Extracting Kolmogorov Complexity with Applications to Dimension Zero-One Laws

Lance Fortnow*
Department of Computer Science
University of Chicago
fortnow@cs.uchicago.edu

John M. Hitchcock†
Department of Computer Science
University of Wyoming
jhitchco@cs.uwyo.edu

A. Pavan‡
Department of Computer Science
Iowa State University
pavan@cs.iastate.edu

N. V. Vinodchandran§
Department of Computer Science and Engineering
University of Nebraska-Lincoln
vinod@cse.unl.edu

Fengming Wang¶
Department of Computer Science
Rutgers University
fengming@cs.rutgers.edu

Abstract

We apply results on extracting randomness from independent sources to "extract" Kolmogorov complexity. For any $\alpha, \epsilon > 0$, given a string x with $K(x) > \alpha|x|$, we show how to use a constant number of advice bits to efficiently compute another string y, $|y| = \Omega(|x|)$, with $K(y) > (1-\epsilon)|y|$. This result holds for both unbounded and space-bounded Kolmogorov complexity.

We use the extraction procedure for space-bounded complexity to establish zero-one laws for the strong dimensions of complexity classes within ESPACE. The unbounded extraction procedure yields a zero-one law for the constructive strong dimensions of Turing degrees.

1 Introduction

Kolmogorov complexity quantifies the amount of randomness in an individual string. If a string x has Kolmogorov complexity m, then x is often said to contain m bits of randomness. Can we efficiently extract the Kolmogorov randomness from a string? That is, given x, is it possible to compute a string of length m that is Kolmogorov-random?

*Research supported in part by NSF grants 0652601 and 0829754.

†Research supported in part by NSF grants 0515313 and 0652601 and by an NWO travel grant. Part of this research was done while this author was on sabbatical at CWI.

‡Research supported in part by NSF grants 0430807 and 0830479.

§Research supported in part by NSF grants 0430991 and 0830730.

¶Research supported in part by NSF grant 0430807. Work done while this author was at Iowa State University.

Vereshchagin and Vyugin showed that this is not possible in general [30], i.e., they showed that there is no algorithm that can extract Kolmogorov complexity. Buhrman, Fortnow, Newman and Vereshchagin [5] showed that if one allows a small amount of extra information then Kolmogorov extraction is indeed possible. More specifically, they showed there is an efficient procedure $\mathcal{A}$ such that for every x with Kolmogorov complexity αn, there exists a string a_x, such that $\mathcal{A}(x, a_x)$ outputs a nearly Kolmogorov random string whose length is close to αn. Moreover, the length of a_x is $O(\log |x|)$, and contents of a_x depend on x.

In this paper we show that we can extract Kolmogorov complexity with only a *constant* constant number of bits of additional information. We give a *polynomial-time computable procedure* which takes x with an additional constant amount of advice and outputs a nearly Kolmogorov-random string whose length is linear in $|x|$. We defer to section 2 for the precise definition of Kolmogorov complexity and other technical concepts. Formally, for any $\alpha, \epsilon > 0$, given a string x with $K(x) > \alpha|x|$, we show how to use a constant number of advice bits to compute another string y, $|y| = \Omega(|x|)$, in polynomial-time that satisfies $K(y) > (1-\epsilon)|y|$. The number of advice bits depends only on α and ϵ, but the content of the advice depends on x. This computation needs only polynomial time, and yet it extracts unbounded Kolmogorov complexity.

Our proofs use a construction of a *multi-source extractor*. Traditional extractor results [6, 13, 19, 20, 23–29, 34] show how to take a distribution with high min-entropy and some truly random bits to create a close to uniform distribution. A multi-source extractor takes several independent distributions with high min-entropy and creates a close to uniform distribution. Thus multi-source extractors eliminate the need for a truly random source. Substantial progress has been made recently in the construction of efficient multi-source extractors [2, 3, 21, 22]. In this paper we use the construction due to Barak, Impagliazzo, and Wigderson [2] for our main result on extracting Kolmogorov complexity.

To make the connection, consider the uniform distribution on the set of strings x whose Kolmogorov complexity is at most m. This distribution has min-entropy about m and x acts like a random member of this set. We can define a set of strings $x_1, \ldots, x_k$ to be independent if $K(x_1 \cdots x_k) \approx K(x_1) + \cdots + K(x_k)$. By symmetry of information this implies $K(x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \approx K(x_i)$. Suppose we are given independent Kolmogorov random strings $x_1, \ldots x_k$, each of which has Kolmogorov complexity m. We view them as arising from k independent distributions each with min-entropy m. We then argue that a multi-source extractor with small error can be used to output a nearly Kolmogorov random string.

To extract the randomness from a single string x, we break x into a number of substrings $x_1, \ldots, x_l$, and view each substring x_i as coming from a different random source. Of course, these substrings may not be independently random in the Kolmogorov sense, thus we can not view these strings as coming from independent sources. A useful concept is to quantify the *dependency within* x as $\sum_{i=1}^{l} K(x_i) - K(x)$. We show that if the dependency within x is small, then the output of the multi-source extractor on its substrings is a nearly Kolmogorov random string. Another technical problem is that the randomness in x may not be nicely distributed among the substrings; for this we need to use a small (constant) number of nonuniform advice bits.

This result about extracting Kolmogorov-randomness also holds for polynomial-space bounded Kolmogorov complexity. We apply this to obtain zero-one laws for the strong dimensions of certain complexity classes. Resource-bounded dimension [14] and strong dimension [1] were developed as extensions of the classical Hausdorff and packing fractal dimensions to study the structure of complexity classes. Dimension and strong dimension both refine resource-bounded measure and are duals of each other in many ways. Strong dimension is also related to resource-bounded category [11]. In this paper we focus on strong dimension.

The strong dimension of each complexity class is a real number between zero and one inclusive. While there are examples of nonstandard complexity classes with fractional dimensions [1], we do not know of a standard complexity class with this property. Can a natural complexity class have a fractional dimension? In particular consider the class E. Determining its strong dimension within ESPACE would imply a major separation (either E $\not\subseteq$ PSPACE or E $\neq$ ESPACE). However, we are able to use our Kolmogorov-randomness extraction procedure to obtain a zero-one law ruling out the intermediate fractional possibility. Formally, we show that the strong dimension Dim(E | ESPACE) is either 0 or 1. The zero-one law also holds for various other complexity classes.

Our techniques also apply in the constructive dimension setting [15]. Miller and Nies [18] asked if it is possible to compute a set of higher constructive dimension from an arbitrary set of positive constructive dimension. We answer the strong dimension variant of this question in the negative, obtaining a zero-one law: for every Turing degree $\mathcal{D}$, the constructive strong dimension $\mathrm{Dim}(\mathcal{D})$ is either 0 or 1.

After the preliminary version of the paper appeared [7], there has been further work on the problem of Kolmogorov extraction and relations between Kolmogorov extraction and randomness extraction [8, 31–33]. Zimand [31] showed that there is a computable function f such that if x and y are two n-bit strings and the dependency within xy is small, then $f(x, y)$ is close to being a Kolmogorov random string. Hitchcock, Pavan and Vinodchandran [8] showed that every computable function that works as a Kolmogorov extractor is also an almost randomness extractor.

2 Preliminaries

2.1 Kolmogorov Complexity

We use $\Sigma = \{0, 1\}$ to denote the binary alphabet. Let M be a Turing machine. Let $f : \mathbb{N} \to \mathbb{N}$. For any $x \in \Sigma^*$, define

$$K_M(x) = \min\{|\pi| \mid M(\pi) \text{ prints } x\}$$

and

$$KS^f_M(x) = \min\{|\pi| \mid M(\pi) \text{ prints } x \text{ using at most } f(|x|) \text{ space}\}.$$

There is a universal machine U such that for every machine M and every reasonable space bound f, there is some constant c such that for all x, $K_U(x) \leq K_M(x)+c$ and $KS^{cf+c}_U(x) \leq KS^f_M(x)+c$ [12]. We fix such a machine U and drop the subscript, writing $K(x)$ and $KS^f(x)$, which are called the *(plain) Kolmogorov complexity of* x and f*-bounded (plain) Kolmogorov complexity of* x. While we use plain complexity in this paper, our results also hold for prefix-free complexity.

The following definition quantifies the fraction of randomness in a string.

Definition. For a string x, the *rate* of x is $\mathit{rate}(x) = K(x)/|x|$. For a polynomial g, the g*-rate of* x is $\mathit{rate}^g(x) = KS^g(x)/|x|$.

We denote the uniform distribution over Σ^n with U_n. Two distributions X and Y over Σ^n, are ϵ*-close* if

$$\frac{1}{2}\sum_{x\in\Sigma^n} |X(x) - Y(x)| \leq \epsilon.$$

Definition. Let X be a distribution over Σ^n and $Sup(X)$ denotes the set $\{x \in \Sigma^n \mid \Pr[X = x] \neq 0\}$. The *min-entropy* of X is

$$\min_{x \in Sup(X)} \log \frac{1}{\Pr[X = x]}.$$

2.2 Polynomial-Space Dimension

We now review the definitions of polynomial-space dimension [14] and strong dimension [1]. For more background we refer to these papers and the survey paper [10].

Let $s > 0$. An *s-gale* is a function $d : \{0,1\}^* \to [0, \infty)$ satisfying $2^s d(w) = d(w0) + d(w1)$ for all $w \in \{0,1\}^*$.

For a language A, we write $A \upharpoonright n$ for the first n bits of A's characteristic sequence (according to the standard enumeration of $\{0,1\}^*$) and $A \upharpoonright [i,j]$ for the subsequence beginning from the ith bit and ending at the jth bit. A language is sometimes also called a sequence. An s-gale d *succeeds on* a language A if $\limsup_{n\to\infty} d(A \upharpoonright n) = \infty$ and d *succeeds strongly on* A if $\liminf_{n\to\infty} d(A \upharpoonright n) = \infty$. The *success set* of d is $S^\infty[d] = \{A \mid d \text{ succeeds on } A\}$. The *strong success set* of d is $S^\infty_{\text{str}}[d] = \{A \mid d \text{ succeeds strongly on } A\}$.

Definition. Let X be a class of languages.

1. The pspace-*dimension* of X is

$$\dim_{\text{pspace}}(X) = \inf \left\{ s \;\middle|\; \begin{array}{l} \text{there is a polynomial-space computable} \\ s\text{-gale } d \text{ such that } X \subseteq S^\infty[d] \end{array} \right\}.$$

2. The *strong* pspace-*dimension* of X is

$$\text{Dim}_{\text{pspace}}(X) = \inf \left\{ s \;\middle|\; \begin{array}{l} \text{there is a polynomial-space computable} \\ s\text{-gale } d \text{ such that } X \subseteq S^\infty_{\text{str}}[d] \end{array} \right\}.$$

For every X, $0 \leq \dim_{\text{pspace}}(X) \leq \text{Dim}_{\text{pspace}}(X) \leq 1$. An important fact is that ESPACE has pspace-dimension 1, which suggests the following definitions.

Definition. Let X be a class of languages.

1. The *dimension of* X *within* ESPACE is

$$\dim(X \mid \text{ESPACE}) = \dim_{\text{pspace}}(X \cap \text{ESPACE}).$$

2. The *strong dimension of* X *within* ESPACE is

$$\text{Dim}(X \mid \text{ESPACE}) = \text{Dim}_{\text{pspace}}(X \cap \text{ESPACE}).$$

In this paper we will use an equivalent definition of these dimensions in terms of space-bounded Kolmogorov complexity.

Definition. Given a language L and a polynomial g the *g-rate of L* is

$$rate^g(L) = \liminf_{n\to\infty} rate^g(L\restriction n).$$

strong g-rate of L is

$$Rate^g(L) = \limsup_{n\to\infty} rate^g(L\restriction n).$$

Theorem 2.1. ([9, 16]) *Let* poly *denote all polynomials. For every class X of languages,*

$$\dim_{\text{pspace}}(X) = \inf_{g\in\text{poly}} \sup_{L\in X} rate^g(L).$$

and

$$\text{Dim}_{\text{pspace}}(X) = \inf_{g\in\text{poly}} \sup_{L\in X} Rate^g(L).$$

3 Extracting Kolmogorov Complexity

Barak, Impagliazzo, and Wigderson [2] gave an explicit multi-source extractor.

Theorem 3.1. *([2]) For every constant $0 < \sigma < 1$, and $c > 1$ there exist $l = poly(1/\sigma, c)$, a constant r and a computable function $E : \Sigma^{\ell n} \to \Sigma^n$ such that if $H_1, \cdots, H_l$ are independent distributions over Σ^n, each with min entropy at least σn, then $E(H_1, \cdots, H_l)$ is 2^{-cn}-close to U_n, where U_n is the uniform distribution over Σ^n. Moreover, E runs in time n^r.*

We show that this extractor can be used to produce nearly Kolmogorov-random strings from strings with high enough complexity. The following notion of dependency is useful for quantifying the performance of the extractor.

Definition. Let $x = x_1x_2\cdots x_k$, where each x_i is an n-bit string. The *dependency within x*, *dep*(x), is defined as $\sum_{i=1}^k K(x_i) - K(x)$.

Theorem 3.2. *For every $0 < \sigma < 1$ there exist constants n_0, $l > 1$ and a polynomial-time computable function E such that for every $n \geq n_0$, if $x_1, x_2, \cdots x_l$ are n-bit strings with $K(x_i) \geq \sigma n$, $1 \leq i \leq l$, then*

$$K(E(x_1, \cdots, x_l)) \geq n - 10l\log n - dep(x),$$

where $x = x_1x_2\cdots x_l$. Then length of $E(x_1, \ldots, x_l)$ is n.

Proof. Let $\sigma' = \sigma/2$. By Theorem 3.1, there is a constant l and a polynomial-time computable multi-source extractor E such that if $H_1, \cdots, H_l$ are independent sources each with min-entropy at least $\sigma' n$, then $E(H_1, \cdots, H_l)$ is 2^{-5n} close to U_n.

We show that this extractor also extracts Kolmogorov complexity. We prove by contradiction. Suppose the conclusion is false, i.e,

$$K(E(x_1, \cdots x_l)) < n - 10l\log n - dep(x).$$

Let $K(x_i) = m_i$, $1 \leq i \leq l$. Define the following sets:

$$I_i = \{y \mid y \in \Sigma^n, K(y) \leq m_i\},$$

$$Z = \{z \in \Sigma^n \mid K(z) < n - 10l \log n - dep(x)\},$$

$$Small = \{\langle y_1, \cdots, y_l\rangle \mid y_i \in I_i, \text{ and } E(y_1, \cdots y_l) \in Z\}.$$

By our assumption $\langle x_1, \cdots x_l\rangle$ belongs to $Small$. We use this to arrive at a contradiction regarding the Kolmogorov complexity of $x = x_1 x_2 \cdots x_l$. We first calculate an upper bound on the size of $Small$.

Every string from the set $S = \{xy \mid x \in \Sigma^{\lceil \sigma' n\rceil}, y = 0^{n-\lceil \sigma' n\rceil}\}$ has Kolmogorov complexity at most $\lceil \sigma' n\rceil + c \log n$ for some fixed constant c. Since $\sigma' = \sigma/2$, when n is large enough this quantity is at most σn. Thus the set S is a subset of each of I_i. Thus the cardinality of each of I_i is at least $2^{\sigma' n}$. Let H_i be the uniform distribution on I_i. Thus the min-entropy of H_i is at least $\sigma' n$.

Since H_i's have min-entropy at least $\sigma' n$, $E(H_1, \cdots, H_l)$ is 2^{-5n}-close to U_n. Then

$$\Big| P[E(H_1, \ldots, H_l) \in Z] - P[U_n \in Z] \Big| \leq 2^{-5n}. \tag{1}$$

Note that the cardinality of I_i is at most 2^{m_i+1}, as there are at most 2^{m_i+1} strings with Kolmogorov complexity at most m_i. Thus H_i places a weight of at least 2^{-m_i-1} on each string from I_i. Thus $H_1 \times \cdots \times H_l$ places a weight of at least $2^{-(m_1+\cdots+m_l+l)}$ on each element of $Small$. Therefore,

$$P[E(H_1, \ldots, H_l) \in Z] = P[(H_1, \ldots, H_l) \in Small] \geq |Small| \cdot 2^{-(m_1+\cdots+m_l+l)},$$

and since $|Z| \leq 2^{n-10l\log n - dep(x)}$, from (1) we obtain

$$|Small| \quad < \quad 2^{m_1+1} \times \cdots \times 2^{m_l+1} \times \left(\frac{2^{n-10l \log n - dep(x)}}{2^n} + 2^{-5n} \right).$$

Without loss of generality we can take $dep(x) < n$, otherwise the theorem is trivially true. Thus $2^{-5n} < 2^{-10l\log n - dep(x)}$ for sufficiently large n. Using this inequality and the fact that l is a constant independent of n, we obtain

$$|Small| < 2^{m_1+\cdots+m_l - dep(x) - 8l \log n},$$

when n is large enough. Since $K(x) = K(x_1) + \cdots + K(x_l) - dep(x)$,

$$|Small| < 2^{K(x) - 8l \log n}.$$

We first observe that there is a program Q that, given the values of m_i's, n, l, and $dep(x)$ as auxiliary inputs, recognizes the set $Small$. This program works as follows: Let $z = z_1 \cdots z_l$, where $|z_i| = n$. For each program P_i of length at most m_i check whether P_i outputs z_i, by running the P_i's in a dovetail fashion. If it is discovered that for each of z_i, $K(z_i) \leq m_i$, then compute $y = E(z_1, \cdots, z_l)$. Now verify that $K(y)$ is at most $n - dep(x) - 10l \log n$. This again can be done by running programs of the length at most $n - dep(x) - 10l \log n$ in a dovetail manner. If it is discovered that $K(y)$ is at most $n - dep(x) - 10l \log n$, then accept z.

So given the values of parameters n, $dep(x)$, l and m_i's, there is a program P that enumerates all elements of $Small$. Since by our assumption x belongs to $Small$, x appears in this enumeration. Let i be the position of x in this enumeration. Since $|Small|$ is at most $2^{K(x)-8l\log n}$, i can be described using $K(x) - 8l \log n$ bits.

Thus there is a program P' based on P that outputs x. This program takes i, $dep(x)$, n, $m_1, \cdots, m_l$, and l, as auxiliary inputs. Since the m_i's and $dep(x)$ are bounded by n,

$$\begin{aligned} K(x) &\leq K(x) - 8l\log n + 2\log n + l\log n + O(1) \\ &\leq K(x) - 5l\log n + O(1), \end{aligned}$$

which is a contradiction. □

Corollary 3.3. *For every constant $0 < \sigma < 1$, there exist constants l and n_0, and a polynomial-time computable function E with the following property:*

- *Let $x_1, \cdots x_l$ be n-bit strings such that $n \geq n_0$, $K(x_i) \geq \sigma n$, and $K(x_1x_2\cdots x_l) = \sum K(x_i) - O(\log n)$*
- *$E(x_1, \cdots, x_l)$ is Kolmogorov random in the sense that*

$$K(E(x_1, \cdots, x_l)) > n - O(\log n).$$

Theorem 3.2 says that given $x \in \Sigma^{ln}$, if each piece x_i has high enough complexity and the dependency with x is small, then we can output a string y whose Kolmogorov rate is higher than the Kolmogorov rate of x, i.e, y is relatively more random than x. What if we only knew that x has high enough complexity but knew nothing about the complexity of individual pieces or the dependency within x? Our next theorem states that in this case also there is a procedure producing a string whose rate is higher than the rate of x. However, this procedure needs a constant number of advice bits.

Theorem 3.4. *For all real numbers $0 < \alpha < \beta < 1$ there exist a constant $0 < \delta < 1$, constants $c, l, n_0 \geq 1$, and a procedure R such that the following holds. For any string x with $|x| \geq n_0$ and $rate(x) \geq \alpha$, there exists an advice string a_x such that*

$$rate(R(x, a_x)) \geq \min\{rate(x) + \delta, \beta\}$$

where $|a_x| = c$. Moreover, R runs in polynomial time, and $|R(x, a_x)| = \lfloor |x|/l \rfloor$.

The number c depends only on α, β and is independent of x. However, the contents of a_x depend on x.

Before we give a formal proof, we briefly explain the proof idea. Given a string x, we split it into l substrings $x_1, x_2, \cdots, x_l$. Consider the function E from Theorem 3.2. If $dep(x_1x_2, \cdots x_l)$ is small, then by Theorem 3.2 the rate of $E(x_1, \cdots, x_l)$ is higher than the rate of x. The crucial observation is that if $dep(x_1x_2\cdots x_l)$ is not small, then one of the substrings x_i must have a higher rate than the rate of x. Thus one of $x_1, x_2, \cdots, x_l, E(x_1, \cdots, x_l)$ has a higher rate than the rate of x. Since l is constant, a constant number of advice bits suffices to specify the string with higher rate. We now give a formal proof.

Proof. Let $0 < \alpha' < \alpha$ and $0 < \epsilon < \min\{1 - \beta, \alpha'\}$. Let $\sigma = (1 - \epsilon)\alpha'$. Using parameter σ in Theorem 3.2, we obtain a constant $l > 1$ and a polynomial-time computable function E that extracts Kolmogorov complexity.

Let $\beta' = 1 - \frac{\epsilon}{2}$, and $\gamma = \frac{\epsilon^2}{2l}$. Observe that $\gamma \leq \frac{1-\beta'}{l}$ and $\gamma < \frac{\alpha'-\sigma}{l}$.

Let x have $rate(x) = \nu \geq \alpha$. Let $n, k \geq 0$ such that $|x| = ln + k$ and $k < l$. We strip the last k bits from x and write $x = x_1 \cdots x_l$ where each $|x_i| = n$. Let $\nu' = rate(x)$ after this change. We have $\nu' > \nu - \gamma/2$ and $\nu' > \alpha'$ if $|x|$ is sufficiently large.

We consider three cases.

Case 1. There exists j, $1 \leq j \leq l$ such that $K(x_j) < \sigma n$.
Case 2. Case 1 does not hold and $dep(x) \geq \gamma ln$.
Case 3. Case 1 does not hold and $dep(x) < \gamma ln$.

We have two claims about Cases 1 and 2:

Claim 3.4.1. *Assume Case 1 holds. There exists i, $1 \leq i \leq l$, such that $rate(x_i) \geq \nu' + \gamma$.*

Proof of Claim 3.4.1. Suppose not. Then for every $i \neq j$, $1 \leq i \leq l$, $K(x_i) \leq (\nu' + \gamma)n$. We can describe x by describing x_j which takes σn bits, and all the x_i's, $i \neq j$. Thus the total complexity of x would be at most

$$(\nu' + \gamma)(l - 1)n + \sigma n + O(\log n)$$

Since $\gamma < \frac{\alpha' - \sigma}{l}$ and $\alpha' < \nu'$ this quantity is less than $\nu' ln$. Since the rate of x is ν', this is a contradiction. □ *Claim 3.4.1*

Claim 3.4.2. *Assume Case 2 holds. There exists i, $1 \leq i \leq l$, $rate(x_i) \geq \nu' + \gamma$.*

Proof of Claim 3.4.2. By definition,

$$K(x) = \sum_{i=1}^{l} K(x_i) - dep(x)$$

Since $dep(x) \geq \gamma ln$ and $K(x) \geq \nu' ln$,

$$\sum_{i=1}^{l} K(x_i) \geq (\nu' + \gamma) ln.$$

Thus there exists i such that $rate(x_i) \geq \nu' + \gamma$. □ *Claim 3.4.2*

We can now describe the constant number of advice bits. The advice a_x contains the following information: which of the three cases described above holds, and

- If Case 1 holds, then from Claim 3.4.1 the index i such that $rate(x_i) \geq \nu' + \gamma$.
- If Case 2 holds, then from Claim 3.4.2 the index i such that $rate(x_i) \geq \nu' + \gamma$.

Since $1 \leq i \leq l$, the number of advice bits is bounded by $O(\log l)$. We now describe procedure R. When R takes an input x, it first examines the advice a_x. If Case 1 or Case 2 holds, then R simply outputs x_i. Otherwise, Case 3 holds, and R outputs $E(x)$. Since E runs in polynomial time, R runs in polynomial time.

If Case 1 or Case 2 holds, then

$$rate(R(x, a_x)) \geq \nu' + \gamma \geq \nu + \tfrac{\gamma}{2}.$$

If Case 3 holds, we have $R(x, a_x) = E(x)$ and by Theorem 3.2, $K(E(x)) \geq n - 10\log n - \gamma ln$. Since $\gamma \leq \frac{1-\beta'}{l}$, in this case

$$rate(R(x, a_x)) \geq \beta' - \frac{10\log n}{n}.$$

For large enough n, this value is at least β. Therefore in all three cases, the rate increases by at least $\gamma/2$ or reaches β. By setting δ to $\gamma/2$, we have the theorem. □

We now prove our main theorem.

Theorem 3.5. *Let α and β be constants with $0 < \alpha < \beta < 1$. There exist a polynomial-time procedure $P(\cdot, \cdot)$ and constants C_1, C_2, n_1 such that for every x with $|x| \geq n_1$ and $rate(x) \geq \alpha$ there exists a string a_x with $|a_x| = C_1$ such that*

$$rate(P(x, a_x)) \geq \beta$$

and $|P(x, a_x)| \geq |x|/C_2$.

Proof. We apply the procedure R from Theorem 3.4 iteratively. Each application of R outputs a string whose rate is at least β or is at least δ more than the rate of the input string. Applying R at most $k = \lceil(\beta - \alpha)/\delta\rceil$ times, we obtain a string whose rate is at least β.

Note that $R(y, a_y)$ has output length $|R(y, a_y)| = \lfloor |y|/l \rfloor$ and increases the rate of y if $|y| \geq n_0$. If we take $n_1 = (n_0 + 1)kl$, we ensure that in each application of R we have a string whose length is at least n_0. Each iteration of R requires c bits of advice, so the total number of advice bits needed is $C_1 = kc$. Thus C_1 depends only on α and β. Each application of R decreases the length by a constant fraction, so there is a constant C_2 such that the length of the final outputs string is at least $|x|/C_2$. □

The proofs in this section also work for space-bounded Kolmogorov complexity. For this we need a space-bounded version of dependency.

Definition. Let $x = x_1 x_2 \cdots x_k$ where each x_i is an n-bit string, let f and g be two space bounds. The (f, g)*-bounded dependency within* x, $dep_g^f(x)$, is defined as $\sum_{i=1}^{k} KS^g(x_i) - KS^f(x)$.

We obtain the following version of Theorem 3.2.

Theorem 3.6. *For every polynomial g there exists a polynomial f such that for every $0 < \sigma < 1$, there exist a constant $l > 1$, and a polynomial-time computable function E such that if $x_1, \cdots, x_l$ are n-bit strings with $KS^f(x_i) \geq \sigma n$, $1 \leq i \leq l$, then*

$$KS^g(E(x_1, \cdots, x_l)) \geq n - 10l\log n - dep_g^f(x).$$

Similarly we obtain the following extension of Theorem 3.5.

Theorem 3.7. *Let g be a polynomial and let α and β be constants with $0 < \alpha < \beta < 1$. There exist a polynomial f, polynomial-time procedure $R(\cdot, \cdot)$, and constants C_1, C_2, n_1 such that for every x with $|x| \geq n_1$ and $rate^f(x) \geq \alpha$ there exists a string a_x with $|a_x| = C_1$ such that*

$$rate^g(R(x, a_x)) \geq \beta$$

and $|R(x, a_x)| \geq |x|/C_2$.

4 Zero-One Laws for Complexity Classes

In this section we establish a zero-one law for the strong dimensions of certain complexity classes. Let $\alpha < \theta$. We will first show that if E has a language with $Rate^f(L) \geq \alpha$, then E has a language L' with $Rate^g(L') \geq \theta$.

Let L be a language with $Rate^f(L) \geq \alpha$ for some function f. We will first show that the characteristic sequence of L is of the form $y_1 y_2 \cdots$ such that for infinitely many i, $rate^f(y_i) \geq \alpha/4$. Let R be the procedure from Theorem 3.7. If we define $R(y_1, a_{y_1})R(y_2, a_{y_2})\cdots$ as the characteristic sequence of a new language L'', then for infinitely many i, the rate of $R(y_i, a_{y_i})$ is bigger than α. If we ensure that length of y_i is reasonably bigger than the length of y_{i-1}, then it follows that $Rate^g(L')$ is at least θ. The following lemma makes these ideas precise.

Lemma 4.1. *Let g be any polynomial and α, θ be rational numbers with $0 < \alpha < \theta < 1$. Then there is a polynomial f such that if there exists $L \in \mathrm{E}$ with $Rate^f(L) > \alpha$, then there exists $L' \in \mathrm{E}$ with $Rate^g(L') \geq \theta$.*

Proof. Let β be a real number bigger than θ and smaller than 1 and $f = \omega(g)$. Pick positive integers C and K such that $(C-1)/K < 3\alpha/4$, and $\frac{(C-1)\beta}{C} > \theta$. Let $n_1 = 1$, $n_{i+1} = Cn_i$.

We now define strings $y_1, y_2, \cdots$ such that each y_i is a substring of the characteristic sequence of L or is in 0^*, and $|y_i| = (C-1)n_i/K$. While defining these strings we will ensure that for infinitely many i, $rate^f(y_i) \geq \alpha/4$.

We now define y_i. We consider three cases.

Case 1. $rate^f(L\restriction n_i) \geq \alpha/4$. Divide $L\restriction n_i$ in to $K/(C-1)$ segments such that the length of each segment is $(C-1)n_i/K$. It is easy to see that at least for one segment the f-rate is at least $\alpha/4$. Define y_i to be a segment with $rate^f(y_i) \geq \alpha/4$.

Case 2. Case 1 does not hold and for every j, $n_i < j < n_{i+1}$, $rate^f(L\restriction j) < \alpha$. In this case we punt and define $y_i = 0^{\frac{(C-1)n_i}{K}}$.

Case 3. Case 1 does not hold and there exists j, $n_i < j < n_{i+1}$ such that $rate^f(L\restriction j) > \alpha$. Divide $L\restriction[n_i, n_{i+1}]$ into K segments. Since $n_{i+1} = Cn_i$, length of each segment is $(C-1)n_i/K$.

Then it is easy to show that some segment has f-rate at least $\alpha/4$. We define y_i to be this segment.

Since for infinitely many j, $rate^f(L\restriction j) \geq \alpha$, for infinitely many i either Case 1 or Case 3 holds. Thus for infinitely many i, $rate^f(y_i) \geq \alpha/4$.

By Theorem 3.7, there is a procedure R with such that given a string x with $rate^f(x) \geq \alpha/4$, and the advice a_x, $rate^g(R(x, a_x)) \geq \beta$.

Let $w_i = R(y_i, a_{y_i})$. Since for infinitely many i, $rate^f(y_i) \geq \alpha/4$, for infinitely many i, $rate^g(w_i) \geq \beta$. Also recall that $|w_i| = |y_i|/C_2$ for an absolute constant C_2.

Claim 4.1.1. $|w_{i+1}| \geq (C-1)\sum_{j=1}^{i} |w_j|$.

Proof of Claim 4.1.1. We have

$$\sum_{j=1}^{i} |w_j| \leq \frac{C-1}{KC_2}\sum_{j=1}^{i} n_j = \frac{C-1}{KC_2}\frac{(C^i-1)n_1}{C-1},$$

with the equality holding because $n_{j+1} = Cn_j$. Also,

$$|w_{i+1}| = \frac{(C-1)n_{i+1}}{KC_2} \geq \frac{(C-1)C^i n_1}{KC_2}.$$

Thus

$$\frac{|w_{i+1}|}{\sum_{j=1}^{i}|w_j|} > (C-1).$$

□ *Claim 4.1.1*

Claim 4.1.2. *For infinitely many i, $rate^g(w_1 \cdots w_i) \geq \theta$.*

Proof of Claim 4.1.2. For infinitely many i, $rate^g(w_i) \geq \beta$, which means $KS^g(w_i) \geq \beta|w_i|$ and therefore

$$KS^g(w_1 \cdots w_i) \geq \beta|w_i| - O(1).$$

By Claim 4.1.1, $|w_i| \geq (C-1)(|w_1| + \cdots + |w_{i-1}|)$. Thus for infinitely many i, $rate^g(w_1 \cdots w_i) \geq \frac{(C-1)\beta}{C} - o(1) \geq \theta$. □ *Claim 4.1.2*

Let L' be the language with characteristic sequence $w_1w_2\cdots$. Then by Claim 4.1.2, $Rate^g(L') \geq \theta$.

Next, we argue that if L is in E, then L' is in E$/O(1)$. Observe that w_i depends on y_i and a_{y_i}, thus each bit of w_i can be computed by knowing y_i and a_{y_i}. Recall that y_i is either a subsegment of the characteristic sequence of L or 0^{n_i}. We will know y_i if we know which of the three cases mentioned above hold. This can be given as advice. Also observe that y_i is a subsequence of $L \upharpoonright n_{i+1}$. Also recall that w_i can be computed from y_i in time polynomial in $|y_i|$ using constant bits of advice a_{y_i}. Since $|w_i| = |y_i|/C_2$ for some absolute constant C_2, the running time needed to compute w_i is also polynomial in $|w_i|$. Since L is in E, this places L' in E$/O(1)$.

Finally, we observe that the advice can be removed to obtain a language in E. Let A be the length of the advice needed to compute L' in exponential time. Recall that A is finite. Let $I = \{i \mid rate^f(y_i) \geq \alpha/4\}$. Given a potential advice a of length A let

$$I_a = \{i \mid i \in I, R(y_i, a) = w_i\}.$$

Since I is infinite and the set of all advices is finite, there is an advice a such that I_a is infinite. From now we will fix one such a. Define our new language L'' as follows: Let $w''_i = R(y_i, a)$, and $w''_1w''_2w''_3\cdots$ is the characteristic sequence of the language L''. Now for every $i \in I_a$, $rate^g(w''_i) \geq \beta$. The proof of Claim 4.1.2, also shows that for every $i \in I_a$ $rate(w''_1w''_2\cdots w''_i) \geq \theta$. Thus $Rate^g(L'') \geq \theta$.

Now we have to argue that L'' is in E. Observe that if know that correct value of a, then we can compute L'' in exponential time. Each possible value for a gives an exponential time algorithm. Since there are only finitely many possible values for a, we have finitely many algorithms and one of them correctly decides L''. This shows that L'' is in E. This completes the proof of Lemma 4.1. □

Theorem 4.2. Dim(E | ESPACE) *is either 0 or 1.*

Proof. Because E $\subseteq$ ESPACE, Dim(E | ESPACE) = $\mathrm{Dim}_{\mathrm{pspace}}$(E). We will show that if $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}) > 0$, then $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}) = 1$. For this, it suffices to show that for every polynomial g and real number $0 < \theta < 1$, there is a language L' in E with $Rate^g(L') \geq \theta$. By Theorem 2.1, this will show that the strong pspace-dimension of E is 1.

The assumption states that the strong pspace-dimension of E is greater than 0. If the strong pspace-dimension of E is actually one, then we are done. If not, let α be a positive rational number that is less than $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E})$. By Theorem 2.1, for every polynomial f, there exists a language $L \in \mathrm{E}$ with $\mathit{Rate}^f(L) \geq \alpha$.

By Lemma 4.1, from such a language L we obtain a language L' in E with $\mathit{Rate}^g(L') \geq \theta$. Thus the strong pspace-dimension of E is 1. □

The zero-one law in Theorem 4.2 also holds for many other complexity classes.

Theorem 4.3. *Let $\mathcal{C}$ be a class that is closed under exponential-time truth-table reductions. Then* $\mathrm{Dim}(\mathcal{C} \mid \mathrm{ESPACE})$ *is either 0 or 1.*

Therefore additional examples of classes the zero-one law holds for include $\mathrm{NE} \cap \mathrm{coNE}$, BPE, and E^{NP}.

Remark. Theorem 4.2 concerns strong dimension. For dimension, the situation is considerably more complicated. With our techniques we can prove that if $\dim_{\mathrm{pspace}}(\mathrm{E}) > 0$, then $\dim_{\mathrm{pspace}}(\mathrm{E}/O(1)) \geq 1/2$. It appears that a different method is needed to eliminate the advice or increase the dimension past 1/2.

5 Zero-One Law for Constructive Strong Dimension

Miller and Nies [18] asked if every sequence of positive constructive dimension computes (by way of a Turing reduction) a sequence of higher constructive dimension. Our techniques yield a positive answer for the variant of this question using strong dimension instead of dimension.

For a sequence S, the constructive dimension of S is

$$\dim(S) = \liminf_{n\to\infty} \mathit{rate}(S \upharpoonright n)$$

and the constructive strong dimension of S is

$$\mathrm{Dim}(S) = \limsup_{n\to\infty} \mathit{rate}(S \upharpoonright n).$$

The definitions extend to a class X of sequences by

$$\dim(X) = \sup_{S\in X} \dim(S)$$

and

$$\mathrm{Dim}(X) = \sup_{S\in X} \mathrm{Dim}(S).$$

We refer to [1, 15] for more background on these dimensions.

Theorem 5.1. *If* $\mathrm{Dim}(S) > 0$*, then for every* $\epsilon > 0$*, there exists* $R \leq_{\mathrm{T}} S$ *such that* $\mathrm{Dim}(R) > 1-\epsilon$.

The proof of Theorem 5.1 is the same as Lemma 4.1, except instead of Theorem 3.7 we use Theorem 3.5. The 0-1 law for the Turing degrees follows:

Theorem 5.2. *For every Turing degree $\mathcal{D}$,* $\mathrm{Dim}(\mathcal{D})$ *is either 0 or 1.*

Proof. Suppose that a Turing degree $\mathcal{D}$ has positive constructive strong dimension and choose $S \in \mathcal{D}$ with $\mathrm{Dim}(S) > 0$. Let $\epsilon > 0$. From Theorem 5.1 we obtain a sequence R_ϵ with $\mathrm{Dim}(R_\epsilon) > 1 - \epsilon$ and $R_\epsilon \leq_{\mathrm{T}} S$. We can encode S into R_ϵ in a sparse way to obtain a sequence R'_ϵ with $S \leq_{\mathrm{T}} R'_\epsilon$, $R'_\epsilon \leq_{\mathrm{T}} S$, and $\mathrm{Dim}(R'_\epsilon) = \mathrm{Dim}(R_\epsilon)$. Therefore $R'_\epsilon \in \mathcal{D}$ and $\mathrm{Dim}(\mathcal{D}) > 1 - \epsilon$. As this holds for all $\epsilon > 0$, it follows that $\mathrm{Dim}(\mathcal{D}) = 1$. □

We note that the reduction we obtain in Theorem 5.1 is actually an exponential-time truth-table reduction, so in particular it is a truth-table reduction. Therefore we also have a 0-1 law for the truth-table degrees.

Subsequent to the conference version of this paper, Bienvenu, Doty, and Stephan [4] obtained a different proof of Theorem 5.1 and other related results using quite different techniques. In contrast, Miller [17] recently showed that there is no analogous 0-1 law for constructive dimension: there exists S with $\dim(S) = 1/2$ such that every sequence $R \leq_{\mathrm{T}} S$ has $\dim(R) \leq 1/2$.

Acknowledgments

We thank Xiaoyang Gu and Philippe Moser for several helpful discussions.

References

[1] K. B. Athreya, J. M. Hitchcock, J. H. Lutz, and E. Mayordomo. Effective strong dimension in algorithmic information and computational complexity. *SIAM Journal on Computing*, 37(3):671–705, 2007.

[2] B. Barak, R. Impagliazzo, and A. Wigderson. Extracting randomness using few independent sources. In *Proceedings of the 45th Annual Symposium on Foundations of Computer Science*, pages 384–393. IEEE Computer Society, 2004.

[3] B. Barak, G. Kindler, R. Shaltiel, B. Sudakov, and A. Wigderson. Simulating independence: new constructions of condensers, Ramsey graphs, dispersers, and extractors. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, pages 1–10. ACM, 2005.

[4] L. Bienvenu, D. Doty, and F. Stephan. Constructive dimension and weak truth-table degrees. In *Proceedings of the Third Conference on Computability in Europe*, pages 63–72. Springer-Verlag, 2007.

[5] H. Buhrman, L. Fortnow, I. Newman, and N. Vereshchagin. Increasing Kolmogorov complexity. In *Proceedings of the 22nd Symposium on Theoretical Aspects of Computer Science*, pages 412–421. Springer-Verlag, 2005.

[6] B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science*, pages 429–442. IEEE Computer Society, 1985.

[7] L. Fortnow, J. Hitchcock, A. Pavan, N. V. Vinodchandran, and F. Wang. Extracting Kolmogorov complexity with applications to dimension zero-one laws. In *Proceedings of the 33rd International Colloquium on Automata, Languages, and Programming*, volume 4051 of *Lecture Notes in Computer Science*, pages 335–345, 2006.

[8] J. Hitchcock, A. Pavan, and N. V. Vinodchandran. Kolmogorov complexity in randomness extraction. In *29th Conference on Foundations of Software Technology and Theoretical Computer Science*, volume 4 of *LIPIcs*, pages 215–226. chloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2009.

[9] J. M. Hitchcock. *Effective Fractal Dimension: Foundations and Applications*. PhD thesis, Iowa State University, 2003.

[10] J. M. Hitchcock, J. H. Lutz, and E. Mayordomo. The fractal geometry of complexity classes. *SIGACT News*, 36(3):24–38, September 2005.

[11] J. M. Hitchcock and A. Pavan. Resource-bounded strong dimension versus resource-bounded category. *Information Processing Letters*, 95(3):377–381, 2005.

[12] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, Berlin, 1997. Second Edition.

[13] C-J. Lu, O. Reingold, S. Vadhan, and A. Wigderson. Extractors: Optimal up to a constant factor. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, pages 602–611. ACM, 2003.

[14] J. H. Lutz. Dimension in complexity classes. *SIAM Journal on Computing*, 32(5):1236–1259, 2003.

[15] J. H. Lutz. The dimensions of individual strings and sequences. *Information and Computation*, 187(1):49–79, 2003.

[16] E. Mayordomo. A Kolmogorov complexity characterization of constructive Hausdorff dimension. *Information Processing Letters*, 84(1):1–3, 2002.

[17] J. S. Miller. Extracting information is hard. *Advances in Mathematics*. To appear.

[18] J. S. Miller and A. Nies. Randomness and computability: open questions. *Bulletin of Symbolic Logic*, 12(3):390–410, 2006.

[19] N. Nisan and A. Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 42(2):149–167, 1999.

[20] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.

[21] A. Rao. Extractors for a constant number of polynomially small min-entropy independent sources. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 497–506. ACM, 2006.

[22] R. Raz. Extractors with weak random seeds. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, pages 11–20. ACM, 2005.

[23] O. Reingold, R. Shaltiel, and A. Wigderson. Extracting randomness via repeated condensing. In *Proceedings of the 41st Annual Conference on Foundations of Computer Science*, pages 22–31. IEEE Computer Society, 2000.

[24] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings of the 41st Annual Symposium on Foundations of Computer Science*, pages 3–13. IEEE Computer Society, 2000.

[25] M. Santha and U. Vazirani. Generating quasi-random sequences from slightly random sources. In *Proceedings of the 25th Annual Symposium on Foundations of Computer Science*, pages 434–440. IEEE Computer Society, 1984.

[26] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 648–657. IEEE Computer Society, 2001.

[27] A. Srinivasan and D. Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28(4):1433–1459, 1999.

[28] A. Ta-Shma, D. Zuckerman, and M. Safra. Extractors from Reed-Muller codes. In *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, pages 638–647. IEEE Computer Society, 2001.

[29] L. Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(1):860–879, 2001.

[30] N. Vereshchagin and M. Vyugin. Independent minimum length programs to translate between given strings. *Theoretical Computer Science*, 271(1–2):131–143, 2002.

[31] M. Zimand. Extracting the kolmogorov complexity of strings and sequences from sources with limited independence. In *26th International Conference on Symposium on Theoretical Aspects of Computer Science*, volume 3 of *LIPIcs*, pages 697–708. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.

[32] M. Zimand. On generating independent random strings. In *5th Conference on Computability in Europe*, number 5635 in Lecture Notes in Computer Science, pages 499–508, 2009.

[33] M. Zimand. Two sources are better than one for increasing Kolmogorov complexity of infinite sequences. *Theory of Computing Systems*, 46(4):707–722, 2010.

[34] D. Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11(4):345–367, 1997.