Extractors and Pseudorandom Generators*

Luca Trevisan†

May 11, 2000

Abstract

We introduce a new approach to constructing extractors. Extractors are algorithms that transform a "weakly random" distribution into an almost uniform distribution. Explicit constructions of extractors have a variety of important applications, and tend to be very difficult to obtain.

We demonstrate an unsuspected connection between extractors and pseudorandom generators. In fact, we show that every pseudorandom generator of a certain kind is an extractor.

A pseudorandom generator construction due to Impagliazzo and Wigderson, once reinterpreted via our connection, is already an extractor that beats most known constructions and solves an important open question. We also show that, using the simpler Nisan-Wigderson generator and standard error-correcting codes, one can build even better extractors with the additional advantage that both the construction and the analysis are simple and admit a short self-contained description.

1 Introduction

An *extractor* is an algorithm that converts a "weak source of randomness" into an almost uniform distribution by using a small number of additional truly random bits. Extractors have several important applications (see e.g. [Nis96]). In this paper we show that pseudorandom generator constructions of a certain kind are extractors. Using our connection and some new ideas we describe constructions of extractors that improve most previously known constructions and that are simpler than previous ones.

1.1 Definitions

We now give the formal definition of an extractor and state some previous results. We first need to define the notions of *min-entropy* and *statistical difference*.

We say that (the distribution of) a random variable X of range $\{0,1\}^n$ has min-entropy at least k if for every $x \in \{0,1\}^n$ it holds $\mathbf{Pr}[X = x] \leq 2^{-k}$. If 2^k is an integer, then a canonical example of a distribution having min-entropy k is the uniform distribution over a set $S \subseteq \{0,1\}^n$ of cardinality 2^k. Indeed, it is implicit in [CG88] that if a distribution has min-entropy k then it is a convex combination of distributions each one of which is uniform over a set of size 2^k. We will consider distributions of min-entropy k as the formalization of the notion of weak sources of randomness containing k "hidden" bits of randomness. In the rest of this paper we will often call (n,k)-*source* a

*An extended abstract of this paper is to appear in the Proceedings of the 31st ACM Symposium on Theory of Computing [Tre99].

†`luca@cs.columbia.edu`. Computer Science Department, Columbia University.

random variable X ranging over $\{0,1\}^n$ and having min-entropy at least k. The use of min-entropy to measure "hidden randomness" has been advocated by Chor and Goldreich [CG88] and, in full generality, by Zuckerman [Zuc90]. The statistical difference between two random variables X and Y with range $\{0,1\}^n$ is defined as

$$||X - Y|| = \max_{T:\{0,1\}^n \to \{0,1\}} |\,\mathbf{Pr}[T(X) = 1] - \mathbf{Pr}[T(Y) = 1]| = \frac{1}{2}\sum_v |\,\mathbf{Pr}[X = v] - \mathbf{Pr}[Y = v]|$$

and we say that X and Y are ε-close if $||X - Y|| \leq \varepsilon$. For an integer l we denote by U_l a random variable that is uniform over $\{0,1\}^l$.

A function $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ is a (k,ε)-extractor if for every random variable X of min entropy at least k it holds that $Ext(X, U_t)$ is ε-close to the uniform distribution over $\{0,1\}^m$. A weaker kind of combinatorial construction has also been considered: A function $Disp : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ is a (k,ε)-disperser if for every subset $S \subseteq \{0,1\}^m$ such that $|S| > \varepsilon 2^m$ and for every X of min-entropy k it holds $\mathbf{Pr}[Disp(X, U_t) \in S] > 0$.

One would like to have, for every n and k, constructions where t is very small and m is as close to k as possible. There are some limitations towards this goal: One can show that, if $k < n - 1$ and $\varepsilon < 1/2$, then it must be the case that $t \geq \max\{\log(1/\varepsilon) - 1\ ,\ \log(n-k)\}$ [NZ93], and also it must be the case that $m \leq k + t - 2\log(1/\varepsilon) + O(1)$ [RTS97]. It is possible to show (non-constructively) that for every n, k, ε, there is a (k,ε)-extractor $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ where $t = O(\log n/\varepsilon)$ and $m = k + t - 2\log(1/\varepsilon) - O(1)$. It is an open question to match such bounds with polynomial-time computable functions Ext.

1.2 Previous Work and Applications

The natural application of extractors is to allow the simulation of randomized algorithms even in (realistic) settings where only weak sources of randomness are available. This line of research has a long history, that dates back at least to von Neumann's algorithm for generating a sequence of unbiased bits from a source of biased but identically distributed and independent bits [vN51]. More recent work by Santha and Vazirani [SV86] and Vazirani and Vazirani [VV85] considered much weaker sources of randomness (that they call "slightly random" sources) that are still sufficient to allow simulations of arbitrary randomized algorithms. These results were generalized by Chor and Goldreich [CG88] and Cohen and Wigderson [CW89], and finally by Zuckerman [Zuc90], who introduced the current definition (based on min-entropy) of weak random sources and a construction of extractors (although the term *extractor* was coined later, in [NZ93]). The main question about simulation of randomized algorithms using weak random sources can be stated as follows: suppose that for every n we have access to a $(n, k(n))$-source, and that we are given a polynomial-time randomized algorithm that we want to simulate given only one access to one of the sources: what is the most slowly growing function $k(\cdot)$ such that we can have a polynomial-time simulations? For a "black-box" simulation, where the randomized algorithm is given as an oracle, it is impossible to solve the simulation problem in polynomial time with a family of $(n, n^{o(1)})$-sources. The best one can hope to achieve is to have, for every $\delta > 0$, a simulation that works in polynomial time given a (n, n^δ)-source. We will call such a simulation an entropy-rate optimal simulation. Improved constructions of extractors appeared in [NZ93, SZ94, TS96, Zuc96b], but none of these constructions implies an entropy-rate optimal simulation of randomized algorithms. Dispersers are objects similar to, but less powerful than, extractors. Randomized algorithms having one-sided error probability can be simulated by using weak random sources and dispersers. Saks et al. [SSZ98] give a construction of dispersers that implies an entropy-rate optimal simulation of one-sided error randomized

Reference	Min entropy k	Output length m	Additional randomness t	Type
[GW97]	$n-a$	$n-\Theta(a)$	$O(a)$	Extractor
[Zuc96b]	$k=\Omega(n)$	$m=(1-\alpha)k$	$t=O(\log n)$	Extractor
[TS96]	any k	$m=k$	$t=O((\log n)^9)$	Extractor
[TS96]	$k=n^{\Omega(1)}$	$m=k^{1-\alpha}$	$t=O(\log n \log \cdots \log n)$	Extractor
[SSZ98]	$k=n^{\Omega(1)}$	$m=k^{1-\alpha}$	$t=O(\log n)$	Disperser
[TS98]	any k	$m=k-\text{poly}\log n$	$t=O(\log n)$	Disperser
This paper	$k=n^{\Omega(1)}$	$m=k^{1-\alpha}$	$t=O(\log n)$	Extractor
	any k	$m=k^{1-\alpha}$	$t=O((\log^2 n)/\log k)$	Extractor

Table 1: Parameters in previous constructions and our construction of (k,ε) extractors $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$. In the expressions, ε is fixed and arbitrarily small, and $\alpha > 0$ is an arbitrarily small constant. $O(\cdot)$ notations hide dependencies on ε and α.

algorithms with weak random sources. Andreev et al. [ACRT99] show how to use the dispersers of Saks et al. in order to give entropy-rate optimal simulations of general randomized algorithms using weak random sources. The result of Andreev et al. leaves open the question of whether there exists a construction of extractors that is good enough to imply directly such entropy-rate optimal simulations.

Extractors are also used to derandomize randomized space-bounded computations [NZ93] and for randomness-efficient reduction of error in randomized algorithms (see [Zuc96b, GZ97] and references therein). They yield oblivious samplers (as defined in [BR94]), that have applications to interactive proofs (see [Zuc96b] and references therein). They also yield expander graphs, as discovered by Wigderson and Zuckerman [WZ93], that in turn have applications to superconcentrators, sorting in rounds, and routing in optical networks. Constructions of expanders via constructions of extractors and the Wigderson-Zuckerman connection appeared in [NZ93, SZ94, TS96], among others. Extractors can also be used to give simple proofs of certain complexity-theoretic results [GZ97], and to prove certain hardness of approximation results [Zuc96a]. An excellent introduction to extractors and their applications is given by a recent survey by Nisan [Nis96] (see also [NTS98], and [Gol99] for a broader perspective).

In Table 1 we summarize the parameters of the previous best constructions, and we state two special cases of the parameters arising in our construction.

1.3 Our Result

The extractors constructed in this paper work for any min-entropy $k = n^{\Omega(1)}$, extract a slightly sub-linear fraction of the original randomness (i.e. the length of the output is $m = k^{1-\alpha}$ for an arbitrarily small $\alpha > 0$) and use $O(\log n)$ bits of true randomness. In fact, a more general result holds, as formalized below.

Theorem 1 (Main) *There is an algorithm that on input parameters n, $k \le n$, $22 \le m < k/2$, $0 < \varepsilon < 2^{-k/12}$ computes in* $\text{poly}(n, 2^t)$ *time a (k,ε)-extractor $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ where*

$$t = O\left(\frac{(\log n/\varepsilon)^2}{\log(k/2m)} \cdot e^{\frac{\ln m}{\log(k/2m)}} \right) .$$

In particular, for any fixed constants $\varepsilon > 0$ and $0 < \gamma' < \gamma < 1$ we have for every n an explicit polynomial-time construction of an (n^γ, ε)-extractor $Ext : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to \{0,1\}^{n^{\gamma'}}$.

It should be noted that the running time of our extractor is exponential in the parameter t (additional randomness), and so the running time is super-polynomial when the additional randomness is super-logarithmic. However, the 2^t factors in the running time of the extractor is payed only once, to construct a combinatorial object (a "design") used by the extractor. After the design is computed, each evaluation of the extractor can be implemented in linear time plus the time that it takes to compute an error-correcting encoding of the input of the extractor. It is possible to generate designs more efficiently, and so to have a polynomial-time extractor construction for every min-entropy. We omit the details of such construction, since the construction of "weak designs" in [RRV99] (see below) give better extractors and is also more efficiently computable.

Our construction improves on the construction of Saks, Srinivasan and Zhou [SSZ98] since we construct an extractor rather than a disperser, and improves over the constructions of Ta-Shma [TS96] since the additional randomness is logarithmic instead of slightly super-logarithmic. The best previous construction of extractors using $O(\log n)$ additional randomness was the one of Zuckerman [Zuc96b], that only works when the min-entropy is a constant fraction of the input length, while in our construction every min-entropy of the form n^γ is admissible. (On the other hand, the extractor of [Zuc96b] extracts a constant factor of the entropy, while we only extract a constant root of it.) Our construction yields an entropy-rate optimal simulation of randomized algorithms using weak random sources. In contrast to the result of [ACRT99] we can use a weak random source to generate almost uniformly distributed random bits independently of the purpose for which the random bits are to be used.[1]

Our construction is not yet the best possible, since we lose part of the randomness of the source and because the additional randomness is logarithmic only as long as $k = n^{\Omega(1)}$. (See also discussion in Section 1.6 below.)

1.4 Techniques

This paper contains two main contributions.

The first one is a connection (outlined in Section 2) between pseudorandom generators of a certain kind and extractors. Our connection applies to certain pseudorandom generator constructions that are based on the (conjectured) existence of predicates (decision problems) that can be uniformly computed in time $t(n)$ but cannot be solved by circuits of size much smaller than $t(n)$. The analysis of such constructions shows that if the predicate is hard, then it is also hard to distinguish the output of the generator from the uniform distribution. This implication is proved by means of a reduction showing how a circuit that is able to distinguish the output of the generator from the uniform distribution can be transformed into a slightly larger circuit that computes the predicate. (Impagliazzo and Wigderson [IW97] present one such construction with very strong parameters.) Our result is that if the (truth table of the) predicate is chosen *randomly*, according to a distribution with sufficiently high min-entropy, then the output of the generator is *statistically close* to uniform. This statement is incomparable with standard analyses: we use a stronger assumption (that the predicate is *random* instead of *fixed and hard*) and prove a stronger conclusion (that the output is *statistically close* to, instead of *indistinguishable* from, the uniform distribution). An immediate application is that a pseudorandom generator construction of this kind *is* an extractor. Our result

[1] Andreev et al. [ACRT99] show how to produce a sequence of bits that "look random" to a specific algorithm, and their construction works by having oracle access to the algorithm. So it is not possible to generate random bits "offline" before fixing the application where the bits will be used.

has a straightforward proof, based on a simple counting argument. The main contribution, indeed, is the *statement* of the result, rather than its *proof*, since it involves a new, more general, way of looking at pseudorandom generator constructions. The Impagliazzo-Wigderson generator, using our connection, is an extractor that beats some previous constructions and that is good enough to imply entropy-rate optimal simulations of randomized algorithms. We stress that although the Impagliazzo-Wigderson generator is known to be a pseudorandom generator only under unproved conjectures, it is *unconditionally* a good extractor (i.e. we do not use any complexity-theoretic assumption in our work).

Our second contribution is a construction that is simpler to describe and analyse (the generator of Impagliazzo and Wigderson is quite complicated) and that has somewhat better parameters. Our idea is to use a pseudorandom generator construction due to Nisan and Wigderson [NW94], that is considerably simpler than the one of Impagliazzo and Wigderson (indeed the construction of Impagliazzo and Wigderson contains the one of Nisan and Wigderson as one of its many components). The Nisan-Wigderson generator has weaker properties than the Impagliazzo-Wigderson generator, and our ideas outlined in Section 2 would not imply that it is an extractor as well. In Section 3 we show how to use error-correcting codes in order to turn the Nisan-Wigderson generator into a very good extractor. Section 3 contains a completely self-contained treatment of the construction and the analysis.

1.5 Perspective

For starters, our construction improves upon previous ones and solves the question of constructing extractors that use a logarithmic amount of randomness, work for any min-entropy that is polynomially related to the length of the input and have an output that is polynomially related to the amount of entropy. Such a construction has been considered an important open question (e.g. in [NTS98, Gol99]), even after Andreev et al. [ACRT99] showed that one does not need such extractors in order to develop an entropy-rate optimal simulation of randomized algorithms via weak random sources. Indeed, it was not clear whether the novel approach introduced by Andreev et al. was *necessary* in order to have optimal simulations, or whether a more traditional approach based on extractors was still possible. Our result clarifies this point, by showing that the traditional approach suffices.

Perhaps more importantly, our construction is simpler to describe and analyse then the most recent previous constructions, and it uses a very different approach. Hopefully, our approach offers more room for improvement than previous, deeply exploited, ones. Raz et al. [RRV99] have already found improvements to our construction (see below). Tight results may come from some combination of our ideas and previous ones.

Our use of results about pseudorandomness in the construction of extractors may come as a surprise: pseudorandom generation deals with (and takes advantage of) a *computational* definition of randomness, while extractors are combinatorial objects used in a framework where *information-theoretic* randomness is being considered. In the past there have been some instances of (highly non-trivial) results about computational randomness inspired by (typically trivial) information-theoretic analogs, e.g. the celebrated Yao's XOR Lemma and various kind of "direct product" results (see e.g. [GNW95]). On the other hand, it seemed "clear" that one could not go the other way, and have information-theoretic applications of computational results. This prejudice might be one reason why the connection discovered in this paper has been missed by the several people who worked on weak random sources and on pseudorandomness in the past decade (including those who did foundational work in both areas). Perhaps other important results might be proved along similar lines.

1.6 Related Papers

The starting point of this paper was an attempt to show that every disperser can be modified into an extractor having similar parameters. This was inspired by the fact (noted by several people, including Andrea Clementi and Avi Wigderson) that every hitting set generator can be transformed into a pseudorandom generator with related parameters, since the existence of hitting set generators implies the existence of problems solvable in exponential time and having high circuit complexity [ACR98] and the existence of such problems can be used to build pseudorandom generators [BFNW93, IW97]. The fact that an information-theoretic analog of this result could be true was suggested by the work done in [ACRT99], where proof-techniques from [ACR98] were adapted in an information-theoretic setting. We were indeed able to use the Impagliazzo-Wigderson generator in order to show that any construction of dispersers yields a construction of extractors with slightly worse parameters. Oded Goldreich then pointed out that we were not making any essential use of the disperser in our construction, and that we were effectively proving that the Impagliazzo-Wigderson generator is an extractor (this result is described in Section 2). The use of error-correcting codes and of the Nisan-Wigderson generator (as in Section 3) was inspired by an alternative proof of some of the results of [IW97] due to Sudan et al. [STV99].

Shortly after the announcement of the results of this paper, Raz, Reingold and Vadhan [RRV99] devised an improvement to our construction. In our construction, if the input has min-entropy k and the output is required to be of length m, and ε is a constant, then the additional randomness is $O(m^{1/\log(k/2m)}(\log n)^2/\log(k/2m))$, which is very bad if, say, $m = k/3$. In [RRV99], the dependency is $O((\log n)^2/\log(k/m))$, so the randomness is bounded by $O(\log^2 n)$ even when $k = m/3$. Furthermore, the running of the extractors of [RRV99] is poly(n, t) rather than poly$(n, 2^t)$ as in the construction presented in this paper. Raz et al. [RRV99] also show how to recursively compose their construction with itself (along the lines of [WZ93]) and obtain another construction where $k = m$ and the additional randomness is $O(\log^3 n)$. Constructions of extractors with parameters $k = m$ have applications to the explicit construction of expander graphs [WZ93]. In particular, Raz et al. [RRV99] present constructions of expander graphs and of superconcentrators that improve previous ones by Ta-Shma [TS96]. Raz et al. [RRV99] also improve the dependency that we have between additional randomness and error parameter ε.

Organization of the Paper

We present in Section 2 our connection between pseudorandom generator constructions and extractors. The main result of Section 2 is that the Impagliazzo-Wigderson generator [IW97] is a good extractor. In Section 3 we describe and analyse a simpler construction based on the Nisan-Wigderson generator [NW94] and on error correcting codes. Section 3 might be read independently of Section 2.

2 The Connection Between Pseudorandom Generators and Extractors

This section describes our main idea of how to view a certain kind of pseudorandom generator as an extractor. Our presentation is specialized on the Impagliazzo-Wigderson generator, but results might be stated in a more general fashion.

2.1 Computational Indistinguishability and Pseudorandom Generators

We start by defining the notion of computational indistinguishability, and pseudorandom generators, due to Blum, Goldwasser, Micali and Yao [GM84, BM84, Yao82].

Recall that we denote by U_n the uniform distribution over $\{0,1\}^n$. We say that two random variables X and Y with the same range $\{0,1\}^n$ are (S,ε)-indistinguishable if for every $T : \{0,1\}^n \to \{0,1\}$ computable by a circuit of size S it holds

$$|\,\mathbf{Pr}[T(X)=1] - \mathbf{Pr}[T(Y)=1]| \le \varepsilon$$

One may view the notion of ε-closeness (that is, of statistical difference less than ε) as the "limit" of the notion of (S,ε)-indistinguishability for unbounded S.

Informally, a pseudorandom generator is an algorithm $G : \{0,1\}^t \to \{0,1\}^m$ where $t << m$ and $G(U_t)$ is (S,ε)-indistinguishable from U_m, for large S and small ε. For derandomization of randomized algorithms, one looks for generators, say, $G : \{0,1\}^{O(\log m)} \to \{0,1\}^m$ where $G(U_{O(\log m)})$ is $(m^{O(1)}, 1/3)$-indistinguishable from U_m. Such generators (if they were uniformly computable in time $\mathrm{poly}(m)$) would imply deterministic polynomial-time simulations of randomized polynomial-time algorithms.

2.2 Constructions of Pseudorandom Generators Based on Hard Predicates

Given current techniques, all interesting constructions of pseudorandom generators have to rely on complexity-theoretic conjectures. For example the Blum-Micali-Yao [BM84, Yao82] construction (that has different parameters from the ones exemplified above) requires the existence of strong one-way permutations. In a line of work initiated by Nisan and Wigderson [Nis91, NW94], there have been results showing that the existence of hard-on-average decision problems in E is sufficient to construct pseudorandom generators. (Recall that E is the class of decision problems solvable deterministically in time $2^{O(n)}$ where n is the length of the input.) Babai et al. [BFNW93] present a construction of pseudorandom generators that only requires the existence of decision problems in E having high worst-case complexity. An improved construction of pseudorandom generators from worst-case hard problems is due to Impagliazzo and Wigderson [IW97], and it will be the main focus of this section. The constructions of [NW94, BFNW93, IW97] require *non-uniform* hardness, that is, use *circuit size* as a complexity measure. (Recent work demonstrated that non-trivial constructions can be based on *uniform* worst-case conditions [IW98].)

The main result of [IW97] is that if there is a decision problem solvable in time $2^{O(n)}$ that cannot be solved by circuits of size less than $2^{\gamma n}$, for some $\gamma > 0$, then $P = BPP$, i.e. every randomized polynomial time algorithm has a deterministic polynomial-time simulation. A precise statement of the result of Impagliazzo and Wigderson follows.

Theorem 2 ([IW97]) *Suppose that there exists a family $\{P_l\}_{l\ge 0}$ of predicates $P_l : \{0,1\}^l \to \{0,1\}$ that is decidable in time $2^{O(l)}$, and a constant $\gamma > 0$ such that, for every sufficiently large l, P_l has circuit complexity at least $2^{\gamma l}$.*

Then for every constant $\varepsilon > 0$ and parameter m there exists a pseudorandom generator $IW^{(m)} : \{0,1\}^{O(\log m)} \to \{0,1\}^m$ computable in time $\mathrm{poly}(m)$ such that $IW^{(m)}(U_{O(\log m)})$ is $(O(m),\varepsilon)$-indistinguishable from the uniform distribution, and $P = BPP$.

Results about pseudorandomness are typically proved by contradiction. Impagliazzo and Wigderson prove Theorem 2 by establishing the following result.

Lemma 3 ([IW97]) *For every pair of constants $\varepsilon > 0$ and $\delta > 0$ there exists a positive constant $0 < \alpha = \alpha(\varepsilon, \delta) < \delta$ and an algorithm that on input a length parameter l and having oracle access to a predicate $P : \{0,1\}^l \to \{0,1\}$ computes a function $IW_P : \{0,1\}^t \to \{0,1\}^m$ in* poly(m) *time, where $t = O(l)$ and $m = 2^{\alpha l}$ such that for every $T : \{0,1\}^m \to \{0,1\}$, if*

$$|\,\mathbf{Pr}[T(IW_P(U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| > \varepsilon$$

then P is computed by a circuit A that uses T-gates and whose size is at most $2^{\delta l}$.

By a "circuit with T-gates" we mean a circuit that can use ordinary fan-in-2 AND and OR gates and fan-in-1 NOT gates, as well as a special gate (of fan-in m) that computes T with unit cost. This is the non-uniform analog of a computation that makes oracle access to T.

It might not be immediate to see how Theorem 2 follows from Lemma 3. The idea is that if we have a predicate $P : \{0,1\}^l \to \{0,1\}$ that cannot be computed by circuits of size $2^{2\delta l}$, then $IW_P(U_t)$ is $(2^{\delta l}, \varepsilon)$-indistinguishable from uniform. This can be proved by contradiction: if T is computed by a circuit of size $2^{\delta l}$ and is such that

$$|\,\mathbf{Pr}[T(IW_P(U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| > \varepsilon$$

then there exists a circuit A of size at most $2^{\delta l}$ that uses T-gates such that A computes P. If each T-gate is replaced by the circuit that computes T, we end up with a circuit of size at most $2^{2\delta l}$ that computes P, a contradiction to our initial assumption.

We stress that Lemma 3 has not been stated in this form by Impagliazzo and Wigderson [IW97]. In particular, the fact that their analysis applies to every predicate P and to every function T, *regardless of their circuit-complexity*, was not explicitely stated. On the other hand, this is not a peculiar or surprising property of the Impagliazzo-Wigderson construction: in general in complexity theory and in cryptography the correctness of a transformation of an object with certain assumed properties (e.g. a predicate with large circuit complexity) into an object with other properties (e.g. a generator whose output is indistinguishable from uniform) is proved by "black-box" reductions, that work by making "oracle access" to the object and making no assumptions about it.

We also mention that three recent papers exploit the hidden generality of the pseudorandom generator construction of Impagliazzo and Wigderson, and of the earlier one by Nisan and Wigderson [NW94]. Arvind and Köbler [AK97] observe that the analysis of the Nisan-Wigderson generator extends to "non-deterministic circuits," which implies the existence of pseudorandom generators for non-deterministic computations, under certain assumptions. Klivans and Van Melkebeek [KvM99] observe similar generalizations of the Impagliazzo-Wigderson generator for arbitrary non-uniform complexity measures having certain closure properties (which does not include non-deterministic circuit size, but includes the related measure "size of circuits having an NP-oracle"). The construction of pseudorandom generators under uniform assumptions by Impagliazzo and Wigderson [IW98] is also based on the observation that the results of [BFNW93] can be stated in a general form where the hard predicate is given as an oracle, and the proof of security can also be seen as the existence of an oracle machine with certain properties.

A novel aspect in our view (that is not explicit in [IW97, AK97, KvM99, IW98]) is to see the Impagliazzo-Wigderson construction as an algorithm that takes two inputs: the truth-table of a predicate and a seed. The Impagliazzo-Wigderson analysis says something meaningful even when the predicate is not *fixed and hard* (for an appropriate complexity measure), but rather *supplied dynamically* to the generator. In the rest of this section, we will see that if the (truth table of the) predicate is sampled from a weak random source of sufficiently large min-entropy, then the output of the Impagliazzo-Wigderson generator is statistically close to uniform: that is, the Impagliazzo-Wigderson generator *is* an extractor.

2.3 Using a Random Predicate Instead of a Hard One

Let us introduce the following additional piece of notation: let $n = 2^l$, for a string $x \in \{0,1\}^n$ we denote by $\langle x \rangle : \{0,1\}^l \to \{0,1\}$ the predicate whose truth-table is x.

Lemma 4 *Fix constants $\varepsilon, \delta > 0$, and an integer parameter l. Consider the function $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ defined as*

$$Ext(x, s) = IW^{(m)}_{\langle x \rangle}(s) \tag{1}$$

where $t = O(l) = O(\log n)$ and $m = 2^{\alpha(\delta,\varepsilon)l} = n^{\alpha}$. Then Ext, as defined in Equation (1) is a $(m\delta n^{\delta} \log n + \log 1/\varepsilon, 2\varepsilon)$-extractor.

Proof: We have to prove that for every random variable X of min entropy $k \geq m\delta n^{\delta} \log n + \log 1/\varepsilon$ and for every $T : \{0,1\}^m \to \{0,1\}$ we have

$$|\,\mathbf{Pr}[T(Ext(X, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \leq 2\varepsilon \tag{2}$$

Let us fix X and T and prove that Expression (2) holds for them. Let us call $B \subseteq \{0,1\}^n$ the set of *bad* strings x for which

$$|\,\mathbf{Pr}[T(Ext(x, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| > \varepsilon \tag{3}$$

For each such x, the analysis of Impagliazzo and Wigderson implies that there is a circuit of size $2^{\delta l} = n^{\delta}$ that uses T-gates and that computes $\langle x \rangle$. Since T is fixed, and any two different predicates are computed by two different circuits, we can conclude that the total number of elements of B is at most the number of circuits of size $S = 2^{\delta l}$ with gates of fan-in at most m. So we have $|B| \leq 2^{mS \log S} = 2^{m\delta n^{\delta} \log n}$.

Since X has high min-entropy, and B is small, the probability of picking an element of B when sampling from X is small, that is

$$\mathbf{Pr}[X \in B] \leq |B| \cdot 2^{-k} \leq 2^{m\delta n^{\delta} \log n} \cdot 2^{-(m\delta n^{\delta} \log n + \log 1/\varepsilon)} = \varepsilon\ . \tag{4}$$

Then we have

$$\begin{aligned}
& |\,\mathbf{Pr}[T(Ext(X, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \\
\leq\ & \mathop{\mathbf{E}}_{x \in X}[|\,\mathbf{Pr}[T(Ext(x, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]|] \\
=\ & \sum_{x \in B} \mathbf{Pr}[X = x] \cdot |\,\mathbf{Pr}[T(Ext(x, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \\
& + \sum_{x \notin B} \mathbf{Pr}[X = x] \cdot |\,\mathbf{Pr}[T(Ext(x, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \\
\leq\ & 2\varepsilon
\end{aligned}$$

where the first inequality is an application of the triangle inequality and the third inequality follows from Expression (4) and the definition of B. □

If we translate the parameters in a more traditional format we have the following extractor construction.

Theorem 5 *For every positive constants γ and ε there is a $\gamma' > 0$ and an explicit construction of (k, ε)-extractor $Ext_n : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ where $t = O(\log n)$, $k = n^\gamma$ and $m = k^{\gamma'}$.*

Proof: We proved that for every constant $\varepsilon > 0$ and $\delta > 0$ there is a $0 < \alpha < \delta$ such that there is a $(k, 2\varepsilon)$-extractor construction where $k = O(n^{\alpha+\delta}) = O(n^{2\delta})$ and the output length is $m = n^\alpha$. We can then set $\delta = \gamma/2$ and $\gamma' = \alpha$ to get the parameters claimed in the theorem. □

This is already a very good construction of extractors, and it implies an entropy-rate optimal simulation of randomized algorithms using weak random sources.

We mentioned in Section 2.2 that Babai et al. [BFNW93] were the first to give a construction of pseudorandom generators based on worst-case hard predicates. In particular, a weaker version of Lemma 3 is proved in [BFNW93], with similar parameters except that $t = O(l^2)$ instead of $t = O(l)$. The analysis of this section applies to the construction of Babai et al. [BFNW93], and one can show that their construction gives extractors with the same parameters as in Theorem 5, except that one would have $t = O((\log n)^2)$.

By deriving a more accurate estimation of the parameters in the Impagliazzo-Wigderson construction, it would be possible to improve on the statement of Theorem 5. More specifically, it could be possible to have an explicit dependency of the parameter t from δ and ε, and an explicit expression for $\alpha(\delta, \gamma)$. However, such improved analysis would not be better than the analysis of the construction that we present in the next section, and so we do not pursue this direction.

3 Main Result

In this section we present a construction of extractors based on the Nisan-Wigderson generator and error-correcting codes. The Nisan-Wigderson generator is simpler than the Impagliazzo Wigderson generator considered in the previous section, and this simplicity will allow us to gain in efficiency.

The advantages of the construction of this section over the construction of the previous section are better quantitative parameters and the possibility of giving a self-contained and relatively simple presentation. The subsequent work of Raz et al [RRV99] took the construction of this section as a starting point, and improved the primitives that we use.

3.1 Overview

The Nisan-Wigderson generator works similarly to the Impagliazzo-Wigderson one: it has access to a predicate, and its output is indistinguishable from uniform provided that the predicate is hard (but, as will be explained in a moment, a stronger notion of hardness is being used). This is proved by means of a reduction that shows that if T is a test that distinguishes the output of the generator with predicate P from uniform, then there is a small circuit with one T-gate that *approximately* computes P. That is, the circuit computes a predicate that agrees with P on a fraction of inputs noticeably bounded away from 1/2.

Due to this analysis, we can say that the output of the Nisan-Wigderson generator is indistinguishable from uniform provided that the predicate being used is hard to compute approximately, as opposed to merely hard to compute exactly, as in the case of the Impagliazzo-Wigderson generator.

We may be tempted to define and analyse an extractor Ext based on the Nisan-Wigderson generator using exactly the same approach of the previous section. Then, as before, we would assume for the sake of contradiction that a test T distinguishes the output $Ext(X, U_t)$ of the extractor from the uniform distribution, and we would look at how many strings x are there such that $|\mathbf{Pr}[T(Ext(x, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]|$ can be large. For each such x we can say that there is a circuit of size S that describes a string *whose Hamming distance from* x is noticeably less than $1/2$. Since there are about 2^S such circuits, the total number of bad strings x is at most 2^S times the number of strings that can belong to a Hamming sphere of radius about $1/2$. If X is sampled from a distribution whose min-entropy is much bigger then the logarithm of the number of possible bad x, then we reach a contradiction to the assumption that T was distinguishing $Ext(X, U_t)$ from uniform. When this calculation is done with the actual parameters of the Nisan-Wigderson generator, the result is very bad, because the number of strings that belong to a Hamming sphere of the proper radius is huge. This is, however, the *only* point where the approach of the previous section breaks down.

Our solution is to use *error-correcting codes*, specifically, codes with the property that every Hamming sphere of a certain radius contains a small number of codewords. We then define an extractor Ext that on input x and s first encodes x using the error-correcting code, and then applies the Nisan-Wigderson generator using s as a seed and the encoding of x as the truth table of the predicate. Thanks to the property of the error-correcting code, the counting argument of the previous section works well again.

3.2 Preliminaries

In this section we state some known technical results that will be used in the analysis of our extractor. For an integer n we denote by $[n]$ the set $\{1, \ldots, n\}$. We denote by $u_1 u_2$ the string obtained by concatenating the strings u_1 and u_2.

3.2.1 Error-correcting codes

Error-correcting codes are one of the basic primitives in our construction. We need the existence of codes such that few codewords belong to any given Hamming ball of sufficiently small radius.

Lemma 6 (Error Correcting Codes) *For every n and δ there is a polynomial-time computable encoding $EC : \{0,1\}^n \to \{0,1\}^{\bar{n}}$ where $\bar{n} = \text{poly}(n, 1/\delta)$ such that every ball of Hamming radius $(1/2 - \delta)\bar{n}$ in $\{0,1\}^{\bar{n}}$ contains at most $1/\delta^2$ codewords. Furthermore $\bar{n}$ can be assumed to be a power of 2.*

Stronger parameters are achievable. In particular the length of the encoding can be $\bar{n} = n\text{poly}(1/\delta)$. However, the stronger bounds would not improve our constructions. Standard codes are good enough to prove Lemma 6. We sketch a proof of the lemma in the Appendix.

3.2.2 Designs and the Nisan-Wigderson Generator

In this section we cite some results from [NW94] in a form that is convenient for our application. Since the statements of the results in this section are slightly different from the corresponding statements in [NW94], we also present full proofs.

Definition 7 (Design) *For positive integers m, l, $a \le l$, and $t \ge l$, a (m, t, l, a) design is a family $\mathcal{S} = S_1, \ldots, S_m$ of sets such that*

- $S_i \subseteq [t]$,
- $|S_i| = l$,
- *for every* $i \neq j \in [m]$, $|S_i \cap S_j| \le a$.

Lemma 8 (Construction of Design [NW94]) *For every positive integers m, l, and $a \leq l$ there exists a (m, t, l, a) design where $t = e^{\frac{\ln m}{a}+1} \cdot \frac{l^2}{a}$. Such a design can be computed deterministically in $O(2^t m)$ time.*

The deterministic construction in Lemma 8 was presented in [NW94] for the special case of $a = \log m$. The case for general a can be proved by using the same proof, but a little care is required while doing a certain probabilistic argument. The proof of Lemma 8 is given in Appendix A.2.

The following notation will be useful in the next definition: if $S \subseteq [t]$, with $S = \{s_1, \ldots, s_l\}$ (where $s_1 < s_2 < \cdots < s_l$) and $y \in \{0,1\}^t$, then we denote by $y_{|S} \in \{0,1\}^l$ the string $y_{s_1} y_{s_2} \cdots y_{s_l}$.

Definition 9 (NW Generator [NW94]) *For a function $f : \{0,1\}^l \to \{0,1\}$ and an (m, t, l, a)-design $\mathcal{S} = (S_1, \ldots, S_m)$, the Nisan-Wigderson generator $NW_{f,\mathcal{S}} : \{0,1\}^t \to \{0,1\}^m$ is defined as*

$$NW_{f,\mathcal{S}}(y) = f(y_{|S_1}) \cdots f(y_{|S_m})$$

Intuitively, if f is a hard-on-average function, then $f(\cdot)$ evaluated on a random point x is an "unpredictable bit" that, to a bounded adversary, "looks like" a random bit. The basic idea in the Nisan-Wigderson generator is to evaluate $f(\cdot)$ at several points, thus generating several unpredictable output bits. In order to have a short seed, evaluation points are not chosen independently, but rather in such a way that any two points have "low correlation." This is where the definition of design is useful: the random seed y for the generator associates a truly random bit to any element of the universe $[t]$ of the design. Then the i-th evaluation point is chosen as the subset of the bits of y corresponding to set S_i in the design. Then the "correlation" between the i-th and the j-th evaluation point is given by the $\leq a$ bits that are in $S_i \cap S_j$. It remains to be seen that the evaluation of f at several points having low correlation looks like a sequence of random bits to a bounded adversary. This is proved in [NW94, Lemma 2.4], and we will state the result in a slightly different form in Lemma 10 below.

For two functions $f, g : \{0,1\}^l \to \{0,1\}$ and a number $0 \leq \rho \leq 1$ we say that g *approximates* f within a factor ρ if f and g agree on at least a fraction ρ of their domain, i.e. $\mathbf{Pr}_x[f(x) = g(x)] \geq \rho$.

Lemma 10 (Analysis of the NW Generator [NW94]) *Let $\mathcal{S}$ be an (m, l, a)-design, and $T : \{0,1\}^m \to \{0,1\}$. Then there exists a family $\mathcal{G}_T$ (depending on T and $\mathcal{S}$) of at most $2^{m2^a + \log m + 2}$ functions such that for every function $f : \{0,1\}^l \to \{0,1\}$ satisfying*

$$\left| \mathbf{Pr}_{y \in \{0,1\}^t}[T(NW_{f,\mathcal{S}}(y)) = 1] - \mathbf{Pr}_{r \in \{0,1\}^m}[T(r) = 1] \right| \geq \varepsilon \ .$$

there exists a function $g : \{0,1\}^l \to \{0,1\}$, $g \in \mathcal{G}_T$, such that $g(\cdot)$ approximates $f(\cdot)$ within $1/2 + \varepsilon/m$.

Proof: [Of Lemma 10] We follow the proof of Lemma 2.4 in [NW94]. The main idea is that if T distinguishes $NW_{f,\mathcal{S}}(\cdot)$ from the uniform distribution, then we can find a bit of the output where this distinction is noticeable.

In order to find the "right bit", we will use the so-called *hybrid argument* of Goldwasser and Micali [GM84]. We define $m+1$ distributions $D_0, \ldots, D_m$; D_i is defined as follows: sample a string $v = NW_{f,\mathcal{S}}(y)$ for a random y, and then sample a string $r \in \{0,1\}^m$ according to the uniform distribution, then concatenate the first i bits of v with the last $m - i$ bits of r. By definition, D_m is distributed as $NW_{f,\mathcal{S}}(y)$ and D_0 is the uniform distribution over $\{0,1\}^m$.

Using the hypothesis of the Lemma, we know that there is a bit $b_0 \in \{0,1\}$ such that

$$\Pr_y[T'(NW_{f,\mathcal{S}}(y)) = 1] - \Pr_r[T'(r)] > \varepsilon$$

where $T'(x) = b_0 \oplus T(x)$.

We then observe that

$$\begin{aligned} \varepsilon &\leq \Pr_y[T'(NW_{f,\mathcal{S}}(y)) = 1] - \Pr_r[T'(r)] \\ &= \mathbf{Pr}[T'(D_m) = 1] - \mathbf{Pr}[T'(D_0) = 1] \\ &= \sum_{i=1}^{m} (\mathbf{Pr}[T'(D_i) = 1] - \mathbf{Pr}[T'(D_{i-1}) = 1]) \end{aligned}$$

In particular, there exists an index i such that

$$\mathbf{Pr}[T'(D_i) = 1] - \mathbf{Pr}[T'(D_{i+1}) = 1] \geq \varepsilon/m \tag{5}$$

Now, recall that

$$D_{i-1} = f(y_{|S_1}) \cdots f(y_{|S_{i-1}}) r_i r_{i+1} \cdot r_m$$

and

$$D_i = f(y_{|S_1}) \cdots f(y_{|S_{i-1}}) f(y_{|S_i}) r_{i+1} \cdot r_m \ .$$

We can assume without loss of generality (up to a renaming of the indices) that $S_i = \{1, \ldots, l\}$. Then we can see $y \in \{0,1\}^t$ as a pair (x, z) where $x = y_{|S_i} \in \{0,1\}^l$ and $z = y_{|[t]-S_i} \in \{0,1\}^{t-l}$. For every $j < i$ and $y = (x, z)$, let us define $h_j(x, z) = y_{|S_j}$: note that $h_j(x, z)$ depends on $|S_i \cap S_j| \leq a$ bits of x and on $l - |S_i \cap S_j| \geq l - a$ bits of z.

Using this notation (and observing that for a 0/1 random variable the probability that the random variable is 1 is equal to the expectation of the random variable) we can rewrite Expression (5) as

$$\begin{aligned} &\mathop{\mathbf{E}}_{r_i,\ldots,r_m,x,z}[T'(f(h_1(x,z)), f(h_2(x,z)), \ldots f(h_{i-1}(x,z)), f(x), \ldots, r_m)] \\ &- \mathop{\mathbf{E}}_{r_i,\ldots,r_m,x,z}[T'(f(h_1(x,z)), f(h_2(x,z)), \ldots f(h_{i-1}(x,z)), r_i, \ldots, r_m)] \\ &= \mathop{\mathbf{E}}_{r_i,\ldots,r_m,x,z}[T'(f(h_1(x,z)), f(h_2(x,z)), \ldots f(h_{i-1}(x,z)), f(x), \ldots, r_m) \\ &\quad -T'(f(h_1(x,z)), f(h_2(x,z)), \ldots f(h_{i-1}(x,z)), r_i, \ldots, r_m)] \geq \varepsilon/m \end{aligned}$$

We can use an averaging argument to claim that we can fix $r_{i+1}, \ldots, r_m$ to some values $c_{i+1} \cdots c_m$, as well as fix z to some value w, and still have

$$\begin{aligned} &\mathop{\mathbf{E}}_{r_i,x}[T'(f(h_1(x,w)), f(h_2(x,w)), \ldots f(h_{i-1}(x,w)), f(x), c_{i+1} \ldots, c_m) \\ &-T'(f(h_1(x,w)), f(h_2(x,w)), \ldots f(h_{i-1}(x,w)), r_i, c_{i+1}, \ldots, c_m)] \geq \varepsilon/m \end{aligned} \tag{6}$$

Let us now consider a new function $F : \{0,1\}^{l+1} \rightarrow \{0,1\}^m$ defined as $F(x, b) = f(h_1(x,w)), f(h_2(x,w)), \ldots f(h_{i-1}(x,w)), b, c_{i+1} \ldots, c_m$. Using F, renaming r_i as b, and moving back to probability notation, we can rewrite Expression (6) as

$$\mathbf{Pr}_{x,b}[T'(F(x,f(x)))=1]-\mathbf{Pr}_{x,b}[T'(F(x,b))=1]>\varepsilon/m$$

That is, using T' and F it is possible to distinguish a pair of the form $(x,f(x))$ from a uniform string of length $l+1$. We will now see that, given $F(\cdot)$ and $T'(\cdot)$, it is possible to describe a function $g(\cdot)$ that agrees with $f(\cdot)$ on a fraction $1/2+\varepsilon/m$ of the domain.

Consider the following approach: on input x, pick a random $b\in\{0,1\}$, and compute $T'(F(x,b))$; if $T'(F(x,b))=1$ then output b, otherwise output $1-b$. Let us call $g_b(x)$ the function performing the above computation, and let us estimate the agreement between $f(\cdot)$ and $g_b(\cdot)$, averaged over the choice of b.

$$\begin{aligned}
\mathbf{Pr}_{b,x}[g_b(x)=f(x)] &= \mathbf{Pr}_{b,x}[g_b(x)=f(x)|b=f(x)]\,\mathbf{Pr}_{b,x}[b=f(x)]+\mathbf{Pr}_{b,x}[g_b(x)=f(x)|b\neq f(x)]\,\mathbf{Pr}_{b,x}[b\neq f(x)]\\
&= \frac{1}{2}\mathbf{Pr}_{b,x}[T'(F(x,b))=1|f(x)=b]+\frac{1}{2}\mathbf{Pr}_{b,x}[T'(F(x,b))=0|f(x)\neq b]\\
&= \frac{1}{2}+\frac{1}{2}\left(\mathbf{Pr}_{b,x}[T'(F(x,b))=1|f(x)=b]-\mathbf{Pr}_{b,x}[T'(F(x,b))=1|f(x)\neq b]\right)\\
&= \frac{1}{2}+\mathbf{Pr}_{x,b}[T'(F(x,f(x)))=1]-\mathbf{Pr}_{x,b}[T'(F(x,b))=1]\\
&\geq \frac{1}{2}+\frac{\varepsilon}{m}
\end{aligned}$$

Over the choices of x and b, the probability that we guess $f(x)$ is $\geq 1/2+\varepsilon/m$, hence there is a bit $b_1\in\{0,1\}$ such that g_{b_1} approximates f to within $1/2+\varepsilon/m$. Once T and F are given, we can specify g_{b_1} using two bits of information (the bit b_1, plus the bit b_0 such that $T'(\cdot)=b_0\oplus T(\cdot)$).

We now observe that F can be totally described by using no more than $\log m+(i-1)2^a+(m-i)<\log m+m2^a$ bits of information. Indeed, we use $\log m$ bits to specify i, then for every $j<i$ and for every x we have to specify $f(h_j(x,w))$. Since $h_j(x,w)$ only depends on the bits of x indexed by $S_i\cap S_j$, we just have to specify 2^a values of f for each such j. For $j>i$ we have to specify c_j.

Overall, we have a function $g(\cdot)$ that approximates f to within $1/2+\varepsilon/m$ and that, given T, can be described using $2+\log m+m2^a$ bits. We define $\mathcal{G}_T$ as containing all functions $g(\cdot)$ that can be defined in this way, over all possible description. □

3.3 Construction

The construction has parameters n, $k\leq n$, $22\leq m\leq k/2$ and $0<\varepsilon<2^{-k/12}$. It can be verified that the constraints on the parameters imply that $2+2\log m+3\log(1/\varepsilon)<k/2$ (because we have $k/4>3\log 1/\varepsilon$ and $k/4\geq m/2\geq 2+2\log m$ for $m\geq 22$).

Let $EC:\{0,1\}^n\to\{0,1\}^{\bar{n}}$ be as in Lemma 6, with $\delta=\varepsilon/m$, so that $\bar{n}=\mathrm{poly}(n,1/\varepsilon)$, and define $l=\log\bar{n}=O(\log n/\varepsilon)$.

For an element $u\in\{0,1\}^n$, define $\bar{u}=\langle EC(u)\rangle:\{0,1\}^l\to\{0,1\}$. Let $\mathcal{S}=S_1,\ldots,S_m$ be as in Lemma 8, such that

- $S_i\subseteq[t]$,
- $|S_i|=l$,
- $|S_i\cap S_j|\leq a=\log(k/2m)$, and

- $t = O\left(e^{\frac{\ln m}{\log(k/2m)}} \cdot \frac{l^2}{\log(k/2m)}\right)$.

By our choice of parameters, and by choosing c appropriately, we have that $m > t$.

Then we define $Ext : \{0,1\}^n \times \{0,1\}^t \to \{0,1\}^m$ as

$$Ext(u, y) = NW_{\bar{u},\mathcal{S}}(y) = \bar{u}(y_{|S_1}) \cdots \bar{u}(y_{|S_m}) \ . \tag{7}$$

3.4 Analysis

Lemma 11 *Let Ext be as in Equation (7). For every fixed predicate $T : \{0,1\}^m \to \{0,1\}$, there are at most $2^{2+m2^a} \cdot (m/\varepsilon)^2$ strings $u \in \{0,1\}^n$ such that*

$$|\mathbf{Pr}[T(Ext(u, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \geq \varepsilon \tag{8}$$

Proof: It follows from the definition of Ext and from Lemma 10 that if u is such that (8) holds, then there exists a function $g : \{0,1\}^l \to \{0,1\}^m$ in $\mathcal{G}_T$ and a bit $b \in \{0,1\}$ such that the function $b \oplus T(g(\cdot))$ approximates $\bar{u}(\cdot)$ within $1/2 + \varepsilon/m = 1/2 + \delta$.

There are at most $2^{2+t-l+\log m+m2^a}$ functions $g \in \mathcal{G}_T$, and we have $l > \log n > \log m$. Furthermore, each such function can be within relative distance $1/2 - \varepsilon/m$ from at most $(m/\varepsilon)^2$ functions $\bar{u}(\cdot)$ coming from the error correcting code of Lemma 6.

We conclude that $2^{2+2\log(m/\varepsilon)+t+m2^a}$ is an upper bound on the number of strings u for which Expression (8) can occur. □

Theorem 12 *Ext as defined in Equation (7) is a $(k, 2\varepsilon)$-extractor.*

Proof: We first note that, by our choice of parameters, we have $m2^a = k/2$. Also, $k/2 > 4m$ and $m > t$, $m > 2$, and $m > c \cdot (\log(n/\varepsilon))^4 > 3\log(m/\varepsilon) > \log(m^2/\varepsilon^3)$.

Now, fix a predicate $T : \{0,1\}^m \to \{0,1\}$. From Lemma 11 we have that the probability that sampling a u from a source X of min-entropy k we have

$$|\,\mathbf{Pr}[T(Ext(u, U_t)) = 1] - \mathbf{Pr}_r[T(U_m) = 1]| \geq \varepsilon$$

is at most $2^{2+t+m2^a+\log(m^2/\varepsilon^2)} \cdot 2^{-k}$ which is at most ε by our choice of parameters. A Markov argument shows that

$$|\,\mathbf{Pr}[T(Ext(X, U_t)) = 1] - \mathbf{Pr}[T(U_m) = 1]| \leq 2\varepsilon$$

□

Theorem 1 now follows from Theorem 12 and by the choice of parameters in the previous section.

Acknowledgments

Oded Goldreich contributed an important idea in a critical phase of this research; he also contributed very valuable suggestions on how to present the results of this paper. I thank Oded Goldreich, Shafi Goldwasser, Madhu Sudan, Salil Vadhan, Amnon Ta-Shma, and Avi Wigderson for several helpful conversations. This paper would have not been possible without the help of Adam Klivans, Danny Lewin, Salil Vadhan, Yevgeny Dodis, Venkatesan Guruswami, and Amit Sahai in assimilating the ideas of [NW94, BFNW93, Imp95, IW97]. Thanks also to Madhu Sudan for hosting our reading group in the Spring'98 Complexity Seminars at MIT. This work was mostly done while the author was at MIT, partially supported by a grant of the Italian CNR. Part of this work was also done while the author was at DIMACS, supported by a DIMACS post-doctoral fellowship.

References

[ACR98] A.E. Andreev, A.E.F. Clementi, and J.D.P. Rolim. A new general derandomization method. *Journal of the ACM*, 45(1):179–213, 1998.

[ACRT99] A.E. Andreev, A.E.F. Clementi, J.D.P. Rolim, and L. Trevisan. Weak random sources, hitting sets, and BPP simulations. *SIAM Journal on Computing*, 28(6):2103–2116, 1999. Preliminary version in *Proc of FOCS'97*.

[AK97] V. Arvind and J. Köbler. On resource-bounded measure and pseudorandomness. In *Proceedings of the 17th Conference on Foundations of Software Technology and Theoretical Computer Science*, pages 235–249. LNCS 1346, Springer-Verlag, 1997.

[BFNW93] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

[BGS98] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCP's and non-approximability – towards tight results. *SIAM Journal on Computing*, 27(3):804–915, 1998. Preliminary version in *Proc. of FOCS'95*.

[BM84] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. Preliminary version in *Proc. of FOCS'82*.

[BR94] M. Bellare and J. Rompel. Randomness-efficient oblivious sampling. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 276–287, 1994.

[CG88] B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, April 1988.

[CW89] A. Cohen and A. Wigderson. Dispersers, deterministic amplification, and weak random sources. In *Proceedings of the 30th IEEE Symposium on Foundations of Computer Science*, pages 14–19, 1989.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. Preliminary Version in *Proc. of STOC'82*.

[GNW95] O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR lemma. Technical Report TR95-50, Electronic Colloquium on Computational Complexity, 1995.

[Gol99] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Springer-Verlag, 1999.

[GW97] O. Goldreich and A. Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures and Algorithms*, 11(4):315–343, 1997.

[GZ97] O. Goldreich and D. Zuckerman. Another proof that $BPP \subseteq PH$ (and more). Technical Report TR97-045, Electronic Colloquium on Computational Complexity, 1997.

[Imp95] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of the 36th IEEE Symposium on Foundations of Computer Science*, pages 538–545, 1995.

[IW97] R. Impagliazzo and A. Wigderson. $P = BPP$ unless E has sub-exponential circuits. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 220–229, 1997.

[IW98] R. Impagliazzo and A. Wigderson. Randomness versus time: De-randomization under a uniform assumption. In *Proceedings of the 39th IEEE Symposium on Foundations of Computer Science*, pages 734–743, 1998.

[KvM99] A. Klivans and D. van Milkebeek. Graph non-isomorphism has subexponential size proofs unless the polynomial hierarchy collapses. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, pages 659–667, 1999.

[Lei92] F.T. Leighton. *Introduction to Parallel Algorithms and Architectures.* Morgan Kaufmann, 1992.

[MS77] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 1977.

[Nis91] N. Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 12(4):63–70, 1991.

[Nis96] N. Nisan. Extracting randomness: How and why. In *Proceedings of the 11th IEEE Conference on Computational Complexity*, pages 44–58, 1996.

[NTS98] N. Nisan and A. Ta-Shma. Extrating randomness : A survey and new constructions. *Journal of Computer and System Sciences*, 1998. To appear. Preliminary versions in [Nis96, TS96].

[NW94] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994. Preliminary version in *Proc. of FOCS'88.*

[NZ93] N. Nisan and D. Zuckerman. More deterministic simulation in Logspace. In *Proceedings of the 25th ACM Symposium on Theory of Computing*, pages 235–244, 1993.

[RRV99] R. Raz, O. Reingold, and S. Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, pages 149–158, 1999.

[RTS97] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*, pages 585–594, 1997.

[SSZ98] M. Saks, A. Srinivasan, and S. Zhou. Explicit OR-dispersers with polylogarithmic degree. *Journal of the ACM*, 45(1):123–154, 1998. Preliminary version in *Proc. of STOC'95.*

[STV99] M. Sudan, L. Trevisan, and S. Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, pages 537–546, 1999.

[SV86] M. Santha and U. Vazirani. Generating quasi-random sequences from slightly random sources. *Journal of Computer and System Sciences*, 33:75–87, 1986.

[SZ94] A. Srinivasan and D. Zuckerman. Computing with very weak random sources. In *Proceedings of the 35th IEEE Symposium on Foundations of Computer Science*, pages 264–275, 1994.

[Tre99] L. Trevisan. Construction of extractors using pseudo-random generators. In *Proceedings of the 31st ACM Symposium on Theory of Computing*, pages 141–148, 1999.

[TS96] A. Ta-Shma. On extracting randomness from weak random sources. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 276–285, 1996.

[TS98] A. Ta-Shma. Almost optimal dispersers. In *Proceedings of the 30th ACM Symposium on Theory of Computing*, 1998.

[vN51] J. von Neumann. Various techniques used in connection with random digits. *National Bureau of Standards, Applied Mathematics Series*, 12:36–38, 1951.

[VV85] U. Vazirani and V. Vazirani. Random polynomial time is equal to slightly random polynomial time. In *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pages 417–428, 1985.

[WZ93] A. Wigderson and D. Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. In *Proceedings of the 25th ACM Symposium on Theory of Computing*, pages 245–251, 1993.

[Yao82] A.C. Yao. Theory and applications of trapdoor functions. In *Proceedings of the 23th IEEE Symposium on Foundations of Computer Science*, pages 80–91, 1982.

[Zuc90] D. Zuckerman. General weak random sources. In *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science*, pages 534–543, 1990.

[Zuc96a] D. Zuckerman. On unapproximable versions of *NP*-complete problems. *SIAM Journal on Computing*, 25(6):1293–1304, 1996. Preliminary Version in *Proc. of Structures'93*.

[Zuc96b] D. Zuckerman. Randomness-optimal sampling, extractors and constructive leader election. In *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 286–295, 1996.

A Appendix

A.1 A Discussion on Lemma 6

It is a standard result that if an error-correcting code has large minimum distance then there can be few codewords in every large ball. In particular, the following bound holds.

Lemma 13 *Suppose C is an error-correcting code with (relative) minimum distance $\geq 1/2 - \beta/2$. Then every Hamming ball of (relative) radius $1/2 - \sqrt{\beta}$ contains at most $1/3\beta$ codewords.*

A proof can be found e.g. in [BGS98, Lemma A.1]. The following result is well known, even if we do not know of a source where it is clearly stated in this way.

Lemma 14 *For every δ and n there exists an error-correcting code with 2^n codewords of length $\bar{n} = \mathrm{poly}(n, 1/\delta)$ and with minimum distance $(1/2 - \delta)\bar{n}$. The code admits a polynomial-time encoding algorithm.*

Several constructions meet this requirement. In particular one can use a Reed-Solomon code concatenated with a Hadamard code. See e.g. [MS77] for a treatment of error correcting codes. Lemma 6 follows from Lemmas 13 and 14.

A.2 Proof of Lemma 8

The following version of the Chernoff bound will be used (this is Lemma 1.7 in [Lei92]).

Lemma 15 *Let $X_1, \ldots, X_n$ be 0/1 mutually independent random variables such that $\mathbf{E}\left[\sum_i X_i\right] = \mu$. Then, for every $\alpha > 1$ it holds*

$$\mathbf{Pr}\left[\sum_i X_i \geq \alpha\mu\right] \leq e^{-((\ln \alpha)+\frac{1}{\alpha}-1)\alpha\mu}$$

We can now give the proof of Lemma 8. The proof is essentially the same as in [NW94], and uses some improvements appeared in [RRV99], in particular, the use of a particularly clean probabilistic argument, that is credited to Zuckerman in [RRV99].

We view the set $[t]$ as made by l intervals, each of size t/l. We call a subset $S \subseteq [t]$ *structured* if it contains exactly one element in each interval.

We consider an algorithm that sequentially chooses m structured subsets of $[t]$ such that, at any step, the chosen subset has intersections of size less than a with all the previously chosen subsets.

In order to prove that at each step it is possible to choose a new subset with the required properties, we use a probabilistic argument.

Lemma 16 *Let $S_1, \ldots, S_k$ with $k < m$ be a collection of structured subsets of $[t]$, where $t = \frac{l^2}{a} \cdot e^{(\ln m)/a}$. Then there exists a structured subset $S \subseteq [t]$ such that $|S| = l$ and $|S_i \cap S| \leq a$ for $i = 1, \ldots, k$.*

PROOF: We choose S randomly among structured sets, i.e. for every interval we pick an element at random and we put it into S. Now we claim that for every i, $\mathbf{Pr}[|S \cap S_i| > a] \leq 1/m$. Indeed, the random variable $|S \cap S_i|$ can be seen as the sum of l independent 0/1 random variables, one for every interval (the random variable being 1 iff S and S_i share an element in the corresponding interval). For each of these random variables, the average is precisely the inverse of the length of the interval, that is l/t. It follows that the average of $|S \cap S_i|$ is $\mu = l^2/t$.

Using the Chernoff bound, we have

$$\mathbf{Pr}[|S \cap S_i| > a] \leq e^{-a((\ln a/\mu)+a/\mu-1)} < e^{-a((\ln a/\mu)-1)}$$

and since we have $a/\mu = at/l^2 = e^{1+(\ln m)/a}$, the expression above is equal to

$$e^{-a((1+(\ln m)/a)-1)} = 1/m$$

Using a union bound, we can now conclude that

$$\mathbf{Pr}[\exists i \in \{1, \ldots, k\}. |S \cap S_i| > a] \leq k/m < 1$$

and so there exists a structured set S having small intersections with each of the S_i. □

A set S satisfying the statement of the above Lemma can be found in $\mathrm{poly}(2^t, m)$ time using $O(m + t)$ space, just by trying all possible structured subsets.