An introduction to randomness extractors

Ronen Shaltiel*

University of Haifa

Abstract. We give an introduction to the area of "randomness extraction" and survey the main concepts of this area: deterministic extractors, seeded extractors and multiple sources extractors. For each one we briefly discuss background, definitions, explicit constructions and applications.

1 Introduction

Randomized algorithms and protocols play an important role in many areas of computer science. It is often the case that such algorithms and protocols are more efficient than deterministic ones. Moreover, having access to randomness is essential for Cryptography.

Randomized algorithms and protocols are designed under the assumption that computers have access to a sequence of truly random bits (that is a sequence of independent and unbiased coin tosses). In actual implementations this sequence is generated by taking a sample from some "source of randomness". Examples are:

- Generating and measuring electromagnetic or radioactive noise.
- Measuring timing of past events (e.g., how much time did the last disk operation take?).
- Measuring user dependent behavior (e.g., timing of the key strokes of users).

While these sources seem to "contain randomness" in the sense that they have entropy, a sample from such sources is not of the form of truly random bits. Randomness extractors are algorithms that when given one sample from a weak random source, produce a sequence of truly random bits.

The motivation described above led to a wide body of research concentrating on three main concepts:

- Deterministic extractors, which we discuss in Section 2.
- Seeded extractors, which we discuss in Section 3.
- Multiple source extractors, which we discuss in Section 4.

It turns out that extractors have many applications beyond the original motivation presented in Section 1. We present few of these applications as we go along (and stress that there are many other applications in the literature).

Our aim is to provide a brief introduction to the area. This article does not attempt to be comprehensive and the reader is referred to [NTS99,Sha02,Vad07,AB09] for some other survey articles.

* The author is supported by ISF grant 686/07.

2 Deterministic extractors

In this section we discuss "deterministic extractors". The term "deterministic" is used to distinguish these extractors from "seeded extractors" that we discuss in Section 3. We begin with some notation. Throughout this manuscript we use the terms "source" and "distribution" interchangeably.

Definition 1 (Statistical distance). *Two distributions X, Y over the same domain are* ϵ-close *if for every event A, $|\Pr[X \in A] - \Pr[Y \in A]| \leq \epsilon$. The support of a distribution X is $\mathsf{Supp}(X) = \{x : \Pr[X = x] > 0\}$. The uniform distribution over $\{0,1\}^m$ is denoted by U_m and we say that X is ϵ-close to uniform if it is ϵ-close to U_m.*

Two distributions that are ϵ-close assign essentially the same probability to all events. In particular, randomized algorithms and protocols retain their useful properties when run with distributions that are close to uniform (rather than uniform). The motivation given in Section 1 leads to the following formal definition of an extractor (we also define a weaker object called a "disperser").

Definition 2 (deterministic extractors and dispersers). *Let $m \leq n$ be integers and let $\epsilon \geq 0$ be a parameter. Let $E : \{0,1\}^n \to \{0,1\}^m$ be a function and X be a distribution over $\{0,1\}^n$.*

- *E is an* ϵ-extractor *for X if the distribution $E(X)$ is ϵ-close to U_m.*
- *E is an* ϵ-disperser *for X if $|\mathsf{Supp}(E(X))| \geq (1-\epsilon) \cdot 2^m$.*

Let $\mathcal{C}$ be a class of probability distributions over $\{0,1\}^n$.

- *E is an* ϵ-extractor *for $\mathcal{C}$ if E is an ϵ-extractor for every X in $\mathcal{C}$.*
- *E is an* ϵ-disperser *for $\mathcal{C}$ if E is an ϵ-disperser for every X in $\mathcal{C}$.*

Note that every ϵ-extractor is in particular an ϵ-disperser. We plan to extract randomness from weak random sources and use this randomness in randomized algorithms and protocols. In the scenario described in Section 1 the "computer designer" can choose an implementation of the weak random source. Nevertheless, note that in the examples given there, this does not necessarily determines the distribution of the source (as the environment in which the computer operates may change). This leads to the following goal.

Goal: Design extractors for "large" families of "interesting" sources.

2.1 Min-entropy: measuring the number of random bits in a source

Let us start with a simple observation. If $E : \{0,1\}^n \to \{0,1\}^m$ is a 0-extractor for X then for every $x \in \mathsf{Supp}(X)$, $\Pr[X = x] \leq 2^{-m}$. (As otherwise, for an x' with $\Pr[X = x'] > 2^{-m}$, we have that $\Pr[E(X) = E(x')] > 2^{-m}$ contradicting the correctness of the extractor as $\Pr[U_m = E(x')] = 2^{-m}$ and the two distributions $E(X)$ and U_m assign different probabilities to some event). Thus, a necessary condition for extracting m random bits from a distribution X is that for every $x \in \mathsf{Supp}(X)$, $\Pr[X = x] \leq 2^{-m}$. This leads to the following concept of entropy.

Definition 3 (min-entropy). *Let X be a distribution. The* min-entropy *of X (denoted by $H_\infty(X)$) is $H_\infty(X) = min_{x \in \mathsf{Supp}(X)} \log \frac{1}{\Pr[X=x]}$.*

We use min-entropy to measure the amount of random bits that can be extracted from a source.[1] Note that a distribution with min-entropy at least m has that for every $x \in \mathsf{Supp}(X)$, $\Pr[X = x] \leq 2^{-m}$. By the previous discussion having min-entropy at least m is a necessary condition for extracting m bits of randomness.[2] We could hope that it is a sufficient condition and that there exists an extractor $E : \{0,1\}^n \to \{0,1\}^m$ for all distributions with min-entropy at least m. However, this does not hold. In fact, for every function $E : \{0,1\}^n \to \{0,1\}$ there exists a distribution X over $\{0,1\}^n$ such that $H_\infty(X) \geq n - 1$ and yet $E(X)$ is completely fixed. (For this, take X to be the uniform distribution over $S = \{x : E(x) = b\}$ for $b \in \{0,1\}$ which gives $|S| \geq 2^n/2$).

Summing up, we cannot have an extractor that extracts even a single bit from all distributions with very large min-entropy. Furthermore, if we plan to use function E as an extractor for $\mathcal{C}$, we cannot allow distributions that are uniform on $\{x : E(x) = b\}$ to be in the family $\mathcal{C}$.

2.2 Explicitness

By the previous discussion, deterministic extractors and dispersers $E : \{0,1\}^n \to \{0,1\}^m$ only exist for classes $\mathcal{C}$ of sources with some "special structure" where each X in $\mathcal{C}$ has $H_\infty(X) \geq m$. By the probabilistic method it is easy to show existence of extractors for such classes $\mathcal{C}$ which contain "few sources".

Existence of deterministic extractors: Let $m \leq n$ be integers, let $\epsilon > 0$ and let $\mathcal{C}$ be a class with at most $2^{\mathrm{poly}(n/\epsilon)}$ distributions over $\{0,1\}^n$. There exist $k = m + O(\log n + \log(1/\epsilon))$ such that if every X in $\mathcal{C}$ has $H_\infty(X) \geq k$ then there exists $E : \{0,1\}^n \to \{0,1\}^m$ that is an ϵ-extractor for $\mathcal{C}$.

However, for our intended application (as well as other applications that we will consider) we require extractors that can be efficiently computed. In this article we identify efficient computation with polynomial-time and this leads to the following definition of explicitness.

[1] It is natural to compare min-entropy with the more standard Shannon entropy given by $H(X) = \sum_{x \in \mathsf{Supp}(X)} \Pr[X = x] \cdot \log \frac{1}{\Pr[X=x]}$. Note that $H_\infty(X) \leq H(X)$ and equality holds for "flat distributions" (that are uniform over their support). Loosely speaking, min-entropy measures the amount of information in a distribution on the "worst case" when taking a single sample, while Shannon entropy measures the amount of information that a distribution has "on average" when taking many independent samples. Following this intuition, min-entropy is the right choice for our setup as we are interested in the behavior of extractors on a single sample from the source distribution.

[2] The previous discussion only considered 0-extractors. However, it is easy to check that if $E : \{0,1\}^n \to \{0,1\}^m$ is an ϵ-extractor for X then X is ϵ-close to some distribution Y with $H_\infty(Y) \geq m$. Thus, a necessary condition for extracting m random bits from a distribution X is that X is ϵ-close to some distribution with min-entropy at least m.

Definition 4 (Explicit extractors and dispersers). *Let $m(\cdot), \epsilon(\cdot)$ be functions over integers. A function E from strings to strings is an* explicit $\epsilon(\cdot)$*-extractor (resp. disperser) if it can be computed in polynomial time, and for every sufficiently large n, when restricted to inputs of length n, E outputs strings of length $m(n)$ and is an $\epsilon(n)$-extractor (resp. disperser).*

In the remainder of this section we survey some of the families of sources that are considered in the literature on deterministic extractors and dispersers.

2.3 Deterministic extractors for von-Neumann sources

The notion of deterministic extractors can be traced back to von-Neumann [vN51] who considered sequences of independent tosses of a biassed coin with unknown bias.

Definition 5. *A distribution X over $\{0,1\}^n$ is a vN-source with probability threshold $p_0 > 0$ if $X_1, \ldots, X_n$ are independent, and there exists $p_0 \leq p \leq 1 - p_0$ such that for every $1 \leq i \leq n$, $\Pr[X_i = 1] = p$.*

In order to extract one bit from such sources, von-Neumann [vN51] divided the n input bits into pairs. The extractor scans the pairs one by one and if it finds a pair of bits that are different, it stops and outputs the first bit in the pair.[3] The correctness of this scheme follows from the observation that for two independent coin tosses X_1, X_2 of a coin with bias p, $\Pr[X_1 = 0 \wedge X_2 = 1] = \Pr[X_1 = 1 \wedge X_2 = 0] = p \cdot (1 - p)$. Moreover, the probability that $n/2$ independent pairs do not produce an output bit is bounded by $(p_0^2 + (1 - p_0)^2)^{n/2} \leq \epsilon$ if $p_0 \geq \frac{\log(1/\epsilon)}{n}$. It is easy to extend this approach to extract many bits. There is also work on extracting a number of bits that approaches the information theoretic limit [Eli72,Per92].

2.4 Impossibility of extraction from Santha-Vazirani sources

It is natural to try and relax the assumption of independence between bits in a vN-source. Santha and Vazirani [SV86] considered a generalization of a vN-source in which for every $1 \leq i \leq n$, and every "prefix" $x_1, \ldots, x_{i-1} \in \{0,1\}$,

$$p_0 \leq \Pr[X_i = 1 | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \leq 1 - p_0.$$

A discouraging result of [SV86] is that there does not exist a deterministic extractor that extract a single bit from such sources. In other words, there are families of sources that are very structured and still do not allow deterministic extraction. This is bad news for the approach of simulating randomized algorithms

[3] The model considered by [vN51] is slightly different in that it places no bound on p. Instead, it allows the extractor to access an unbounded "stream" of independent coin tosses and requires that the extractor will stop and output completely uniform bits in expected time that depends on p. Nevertheless, the same algorithm applies in both cases.

and protocols with weak sources by deterministic extraction. Historically, this led to the notion of seeded extractors that we describe in Section 3. Nevertheless, as we discuss below, deterministic extraction has many applications beyond the original motivation of Section 1.

2.5 Bit-fixing sources and privacy amplification

We now consider the family of bit-fixing sources. In such a source some bits are independent unbiased coin tosses, while others are constants.

Definition 6 (bit-fixing sources). *A distribution X over $\{0,1\}^n$ is a k-bit-fixing source if there exist k distinct indices $i_1, \ldots, i_k$ such that the distribution $(X_{i_1}, \ldots, X_{i_k})$ is distributed like U_k and for $i \notin \{i_1, \ldots, i_k\}$, X_i is a fixed constant.*[4]

Bit-fixing sources do not seem to arise naturally in the scenario considered in Section 1. Nevertheless, as we observe now, they arise naturally in Cryptography [CGH$^+$85]. Consider for example the following scenario. Assume that two parties Alice and Bob share a uniformly chosen random key $K \in \{0,1\}^n$. The key K is *private* in the sense that eavesdropper Eve has no information about it. Alice and Bob can securely encrypt communication between them using a shared private key. Suppose that at some stage, the privacy of K is somehow compromised and Eve learns $f(K)$ where $f(x) = x_{j_1}, \ldots, j_{n/2}$ for some indices $j_1, \ldots, j_{n/2}$ that are *unknown* to Alice and Bob. Alice and Bob would like to recover their privacy and come up with a new shared private key K'.

This can be achieved as follows: Let $E : \{0,1\}^n \to \{0,1\}^m$ be an ϵ-extractor for $(n/2)$-bit-fixing sources. Each party can compute $K' = E(K)$. We claim that K' is private in the sense that the distribution $(K', f(K))$ is ϵ-close to the distribution $(Z, f(K))$ where Z is uniformly distributed and independent of K. In order to see the meaning of thus claim, note that if $\epsilon = 0$ then the claim says that K' is independent of $f(K)$. For general $\epsilon > 0$ the claim implies that even an all-powerful Eve cannot distinguish K' from uniform with advantage greater than ϵ, given her view.

In order to prove the claim we note that for any value $v \in \mathsf{Supp}(f(K))$, the distribution $X = (K|f(K) = v))$ is an $(n/2)$-bit-fixing source. X captures the view that Eve has on K conditioned on the event $\{f(K) = v\}$. It follows that $E(X) = (E(K)|f(K) = v)$ is ϵ-close to uniform (and this holds for every $v \in \mathsf{Supp}(f(K))$. If $\epsilon = 0$ then this implies that $K' = E(K)$ is uniformly distributed and independent of $f(K)$. For general $\epsilon > 0$, this implies the statement in the claim above. (We remark that this argument can be generalized for other choices of "allowed functions" f as explained in Section 2.6.

Note that this application requires E to be explicit. Furthermore, Alice and Bob would like the new key to be as long as possible and this motivates extractors

[4] We remark that in the literature on extractors these sources are sometimes referred to as "oblivious bit-fixing sources" to distinguish them from "non-oblivious bit-fixing sources" in which for $i \notin \{i_1, \ldots, i_k\}$, X_i is some arbitrary function of $X_{i_1}, \ldots, X_{i_k}$.

that extract as many bits as possible from the k random bits that are "present" in a k-bit-fixing source.

It is trivial that the function $E(x) = (\sum_{1 \le i \le n} x_i) \bmod 2$ is a 0-extractor for k-bit-fixing sources for every $k \ge 1$. This simple idea does not easily extend to extract many bits for general k, as it was shown in [CGH$^+$85] that there are no 0-extractors with $m > 1$ for $k < n/3$. The problem of extracting many bits was considered in [CGH$^+$85,KZ07,GRS06,Rao09b] and the current state of the art is that there are explicit extractors that extract $m = (1 - o(1))k$ random bits from k-bit-fixing sources for every $k \ge \text{polylog}(n)$.

2.6 Families of sources in the literature on deterministic extraction

The literature on deterministic extractors and dispersers considers many families of sources. We briefly survey some of these families below. Our emphasis in the discussion below is on the min-entropy threshold parameter k. This is partly because there are general techniques to increase the output length of deterministic extractors [Sha08] and dispersers [GS08]. Using these techniques, it is often the case that extractors and dispersers that extract $\Omega(\log n)$ bits can be transformed into ones that extract $(1 - o(1)) \cdot k$ bits.

Multiple independent sources: Let $n = \ell \cdot n'$ and identify $\{0,1\}^n$ with $(\{0,1\}^{n'})^\ell$. A distribution $X = (X_1, \ldots, X_\ell)$ is an ℓ-independent source if $X_1, \ldots, X_\ell$ are independent. There exist extractors for 2-independent sources assuming $H_\infty(X_1), H_\infty(X_2) \ge \Omega(\log n)$. Most of the literature on deterministic extractors and dispersers focuses on multiple independent sources. We focus on this setup in Section 4.

Affine sources: Let $\mathbb{F}_q$ be the finite field with q elements. Affine sources are distributions that are uniform over some affine subspace of the vector space $\mathbb{F}_q^n$. The min-entropy of such sources coincides with the dimension of the affine space. Most of the research on explicit constructions focuses on the case $q = 2$ [BKS$^+$10,Bou07,Rao09b,BSK09,Yeh10,Li11b,Sha11]. Explicit constructions of extractors and dispersers for affine sources are far from approaching the existential bounds proven using the probabilistic method. The latter show existence of extractors for affine sources with min-entropy at least $k = O(\log n)$. The best known explicit extractor is due to Bourgain [Bou07] and works for affine sources with min-entropy at least $k = o(n)$ (slight improvements to $k = n/\sqrt{\log \log n}$ were given in [Yeh10,Li11b]). It is possible to do better for dispersers and Shaltiel [Sha11] constructs an explicit disperser for affine sources with min-entropy at least $k = n^{o(1)}$. Explicit constructions do much better for "large fields" in which $q = n^{\Theta(1)}$ [GR08,GS08,DG10] and are able to extract from affine sources with min-entropy $O(\log n)$.

Feasibly generated sources: This approach considers families of sources that are specified by placing limitations on the process that generates the source. It was initiated by Blum [Blu86] that considered sources generated by finite Markov chains (see also [Vaz87]). A computational perspective was suggested by Trevisan and Vadhan [TV00] (see also [KM05,KRVZ11]) who consider

sources $X = C(U_r)$ where $C : \{0,1\}^r \to \{0,1\}^n$ is a "computational device". Different families of sources are obtained by placing limitations on the complexity of C. Note that affine sources are captured if C is a degree one polynomial. It is also natural to consider polynomials with larger degree [DGW09].

Feasibly recognizable sources: This approach (explicitly suggested in [Sha09]) considers sources that are uniform over sets of the form $\{x : f(x) = v\}$ for functions f coming from some specified class. Note that bit-fixing sources are captured by considering functions that are projections and affine sources are captured (once again) by considering degree one polynomials. It is also natural to consider polynomials with larger degrees [Dvi09]. Other families of sources can also be captured thus way: Sources recognizable by decision trees are convex combinations of bit-fixing sources and sources recognizable by 2-party communication protocols are convex combinations of 2-independent sources. (This is useful as an extractor for some family $\mathcal{C}$ is also an extractor for convex combinations of sources from the family). We also remark that the argument of Section 2.5 showing that Alice and Bob can recover their privacy when the function f is a projection, immediately extends to any class of functions f if one can explicitly construct an extractor for distributions recognizable by the class. Other applications of such extractors are given in [Sha09,KvMS09].

3 Seeded extractors

These are extractors that in addition to one sample from the source distribution X also receive a second input Y (called "seed") which consists of (few) independent truly random bits.

Definition 7 (seeded extractors). *[NZ96] A function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k,ϵ)-*extractor *if for every distribution X over $\{0,1\}^n$ with $H_\infty(X) \geq k$, $E(X,Y)$ is ϵ-close to uniform (where Y is distributed like U_d and is independent of X). E is a (k,ϵ)-*disperser *if $|\mathsf{Supp}(E(X,Y))| \geq (1-\epsilon)\cdot 2^m$. E is a* strong extractor *if $E'(x,y) = (E(x,y),y)$ is an extractor.*

The definition above does not consider specific families $\mathcal{C}$ of sources. This is because seeded extractors are able to use a logarithmic length seed to extract from the "maximal family" of all distributions with "large" min-entropy.

In this section we survey some of the research on seeded extractors. In Section 3.1 we discuss explicit constructions and lower bounds. In Section 3.2 we explain how seeded extractors can be used to efficiently simulate randomized algorithms with access to a weak source. It turns out that seeded extractors have many other applications beyond the original motivation discussed in Section 1. We discuss some of these applications below. In Section 3.3 we point out a useful connection between seeded extractors and list-deocdable error-correcting codes. In Section 3.4 we interpret seeded extractors as graphs with large "relative expansion". In Section 3.5 we show that seeded extractors can be used to construct graphs with

expansion beating eigenvalue bounds. In Section 3.6 we point out that seeded extractors yield optimal averaging samplers.

3.1 Explicit constructions and lower bounds

By the probabilistic method it is not hard to show that for every n, k, ϵ there exist (k, ϵ)-extractors that use a seed of length $d = \log(n-k) + 2\log(1/\epsilon) + O(1)$, and output $m = k + d - 2\log(1/\epsilon) - O(1)$ bits. There are lower bounds of Radhakrishnan and Ta-Shma [RTS00] showing that this is optimal (except for the constants hidden in the $O(1)$).

The quantity $k + d - m$ is referred to as the "entropy loss" of the extractor (as the input distribution (X, Y) has min-entropy $k + d$). It is obvious that the entropy loss is always non-negative. The lower bounds of [RTS00] stated above show that the entropy loss is at least $2\log(1/\epsilon) - O(1)$. This means that as ϵ decreases, some randomness must be lost in the extraction process.

A long line of research attempts to match the parameters of the existential bounds with explicit constructions.[5] There are explicit constructions that achieve:

Extractors optimal up to constant factors: For every constant $\alpha > 0$ there exists a constant c such that for every n, k, ϵ there are explicit extractors with seed length $d = c \cdot (\log n + \log(1/\epsilon))$, output length $m = (1-\alpha)k$. This was achieved by Lu et al. [LRVW03] for constant error ϵ and by Guruswami, Umans and Vadhan and [GUV09] for general error.

Extractors with sublinear entropy loss for large error: For every constant e there is a constant c such that for every n, k there are explicit extractors with seed length $d = c \cdot \log n$ and output length $m = (1 - \frac{1}{\log^e n}) \cdot k$ and error $\epsilon = 1/\log n$. This was achieved by Dvir et al. [DKSS09].

We remark that it is sometimes possible to do better for specific values of k, and that there are constructions that can push the leading constant c to $1 + o(1)$ while paying a little bit in some of the other parameters. The reader is referred to a survey article on explicit constructions of seeded extractors [Sha02].

3.2 Simulating BPP with access to a weak random source

Unlike deterministic extractors, seeded extractors expect to receive a short seed of truly random bits. Such a seed is not available in the scenario described in Section 1. Nevertheless, we now explain how to simulate any polynomial time randomized algorithm in polynomial time when given access to a general weak random source with sufficient min-entropy.

[5] Explicit seeded extractors are defined in a similar fashion to deterministic extractors: Let $d(\cdot), k(\cdot), m(\cdot)$ and $\epsilon(\cdot)$ be functions over integers. A function E that takes two strings and returns a string is an explicit $(k(\cdot), \epsilon(\cdot))$-extractor if for every sufficiently large n, when the first input x is of length n, E uses seed length $d(n)$, has output length $m(n)$ and is a $(k(n), \epsilon(n))$-extractor.

Let A be a randomized algorithm that runs in polynomial time and solves some decision problem with error $\leq 1/3$ (meaning that for every input, the probability that A answers incorrectly is at most $1/3$ where the probability is over choosing random coins for A). Assume that on an input x' of length n', A requires $m = \text{poly}(n')$ truly random bits. Assume that we can sample from some unknown distribution X over $\{0,1\}^n$ where $n = \text{poly}(n')$ with the guarantee that $H_\infty(X) \geq k$ for $k \geq 2m$. Let $E : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to \{0,1\}^m$ be an explicit $(k, \frac{1}{10})$-extractor (that exists by the discussion in Section 3.1).

We simulate A as follows: When given input $x' \in \{0,1\}^{n'}$ and a sample x from the source X, for every seed y of E, we compute a bit v_y by applying A on input x' using $E(x,y)$ as a sequence of "random coins". The final output is the value v such that $v_y = v$ for most seeds y. It is not hard to see that for every input x' this process solves the decision problem with error less than $1/3 + 1/10 < 1/2$ where the probability is over the choice of x from X.[6]

Note that for this application it is crucial that E is explicit. Moreover, the simulation described above goes over all 2^d seeds of the extractor E. Consequently, it is crucial that $d = c \log n$ for some constant c to obtain a polynomial time simulation. Furthermore, the constant c determines the exponent of the polynomial (which gives motivation for constructing extractors with a small leading constant c) [TSZS06,SU05].

Inapplicability of this approach in cryptographic or distributed settings. It is important to stress that while this approach works for simulating randomized algorithms, it is not applicable when simulating randomized protocols in cryptographic or distributed settings. This is because the approach sketched above requires running the original protocol $2^d = n^{O(1)}$ times with many sequences of random coins. In cryptographic settings this means that adversaries get to participate in interactions in which the key is not necessarily random (which compromises the security of the protocol). In distributed settings, the total overhead incurred in running the initial protocol $n^{O(1)}$ times, leads to protocols that are inefficient and uninteresting. In Section 4.1 we suggest an alternative approach to simulate randomized protocols given access to multiple independent sources.

3.3 Seeded extractors and list-decodable error-correcting codes

List-decodable error-correcting codes have many applications in computer science and are extensively studied. The reader is referred to [Gur07] for a survey articles on this notion and its applications. The definition below uses a nonstandard choice of letters preparing for the application below.

Definition 8 (List-decodable code). *For $x, y \in \{0,1\}^n$, let $\delta(x,y)$ denote the relative Hamming distance of x and y, that is $\delta(x,y) = \frac{|\{i : x_i \neq y_i\}|}{n}$. A function*

[6] In fact, the analysis can be improved and show that the error probability is bounded by $2^{-\Omega(k)}$ if we set the extractor to extract from distributions with min-entropy k' for $m \leq k' \leq \alpha k$ for a constant $\alpha < 1$. This is because of the connection between extractors and averaging samplers that we explain later on in Section 3.6.

$C : \{0,1\}^n \to \{0,1\}^{2^d}$ *is an* (ℓ, ϵ)*-list-decodable code if for every* $z \in \{0,1\}^{2^d}$, $\left|\left\{x : \delta(C(x), z) \leq \frac{1}{2} - \epsilon\right\}\right| \leq \ell$.

The definition above says that if one encodes a string $x \in \{0,1\}^n$ by $C(x)$ and then transmits $C(x)$ on a "noisy channel" that is allowed to adversarially choose $1/2 - \epsilon$ of the indices of $C(x)$ and flip them to obtain a string z, then the receiver (who only sees z) knows a list of at most ℓ messages, such that one of them is the original message x. (The more standard notion of uniquely decodable codes is captured by the special case where $\ell = 1$ and it is known that such codes can only exist for $\epsilon > 1/4$.)

It turns out that strong seeded extractors that extract a single bit are essentially equivalent to list-decodable error correcting codes using the translation $E(x, y) = C(x)_y$ with $k \approx \log \ell$. This was first observed by Trevisan [Tre01]. A precise statement is given below:

- If $C : \{0,1\}^n \to \{0,1\}^{2^d}$ is an (ℓ, ϵ)-error correcting code then $E(x, y) = C(x)_y$ is a strong $(k, 2\epsilon)$-extractor for $k = \log \ell + \log(1/\epsilon) + 1$.
- If $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}$ is a strong (k, ϵ)-extractor then $C(x)$ defined by $C(x)_y = E(x, y)$ is a $(2^k - 1, 2\epsilon)$-list decodable code.

Proof. (sketch) For the second item, we note that if C is not a list-deocdable code, then there exists z which violates Definition 8 and has a list of size at least 2^k. The extractor E can be shown to fail on the distribution X that is uniform on this set. This is because for every $x \in \mathsf{Supp}(X)$, the "predictor function" $p(y) = z_y$ has $\Pr[p(Y) = E(x, Y)] \geq 1/2 + 2\epsilon > 1/2 + \epsilon$. It follows that $\Pr[p(Y) = E(X, Y)] > 1/2 + \epsilon$ which is a contradiction to the correctness of the extractor.

For the first item we note that if E is not a strong extractor, then there exist a source X and an event A such that $\Pr[(E(X, Y), Y) \in A]$ and $\Pr[U_{d+1} \in A]$ differ by more than 2ϵ. By standard arguments, such an event gives rise to a "predictor function" $p : \{0,1\}^d \to \{0,1\}$ that has $\Pr[p(Y) = E(X, Y)] > 1/2 + 2\epsilon$. (Intuitively, this follows because Y is uniformly distributed and so A does not gain from trying to distinguish Y from uniform, and has to be able to predict $E(X, Y)$ when given Y). By an averaging argument, there exist a set S consisting of an ϵ fraction of $x \in \mathsf{Supp}(X)$ such that for every $x \in S$ we have $\Pr[p(Y) = E(x, Y)] > 1/2 + \epsilon$. Function p gives rise to a string $z \in \{0,1\}^{2^d}$ by $z = p(y)_{(y \in \{0,1\}^d)}$ which contradicts Definition 8 as any message in S belongs to the list of z. □

The translation above can be used to present a *unified theory* of extractors, error-correcting codes (as well as other objects such as hash functions and expander graphs). The reader is referred to [Vad07] for such a treatment.

Exploiting this connection in explicit constructions. The connection above immediately gives excellent explicit constructions of strong extractors that extract one bit (by using known list-decodable codes). In order to extract many bits it is natural to have $E(x, y) = C(x)_{y_1}, \ldots, C(x)_{y_m}$ for some mapping $y \to y_1, \ldots, y_m$.

This approach is used by many constructions starting with Trevisan's breakthrough construction [Tre01] (see also [RRV02]) in which the mapping used is the Nisan-Wigderson pseudorandom generator [NW94]. A different mapping is used by Shaltiel and Umans [SU05] which also relies on a specific choice of the code. The aforementioned recent construction of Guruswami, Umans and Vadhan [GUV09] exploits this connection by using recent advances in coding theory (more specifically the Parvaresh-Vardy code [PV05]). The Parvaresh-Vardy code is not a code with Boolean alphabet, and so the translation above does not work directly. Nevertheless, a similar argument to the one given above can be used to construct "unbalanced expander graphs" which in turn yield seeded extractors.

A coding theoretic interpretation of extracting many bits. It was observed by Ta-Shma and Zuckerman [TSZ04] that strong extractors (k, ϵ)-extractors that extract $m > 1$ bits can be viewed (by the translation above) as codes over alphabet $\{0,1\}^m$ that allow list-decoding against channels that are extremely noisy in the following sense: When an encoding $C(x)$ of a message x passes through the channel, for every symbol $C(x)_y = E(x, y)$ of the encoding, the receiver gets a set $S_y \subseteq \{0,1\}^m$ of size say $2^m/2$ with the guarantee that $C(x)_y \in S_y$ for every y. "Extractor codes" allow recovery against such channels in the sense that the receiver knows a list of size 2^k of messages such that one of them is the original message x. In fact, extractor codes are resilient even against stronger channels that also "add errors" and are allowed adversarially choose $1/2 - \epsilon$ indices y in which S_y does not satisfy $C(x)_y \in S_y$.

3.4 Seeded dispersers as graphs with expansion properties

Given a function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ we set $N = 2^n$, $M = 2^m$, $D = 2^d$ and define a bipartite graph G_E where the left hand set of vertices is $\{0,1\}^n$, the right hand side set of vertices is $\{0,1\}^m$, and each vertex $x \in \{0,1\}^n$ is connected to $E(x, y)$ for every $y \in \{0,1\}^d$. Thus, the degree of vertices on the left hand side is D (and note that we allow multiple edges).

For a set $S \subseteq \{0,1\}^n$ on the left hand side, we define $\Gamma(S)$ to be the set of neighbors of S. It follows that if E is a (k, ϵ)-disperser (which follows in case E is a (k, ϵ)-extractor) then every set S of size at least $K = 2^k$ has $|\Gamma(S)| \geq (1-\epsilon) \cdot 2^m$. This notion resembles "vertex expansion" in so called "expander graphs". The reader is referred to [HLW06] for a manuscript on expander graphs and their many applications. We give a definition of vertex expansion below.

Definition 9 (Bipartite expander graphs). *A bipartite graph G is a (K, e)-expander if any set S on the left hand side of size at most K has $|\Gamma(S)| \geq e \cdot |S|$.*

In the definition above we consider bipartite graphs. This requirement is made to easily compare expander graphs with "disperser graphs". Note that standard expander graphs easily translate into bipartite ones. (Given a standard non-bipartite graph G in which every not too large set $S \subseteq V$ expands, we can create two copies of the vertices and imagine that edges go from the left-hand copy to the right-hand copy).

size vs. volume: In bipartite expander graphs sets expand in size in the sense that $|\Gamma(S)| \geq e \cdot |S|$. Disperser graphs may not expand in size and that all sets S have $|\Gamma(S)| < |S|$. Nevertheless, in disperser graphs, the set S expands in "volume" (the ratio of the size of the set and the size of the universe it lives in). More precisely, the volume of $S \subseteq \{0,1\}^n$ is $\frac{|S|}{N}$ (that may be very small and tend to zero), while the volume of $\Gamma(S) \subseteq \{0,1\}^m$ is $\frac{|\Gamma(S)|}{M} \geq (1-\epsilon)$.

Balanced vs. Unbalanced graphs: In many settings (and in particular in the setting of the application considered in Section 3.2) disperser graphs are "unbalanced", meaning that right had side which is of size M is much smaller than the left hand side which is of size N. Bipartite expander graphs are typically balanced in and the left hand side is of the same size as the right hand side. Nevertheless, it is interesting and useful to consider unbalanced bipartite expander graphs [TSUZ07,CRVW02,GUV09].

Constant degree vs. logarithmic degree: A useful property of expander graphs is that it is possible to have such graphs with constant degree. In contrast, the aforementioned lower bounds of [RTS00] imply that extractor graphs and disperser graphs must have degree at least $D \geq \Omega(\log(\frac{N}{K}))$ which is constant for $K = \Omega(N)$ and non-constant if $K = o(N)$. This means that disperser graphs cannot have constant degree if $K = o(N)$ (and the graph is unbalanced). Nevertheless, even bipartite expanders cannot have constant degree when the graph is sufficiently unbalanced. We perceive this observation as saying that the issue here is balanced vs. unbalanced rather than expander vs. disperser.

Sets smaller than K vs. sets larger than K: In expander graphs every set smaller than K expands, while in disperser graphs every set of size larger than K expands. This difference is a consequence of the difference between the notions of size and volume. In expander graphs, sets which are too large do not have "room" to expand in size, while in disperser graphs, sets which are too small cannot possibly expand to volume that is almost one unless the degree is huge.

There are many applications of extractor and disperser graphs in the literature. In some cases, extractors and dispersers give better performance than expander graphs. We present such examples in the next two sections.

3.5 Graphs with expansion beating the eigenvalue bound

We now present a result of Wigderson and Zuckerman [WZ99] showing that dispersers can be used to construct expanders with very strong expansion (for a particular parameter regime). We consider the following problem: Let $A \leq N/10$ be a parameter. Design a graph with small degree on N vertices such that every two sets of N/A vertices have an edge between them. (This is indeed a form of vertex expansion as it means that every set of size N/A sees more than $N - N/A \geq \frac{9N}{10}$ vertices).

Obviously, it is impossible to achieve this property with degree $o(A)$. The probabilistic method shows existence of such graphs with degree $\approx A \cdot \log(A)$. If

one achieves vertex expansion by analyzing the "spectral gap" of the graph, then the best possible degree is $\approx A^2$ in the sense that analysis of the spectral gap gives degree $\approx A^2$, and there exist graphs with optimal spectral gap in which the degree is $\approx A^2$ [Kah06].

Using optimal dispersers it is possible to almost match the probabilistic method and obtain degree $A \cdot \text{polylog}(A)$ (and such bounds can be approached using explicit constructions of dispersers for various choices of A [RVW00,SSZ98,TS02]).

The construction of [WZ99] works as follows: Let $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a $(k, 1/4)$-disperser for $k = \log(N/A)$ and $m = k$. (Furthermore, we need to assume that all nodes on the right hand side of the disperser graph have degree not much larger than the average degree given by ND/M). Let S_1, S_2 be two sets of size $N/A = 2^k$. By the disperser property each of the two sets sees more than half the vertices on the right hand side. Therefore, S_1 and S_2 both see a common neighbor on the right hand side. Define a graph G on $\{0,1\}^n$ in which every two vertices are connected if they share a common neighbor in the disperser graph. By the previous argument every sets S_1, S_2 of size N/A have an edge between them. Moreover, the degree of the graph is given by $D \cdot \frac{ND}{M} = \frac{D^2 N}{M}$ which is indeed $A \cdot \text{polylog}(A)$ if we use a disperser graph with small degree $D = \text{polylog}(N/K)$ (or equivalently short seed $d = O(\log(n-k))$).

Another example of graphs with expansion beating the eigenvalue bound is given by Capalbo et al. [CRVW02].

3.6 Seeded extractors and averaging samplers

Let $Z_1, \ldots, Z_D$ be independent random variables over $\{0,1\}^m$, let $A \subseteq \{0,1\}^m$ and define indicator random variables $R_1, \ldots, R_d$ by $R_i = 1$ if $Z_i \in A$ and $R_i = 0$ otherwise. Let $Z = 2^{-D} \cdot \sum_{1 \le i \le D} Z_i$. The Chernoff bound gives that

$$\Pr[\left| Z - \frac{|A|}{2^m} \right| \ge \epsilon] \le \delta$$

for $\delta = 2^{-\Omega(\epsilon^2 D)}$. We say that random variables $Z_1, \ldots, Z_D$ (that are not necessarily independent) form an *averaging sampler* with estimation error ϵ and sampling error δ if they satisfy the inequality above for the parameters ϵ, δ.

Consider graphs over the vertex set $\{0,1\}^m$. A useful property of such graphs with "large spectral gap" is that if we choose a vertex Z_1 at random and take a random walk of length D on the graph to generate random variables $Z_1, \ldots, Z_D$ then we obtain random variables which form an averaging sampler (with quality that depends on the spectral gap) [AKS87,Gil98]. The advantage of this approach is that if the graph has constant degree then it is possible to generate the random walk variables $Z_1, \ldots, Z_D$ using only $m + O(D)$ random bits (compared to the $m \cdot D$ bits required to generate D independent random variables). This property has many applications in "derandomization theory" as it allows to approximate $|A|/2^m$ with small additive error using few random bits.

It was observed by Zuckerman [Zuc97] that seeded extractors yield averaging samplers with parameters that can beat those given by graphs with large spectral gap. The connection works as follows:

Extractors yield averaging samplers: Let $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (k,ϵ)-extractor. Sample x uniformly from $\{0,1\}^n$, and set $Z_y = E(x,y)$. This gives random variables that are an averaging sampler with estimation error ϵ and sampling error $\delta = 2^{k+1}/2^n$. Moreover, sampling these variables requires n random bits.

Proof. (sketch) For every set $A \subseteq \{0,1\}^m$ we define

$$S_A = \left\{ x : \left| \Pr[E(x,Y) \in A] - \frac{|A|}{2^m} \right| > \epsilon \right\}$$

to be the set of strings with which the averaging sampler fails to estimate A correctly. To bound the size of S_A we note that w.l.o.g. at least half of its elements have the property above without the absolute value. The distribution X that is uniform over these elements is a source on which A distinguishes the output of the extractor from uniform. It follows that X does not have large min-entropy, which implies that S_A is small. □

It turns out that the connection between extractors and averaging samplers can also be reversed. Any procedure that uses n random bits to generate D random variables that form an averaging sampler with estimation error ϵ and sampling error δ, immediately translates into a $(k, 2\epsilon)$-extractor with $k = n - (\log(1/\delta) - \log(1/\epsilon))$ by setting $E(x,y) = Z_y$. Summing up, seeded extractors are essentially equivalent to averaging samplers under the translation $k \approx n - \log(1/\delta)$.

A consequence of this connection is that graphs with large spectral gap yield averaging samplers which in turn yield seeded extractors. This relationship is useful to construct extractors for very large k ($k \geq (1-\alpha)n$ for some constant $\alpha > 0$) but breaks down for smaller values.

4 Extractors for multiple independent sources

Seeded extractors receive one source X (with the guarantee that $H_\infty(X) \geq k$ for some parameter k), and an independent short seed Y (that is uniformly distributed). The assumption that Y is uniformly distributed seems very strong, and we can try and relax it. A natural relaxation is to replace the requirement that Y is uniformly distributed by the weaker requirement that Y has large min-entropy. In this setup, it is no longer necessary to require that Y is short. A more natural setting of parameters is to have Y have the same length and min-entropy threshold as X. This leads to the notion of extractors for two independent sources. In fact, once we consider two independent sources, we may as well consider ℓ independent sources.

Definition 10 (Extractors and dispersers for multiple independent sources). *A function $E : (\{0,1\}^n)^\ell \to \{0,1\}^m$ is a (k,ϵ) ℓ-source extractor if for every ℓ independent random variables $X_1, \ldots, X_\ell$ such that for every $1 \leq i \leq \ell$, $H_\infty(X_i) \geq k$, it holds that $E(X_1, \ldots, X_\ell)$ is ϵ-close to U_m. E is a (k,ϵ) ℓ-source disperser if $|\mathsf{Supp}(E(X_1, \ldots, X_\ell))| \geq (1-\epsilon) \cdot 2^m$.*

We remark that these extractors are often seen as a special case of deterministic extraction (as explained in Section 2.6). We have seen in Section 2.1 that $(n-1,\epsilon)$ 1-source extractors/dispersers do not exist for $\epsilon < 1$. By the probabilistic method, 2-source extractors exist even for $k = O(\log n + \log(1/\epsilon))$.

In this section we survey some of the research on extractors and dispersers for multiple independent sources. In Section 4.1 we explain that multiple source extractors can be used to generate keys for cryptographic protocols. In Section 4.2 we show that explicit 2-source dispersers can be used to construct Ramsey graphs. In Section 4.3 we survey some of the explicit constructions extractors and dispersers for multiple independent sources.

4.1 Generating keys for cryptographic protocols

In Section 3.2 we saw that it is possible to simulate randomized algorithms efficiently given access to *one* source. However, as explained there, that simulation is not applicable in cryptographic or distributed settings. Let us focus on cryptographic protocols. The security of such protocols depends on the ability of honest parties to generate uniformly distributed and private random keys. Moreover, in such settings the computer of an honest party may be operating in a "hostile environment" set up by an adversary that is trying to steal the secrets of the honest party.

ℓ-source extractors enable an honest party to sample a string that is (close to) uniform, assuming the party has access to ℓ independent sources. Each such source may be implemented by one of the approaches explained in Section 1. The requirement from each individual weak source is minimal: It should contain some min-entropy. It is plausible to assume that samples taken from sources that are "unrelated" or "remote" are independent. If we accept this assumption then we can generate keys for cryptographic protocols. Moreover, by the aforementioned discussion we can hope to have $\ell = 2$ and we only require independence between two random variables.

On a philosophical level the ability to generate independent random variables is a pre-requisite of cryptography. This is because a world in which this is not possible is a world in which there are no secrets (as every secret that we generate is correlated with other random variables that may become known to the adversary when we interact with him).

4.2 2-source dispersers and Ramsey graphs

An (undirected) graph G on N vertices is K-Ramsey if there are no cliques or independent sets of size K in G. In 1947 (in the paper that introduced the celebrated probabilistic method) Erdös showed the existence of K-Ramsey graphs for $K \approx \log N$. It is a longstanding challenge to explicitly construct such graphs. Until recently, the best results were achieved by the classical work of Frankl and Wilson [FW81] and matched by several other researchers [Alo98,Gro00] giving K-Ramsey graphs for $K \approx 2^{\sqrt{\log N}}$. Moreover, Gopalan [Gop06] showed that

some of the techniques used to attack this problem cannot beat this bound. We now observe that explicit errorless two-source dispersers that output one bit yield explicit constructions of Ramsey graphs.

2-source dispersers yield Ramsey graphs: An explicit $(k, 0)$ 2-source disperser $D : (\{0,1\}^n)^2 \to \{0,1\}$ translates into an explicit construction of 2^{k+1}-Ramsey graph on 2^n vertices.

In fact, 2-source dispersers yield bipartite-Ramsey graphs (which are even harder to construct than Ramsey graphs). We now explain this argument. Let $D : (\{0,1\}^n)^2 \to \{0,1\}$ be a $(k, 0)$ 2-source disperser. We can use D to define a bipartite graph $B = (L, R, E)$ by interpreting it as an adjacency matrix. More formally, the left hand set of vertices is $L = \{0,1\}^n$, the right hand set is $R = \{0,1\}^n$ and two nodes $v_1 \in L$, $v_2 \in R$ are connected iff $D(v_1, v_2) = 1$. Graph B is a balanced bipartite graph where each of the two sides has $N = 2^n$ vertices. Furthermore, G has the property that for every two sets $A, B \subseteq \{0,1\}^n$ of size $K = 2^k$, $D(A, B) = \{0,1\}$ meaning that every $K \times K$ induced subgraph of G cannot be empty nor complete. Such graphs are called "K-bipartite Ramsey graphs".

In particular, we have that for every set $A \subseteq \{0,1\}^n$ of size $K = 2^k$, $D(A, A) = \{0,1\}$. If D is symmetric (meaning that $D(x, y) = D(y, x)$) then we can also interpret it as the adjacency matrix of a (non-bipartite) undirected graph over vertex set $\{0,1\}^n$ and the property above means that D is a K-Ramsey graph (as every set A of size K is not a clique nor an independent set). We can make D symmetric by ordering the elements in $\{0,1\}^n$ in some arbitrary way and modifying $D(x, y)$ to $D(y, x)$ for $x > y$. (We can also force $D(x, x) = 0$ if we want to avoid self loops). This modification does not spoil D by much. It is easy to see that if D is a $(k, 0)$ 2-source disperser than following the modification it is still a $(k+1, 0)$ 2-source disperser. Summing up, when given disperser D we can symmetrize it and use it to define the adjacency matrix of a Ramsey graph.

4.3 Explicit constructions of ℓ-source extractors and dispersers

2-*source extractors and dispersers.* The discussion above explains some of the difficulty in constructing explicit 2-source extractors and dispersers for small k. We now survey the known constructions. Chor and Goldreich [CG88] showed that $E(x, y) = (\sum_{1 \le i \le n} x_i \cdot y_i) \bmod 2$ is a 2-source extractor with very small error if k is sufficiently larger than $n/2$. (Earlier work by Santha and Vazirani [SV86] considered two independent Santha-Vazirani sources and analyzed the same extractor function). Bourgain [Bou05] (see also [Rao07]) improved the entropy threshold to $k = (1/2 - \alpha)n$ for some small constant $\alpha > 0$. (The seemingly small difference between $n/2$ and $(1/2 - \alpha)n$ plays a crucial role in some of the recent developments in this area). This is the best known extractor construction for two sources with the same min-entropy. Raz [Raz05] constructed 2-source extractors where one source has min-entropy larger than $n/2$ and the other has

logarithmic entropy. Shaltiel [Sha08] used Raz's extractor to give a 2-source extractor which for two sources with min-entropy $k > n/2$ extracts $(2 - o(1)) \cdot k$ random bits out of the $2k$ bits of entropy that are available in the two sources.

Barak et al. [BKS+10] constructed 2-source dispersers that for every $\delta > 0$ achieve $k = \delta n$. Barak et al. [BRSW06] extended the technique of [BKS+10] and constructed 2-source dispersers for $k = 2^{\log^{0.9} n} = n^{o(1)}$. By the discussion in Section 4.2 such dispersers translate into Ramsey graphs that beat the Frankl-Wilson construction [FW81] and give the state of the art on this problem. The constructions of [BKS+10,BRSW06] are quite involved and rely on many components from the extractor literature.

ℓ-source extractors. Barak, Impagliazzo and Wigderson [BIW06] constructed extractors that for every $\delta > 0$, use poly$(1/\delta)$ sources with min-entropy $k = \delta n$. The key was exploiting recent developments in arithmetic combinatorics (in particular, the "sum-product theorem" of [BKT04,Kon03]) to analyze a construction that was previously suggested by Zuckerman. The high level idea is to show that if X, Y, Z are three independent sources with min-entropy δn, where each source is over a finite field $\mathbb{F}$ that has no large subfields (which holds vacuously if $\mathbb{F}$ is a prime field) then $X \cdot Y + Z$ is a distribution that is (close to) having min-entropy $min((\delta + \alpha)n, n)$ for some constant $\alpha > 0$. By iteratively increasing the entropy, this gives an extractor.

Rao [Rao09a] used machinery from seeded extractors to construct extractors that extract randomness from $O(\frac{\log n}{\log k})$ independent sources with min-entropy k. Note that this means that only $O(1/\delta)$ sources are needed for $k = n^{\delta}$. Rao starts by observing that every source X over $\{0,1\}^n$ with $H_\infty(X) \geq k$ can be transformed into a "somewhere-random" source X' consisting of $n^{O(1)}$ blocks of length $\Omega(k)$ where at least one of them is (close to) uniformly distributed. (Each such block B_y is obtained by $B_y = E(X, y)$ where E is a (k, ϵ)-seeded extractor with seed length $O(\log n)$ so that there are $n^{O(1)}$ blocks). This reduces the task of extracting from independent general sources to that of extracting from independent somewhere random sources, and this turns out to be easier.

In recent years there is ongoing research that aims to improve the results above.

5 Some open problems

Below we state some open problems related to explicit constructions of extractors. We remark that there are many other open problems that are related to applications of extractors.

Deterministic extractors.

- Construct affine extractors for the field $\mathbb{F}_2$ with min-entropy $k < \sqrt{n}$. The best known construction achieves $k = n/\sqrt{\log \log n}$ [Bou07,Yeh10,Li11b].
- Construct affine dispersers for the field $\mathbb{F}_2$ with min-entropy $k = \text{polylog}(n)$. The best known construction achieves $k = 2^{\log^{0.9} n}$ [Sha11].

– For every constant c, construct extractors for sources samplable by circuits of size n^c with min-entropy $k < n/2$. It is allowed to use any "plausible hardness assumption". A construction based on worst-case hardness against a strong variant of nondeterministic circuits was given in [TV00]. This construction requires $k = (1 - \alpha)n$ for some constant $\alpha > 0$.

Seeded extractors.

– Construct seeded extractors that match the lower bounds of [RTS00]. These lower bounds and state of the art are described in Section 3.1 (see also [Sha02]).

Multiple source extractors.

– Construct 2-source extractors for min-entropy $k \leq n/3$. The best known construction achieves $k = (1/2 - \alpha)n$ for some constant $\alpha > 0$ [Bou05].
– Construct 2-source dispersers for min-entropy $k = \text{polylog}(n)$. The best known construction achieves $k = 2^{\log^{0.9} n}$ [BRSW06].
– Construct ℓ-source extractors for $\ell = O(1)$ and min-entropy $k = \text{polylog}(n)$. The best known construction achieves $\ell = O(\frac{\log n}{\log k})$ which is constant only for $k = n^{\Omega(1)}$ [Rao09a].

References

[AB09] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[AKS87] Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in logspace. In *STOC*, pages 132–140, 1987.

[Alo98] N. Alon. The shannon capacity of a union. *Combinatorica*, 18(3):301–310, 1998.

[BIW06] Boaz Barak, Russell Impagliazzo, and Avi Wigderson. Extracting randomness using few independent sources. *SIAM J. Comput.*, 36(4):1095–1118, 2006.

[BKS+10] Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, ramsey graphs, dispersers, and extractors. *J. ACM*, 57(4), 2010.

[BKT04] Jean Bourgain, Nets Katz, and Terence Tao. A sum-product estimate in finite fields, and applications. *Geom. Funct. Anal.*, 14(1):27–57, 2004.

[Blu86] Manuel Blum. Independent unbiased coin flips from a correlated biased source-a finite stae markov chain. *Combinatorica*, 6(2):97–108, 1986.

[Bou05] Jean Bourgain. More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory*, 1:1–32, 2005.

[Bou07] Jean Bourgain. On the construction of affine extractors. *Geometric And Functional Analysis*, 17(1):33–57, 2007.

[BRSW06] Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *STOC*, pages 671–680, 2006.

[BSK09] Eli Ben-Sasson and Swastik Kopparty. Affine dispersers from subspace polynomials. In *STOC*, pages 65–74, 2009.

[BSZ11] Eli Ben-Sasson and Noga Zewi. From affine to two-source extractors via approximate duality. In *STOC*, 2011.

[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. Special issue on cryptography.

[CGH+85] Benny Chor, Oded Goldreich, Johan Håstad, Joel Friedman, Steven Rudich, and Roman Smolensky. The bit extraction problem of t-resilient functions (preliminary version). In *FOCS*, pages 396–407, 1985.

[CRVW02] Michael R. Capalbo, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Randomness conductors and constant-degree lossless expanders. In *STOC*, pages 659–668, 2002.

[DG10] Matt DeVos and Ariel Gabizon. Simple affine extractors using dimension expansion. In *IEEE Conference on Computational Complexity*, pages 50–57, 2010.

[DGW09] Zeev Dvir, Ariel Gabizon, and Avi Wigderson. Extractors and rank extractors for polynomial sources. *Computational Complexity*, 18(1):1–58, 2009.

[DKSS09] Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *FOCS*, pages 181–190, 2009.

[Dvi09] Zeev Dvir. Extractors for varieties. In *IEEE Conference on Computational Complexity*, pages 102–113, 2009.

[Eli72] P. Elias. The efficient construction of an unbiased random sequence. *Ann. Math. Statist.*, 43:865–870, 1972.

[FW81] P. Frankl and R. M. Wilson. Intersection theorems with geometric consequences. *Combinatorica*, 1(4):357–368, 1981.

[Gil98] David Gillman. A chernoff bound for random walks on expander graphs. *SIAM J. Comput.*, 27(4):1203–1220, 1998.

[Gop06] P. Gopalan. Constructing ramsey graphs from boolean function representations. In *IEEE Conference on Computational Complexity*, pages 115–128, 2006.

[GR08] Ariel Gabizon and Ran Raz. Deterministic extractors for affine sources over large fields. *Combinatorica*, 28(4):415–440, 2008.

[Gro00] V. Grolmusz. Superpolynomial size set-systems with restricted intersections mod 6 and explicit ramsey graphs. *Combinatorica*, 20(1):71–86, 2000.

[GRS06] Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. *SIAM J. Comput.*, 36(4):1072–1094, 2006.

[GS08] Ariel Gabizon and Ronen Shaltiel. Increasing the output length of zero-error dispersers. In *APPROX-RANDOM*, pages 430–443, 2008.

[Gur07] Venkat Guruswami. Algorithmic results in list decoding. *Foundations and Trends in Theoretical Computer Science*, 2(2), 2007.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *J. ACM*, 56(4), 2009.

[HLW06] Shlomo Hoory, Nati Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. Amer. Math Soc.*, 43:439–561, 2006.

[Kah06] N. Kahale. Eigenvalues and expansion of regular graphs. *J. Assoc. Comput. Mach.*, 42:1091–1106, 2006.

[KLR09] Yael Tauman Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *FOCS*, pages 617–626, 2009.

[KLRZ08] Yael Tauman Kalai, Xin Li, Anup Rao, and David Zuckerman. Network extractor protocols. In *FOCS*, pages 654–663, 2008.

[KM05] Robert König and Ueli M. Maurer. Generalized strong extractors and deterministic privacy amplification. In *IMA Int. Conf.*, pages 322–339, 2005.

[Kon03] S.V. Konyagin. A sum-product estimate in fields of prime order. Arxiv technical report, http://arxiv.org/abs/math.NT/0304217, 2003.

[KRVZ11] Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *J. Comput. Syst. Sci.*, 77(1):191–220, 2011.

[KvMS09] Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom generators and typically-correct derandomization. In *APPROX-RANDOM*, pages 574–587, 2009.

[KZ07] Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM J. Comput.*, 36(5):1231–1247, 2007.

[Li11a] Xin Li. Improved constructions of three source extractors. In *IEEE Conference on Computational Complexity*, 2011.

[Li11b] Xin Li. A new approach to affine extractors and dispersers. In *IEEE Conference on Computational Complexity*, 2011.

[LRVW03] Chi-Jen Lu, Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Extractors: optimal up to constant factors. In *STOC*, pages 602–611, 2003.

[NTS99] Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *J. Comput. Syst. Sci.*, 58(1):148–173, 1999.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

[Per92] Yuval Peres. Iterating von neumanns procedure for extracting random bits. *Ann. Statist.*, 20:590–597, 1992.

[PV05] Farzad Parvaresh and Alexander Vardy. Correcting errors beyond the guruswami-sudan radius in polynomial time. In *FOCS*, pages 285–294, 2005.

[Rao07] Anup Rao. An exposition of bourgain's 2-source extractor. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(034), 2007.

[Rao09a] Anup Rao. Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM J. Comput.*, 39(1):168–194, 2009.

[Rao09b] Anup Rao. Extractors for low-weight affine sources. In *IEEE Conference on Computational Complexity*, pages 95–101, 2009.

[Raz05] Ran Raz. Extractors with weak random seeds. In *STOC*, pages 11–20, 2005.

[RRV02] Ran Raz, Omer Reingold, and Salil P. Vadhan. Extracting all the randomness and reducing the error in trevisan's extractors. *J. Comput. Syst. Sci.*, 65(1):97–128, 2002.

[RTS00] J. Radhakrishnan and A. Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM Journal on Discrete Mathematics*, 13(1):2–24, February 2000.

[RVW00] Omer Reingold, Salil P. Vadhan, and Avi Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *FOCS*, pages 3–13, 2000.

[RZ08] Anup Rao and David Zuckerman. Extractors for three uneven-length sources. In *APPROX-RANDOM*, pages 557–570, 2008.

[Sha02] Ronen Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the EATCS*, 77:67–95, 2002.

[Sha08] Ronen Shaltiel. How to get more mileage from randomness extractors. *Random Struct. Algorithms*, 33(2):157–186, 2008.

[Sha09] Ronen Shaltiel. Weak derandomization of weak algorithms: Explicit versions of yao's lemma. In *IEEE Conference on Computational Complexity*, pages 114–125, 2009.

[Sha11] Ronen Shaltiel. Dispersers for affine sources with sub-polynomial entropy. *Unpublished*, 2011.

[SSZ98] Michael E. Saks, Aravind Srinivasan, and Shiyu Zhou. Explicit or-dispersers with polylogarithmic degree. *J. ACM*, 45(1):123–154, 1998.

[SU05] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *J. Comput. Syst. Sci.*, 33(1):75–87, 1986.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

[TS02] Amnon Ta-Shma. Almost optimal dispersers. *Combinatorica*, 22(1):123–145, 2002.

[TSUZ07] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless condensers, unbalanced expanders, and extractors. *Combinatorica*, 27(2):213–240, 2007.

[TSZ04] Amnon Ta-Shma and David Zuckerman. Extractor codes. *IEEE Transactions on Information Theory*, 50(12):3015–3025, 2004.

[TSZS06] Amnon Ta-Shma, David Zuckerman, and Shmuel Safra. Extractors from reed-muller codes. *J. Comput. Syst. Sci.*, 72(5):786–812, 2006.

[TV00] Luca Trevisan and Salil P. Vadhan. Extracting randomness from samplable distributions. In *FOCS*, pages 32–42, 2000.

[Vad07] Salil P. Vadhan. The unified theory of pseudorandomness. *SIGACT News*, 38(3):39–54, 2007.

[Vaz87] Umesh V. Vazirani. Efficiency considerations in using semi-random sources (extended abstract). In *STOC*, pages 160–168, 1987.

[vN51] John von Neumann. Various techniques used in connection with random digits. *Applied Math Series*, 12:36–38, 1951.

[VV85] Umesh V. Vazirani and Vijay V. Vazirani. Random polynomial time is equal to slightly-random polynomial time. In *FOCS*, pages 417–428, 1985.

[WZ99] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. *Combinatorica*, 19(1):125–138, 1999.

[Yeh10] Amir Yehudayoff. Affine extractors over prime fields. *Manuscript*, 2010.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Struct. Algorithms*, 11(4):345–367, 1997.