds to n different strings in the domain of M_a. If a U-description that is already used at stage s becomes used again at stage $s+1$ we say that it was *reused*. Let S_0^a contain the descriptions in U that are A-used at least once. For each $k > 0$ let S_k^a contain the descriptions in the domain of U which are A-used at least $k+1$ times. Note that $S_{i+1}^a \subseteq S_i^a$ for each i. According to the correspondence between the domains of U and M^a, a string σ in the domain of U that is A-used k times incurs weight $k \cdot 2^{-|\sigma|}$ to the domain of M^a. Similar terminology and observations apply on D and M_d. Hence we have (4.5).

$$\mathtt{wgt}(M_a) \leqslant \sum_k \mathtt{wgt}\big(S_k^a\big) \quad \text{and} \quad \mathtt{wgt}(M_d) \leqslant \sum_k \mathtt{wgt}\big(S_k^d\big). \tag{4.5}$$

Note here that we avoided a multiplicative factor k in the above sums. This is not needed as each description in S_k^a will be counted $k+1$ in the above sum (and similarly with S_k^d). This, in turn, is a consequence of the fact that the sets in the sequences (S_k^a) and (S_k^d) are nested.

A U-description is called *A-active* at stage s if $U(\sigma)[s] \subseteq A_s$. By the construction, only currently A-active strings may move from S_k^a to S_{k+1}^a and only currently D-active strings may move from S_k^d to S_{k+1}^d at any given stage.

The sets S_k^a and S_k^d may be visualized as the containers of two independent decanter models that are identical to the one illustrated in Fig. 4. If at some stage a marker m_i moves (while m_j, $j < i$ remain stable) some strings from S_k^a enter S_{k+1}^a and some strings from S_k^d enter S_{k+1}^d for various $k \in \mathbb{N}$. In this case we say that these strings were A-reused and D-reused respectively by m_i.

The justification of the following lemma is analogous to Lemma 3.3 of Section 3. However it also deals with the non-uniformity that was discussed in Section 4.2, so it is not identical to the argument that was used in the proof of Lemma 3.3.

Lemma 4.3. *If during the interval of stages $[s, r]$ a marker m_n is not injured and $n \notin \emptyset'_r$ then the weight of the strings that are A-reused by m_n during this interval which remain active at stage r is at most $2^{-c_n[r]} + p^a_n[r]$.*

Proof. Let (s_i) be the sequence of stages in $[s, r]$ where m_n moves *and* an enumeration into N^a_n occurs. Note that m_n may move without an enumeration into N^a_n taking place. Moreover, at each stage s_i, the construction sets $p^a_n[s_i] = 0$. We claim that it suffices to show the lemma for the special set of stages (s_i). Indeed, by the construction, the weight of the strings that are A-reused by m_n during a stage in (s_i, s_{i+1}) is bounded by the increase in p^a_n. Hence if we prove that at each stage s_i the weight of the strings that are A-reused by m_n and remain active at stage r is bounded by $2^{-c_n[r]}$, we also have the result of the lemma for each stage in $[s, r]$.

In order to establish at each stage s_i the bound $2^{-c_n[r]}$ for the weight of the strings that are A-reused by m_n and remain active at stage r we will follow the argument that was given in the proof of Lemma 3.3. Note that instead of q_n, t_n we now have q^a_n, t^a_n and instead of the facts (3.5), (3.6) we now have (4.3), (4.4) respectively.

By the hypothesis of the lemma, the parameter c_n remains constant throughout the interval $[s, r]$. At stage $s_i \in [s, r]$ marker m_n moves. By the definition of stages s_{i-1}, s_i, no N^a_n enumeration takes place in the interval (s_{i-1}, s_i), except perhaps for the computations from clause (3.5) of the construction. Since m_n did not move during the stages in (s_{i-1}, s_i) for the sake of clause (b) of Section 4.2, it follows that

$$\sum_{j>m_n[s_{i-1}]} 2^{-K(A\upharpoonright j)[s]} < q_n[s] \quad \text{for } s \in (s_{i-1}, s_i). \tag{4.6}$$

Note that when m_n moves at stage $x+1$, the weight of the U-descriptions that it reuses is at most $\sum_{j>m_n[s_{i-1}]} 2^{-K(A\upharpoonright j)[s_i-1]}$ (and not $\sum_{j>m_n[s_{i-1}]} 2^{-K(A\upharpoonright j)[s_i]}$). This happens because the construction first moves marker m_n and then enumerates additional computations in M. In other words, the descriptions that m_n reuses at s_i correspond to M-computations that occurred in the previous stages, not the M-computations that may occur by the end of stage s_i. Hence by (4.6), the weight of the U-descriptions that are reused by m_n at stage s_i are bounded by $q^a_n[s_i - 1]$, which is $2^{-K(t^a_n)[s_i-1]-c_n[s_i-2]}$.

Now let us consider the overall effect of the movement of m_n during the stages (s_i). If at least one of the descriptions in U that m_n reused at some stage s_i continues to be active at stage r, then $A_{s_i} \upharpoonright_{m_n[s_i-1]+1} = A_r \upharpoonright_{m_n[s_i-1]+1}$. By (4.3), under the same assumptions this implies

$$A_{s_i} \upharpoonright_{t^a_n[s_i-1]+1} = A_r \upharpoonright_{t^a_n[s_i-1]+1}.$$

By $(*)$ of the construction (i.e. the enumeration of a computation in N^a_i upon the movement of a marker), since at stage s_i the marker m_n moved, we have $t^a_n[s_i] = t^a_n[s_i - 1] + 1$. Hence by (4.4) we get that

$$t^a_n[y] \geqslant t^a_n[s_i] > t^a_n[s_i - 1] \quad \text{for all } y \in [s_i, r].$$

The above observation along with the bound that we established in the previous paragraph on the weight of the U-descriptions that are reused by m_n at a stage in $[s, r]$, imply the following fact.

> At the stages (s_i) the weight of the descriptions in U that are A-used due to m_n and remain active at stage r, are bounded by $2^{-K(t^a_n)-c_n}$, where t^a_n is larger and larger and c_n remains equal to $c_n[s]$

(while the Kolmogorov function follows its usual approximation). More formally, $t^a_n[s_i - 1] < t^a_n[y_i]$ and the weight of U-descriptions that m_n reuses at stage s_i and remain active at stage r is at most $2^{-K(t^a_n)[s_i-1]-c_n[s]}$. So the total weight of the U-descriptions that m_n uses during the stages in $[s, r]$ and which remain active at stage r is less than

$$\sum_i 2^{-K(i)-c_n[s]}.$$

Since the above sum is bounded by $2^{-c_n[s]}$, this concludes the proof. □

The same argument applies symmetrically to the strings that are D-used, providing the bound $2^{-c_n[s]} + p^d_n[s]$.

Lemma 4.4. *If during the interval of stages $[s, r]$ a marker m_n is not injured then the weight of the strings that are D-reused by m_n during this interval which remain active at stage r is at most $2^{-c_n[s]} + p^d_n[s]$.*

Note that $p^a_n[s] \leqslant q^a_n[s]$ for each n and all stages s. This follows from clause (b) in Section 4.2 and the fact that whenever m_n moves due to this clause (or is injured) parameter p^a_n takes value 0. On the other hand by the definition of q^a_n we have $q^a_n[s] < 2^{-c_n[s]}$, so $p^a_n[s] \leqslant 2^{-c_n[s]}$. Hence the bound in Lemma 4.3 can be replaced with $2^{-c_n[s]+1}$. A similar argument applies to $p^d_n[s]$. The proof of Lemma 4.5 uses this observation in an adaptation of the proof of the analogous Lemma 3.4.

Lemma 4.5. *The weight of the requests that are enumerated in M_a is finite; the same holds for M_d.*

Proof. We give the proof for M_a; the proof for M_d is entirely symmetric. According to the correspondence between the domains of U and M_a that we discussed, we can bound the weight of M_a via (4.5). Note that each description in S_k^a is counted $k+1$ times in this sum as it belongs to all S_i^a, $i \leqslant k$. So it suffices to show that

$$\mathtt{wgt}\big(S_k^a\big) < 2^{-k-1} \quad \text{for each } k \geqslant 0. \tag{4.7}$$

Since only strings in the domain of U are used, $\mathtt{wgt}(S_0^a) < 2^{-2}$. Since $S_1^a \subseteq S_0^a$, condition (4.7) holds for $k \leqslant 1$. Let $k > 1$. Every entry of a string into S_k^a is due to a marker m_x which A-reused it when it was already in S_{k-1}^a. Since $k > 1$, this string entered S_{k-1}^a due to another marker m_y with $y > x \geqslant 0$. Inductively, that string entered S_1^a due to a marker m_z with $z \geqslant k-1$. Fix z, and let $S_k^a(z)$ contain the strings in S_k^a that entered S_1^a due to marker m_z. Then $S_k^a = \bigcup_{z \geqslant k-1} S_k^a(z)$ and $S_{k+1}^a(z) \subseteq S_k^a(z)$ for each $k > 1$. Hence

$$\mathtt{wgt}\big(S_k^a(z)\big) \leqslant \sum_{z \geqslant k-1} \mathtt{wgt}\big(S_k^a(z)\big) \quad \text{for each } k > 1.$$

So in order to prove (4.7) for $k > 1$ it suffices to show that

$$\mathtt{wgt}\big(S_k^a(z)\big) < 2^{-z-2} \quad \text{for each } z \geqslant 0. \tag{4.8}$$

Let (s_i) be the increasing sequence of stages where m_z is injured. Note that at this point we do not assume that (s_j) is a finite sequence. We may count the weight of $S_k^a(z)$ by counting the weight of the bunches of descriptions that enter in $S_1^a(z)$ and then enter in $S_2^a(z)$ (necessarily by some m_j with $j < z$). This is justified because every description that enters $S_k^a(z)$ must have passed from $S_2^a(z)$ first.

Since the movement of a marker m_i injures all m_j, $j > i$, the only stages were strings move from $S_1^a(z)$ to $S_2^a(z)$ are the stages (s_i). Moreover since only active strings move from $S_1^a(z)$ to $S_2^a(z)$ at stage s_i, according to Lemma 4.3 (and the observation straight after it) their weight is bounded by $2^{-c_z[s_i-1]+1}$. So the weight of the strings that enter $S_2^a(z)$ from $S_1^a(z)$ is bounded above by $\sum_j 2^{-c_z[s_j-1]}$. Since $c_z[s_{j+1}-1] = c_z[s_j] < c_z[s_j-1]$ for all j, this weight is bounded by $\sum_j 2^{-c_z[0]-j} = 2^{-c_z[0]+1}$. Since $c_z[0] = z+4$ this bound becomes 2^{-z-2}, which establishes (4.8) and concludes the proof. □

We conclude with the proof that (4.1) is met.

Lemma 4.6. *The following hold for each i:*

- *there is an M_a-description of $B \restriction_i$ of length $\leqslant K(A \restriction_i)$;*
- *there is an M_d-description of $B \restriction_i$ of length $\leqslant K(D \restriction_i)$.*

Proof. We argue by induction on i. Suppose that the lemma holds for $i \in \mathbb{N}$. Then there is some stage s_0 at which marker m_i is defined and has stopped moving and for each $(X, x) \in \{(A, a), (D, d)\}$

- the approximations to $X \restriction_{i+1}$, $B \restriction_{i+1}$, $K(X \restriction_{i+1})$, $K_{M_x}(B \restriction_i)$ have settled;
- $K_{M_x}(B \restriction_i)[s_0] \leqslant K(X \restriction_i)[s_0]$.

For each $(X, x) \in \{(A, a), (D, d)\}$, if $K_{M_x}(B \restriction_{i+1})[s_0] > K(X \restriction_{i+1})[s_0]$ the construction at stage $s_0 + 1$ will enumerate an M_x-computation that describes $B \restriction_{i+1}$ with a string of length $K(X \restriction_{i+1})$. □

By Lemma 4.2 and the construction we get that the movement of the markers satisfies properties (i)–(v) of Section 2. Hence $\emptyset' \leqslant_T B$. We conclude the proof of Theorem 1.1 by observing that (4.1) is met. By Lemma 4.6 the construction enumerates the required requests in M_a which ask for a description of $B \restriction_i$ with a string of length at most $K(A \restriction_i)$, for each i. Moreover the same holds for D in place of A and M_d in place of M_a. On the other hand Lemma 4.5 establishes that these request sets correspond to prefix-free machine, via the Kraft–Chaitin lemma. Hence (4.1) is met, which concludes the verification of the construction and the proof of Theorem 1.2.

5. Concluding remarks

We have demonstrated that computably enumerable sets can have a lot of information (for example, a solution to the halting problem) yet have very simple initial segments. On the other hand, as we discussed, it is known that such sets cannot have trivial initial segment complexity. In other words, their initial segments are more complex that the initial segments of an infinite sequence of 0s. Our result has had numerous applications, which were discussed in Section 1.5.

The methods that we used have novel features, but are not completely new. The bulk of the argument is depicted in Fig. 3 which indicates the dynamic relationships between each pair of the three pairs from the following actions:

(a) bound the complexity constructed set;
(b) challenge the non-triviality of the given set;
(c) code information into the constructed set.

After some abstraction, this type of argument can be found in other places in the recent literature (some times in simpler forms) where a set with nontrivial algorithmic–theoretic complexity is given and one is required to construct a set with lesser complexity which encodes certain kinds of information. Examples of such arguments can be found in [11,5,4,9]. However in the present paper we have made a conscious effort to explain the intuition and the dynamics of the argument in concrete terms. Despite the common form of these arguments, however, each case has its own unique features that stem from the particular measures of complexity that are involved. As an example in the LK-degrees, in [5] it was shown that every non-zero Δ^0_2 degree has uncountably many predecessors and in [4] it was shown that there are no minimal pairs of Δ^0_2 degrees. However, as we discussed, in the K-degrees every c.e. degree has only countably many predecessors. Moreover, although we showed that there is no minimal pair of K-degrees of c.e. sets, the same question for Δ^0_2 sets remains open.

References

[1] Klaus Ambos-Spies, Decheng Ding, Yun Fan, Wolfgang Merkle, Maximal pairs of computably enumerable sets in the computably Lipschitz degrees, Theory Comput. Syst. 52 (1) (2013) 2–27.
[2] Janis M. Barzdin, Complexity of programs to determine whether natural numbers not greater than n belong to recursively enumerable set, Sov. Math. Dokl. 9 (1968) 1251–1254.
[3] George Barmpalias, Computably enumerable sets in the Solovay and the strong weak truth table degrees, in: S. Barry Cooper, Benedikt Löwe, Leen Torenvliet (Eds.), CiE, in: Lect. Notes Comput. Sci., vol. 3526, Springer, 2005, pp. 8–17.
[4] George Barmpalias, Elementary differences between the degrees of unsolvability and the degrees of compressibility, Ann. Pure Appl. Log. 161 (7) (2010) 923–934.
[5] George Barmpalias, Relative randomness and cardinality, Notre Dame J. Form. Log. 51 (2) (2010).
[6] George Barmpalias, On strings with trivial Kolmogorov complexity, Int. J. Software Informatics 5 (4) (2011) 609–623.
[7] Martijn Baartse, George Barmpalias, On the gap between trivial and nontrivial initial segment prefix-free complexity, Theory Comput. Syst. 52 (2013) 28–47.
[8] George Barmpalias, Rupert Hölzl, Andrew E.M. Lewis, Wolfgang Merkle, Analogues of Chaitin's Omega in the computably enumerable sets, Inf. Process. Lett. 113 (5–6) (2013) 171–178.
[9] George Barmpalias, Andrew E.M. Lewis, Chaitin's halting probability and the compression of strings using oracles, Proc. R. Soc. A 467 (2011) 2912–2926.
[10] George Barmpalias, Angsheng Li, Kolmogorov complexity and computably enumerable sets, Ann. Pure Appl. Log. 164 (2013) 1187–1200.
[11] George Barmpalias, Anthony Morphett, Non-cupping, measure and computably enumerable splittings, Math. Struct. Comput. Sci. 19 (1) (2009) 25–43.
[12] Laurent Bienvenu, Wolfgang Merkle, André Nies, Solovay functions and K-triviality, in: STACS, 2011, pp. 452–463.
[13] George Barmpalias, C.S. Vlek, Kolmogorov complexity of initial segments of sequences and arithmetical definability, Theor. Comput. Sci. 412 (41) (2011) 5656–5667.
[14] Gregory J. Chaitin, A theory of program size formally identical to information theory, J. Assoc. Comput. Mach. 22 (1975) 329–340.
[15] Barbara F. Csima, Antonio Montalbán, A minimal pair of K-degrees, Proc. Am. Math. Soc. 134 (5) (2006) 1499–1502 (electronic).
[16] Rodney G. Downey, Denis Hirshfeldt, Algorithmic Randomness and Complexity, Springer, 2010.
[17] Rodney G. Downey, Denis R. Hirschfeldt, Geoff LaForte, Randomness and reducibility, J. Comput. Syst. Sci. 68 (1) (2004) 96–114.
[18] Rodney G. Downey, Denis R. Hirschfeldt, Geoffrey LaForte, Undecidability of the structure of the Solovay degrees of c.e. reals, J. Comput. Syst. Sci. 73 (5) (2007) 769–787.
[19] Rodney R. Downey, Denis Hirschfeldt, André Nies, Randomness, computability and density, SIAM J. Comput. 31 (2002) 1169–1183.
[20] Rodney G. Downey, Denis R. Hirschfeldt, André Nies, Frank Stephan, Trivial reals, in: Proceedings of the 7th and 8th Asian Logic Conferences, Singapore Univ. Press, Singapore, 2003, pp. 103–131.
[21] Bjørn Kjos-Hanssen, Joseph S. Miller, Reed Solomon, Lowness notions, measure and domination, J. Lond. Math. Soc. (2013), in press.
[22] Andrey N. Kolmogorov, Three approaches to the quantitative definition of information, Probl. Inf. Transm. 1 (1–7) (1965) 3–11.
[23] Leonid A. Levin, The concept of a random sequence, Dokl. Akad. Nauk SSSR 212 (1973) 548–550.
[24] Per Martin-Löf, The definition of random sequences, Inf. Control 9 (1966) 602–619.
[25] Wolfgang Merkle, Frank Stephan, On C-degrees, H-degrees and T-degrees, in: Twenty-Second Annual IEEE Conference on Computational Complexity (CCC 2007), San Diego, USA, 12–16 June 2007, IEEE Computer Society, Los Alamitos, CA, USA, 2007, pp. 60–69.
[26] Joseph S. Miller, Liang Yu, On initial segment complexity and degrees of randomness, Trans. Am. Math. Soc. 360 (6) (2008) 3193–3210.
[27] Joseph S. Miller, Liang Yu, Oscillation in the initial segment complexity of random reals, Adv. Math. 226 (6) (2011) 4816–4840.
[28] André Nies, Lowness properties and randomness, Adv. Math. 197 (1) (2005) 274–305.
[29] André Nies, Computability and Randomness, Oxford University Press, 2009.
[30] Robert Solovay, Handwritten Manuscript Related to Chaitin's Work, IBM Thomas J. Watson Research Center, Yorktown Heights, NY, 1975, 215 pp.
[31] Liang Yu, Decheng Ding, Rod Downey, The Kolmogorov complexity of the random reals, Ann. Pure Appl. Log. 129 (1–3) (2004) 163–180.