

Kolmogorov extraction and resource-bounded zero-one laws

by

Fengming Wang

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Pavan Aduri, Major Professor
Jack H. Lutz
Srikanta Tirthapura

Iowa State University

Ames, Iowa

2006

Copyright © Fengming Wang, 2006. All rights reserved.

Graduate College
Iowa State University

This is to certify that the master's thesis of
Fengming Wang
has met the thesis requirements of Iowa State University

Major Professor

For the Major Program

DEDICATION

I would like to dedicate this thesis to my parents and to my love Cui without whose support I would not have been able to complete this work.

TABLE OF CONTENTS

ABSTRACT	vi
CHAPTER 1 Introduction	1
1.1 Kolmogorov complexity	1
1.1.1 Kolmogorov randomness	1
1.1.2 Extractors	2
1.1.3 Connections between Kolmogorov complexity and extractors	3
1.2 Resource-bounded dimension	4
1.2.1 History	4
1.2.2 Zero-one laws	4
1.3 Organization of the thesis	5
CHAPTER 2 Preliminaries	6
2.1 Kolmogorov complexity	6
2.2 Polynomial-space dimension	7
2.3 Extractors	9
CHAPTER 3 Extracting Kolmogorov complexity	11
3.1 Extraction using multi-source extractor	11
3.2 Extraction using single-source extractor	19

CHAPTER 4	Zero-one laws	22
4.1	Dimension zero-one laws	22
4.2	Scaled dimension zero-one Law	25
4.3	Open questions	26
BIBLIOGRAPHY		31
ACKNOWLEDGEMENTS		32

ABSTRACT

Traditional extractors show how to efficiently extract randomness from weak random sources with help of small truly random bits. Recent breakthroughs on multi-source extractors gave an efficient way to extract randomness from independent sources. We apply these techniques to “extract” Kolmogorov complexity. More formally,

1. for any $\alpha > 0$, given a string x with $K(x) > |x|^\alpha$, we show how to use $O(\log |x|)$ bits of advice to efficiently compute another string y , $|y| = |x|^{\Omega(1)}$, with $K(y) > |y| - O(\log |y|)$
2. 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 above extraction procedure for space-bounded complexity to establish zero-one laws for both polynomial-space strong dimension and strong scaled dimension. Our results include:

- (i) If $\text{Dim}_{\text{pspace}}(\mathbb{E}) > 0$, then $\text{Dim}_{\text{pspace}}(\mathbb{E}/O(1)) = 1$.
- (ii) $\text{Dim}(\mathbb{E}/O(1) \mid \text{ESPACE})$ is either 0 or 1.
- (iii) $\text{Dim}(\mathbb{E}/\text{poly} \mid \text{ESPACE})$ is either 0 or 1.
- (iv) Either $\text{Dim}_{\text{pspace}}^{(1)}(\mathbb{E}/O(n)) = 0$ or $\text{Dim}_{\text{pspace}}^{(-1)}(\mathbb{E}/O(n)) = 1$.

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 or maximally complex within $ESPACE$.

CHAPTER 1 Introduction

This thesis studies two related topics: resource-bounded dimension and Kolmogorov complexity. The latter topic serves as the main technical tool to obtain the results in the former one.

1.1 Kolmogorov complexity

1.1.1 Kolmogorov randomness

Informally the Kolmogorov complexity of a string x , denoted as $K(x)$, is the number of bits required to describe x . More formally, $K(x)$ is the length of the shortest program that prints x . If a string has a noticeable pattern, then a short program could use this pattern to print the string. Thus for such strings $K(x)$ is less than $|x|$. For example, $0^n 1^n$ has very low Kolmogorov complexity, approximately $O(\log n)$. If a string has no noticeable patterns, then the best way to describe the string is itself. Typically such strings are called "random strings". Therefore Kolmogorov complexity of a string measures the amount of randomness within a string. A random string has Kolmogorov complexity close to its length while the Kolmogorov complexity of a non-random string is much less than its length. If a string x has Kolmogorov complexity m , then x is often said to contain m bits of randomness. Naturally this raises the following question:

Given x with Kolmogorov complexity m , is it possible to compute a string of length m that is Kolmogorov-random?

Vereshchagin and Vyugin [28] showed that there are no such uniform algorithms which could perform the above task. However we show that if the algorithm is given access to some advice bits, we can achieve this task. 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 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 . This computation needs only polynomial time, and yet it extracts unbounded Kolmogorov complexity. To obtain this result, we apply results from extractors.

1.1.2 Extractors

Probabilistic computations, including randomized algorithms and interactive protocols, have demonstrated many advantages over deterministic computations. For instance, randomized algorithms might give better time bounds and interactive protocols could perform tasks which are impossible in the deterministic world. All these probabilistic computations requires "truly random bits", that is a sequence of coin flips which are uniformly distributed and independent of each other. To simulate the probabilistic algorithms in practice, we need a good random source that generates "truly random bits". So the following question naturally arises: How could we obtain truly random bits? There are physical devices such as Zener Diodes that output random bits. While these sources have some randomness, we do not know whether they are truly random. Such sources are called *weak random sources*. Can we simulate probabilistic algorithms using *weak random sources*? One way to achieve this is by converting *weak random sources* to truly random sources. Such conversion procedures are called extractors.

To formalize this problem we need a way to measure the amount of randomness in *weak*

random sources. The notion of *min-entropy* turns out to be an appropriate measure. If a source has min-entropy k , then we say the source has k bits of randomness. Therefore we can formally phrase the problem as follows: If given a distribution X over Σ^n with *min-entropy* k , can we generate the uniform distribution over Σ^k ?

Deterministically it is impossible, yet with the help of small amount of truly random bits, we could design such procedures, namely *extractor*. Informally traditional single-source extractors take a distribution from Σ^n with high min-entropy and some truly random bits (approximately $O(\log n)$) to create a close to uniform distribution. Many results have been obtained along this line of research [19, 29, 24, 18, 26, 20, 21, 25, 11, 23, 22, 3]. Recently, Barak, Impagliazzo, and Wigderson [2] showed how to eliminate the need for a truly random source when several independent random sources are available.

1.1.3 Connections between Kolmogorov complexity and extractors

We make use of the above extractors 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. Consider an extractor E that converts this distribution into a uniform distribution over Σ^m . We argue that if every output of E has low Kolmogorov complexity, then its output distribution can not be uniform. From this we show how to obtain Kolmogorov-random strings of length $O(m)$.

1.2 Resource-bounded dimension

1.2.1 History

Lutz [12] proposed resource-bounded measure, a quantitative method to investigate the structure of exponential time complexity classes, which is an analogue of classical Lebesgue measure. For example, the p -measure for NP, $\mu_p(\text{NP})$, reflects its relative "size" compared to E. Similarly $\mu_{p\text{space}}(\text{E})$ is the relative "size" of E compared to ESPACE. It is known that $\mu_p(\text{P}) = 0$ as well as $\mu_p(\text{E}) \neq 0$, thus deciding $\mu_p(\text{NP})$ could give us a major separation in complexity theory, either $\text{P} \neq \text{NP}$ or $\text{NP} \neq \text{E}$. Similarly determining the value of $\mu_{p\text{space}}(\text{E})$ would add another important separation to our knowledge, either $\text{PSPACE} \neq \text{E}$ or $\text{E} \neq \text{ESPACE}$. For more details, we refer the readers to the survey papers by Lutz and Mayordomo [15] and Lutz [13].

Several years later, Lutz [14] refined resource-bounded measure to introduce resource-bounded dimension for complexity theory as an effectivization of classical Hausdorff dimension. Just in the same manner that Hausdorff dimension generalized Lebesgue measure, resource-bounded dimension generalizes resource-bounded measure (See [8] for more details.). Deciding p -dimension of NP, denoted as $\text{dim}_p(\text{NP})$, and $p\text{space}$ -dimension of E, $\text{dim}_{p\text{space}}(\text{E})$ would imply the same separations as above.

1.2.2 Zero-one laws

It is known that any reasonable complexity class can have p -measure 0, 1, or non- p -measurable. In particular, there is no reasonable class C with $0 < \mu_p(C) < 1$. For some classes, we can eliminate the third possibility. Therefore we obtain resource-bounded zero-one laws for those classes. Dieter van Melkebeek [27] showed that the p -measure zero-one law holds for BPP. Afterwards Impagliazzo and Moser [9] obtained a similar zero-one law for RP.

Similarly A zero-one law for resource-bounded dimension means that the dimension of a complexity class has only two outcomes, either zero or one. However, unlike resource-bounded measure, it is possible for a complexity class to have resource-bounded dimension between zero and one. For example, Lutz [14] showed that $\dim_{\text{pspace}}(\text{SIZE}(\alpha \frac{2^n}{n})) = \alpha$, where $\text{SIZE}(\alpha \frac{2^n}{n})$ denotes the set of languages decided by family of circuits no larger than $\alpha \frac{2^n}{n}$. Therefore the following intriguing question arises:

Are there any standard complexity classes such as BPP, NP, or E which admit resource-bounded dimension zero-one laws? For example, a zero-one law for E is that *pspace*-dimension for E is either 0 or 1, which means E is either minimally complex or maximally complex within ESPACE. Moser [17] proved that BPP has effective dimension at most $\frac{1}{2}$ unless BPP equals to EXP. Fortnow and Lutz [4] observed that in fact there is a dimension zero-one law for BPP.

In this thesis, we apply the technique of Kolmogorov-randomness extraction to achieve the following results:

- (i) If strong *pspace*-dimension for E is greater than 0, then strong *pspace*-dimension for $E/O(1)$ is equal to 1.
- (ii) strong *pspace*-dimension for $E/O(1)$ within ESPACE is either 0 or 1.
- (iii) strong *pspace*-dimension for E/poly within ESPACE is either 0 or 1.

1.3 Organization of the thesis

Chapter 2 gives essential preliminaries on Kolmogorov complexity, space-bounded dimension as well as the notations that we use throughout this thesis. Next we describe Kolmogorov-randomness extraction using both Trevisan's Extractor [26] and recent multi-source extractor [2] in Chapter 3. In Chapter 4, we present dimension zero-one laws.

CHAPTER 2 Preliminaries

2.1 Kolmogorov complexity

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

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

and

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

There is a universal machine U such that for every machine M , there is some constant c such that for all x , $K_U(x) \leq K_M(x) + c$ and $KS_U^f(x) \leq KS_M^{cf+c}(x) + c$ [10]. 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 the *rate of x* , $rate(x)$, is $KS(x)/|x|$,

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 [14] and strong dimension [1]. For more background we refer to these papers and the recent survey paper [8].

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

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 Σ^*), $A \upharpoonright [i, j]$ for the subsequence beginning from the i th bit and ending at the j th bit. An s -gale d *succeeds on* a language A if $\limsup_{n \rightarrow \infty} d(A \upharpoonright n) = \infty$ and d *succeeds strongly on* A if $\liminf_{n \rightarrow \infty} d(A \upharpoonright n) = \infty$. The *success set* of d is $S^\infty[d] = \{A \mid d \text{ succeeds on } A\}$. The *strong success set* of d is $S_{\text{str}}^\infty[d] = \{A \mid d \text{ succeeds strongly on } A\}$.

Definition. Let X be a class of languages.

1. The *pspace-dimension* of X is

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

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

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

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 the above dimensions in terms of Kolmogorov complexity.

Definition. Given a language L the *rate of L* is

$$\text{rate}(L) = \liminf_{n \rightarrow \infty} \text{rate}(L \upharpoonright n).$$

strong rate of L is

$$\text{Rate}(L) = \limsup_{n \rightarrow \infty} \text{rate}(L \upharpoonright n).$$

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

$$\text{rate}^g(L) = \liminf_{n \rightarrow \infty} \text{rate}^g(L \upharpoonright n).$$

strong g -rate of L is

$$\text{Rate}^g(L) = \limsup_{n \rightarrow \infty} \text{rate}^g(L \upharpoonright n).$$

Mayordomo [16] give a Komolgorov characterization of constructive dimension.

Theorem 2.2.1 (Mayordomo [16]). *For every class X of languages,*

$$\text{dim}(X) = \inf \sup_{L \in X} \text{rate}(L).$$

and

$$\text{Dim}(X) = \inf \sup_{L \in X} \text{Rate}(L).$$

Theorem 2.2.2. (Hitchcock [5]) *Let poly denote all polynomials. For every class X of languages,*

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

and

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

2.3 Extractors

We review definition of extractors.

Definition. Suppose we have two distributions X_1, X_2 over Σ^n , the statistical difference between X_1 and X_2 is defined as

$$\text{stat}(X_1, X_2) = \sum_{x \in \Sigma^n} |Pr[x \in X_1] - Pr[x \in X_2]|$$

We say that X_1 is ϵ -close to X_2 if $\text{stat}(X_1, X_2) \leq \epsilon$

Definition. For a probability distribution X over Σ^n , define *min-entropy* d of X as

$$d = \log \frac{1}{\max_{x \in \Sigma^n} Pr[x \in X]}$$

We say X is an (n, d) -source if X is a distribution over Σ^n and the min-entropy of X is at least d .

Let U_n denote the uniform distribution over Σ^n .

Definition (Single-source Extractor). An (n, d, l, m, ϵ) single-source extractor is a polynomial-time computable function $E : \Sigma^n \times \Sigma^l \rightarrow \Sigma^m$ which satisfies the following properties:

for any (n, d) -source X , $\text{stat}(E(X, U_l), U_m) \leq \epsilon$,

We say that E is a single-source extractor with error parameter ϵ and min-entropy threshold d , and its seed length is l .

Definition (Multi-source Extractor). An (n, d, l, m, ϵ) multi-source extractor is a polynomial-time computable function $E : \Sigma^{l \times n} \rightarrow \Sigma^m$, which satisfies the following properties:

for any l independent (n, d) -sources X_1, X_2, \dots, X_l ,

$$\text{stat}(E(X_1, X_2, \dots, X_l), U_m) \leq \epsilon$$

we call E a multi-source extractor with error parameter ϵ and min-entropy threshold d .

CHAPTER 3 Extracting Kolmogorov complexity

In this chapter, we consider the following question: is there a polynomial-time procedure E such that for every x with $K(x) \geq m$, $K(E(x, s)) = O(m)$ and $|E(x, s)| = O(m)$? (s is the small amount of advice such that $s = O(\log |x|)$ or $O(1)$).

3.1 Extraction using multi-source extractor

We show that if a string x has linear Kolmogorov rate ($K(x) = \Omega(|x|)$), with the help of constant bits of advice we could output a nearly Kolmogorov-random string of length equal to $O(|x|)$.

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

Theorem 3.1.1. ([2]) *For every constant $0 < \sigma < 1$, and $c > 1$ there exist $l = \text{poly}(1/\sigma, c)$, a constant r and an $(n, \sigma n, l, n, 2^{-cn})$ -extractor E . Moreover, E runs in time n^r .*

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

Theorem 3.1.2. *For every $0 < \sigma < 1$ and large enough n , 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

$$K(E(x_1, \dots, x_l)) \geq n - 10l \log n - \text{dep}(x).$$

Proof. Let $0 < \sigma' < \sigma$. By Theorem 3.1.1, there is a constant l and a polynomial-time computable multi-source extractor E such that if H_1, \dots, H_l are independent sources each with min-entropy at least $\sigma'n$, then $E(H_1, \dots, H_l)$ 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,

$$K(E(x_1, \dots, x_l)) < n - 10l \log n - \text{dep}(x).$$

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

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

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

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

By our assumption $\langle x_1, \dots, x_l \rangle$ belongs to *Small*. We use this to arrive at a contradiction regarding the Kolmogorov complexity of $x = x_1 x_2 \dots x_l$. 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 $\sigma'n$, $E(H_1, \dots, H_l)$ is 2^{-5n} -close to U_n . Then

$$\left| Pr[E(H_1, \dots, H_l) \in Z] - Pr[U_n \in Z] \right| \leq 2^{-5n}. \quad (3.1)$$

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 $H_1 \times \dots \times H_l$ places a weight of at least $2^{-(m_1+\dots+m_l+l)}$ on each element of $Small$. Therefore,

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

and since $|Z| \leq 2^{n-10l \log n - dep(x)}$, from (3.1) we obtain

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

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

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

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

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

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 $z = z_1 \dots z_l$, 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 $y = E(z_1, \dots, z_l)$. Now verify that $K(y)$ is at most $n - \text{dep}(x) - 10l \log n$. This again can be done by running programs of the length at most $n - \text{dep}(x) - 10l \log n$ in a dovetail manner. If it is discovered that $K(y)$ is at most $n - \text{dep}(x) - 10l \log n$, then accept z .

So given the values of parameters n , $\text{dep}(x)$, 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 , $\text{dep}(x)$, n , m_1, \dots, m_l , and l , as auxiliary inputs. Since the m_i 's and $\text{dep}(x)$ 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, \dots, x_l are independent strings with $K(x_i) \geq \sigma n$, then $E(x_1, \dots, x_l)$ is a Kolmogorov random string of length n .

Corollary 3.1.3. *For every constant $0 < \sigma < 1$, there exists a constant l , and a polynomial-time computable function E such that if x_1, \dots, x_l are n -bit strings such $K(x_i) \geq \sigma n$, and $K(x_1 x_2 \dots x_l) = \sum K(x_i) - O(\log n)$, then $E(x_1, \dots, x_l)$ is Kolmogorov random, i.e.,*

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

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 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 constant bits of advice.

Theorem 3.1.4. *For all real numbers $0 < \alpha < \beta < 1$ there exist a constant $0 < \gamma < 1$, 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 $\text{rate}(x) \geq \alpha$, there exists an advice string a_x such that*

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

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 .

Proof. Let $\alpha' < \alpha$ and $\epsilon < \min\{1 - \beta, \alpha'\}$. Let $\sigma = (1 - \epsilon)\alpha'$. Using parameter σ in Theorem 3.1.2, 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 $\text{rate}(x) = \nu \geq \alpha$. 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 $\nu' = \text{rate}(x)$ 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 $\text{dep}(x) \geq \gamma ln$.

Case 3. Case 1 does not hold and $\text{dep}(x) < \gamma ln$.

We have two claims about Cases 1 and 2:

Claim 3.1.4.1. *Assume Case 1 holds. There exists i , $1 \leq i \leq l$, such that $\text{rate}(x_i) \geq \nu' + \gamma$.*

Proof. Proof of claim 3.1.4.1 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. \square

Claim 3.1.4.2. *Assume Case 2 holds. There exists i , $1 \leq i \leq l$, $\text{rate}(x_i) \geq \nu' + \gamma$.*

Proof. Proof of claim 3.1.4.2 By definition,

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

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

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

Thus there exists i such that $\text{rate}(x_i) \geq \nu' + \gamma$. \square

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 3.1.4.1 the index i such that $\text{rate}(x_i) \geq \nu' + \gamma$.

- If Case 2 holds, then from Claim 3.1.4.2 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

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

If Case 3 holds, we have $R(x, a_x) = E(x)$ and by Theorem 3.1.2, $K(E(x)) \geq n - 10 \log n - \gamma \ln n$.

Since $\gamma \leq \frac{1-\beta'}{t}$, 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 . \square

We now prove our main theorem.

Theorem 3.1.5. *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 3.1.4 iteratively. Each application of R outputs a string whose rate is at least β or is at least γ more than the rate of the input string. Applying R at most $k = \lceil (\beta - \alpha)/\gamma \rceil$ 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 $|x|/C_2$. \square

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

Definition. Let $x = x_1x_2 \cdots x_k$ 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 3.1.2.

Theorem 3.1.6. *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, \dots, x_l are n -bit strings with $KS^f(x_i) \geq \sigma n$, $1 \leq i \leq l$, then*

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

Similarly we obtain the following extension of Theorem 3.1.5.

Theorem 3.1.7. *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 $|R(x, a_x)| \geq |x|/C_2$.

3.2 Extraction using single-source extractor

The previous extraction requires that the original string x has linear Kolmogorov rate. What if x has sub-linear Kolmogorov rate? We show that in this case we can also convert x to a Kolmogorov-random string. However, this process needs $O(\log n)$ bits of advice.

Theorem 3.2.1 (Trevisan's Extractor [26]). *For any constant $\alpha < \delta < 1$ and for any integer n which is large enough, there is a $(n, n^\delta, O(\log n), n^\alpha, \epsilon)$ -extractor*

$$E : \Sigma^n \times \Sigma^{O(\log n + \log \frac{1}{\epsilon})} \rightarrow \Sigma^{n^\alpha}$$

Applying Trevisan's Extractor, we get the following theorem.

Theorem 3.2.2. *For any constant $\alpha' < \delta' < 1$, there is a poly-time computable procedure E such that for any string x of large enough length n with property that $K(x) \geq n^{\delta'}$, E takes $O(\log n)$ bits advice denoted as s and has the following properties:*

1. $|E(x, s)| = n^{\alpha'}$
2. $K(E(x, s)) > |E(x, s)| - O(\log n)$
3. *the content of s depends on x*

which means that $E(x, s)$ outputs a string of nearly maximal Kolmogorov complexity.

Proof. We take Trevisan's extractor in Theorem 3.2.1 as our poly-time computable procedure E and choose the parameters as follows: let $\alpha = \alpha'$ and δ be a constant smaller than δ' ; Let the error parameter ϵ equal to $\frac{1}{n^\beta}$ for some arbitrary constant $\beta > 10$, therefore the seed length of E is $O(\log n + \beta \log n) = O(\log n)$ denoted as l .

Suppose $K(x) = k$. Define the following sets:

$$Input = \{v \mid v \in \Sigma^n, K(v) \leq k\}$$

$$Z = \{u \mid u \in \Sigma^{n^\alpha}, K(u) \leq n^\alpha - 10 \log n\}$$

$$Small = \{\langle y, w \rangle \mid y \in Input, w \in \Sigma^l, E(y, w) \in Z\}$$

$$AllSmall = \{y \in Input \mid \forall w \in \Sigma^l, \langle y, w \rangle \in Small\}$$

Define the uniform distribution U_{Input} over the set $Input$, it satisfies the min-entropy threshold of E . Consider the set $Less = \{u \mid u = v0^{n-n^\delta}\}$ which is a subset of $Input$. So There are more than many 2^{n^δ} elements within $Input$. By the property of extractor the output distribution $E(U_{Input}, U_l)$ is ϵ -close to U_{n^α} .

From the above observation, we obtain an upper bound for $Small$.

$$|Small| \leq 2^k \times 2^l \times \left(\frac{1}{n^{10}} + \frac{1}{n^\beta}\right) \leq 2^k \times 2^l \times \frac{1}{n^9}$$

As well as an upper bound for $AllSmall$

$$|AllSmall| \leq \frac{|Small|}{2^l} \leq 2^k \times \frac{1}{n^9}$$

So the index of enumeration for $AllSmall$ is no larger than $k - 9 \log n$ bits.

In fact, given the parameters $\alpha, \delta, n, \epsilon, l$ and given the value k , there is an algorithm accepting all of the elements in $AllSmall$. Working in a dovetailing manner, the algorithm simulates the universal Turing machine over all of programs as follows: On input $y \in \Sigma^n$, first check whether $K(y) \leq k$; if it holds, then for every pair $\langle y, w \in \Sigma^l \rangle$, check whether $K(E(y, w)) \in Small$; if all computations return yes, then accepts y .

At the same time, we could have a program P enumerate all of the elements in $AllSmall$ given $\alpha, \delta, n, \epsilon, l$ and k .

To describe P itself, we only need $O(1)$ bits. Additionally we use at most $O(\log n)$ bits for each parameter. So combining the index bits, we may specify every string in the enumeration with the description of no more than $k - 9 \log n + 6 \log n + O(1) = k - 2 \log n$ bits.

Next we show that there exists a seed s for x such that $K(E(x, s)) > |E(x, s)| - 10(\log n)$ and prove it via contradiction.

Suppose not. Then $\forall w \in \Sigma^l, E(x, w) \in Small$. Therefore $x \in AllSmall$, by the above argument, $K(x) \leq k - 2 \log n$, which is contradictory to the fact that k is the Kolmogorov complexity of x .

□

Theorem can also be extended to space-bounded Kolmogorov complexity.

Theorem 3.2.3. *Let g be a polynomial and let α' and δ' be constants with $0 < \alpha' < \delta' < 1$. There exists a polynomial f , polynomial-time procedure $E(\cdot, \cdot)$ such that for every string x of large enough length n with $KS^g(x) > n^{\delta'}$, there exists a short advice s of length $O(\log n)$ such that the following properties hold:*

1. $|E(x, s)| = n^{\alpha'}$
2. $KS^f(E(x, s)) > |E(x, s)| - O(\log n)$
3. *the content of s depends on x .*

CHAPTER 4 Zero-one laws

4.1 Dimension 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 4.1.1. *If $\text{Dim}_{\text{pspace}}(E) > 0$, then $\text{Dim}_{\text{pspace}}(E/O(1)) = 1$.*

For the theorem we use the following lemma, which is proved using Theorem 3.1.7.

Lemma 4.1.2. *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 $L \in E$ with $\text{Rate}^f(L) \geq \alpha$, then there exists $L' \in E/O(1)$ with $\text{Rate}^g(L') \geq \theta$.*

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, \dots 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 , $\text{rate}^f(y_i) \geq \alpha/4$.

We now define y_i . We consider three cases.

Case 1. $\text{rate}^f(L \upharpoonright n_i) \geq \alpha/4$. Divide $L \upharpoonright n_i$ into $K/(C - 1)$ 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 $\text{rate}^f(y_i) \geq \alpha/4$.

Case 2. Case 1 does not hold and for every j , $n_i < j < n_{i+1}$, $\text{rate}^f(L \upharpoonright j) < \alpha$. 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 $\text{rate}^f(L \upharpoonright j) > \alpha$. Divide $L \upharpoonright [n_i, n_{i+1}]$ 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 , $\text{rate}^f(L \upharpoonright j) \geq \alpha$, for infinitely many i either Case 1 or Case 3 holds. Thus for infinitely many i , $\text{rate}^f(y_i) \geq \alpha/4$.

By Theorem 3.1.7, there is a procedure R with such that given a string x with $\text{rate}^f(x) \geq \alpha/4$, and the advice a_x , $\text{rate}^g(R(x, a_x)) \geq \beta$.

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

Claim 4.1.2.1. $|w_{i+1}| \geq (C-1) \sum_{j=1}^i |w_j|$.

Proof of claim 4.1.2.1. 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 4.1.2.2. For infinitely many i , $rate^g(w_1 \cdots w_i) \geq \theta$.

Proof of claim 4.1.2.2. 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 4.1.2.1, $|w_i| \geq (C-1)(|w_1| + \cdots + |w_{i-1}|)$. Thus for infinitely many i , $rate^g(w_1 \cdots w_i) \geq \frac{(C-1)\beta}{C} - o(1) \geq \theta$. \square

We define $w_1 w_2 \cdots$ to be the characteristic sequence of L' . Then by Claim 4.1.2.2, $Rate^g(L') \geq \theta$.

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 $L \upharpoonright n_{i+1}$. 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 $L \upharpoonright n_{i+1}$. Since L is in E , this places L' in $E/O(1)$.

This completes the proof of Lemma 4.1.2. \square

Proof of Theorem 4.1.1. 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 2.2.2, 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 $\text{Dim}_{\text{pspace}}(E)$. By Theorem 2.2.2, for every polynomial f , there exists a language $L \in E$ with $Rate^f(L) \geq \alpha$.

By Lemma 4.1.2, 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. \square

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 $L \in \text{ESPACE}$, it can be shown that $L' \in \text{ESPACE}$. With these observations, we obtain the following zero-one laws.

Theorem 4.1.3. *Each of the following is either 0 or 1.*

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

We remark that in Theorems 4.1.1 and 4.1.3, if we replace E by EXP , the theorems still hold. The proofs also go through for other classes such as BPEXP , $\text{NEXP} \cap \text{coNEXP}$, and $\text{NEXP}/poly$.

Theorems 4.1.1 and 4.1.3 concern strong dimension. For dimension, the situation is more complicated. Using similar techniques, we can prove that

if $\text{dim}_{\text{pspace}}(E) > 0$, then $\text{dim}_{\text{pspace}}(E/O(1)) \geq 1/2$.

Analogously, we can obtain zero-half laws for the pspace-dimension of $E/poly$, etc.

4.2 Scaled dimension zero-one Law

Hitchcock, Lutz and Mayordomo [7] introduced resource-bounded scaled dimension. In [6], Hitchcock et. al gave a Kolmogorov characterization of scaled dimension.

Theorem 4.2.1 (Hitchcock, López-Valdés and Mayordomo [6]). *Let poly denote all polynomials. For every class X of languages,*

$$\text{Dim}_{\text{pspace}}^{(1)}(X) = \inf_{g \in \text{poly}} \sup_{L \in X} \limsup_{n \rightarrow \infty} \frac{\log \text{rate}^g(L \upharpoonright n)}{\log n}.$$

and

$$\text{Dim}_{\text{pspace}}^{(-1)}(X) = \inf_{g \in \text{poly}} \sup_{L \in X} \limsup_{n \rightarrow \infty} \left(1 - \frac{\log \text{rate}^g(L \upharpoonright n)}{\log n}\right).$$

By applying Theorem 3.2.3 and using the similar segmentation technique in section 4.1, we could have a zero-one law regarding scaled strong dimension using single-source extraction in the previous section.

Theorem 4.2.2. *Either of the following conditions holds*

1. $\text{Dim}_{\text{pspace}}^{(1)}(\mathbf{E}/O(n)) = 0$
2. $\text{Dim}_{\text{pspace}}^{(-1)}(\mathbf{E}/O(n)) = 1$.

We omit the tedious proof here and remark that because we have $O(\log n)$ bits of advice, we could adopt more flexible segmentation. The relative gap between different segments might be much larger for Theorem 4.2.2. For instance, suppose at step i , the length of segment is n_i , then at next step $i + 1$, we may have a segment of length n_i^ω for some constant $\omega > 1$. By manipulating ω , we could obtain our desired result.

4.3 Open questions

In the end, we mention several interesting open questions as follows:

1. We have proved the zero-one laws for space-bounded dimension, yet we do not know whether we could extend them to corresponding time-bounded dimension. It seems that

the only bottleneck is the improvement of reconstruction techniques. Can someone push the results to time-bounded dimension?

2. Can condensers be used to obtain results regarding Kolmogorov complexity extraction?

BIBLIOGRAPHY

- [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. 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.
- [4] L. Fortnow and J. H. Lutz. Prediction and dimension. *Journal of Computer and System Sciences*, 70(4):570–589, 2005.
- [5] J. M. Hitchcock. *Effective Fractal Dimension: Foundations and Applications*. PhD thesis, Iowa State University, 2003.
- [6] J. M. Hitchcock, M. López-Valdés, and E. Mayordomo. Scaled dimension and the Kolmogorov complexity of Turing-hard sets. In *Proceedings of the 29th International Symposium on Mathematical Foundations of Computer Science*, pages 476–487. Springer-Verlag, 2004.

- [7] J. M. Hitchcock, J. H. Lutz, and E. Mayordomo. Scaled dimension and nonuniform complexity. *Journal of Computer and System Sciences*, 69(2):97–122, 2004.
- [8] J. M. Hitchcock, J. H. Lutz, and E. Mayordomo. The fractal geometry of complexity classes. *SIGACT News*, 36(3):24–38, September 2005.
- [9] R. Impagliazzo and P. Moser. A zero-one law for RP. In *Proceedings of the 18th IEEE Conference on Computational Complexity*, pages 43–47. IEEE Computer Society, 2003.
- [10] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer-Verlag, Berlin, 1997. Second Edition.
- [11] 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.
- [12] J. H. Lutz. Almost everywhere high nonuniform complexity. *Journal of Computer and System Sciences*, 44(2):220–258, 1992.
- [13] J. H. Lutz. The quantitative structure of exponential time. In L. A. Hemaspaandra and A. L. Selman, editors, *Complexity Theory Retrospective II*, pages 225–254. Springer-Verlag, 1997.
- [14] J. H. Lutz. Dimension in complexity classes. *SIAM Journal on Computing*, 32(5):1236–1259, 2003.
- [15] J. H. Lutz and E. Mayordomo. Twelve problems in resource-bounded measure. *Bulletin of the European Association for Theoretical Computer Science*, 68:64–80, 1999. Also in *Current Trends in Theoretical Computer Science: Entering the 21st Century*, pages 83–101, World Scientific Publishing, 2001.

- [16] E. Mayordomo. A Kolmogorov complexity characterization of constructive Hausdorff dimension. *Information Processing Letters*, 84(1):1–3, 2002.
- [17] P. Moser. BPP has effective dimension at most $1/2$ unless $BPP = EXP$. Technical Report TR03-029, Electronic Colloquium on Computational Complexity, 2003.
- [18] N. Nisan and A. Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 42(2):149–167, 1999.
- [19] N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.
- [20] 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.
- [21] 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.
- [22] 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.
- [23] 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.
- [24] A. Srinivasan and D. Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28(4):1433–1459, 1999.

- [25] 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.
- [26] L. Trevisan. Extractors and pseudorandom generators. *Journal of the ACM*, 48(1):860–879, 2001.
- [27] D. van Melkebeek. The zero-one law holds for BPP. *Theoretical Computer Science*, 244(1–2):283–288, 2000.
- [28] N. Vereshchagin and M. Vyugin. Independent minimum length programs to translate between given strings. *Theoretical Computer Science*, 271:131–143, 2002.
- [29] D. Zuckerman. Randomness-optimal oblivious sampling. *Random Structures and Algorithms*, 11:345–367, 1997.

ACKNOWLEDGEMENTS

I feel very honored to take this opportunity to express my thanks to those who unselfishly helped me with conducting the research and writing this thesis, especially my family and friends.

First and foremost, I would like to thank Dr. Pavan Aduri for bringing me into complexity theory and for his patience, guidance, insights and words of encouragement which benefited me a lot during my journey in this field. I would also like to thank my committee members: Dr. Jack H. Lutz and Dr. Srikanta Tirthapura. I would additionally like to thank Dr. Lutz for his guidance throughout the initial stages of my graduate career and Dr. Tirthapura for his inspirational introduction of randomized algorithm.

Moreover I would like to thank fellows in our theory group for giving me a lot of enlightening discussions.

The research presented in this thesis was supported in part by National Science Foundation Grant #0430807.