Electronic Colloquium on Computational Complexity, Report No. 105 (2005)

Extracting Kolmogorov Complexity with Applications to Dimension Zero-One Laws

Lance Fortnow* John M. Hitchcock† A. Pavan‡ N. V. Vinodchandran§
Fengming Wang¶

September 24, 2005

Abstract

We apply recent 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 classical and space-bounded Kolmogorov complexity.

We use the extraction procedure for space-bounded complexity to establish zero-one laws for polynomial-space strong dimension. Our results include:

(i) If $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}) > 0$, then $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}/O(1)) = 1$.

(ii) $\mathrm{Dim}(\mathrm{E}/O(1) \mid \mathrm{ESPACE})$ is either 0 or 1.

(iii) $\mathrm{Dim}(\mathrm{E/poly} \mid \mathrm{ESPACE})$ is either 0 or 1.

In other words, from a dimension standpoint and with respect to a small amount of advice, the exponential-time class E is either minimally complex (dimension 0) or maximally complex (dimension 1) within ESPACE.

Classification: Computational and Structural Complexity.

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. Given x, is it possible to compute a string of length m that is Kolmogorov-random? In general this is impossible but we do make progress in this direction if we allow a tiny amount of extra information. We

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

†Department of Computer Science, University of Wyoming. jhitchco@cs.uwyo.edu. This research was supported in part by NSF grant 0515313.

‡Department of Computer Science, Iowa State University. pavan@cs.iastate.edu. This research was supported in part by NSF grant 0430807.

§Department of Computer Science and Engineering, University of Nebraska-Lincoln. vinod@cse.unl.edu. This research was supported in part by NSF grant 0430991.

¶Department of Computer Science, Iowa State University. wfengm@cs.iastate.edu. This research was supported in part by NSF grant 0430807.

ISSN 1433-8092

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 m. 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.

Our proofs use a recent construction of extractors using multiple independent sources. Traditional extractor results [13, 22, 19, 12, 21, 15, 16, 20, 9, 18, 17, 4] show how to take a distribution with high min-entropy and some truly random bits to create a close to uniform distribution. Recently, Barak, Impagliazzo, and Wigderson [2] showed how to eliminate the need for a truly random source when several independent random sources are available. We make use of these extractors for our main result on extracting Kolmogorov complexity. Barak et. al. [3] and Raz [14] have further extensions.

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 $K(x_1 \ldots x_k) \approx K(x_1) + \cdots + K(x_k)$. By symmetry of information this implies $K(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k) \approx K(x_i)$. Combining these ideas we are able to apply the extractor constructions for multiple independent sources to Kolmogorov complexity.

To extract the randomness from a string x, we break x into a number of substrings $x_1, \ldots, x_l$, and view each substring x_i as coming from an independent random source. Of course, these substrings may not be independently random in the Kolmogorov sense. We find it a useful concept to quantify the *dependency within* x as $\sum_{i=1}^{l} K(x_i) - K(x)$. Another technical problem is that the randomness in x may not be nicely distributed among these 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 some zero-one laws for the dimensions of complexity classes. Polynomial-space dimension [11] and strong dimension [1] have been developed to study the quantitative structure of classes that lie in E and ESPACE. These dimensions are resource-bounded versions of Hausdorff dimension and packing dimension, respectively, the two most important fractal dimensions. Polynomial-space dimension and strong dimension refine PSPACE-measure [10] and have been shown to be duals of each other in many ways [1]. Additionally, polynomial-space strong dimension is closely related to PSPACE-category [7]. In this paper we focus on polynomial-space strong dimension which quantifies PSPACE and ESPACE in the following way:

- $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{PSPACE}) = 0$.
- $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{ESPACE}) = 1$.

We would like to know the dimension of a complexity class $\mathcal{C}$, contained in ESPACE. The dimension must always exist and be a real number between zero and one inclusive. Can a reasonable complexity class have a fractional dimension? In particular consider the class E. Deciding the polynomial-space dimension of E would imply a major complexity separation but perhaps we can show that E must have dimension either zero or one, a "zero-one" law for dimension.

We can show such a "zero-one" law if we add a small amount of nonuniform advice. An equivalence between space-bounded Kolmogorov complexity rates and strong pspace-dimension allows us to use our Kolmogorov-randomness extraction procedure to show the following results.

(i) If $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}) > 0$, then $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}/O(1)) = 1$.

(ii) $\mathrm{Dim}(\mathrm{E}/O(1) \mid \mathrm{ESPACE})$ is either 0 or 1.

(iii) $\mathrm{Dim}(\mathrm{E/poly} \mid \mathrm{ESPACE})$ is either 0 or 1.

2 Preliminaries

2.1 Kolmogorov Complexity

Let M be a universal Turing machine. Let $f : \mathbb{N} \to \mathbb{N}$. For any $x \in \{0,1\}^*$, define

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

and

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

There is a universal machine U such that for every machine M, there is some constant c such that for all x, $K_U(x) \leq K_M(x)$ and $KS_U^f(x) \leq KS_M^{cf+c}(x) + c$ [8]. 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 space-bounded randomness in a string.

Definition. Given a string x and a polynomial g the g*-rate of* x, $rate^g(x)$, is $KS^g(x)/|x|$,

2.2 Polynomial-Space Dimension

We now review the definitions of polynomial-space dimension [11] and strong dimension [1]. For more background we refer to these papers and the recent survey paper [6].

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 $\{0,1\}^*$). 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 $S^\infty[d] = \{A \mid d \text{ succeeds on } S\}$. The *strong success set* of d is $S^\infty_{\mathrm{str}}[d] = \{A \mid d \text{ succeeds strongly on } S\}$.

Definition. Let X be a class of languages.

1. The pspace-*dimension* of X is

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

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

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

For every X, $0 \leq \dim_{\rm pspace}(X) \leq \mathrm{Dim}_{\rm pspace}(X) \leq 1$. 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 $\dim(X \mid \mathrm{ESPACE}) = \dim_{\rm pspace}(X \cap \mathrm{ESPACE})$.
2. The *strong dimension of X within* ESPACE is $\mathrm{Dim}(X \mid \mathrm{ESPACE}) = \mathrm{Dim}_{\rm pspace}(X \cap \mathrm{ESPACE})$.

In this paper we will use an equivalent definition of the above 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. (Hitchcock [5]) *Let* poly *denote all polynomials. For every class X of languages,*

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

and

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

3 Extracting Kolmogorov Complexity

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

Theorem 3.1. *([2]) For every constants $0 < \sigma < 1$, and $c > 1$ there exists $l = poly(1/\sigma, c)$ and a computable function E 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 the the above 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 $x = x_1 x_2 \cdots x_k$, where each x_i is an n-bit string. Given a function f, 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 a constant $l > 1$, and a polynomial-time computable function E such that 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*

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

Proof. Let $0 < \sigma' < \sigma$. 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 $K(x_i) = m_i$, $1 \le i \le l$. Define the following sets:

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

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

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

By our assumption $\langle x_1, \cdots x_l\rangle$ belongs to $Small$. We use this to arrive 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 $Small$.

Observe that the set $\{xy \mid x \in \Sigma^{\sigma' n}, y = 0^{n-\sigma' n}\}$ 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

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

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,

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

and since $|Z| \le 2^{n-10l \log n - dep(x)}$, 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 $2^{-5n} < 2^{-10l \log n - dep(x)}$. Using this 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 $Small$ is a computably enumerable set. Let$z = z_1 \cdots z_l$, where $|z_i| = n$. The following program accepts z if it belongs to $Small$: For each program P_i of length at most m_i check whether P_i outputs z_i, by running 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 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.

Since $Small$ is computably enumerable, 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 Q 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. □

If $x_1, \cdots x_l$ are independent strings with $K(x_i) \geq \sigma n$, then $E(x_1, \cdots, x_l)$ is a Kolmogorov random string of length n.

Corollary 3.3. *For every constant $0 < \sigma < 1$, there exists a constant l, and a polynomial-time computable function E such that if $x_1, \cdots x_l$ are n-bit strings such $K(x_i) \geq \sigma n$, and $K(x) = \sum K(x_i) - O(\log n)$, then $E(x)$ is Kolmogorov random, i.e.,*

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

We next show that above theorem can be generalized to the space-bounded case. Later we will use the space-bounded version to obtain dimension zero-one laws. 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 $\sum_{i=1}^{k} KS^g(x_i) - KS^f(x)$.

Theorem 3.4. *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 $x_1, x_2, \cdots x_l$ are n-bit strings with $KS^f(x_i) \geq \sigma n$, $1 \leq i \leq l$, then*

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

Proof. For the most part proof is similar to the proof of Theorem 3.2. Here we point the places where the proofs differ. Pick parameters σ' and l as before. This defines an extractor E. Let n^r be a bound on the running time of E. Pick a polynomial $f = \omega(g + n^r)$.

Suppose the conclusion is false, i.e,

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

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

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

$$Small = \{\langle y_1, \cdots, y_l \rangle \mid y_i \in I_i, \text{ and } KS^g(E(y_1, \cdots y_l)) < n - 10l \log n - dep_g^f(x)\}.$$

Arguing exactly as before, we obtain

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

Since $dep_g^f(x) = KS^g(x_1) + \cdots + KS^g(x_l) - KS^f(x)$,

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

Given a string $z = z_1 \cdots z_l$, we can check whether $z \in Small$ within $f(n)$ space as follows: Run every program P_i of length at most m_i within $g(n)$ space. If it is discovered that for each z_i, $KS^g(z_i) \le m_i$, then compute $y = E(z_1, \cdots, z_l)$. Check if $KS^g(y)$ is at most $n - 10l\log n - dep_g^f(x)$. Since E runs in n^r time, and $f = \omega(g + n^r)$, this program takes $f(n)$ space.

Now arguing as in Theorem 3.2, we obtain a contradiction regarding $KS^f(x)$. □

This theorem says that given $x \in \Sigma^{ln}$, if each piece x_i has high enough complexity and the dependency with x is small then, 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 state that in this case also there is a procedure a string whose rate is higher than the rate of x. However, this procedure needs constant bits of advice.

Theorem 3.5. *For every polynomial g and real number $\alpha \in (0,1)$, there exist a polynomial f, a positive integer l, a constant $0 < \gamma < 1$, and a procedure R such that for any string $x \in \Sigma^{ln}$ with $rate^f(x) \ge \alpha$,*

$$rate^g(R(x)) \ge \alpha + \gamma.$$

The procedure R requites C_1 bits of advice, where C_1 depends only on α and is independent of x and $|x|$. Moreover R runs in polynomial time and $|R(x)| = |x|/l$.

Proof. Pick σ such that $0 < \sigma < \alpha$. By Theorem 3.4, there is a constant $l > 1$ and a polynomial-time computable function E that extracts Kolmogorov complexity. Let $x = x_1 x_2 \cdots x_l$ where $|x_i| = n, 1 \le i \le l$, and $rate^f(x) \ge \alpha$. Let $1 > \beta' > \beta > \alpha$. Let $\gamma' \le \frac{1-\beta'}{l}$, $0 < \sigma < \alpha$, and $\delta < \frac{\alpha-\sigma}{l}$. Pick f such that $f = \omega(g + n^r)$, where n^r is the running time of E. We consider three cases.

Case 1. There exists j, $1 \le j \le l$ such that $KS^f(x_j) < \sigma n$.
Case 2. Case 1 does not hold and $dep_g^f(x) \ge \gamma' ln$.
Case 3. Cases 1 does not hold and $dep_g^f(x) < \gamma' ln$.

We have two claims about Cases 1 and 2:

Claim 3.5.1. *Assume Case 1 holds. There exists i, $1 \le i \le l$, such that $rate^g(x_i) \ge rate^f(x) + \delta$.*

Proof. Suppose not. Then for every $i \ne j$, $1 \le i \le l$, $KS^g(x_i) \le (\alpha + \delta)n$. We can describe x by describing j, which takes $\log l$ bits, x_j which takes σn bits, and all the x_i's, $i \ne j$. Thus the total complexity of x would be at most

$$(\alpha + \delta)(l-1)n + \sigma n + \log l$$

Since $\delta < \frac{\alpha-\sigma}{l}$ this quantity is less than αln. Since the f-rate of x is at least α, this is a contradiction. □ *Claim 3.5.1.*

Claim 3.5.2. *Assume Case 2 holds. There exists i, $1 \le i \le l$, $rate^g(x_i) \ge rate^f(x) + \gamma'$.*

Proof. By definition,

$$KS^f(x) = \sum_{i=1}^{l} KS^g(x_i) - dep_g^f(x)$$

Since $dep_g^f(x) \ge \gamma' ln$ and $KS^f(x) \ge \alpha ln$,

$$\sum_{i=1}^{l} KS^g(x_i) \ge (\alpha + \gamma')ln.$$

Thus there exists i such that $rate^g(x) \ge rate^f(x) + \gamma'$. □ *Claim 3.5.2.*

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

- If Case 1 holds, then from Claim 3.5.1 the index i such that $rate^g(x_i) \ge rate^f(x) + \delta$.
- If Case 2 holds, then from Claim 3.5.2 the index i such that $rate^g(x_i) \ge rate^f(x) + \gamma'$.

We now describe procedure R. When R takes an input x, it first examines the advice. If Case 1 or Case 2 holds, then R simply outputs x_i. Otherwise, Case 3 holds, and R outputs $E(x)$.

Clearly if Case 1 holds, then

$$rate^g(R(x)) \ge rate^f(x) + \delta,$$

and if Case 2 holds, then

$$rate^g(R(x)) \ge rate^f(x) + \gamma'.$$

If Case 3 holds, we have $R(x) = E(x)$ and by Theorem 3.4, $KS^g(R(x)) \ge n - 10\log n - \gamma' ln$. Since $\gamma' \le \frac{1-\beta'}{l}$, in this case

$$rate^g(R(x)) \ge \beta' - \tfrac{10\log n}{n}.$$

For large enough n, this value is bigger than β.

Finally, letting $\gamma = \min\{\delta, \gamma', \beta - \alpha\}$, we have

$$rate^g(R(x)) \ge rate^f(x) + \gamma$$

in all cases. Since E runs in polynomial time, R also runs in polynomial time. □

The following theorem follows from the above theorem.

Theorem 3.6. *For every polynomial g, there exist a polynomial f such that given $0 < \alpha < \beta < 1$ and a string x with $rate^f(x) \ge \alpha$, there exist constants C_1, C_2 and a procedure R such that $rate^g(R(x)) \ge \beta$. Moreover P takes C_1 bits of advice and $|R(x)| = |x|/C_2$.*

We will apply Theorem 3.5 iteratively. Each iteration of the Theorem increases the rate by γ. We will stop when we touch the desired rate β. Since in each iteration we increase the rate by a constant, this process terminates in constant number of iterations. However, this argument has a small caveat—it is possible that in each iteration the value of γ decreases and so we may never touch the desired rate β. Observe that the value of γ depends on parameters σ, l, β, and β'. By choosing these parameters carefully, we can ensure that in each iteration the rate is incremented by a sufficient amount, and in constant rounds it touches β. We omit the details.

4 Zero-One Laws

In this section we establish zero-one laws for the dimensions of certain classes within ESPACE. Our most basic result is the following, which says that if E has positive dimension, then the class $E/O(1)$ has maximal dimension.

Theorem 4.1. *If* $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}) > 0$*, then* $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}/O(1)) = 1$.

We first show the following lemma from which the theorem follows easily.

Lemma 4.2. *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 $Rate^f(L) \geq \alpha$, then there exists $L' \in E/O(1)$ with $Rate^g(L') \geq \theta$.*

Proof of Lemma 4.2. Let β be a real number bigger than θ and smaller than 1. 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, 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. $rate^f(L|n_i) \geq \alpha/4$. Divide $L|n_i$ 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, $n_i < j < n_{i+1}$, $rate^f(L|j) < \alpha$. 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 $rate^f(L|j) > \alpha$. Divide $L|[n_i, n_{i+1}]$ 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, $rate^f(L|j) \geq \alpha$, 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 3.6, there is a procedure R such that given a string x with $rate^f(x) \geq \alpha/4$, $rate^g(R(x)) \geq \beta$.

Let $w_i = R(y_i)$. 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 4.2.1. $|w_{i+1}| \geq (C-1)\sum_{j=1}^{i} |w_j|$.

Proof. 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 4.2.1.*

Claim 4.2.2. *For infinitely many i, $rate^g(w_1 \cdots w_i) \geq \theta$.*

Proof. 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 4.2.1, $|w_i| \geq (C-1)(|w_1| + \cdots + |w_{i-1}|)$. Thus for infinitely many i, $rate^g(w_1 \cdots w_i) \geq \frac{(C-1)\beta}{C} - o(1) \geq \theta$. □ *Claim 4.2.2.*

We define $w_1 w_2 \cdots$ to be the characteristic sequence of L'. Then by Claim 4.2.2, $Rate^g(L') \geq \theta$.

Finally, we argue that if L is in E, then L' is in E$/O(1)$. Observe that w_i depends on y_i, thus each bit of w_i can be computed by knowing 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 $L|n_{i+1}$. Also recall that w_i can be computed from y_i in polynomial time (polynomial in $|y_i|$) using constant bits of advice. Also observe that $|w_i| = |y_i|/C_1$ for some absolute constant C_1. Thus w_i can be computed in polynomial time (polynomial in $|w_i|$) given $L|n_{i+1}$. Since L is in E, this places L' in E$/O(1)$.

This completes the proof of Lemma 4.2.

□

We now return to the proof of Theorem 4.1.

Proof of Theorem 4.1. We will show that for every polynomial g, and real number $0 < \theta < 1$, there is a language L' in E$/O(1)$ with $Rate^g(L) \geq \theta$. By Theorem 2.1, this will show that the strong *pspace*-dimension of E$/O(1)$ 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 $Rate^f(L) \geq \alpha$.

By Lemma 4.2, from such a language L we obtain a language L' in E$/O(1)$ with $Rate^g(L') \geq \theta$. Thus the strong *pspace*-dimension of E$/O(1)$ is 1. □

Observe that in the above construction, if the original language L is in E$/O(1)$, then also L' is in E$/O(1)$, and similarly membership in E/poly is preserved. Additionally, if $L \in$ ESPACE, it can be shown that $L' \in$ ESPACE. With these observations, we obtain the following zero-one laws.

Theorem 4.3. *Each of the following is either 0 or 1.*

1. $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E}/O(1))$.
2. $\mathrm{Dim}_{\mathrm{pspace}}(\mathrm{E/poly})$.
3. $\mathrm{Dim}(\mathrm{E}/O(1) \mid \mathrm{ESPACE})$.
4. $\mathrm{Dim}(\mathrm{E/poly} \mid \mathrm{ESPACE})$.

We remark that in Theorems 4.1 and 4.3, if we replace E by EXP, the theorems still hold. The proofs also go through for other classes such as BPEXP, NEXP $\cap$ coNEXP, or NEXP/poly.

Theorems 4.1 and 4.3 concern strong dimension. For dimension, the situation is more complicated. Using similar techniques, we can prove that if $\dim_{\mathrm{pspace}}(\mathrm{E}) > 0$, then $\dim_{\mathrm{pspace}}(\mathrm{E}/O(1)) \geq 1/2$. Analogously, we can obtain zero-half laws for the pspace-dimension of E/poly, etc. We omit the details.

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*. To appear.

[2] B. Barak, R. Impagliazzo, and A. Wigderson. Extracting randomness using few independent sources. In *Proceedings of the 45th Annual IEEE 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, 2005.

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

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

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

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

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

[9] 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, 2003.

[10] J. H. Lutz. Almost everywhere high nonuniform complexity. *Journal of Computer and System Sciences*, 44(2):220–258, 1992.

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

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

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

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

[15] O. Reingold, R. Shaltiel, and A. Wigderson. Extracting randomness via repeated condensing. In *Proceedings of the 41st Annual COnference on Foundations of Computer science*, 2000.

[16] 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 IEEE Conference on Foundations of Computer Science*, 2000.

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

[18] R. Shaltiel and C. Umans. Simple extractors for all min-entropies and a new pseudo-random generator. In *Proceedings of the 42nd Annual Conference on Foundations of Computer Science*, 2001.

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

[20] A. Ta-Shma, D. Zuckerman, and M. Safra. Extractors from reed-muller codes. In *Proceedings of the 42nd Annual Conference on Foundations of Computer Science*, 2001.

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

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

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'