Texts in Computational Complexity: Probabilistic Preliminaries and Advanced Topics in Randomization

Oded Goldreich
Department of Computer Science and Applied Mathematics
Weizmann Institute of Science, Rehovot, ISRAEL.

January 28, 2006

What is this? Chicken Quesadilla and Seafood Salad?
Fine, but in the same plate? This is disgusting!

Johan Håstad at Grendel's, Cambridge (1985)

Summary: These notes lumps together some preliminaries regarding probability theory and some advanced topics related to the role and use of randomness in computation. Needless to say, each of these appears in a separate section.

The probabilistic preliminaries include our conventions regarding random variables, which are used throughout the book. Also included are overviews of three useful inequalities: Markov Inequality, Chebyshev's Inequality, and Chernoff Bound.

The advanced topics include hashing, sampling and randomness extraction. For hashing, we describe constructions of pairwise (and t-wise independent) hashing functions, and variants of the Leftover Hashing Lemma (which are used a couple of times in the main text). We then review the "complexity of sampling": that is, the number of samples and the randomness complexity involved in estimating the average value of an arbitrary function defined over a huge domain. Finally, we provide an overview on the question of extracting almost perfect randomness from sources of weak (or defected) randomness.

Contents

1 Probabilistic preliminaries

Probability plays a central role in complexity theory. We assume that the reader is familiar with the basic notions of probability theory. In this section, we merely present the probabilistic notations that are used throughout the book, and three useful probabilistic inequalities.

1.1 Notational Conventions

Throughout the entire course we will refer only to *discrete* probability distributions. Specifically, the underlying probability space will consist of the set of all strings of a certain length ℓ, taken with uniform probability distribution. That is, the sample space is the set of all ℓ-bit long strings, and each such string is assigned probability measure $2^{-\ell}$. Traditionally, *random variables* are defined as functions from the sample space to the reals. Abusing the traditional terminology, we use the term random variable also when referring to functions mapping the sample space into the set of binary strings. We often do not specify the probability space, but rather talk directly about random variables. For example, we may say that X is a random variable assigned values in the set of all strings such that $\mathsf{Pr}[X=00] = \frac{1}{4}$ and $\mathsf{Pr}[X=111] = \frac{3}{4}$. (Such a random variable may be defined over the sample space $\{0,1\}^2$, so that $X(11) = 00$ and $X(00) = X(01) = X(10) = 111$.) One important case of a random variable is the output of a randomized process (e.g., a probabilistic polynomial-time algorithm).

All our probabilistic statements refer to functions of random variables that are defined beforehand. Typically, we may write $\mathsf{Pr}[f(X) = 1]$, where X is a random variable defined beforehand (and f is a function). An important convention is that *all occurrences of the same symbol in a probabilistic statement refer to the same* (unique) *random variable.* Hence, if $B(\cdot,\cdot)$ is a Boolean expression depending on two variables, and X is a random variable then $\mathsf{Pr}[B(X,X)]$ denotes the probability that $B(x,x)$ holds when x is chosen with probability $\mathsf{Pr}[X=x]$. For example, for every random variable X, we have $\mathsf{Pr}[X=X] = 1$. We stress that if we wish to discuss the probability that $B(x,y)$ holds when x and y are chosen independently with identical probability distribution, then we will define *two* independent random variables each with the same probability distribution. Hence, if X and Y are two independent random variables then $\mathsf{Pr}[B(X,Y)]$ denotes the probability that $B(x,y)$ holds when the pair (x,y) is chosen with probability $\mathsf{Pr}[X=x]\cdot\mathsf{Pr}[Y=y]$. For example, for every two independent random variables, X and Y, we have $\mathsf{Pr}[X=Y] = 1$ only if both X and Y are trivial (i.e., assign the entire probability mass to a single string).

Throughout the entire book, U_n denotes a random variable uniformly distributed over the set of strings of length n. Namely, $\mathsf{Pr}[U_n=\alpha]$ equals 2^{-n} if $\alpha \in \{0,1\}^n$ and equals 0 otherwise. We will often refer to the distribution of U_n as the uniform distribution (neglecting to qualify that it is uniform over $\{0,1\}^n$). In addition, we will occasionally use random variables (arbitrarily) distributed over

$\{0,1\}^n$ or $\{0,1\}^{\ell(n)}$, for some function $\ell:\mathbb{N}\to\mathbb{N}$. Such random variables are typically denoted by X_n, Y_n, Z_n, etc. We stress that in some cases X_n is distributed over $\{0,1\}^n$, whereas in other cases it is distributed over $\{0,1\}^{\ell(n)}$, for some function $\ell(\cdot)$, which is typically a polynomial. We will often talk about probability ensembles, which are infinite sequence of random variables $\{X_n\}_{n\in\mathbb{N}}$ such that each X_n ranges over strings of length bounded by a polynomial in n.

Statistical difference. The statistical distance (a.k.a variation distance) between the random variables X and Y is defined as

$$\frac{1}{2}\cdot\sum_v |\mathsf{Pr}[X=v]-\mathsf{Pr}[Y=v]| \;=\; \max_S\{\mathsf{Pr}[X\in S]-\mathsf{Pr}[Y\in S]\}.$$

We say that X is δ-close (resp., δ-far) to Y if the statistical distance between them is at least (resp., at most) δ.

1.2 Three Inequalities

The following probabilistic inequalities are very useful. All inequalities refer to random variables that are assigned real values and provide upper-bounds on the probability that the random variable deviates from its expectation.

Markov Inequality. The most basic inequality is Markov Inequality that applies to any random variable with bounded maximum or minimum value. For simplicity, it is stated for random variables that are lower-bounded by zero, and reads: *Let X be a non-negative random variable and v a real number. Then*

$$\mathsf{Pr}\left[X\geq v\right]\leq\frac{\mathsf{E}(X)}{v} \tag{1}$$

Equivalently, $\mathsf{Pr}[X\geq r\cdot\mathsf{E}(X)]\leq\frac{1}{r}$. The proof amounts to the following sequence.

$$\begin{aligned}
\mathsf{E}(X) &= \sum_x \mathsf{Pr}[X=x]\cdot x\\
&\geq \sum_{x<v}\mathsf{Pr}[X=x]\cdot 0+\sum_{x\geq v}\mathsf{Pr}[X=x]\cdot v\\
&= \mathsf{Pr}[X\geq v]\cdot v
\end{aligned}$$

Chebyshev's Inequality: Using Markov's inequality, one gets a potentially stronger bound on the deviation of a random variable from its expectation. This bound, called Chebyshev's inequality, is useful when having additional information concerning the random variable (specifically a good upper bound on its variance). For a random variable X of finite expectation, we denote by $\mathsf{Var}(X)\stackrel{\text{def}}{=}\mathsf{E}[(X-\mathsf{E}(X))^2]$ the variance of X, and observe that $\mathsf{Var}(X)=\mathsf{E}(X^2)-\mathsf{E}(X)^2$. Chebyshev's inequality then reads as follows: *Let X be a random variable, and $\delta>0$. Then*

$$\mathsf{Pr}\left[|X-\mathsf{E}(X)|\geq\delta\right]\;\leq\;\frac{\mathsf{Var}(X)}{\delta^2} \tag{2}$$

Proof: We define a random variable $Y\stackrel{\text{def}}{=}(X-\mathsf{E}(X))^2$, and apply Markov inequality. We get

$$\begin{aligned}
\mathsf{Pr}\left[|X-\mathsf{E}(X)|\geq\delta\right] &= \mathsf{Pr}\left[(X-\mathsf{E}(X))^2\geq\delta^2\right]\\
&\leq \frac{\mathsf{E}[(X-\mathsf{E}(X))^2]}{\delta^2}
\end{aligned}$$

and the claim follows. $\blacksquare$

Corollary (Pairwise Independent Sampling): Chebyshev's inequality is particularly useful in the analysis of the error probability of approximation via repeated sampling. It suffices to assume that the samples are picked in a pairwise independent manner, where $X_1, X_2, ..., X_n$ are pairwise independent if for every $i \neq j$ and every α, β it holds that $\mathsf{Pr}[X_i = \alpha \wedge X_j = \beta] = \mathsf{Pr}[X_i = \alpha] \cdot \mathsf{Pr}[X_j = \beta]$. The corollary reads: *Let $X_1, X_2, ..., X_n$ be pairwise independent random variables with identical expectation, denoted μ, and identical variance, denoted σ^2. Then, for every $\varepsilon > 0$,*

$$\mathsf{Pr}\left[\left|\frac{\sum_{i=1}^n X_i}{n} - \mu\right| \geq \varepsilon\right] \leq \frac{\sigma^2}{\varepsilon^2 n} \tag{3}$$

Proof: Define the random variables $\overline{X}_i \stackrel{\text{def}}{=} X_i - \mathsf{E}(X_i)$. Note that the $\overline{X}_i$'s are pairwise independent, and each has zero expectation. Applying Chebyshev's inequality to the random variable $\sum_{i=1}^n \frac{X_i}{n}$, and using the linearity of the expectation operator, we get

$$\begin{aligned} \mathsf{Pr}\left[\left|\sum_{i=1}^n \frac{X_i}{n} - \mu\right| \geq \varepsilon\right] &\leq \frac{\mathsf{Var}\left[\sum_{i=1}^n \frac{X_i}{n}\right]}{\varepsilon^2} \\ &= \frac{\mathsf{E}\left[\left(\sum_{i=1}^n \overline{X}_i\right)^2\right]}{\varepsilon^2 \cdot n^2} \end{aligned}$$

Now (again using the linearity of expectation)

$$\mathsf{E}\left[\left(\sum_{i=1}^n \overline{X}_i\right)^2\right] = \sum_{i=1}^n \mathsf{E}\left[\overline{X}_i^2\right] + \sum_{1 \leq i \neq j \leq n} \mathsf{E}\left[\overline{X}_i \overline{X}_j\right]$$

By the pairwise independence of the $\overline{X}_i$'s, we get $\mathsf{E}[\overline{X}_i \overline{X}_j] = \mathsf{E}[\overline{X}_i] \cdot \mathsf{E}[\overline{X}_j]$, and using $\mathsf{E}[\overline{X}_i] = 0$, we get

$$\mathsf{E}\left[\left(\sum_{i=1}^n \overline{X}_i\right)^2\right] = n \cdot \sigma^2$$

The corollary follows. $\blacksquare$

Chernoff Bound: Using pairwise independent sampling, the error probability in the approximation is decreasing linearly with the number of sample points (see Eq. (3)). Using totally independent sampling points, the error probability in the approximation can be shown to decrease exponentially with the number of sample points. (The random variables $X_1, X_2, ..., X_n$ are said to be totally independent if for every sequence $a_1, a_2, ..., a_n$ it holds that $\mathsf{Pr}[\wedge_{i=1}^n X_i = a_i] = \prod_{i=1}^n \mathsf{Pr}[X_i = a_i]$.) Probability bounds, supporting the above statement, are given next. The first bound, commonly referred to as Chernoff Bound, concerns 0-1 random variables (i.e., random variables that are assigned as values either 0 or 1), and asserts the following. *Let $p \leq \frac{1}{2}$, and $X_1, X_2, ..., X_n$ be independent 0-1 random variables so that $\mathsf{Pr}[X_i = 1] = p$, for each i. Then for all ε, $0 < \varepsilon \leq p(1-p)$, we have*

$$\mathsf{Pr}\left[\left|\frac{\sum_{i=1}^n X_i}{n} - p\right| > \varepsilon\right] < 2 \cdot e^{-\frac{\varepsilon^2}{2p(1-p)} \cdot n} \tag{4}$$

Proof Sketch: We upper-bound $\Pr[\sum_{i=1}^{n} X_i - pn > \varepsilon n]$ and $\Pr[pn - \sum_{i=1}^{n} X_i > \varepsilon n]$ is bounded similarly. Letting $\overline{X}_i \stackrel{\rm def}{=} X_i - \mathsf{E}(X_i)$, we apply Markov Inequality to the random variable $e^{\lambda \sum_{i=1}^{n} \overline{X}_i}$, where $\lambda > 0$ is determined to optimize the expressions that we derive (hint: $\lambda = \theta(\varepsilon/p(1-p))$ will do). Thus, $\Pr[\sum_{i=1}^{n} \overline{X}_i > \varepsilon n]$ is upper-bounded by

$$\frac{\mathsf{E}[e^{\lambda \sum_{i=1}^{n} \overline{X}_i}]}{e^{\lambda \varepsilon n}} = e^{-\lambda \varepsilon n} \cdot \prod_{i=1}^{n} \mathsf{E}[e^{\lambda \overline{X}_i}]$$

where the equality is due to the independence of the random variables. To simplify the rest of the proof, we establish a sub-optimal bound as follows. Using a Taylor expansion of e^x (e.g., $e^x < 1+x+2x^2$) and observing that $\mathsf{E}[\overline{X}_i] = 0$, we get $\mathsf{E}[e^{\lambda \overline{X}_i}] < 1+2\lambda^2 \mathsf{E}[\overline{X}_i^2]$, which equals $1+2\lambda^2 p(1-p)$. Thus, $\Pr[\sum_{i=1}^{n} X_i - pn > \varepsilon n]$ is upper-bounded by $(1+2\lambda^2 p(1-p))^n < \exp(-\lambda \varepsilon n + 2\lambda^2 p(1-p)n)$, which is optimized at $\lambda = \varepsilon/(4p(1-p))$ yielding $\exp(-\frac{\varepsilon^2}{8p(1-p)} \cdot n)$. Needless to say, this method can be applied in more general settings (e.g., for $X_i \in [0,1]$ rather than $X_i \in \{0,1\}$). $\square$

A more general bound, useful in the approximations of the expectation of a general random variable (not necessarily 0-1), is given next (and is commonly referred to as Hoefding Inequality).[1] *Let $X_1, X_2, ..., X_n$ be n independent random variables with identical probability distribution, each ranging over the (real) interval $[a,b]$, and let μ denote the expected value of each of these variables. Then, for every $\varepsilon > 0$,*

$$\Pr\left[\left|\frac{\sum_{i=1}^{n} X_i}{n} - \mu\right| > \varepsilon\right] < 2 \cdot e^{-\frac{2\varepsilon^2}{(b-a)^2} \cdot n} \tag{5}$$

Hoefding Inequality is useful in estimating the average value of a function defined over a large set of values, especially when the desired error probability needs to be negligible (i.e., decrease faster than any polynomial in the relevant parameter). Such an estimate can be obtained provided that we can efficiently sample the set and have a bound on the possible values (of the function).

Pairwise independent vs totally independent sampling. Consider, for simplicity the case that $a = 0 < \mu < b = 1$. In this case, n independent samples give an approximation that deviates by ε from the expectation with probability, denoted δ, that is exponentially decreasing with $\varepsilon^2 n$. Such an approximation is called an (ε, δ)-approximation, and can be achieved using $n = O(\varepsilon^{-2} \cdot \log(1/\delta))$ sample points. Thus, the number of sample points is polynomially related to ε^{-1} and logarithmically related to δ^{-1}. It follows that, when using poly(n) many samples,[2] the error probability (i.e., δ) can be made negligible (as a function in n), but the accuracy of the estimation (i.e., ε) can be only bounded above by any fixed polynomial fraction (and cannot be made negligible). We stress that the dependency of the number of samples on ε is not better than in the case of pairwise independent sampling; the advantage of totally independent samples is only in the dependency of the number of samples on δ.

2 Hashing

Hashing is extensively used in complexity theory in order to map arbitrary (unstructured) sets "almost uniformly" to a smaller structured set of adequate size. Specifically, hashing is supposed to map an arbitrary 2^m-subset (of $\{0,1\}^n$) to $\{0,1\}^m$ in an "almost uniform" manner.

[1] A more general form requires the X_i's to be independent, but not necessarily identical, and uses $\mu \stackrel{\rm def}{=} \frac{1}{n}\sum_{i=1}^{n} \mathsf{E}(X_i)$. See [1, Apdx. A].

[2] Here and in the rest of the book, we denote by poly() some fixed but unspecified polynomial.

For a fixed set S of cardinality 2^m, a 1-1 mapping $f_S : S \to \{0,1\}^m$ does exist, but it is not necessarily an efficient one (e.g., it may require "knowing" the entire set S). Clearly, no fixed function $f : \{0,1\}^n \to \{0,1\}^m$ can map every 2^m subset of $\{0,1\}^n$ to $\{0,1\}^m$ in a 1-1 manner (or even approximately so). However, a random function $f : \{0,1\}^n \to \{0,1\}^m$ has the property that, for every 2^m-subset $S \subset \{0,1\}^n$, with overwhelmingly high probability f maps S to $\{0,1\}^m$ such that no point in the range has many f-preimages in S. The problem is that a truly random function is unlikely to have a succinct representation (let alone an efficient evaluation algorithm). We seek families of functions that have a similar property, but do have a succinct representation as well as an efficient evaluation algorithm.

2.1 Definitions

Motivated by the foregoing discussion, we consider families of functions $\{H_n^m\}_{m<n}$ Such that the following properties hold:

1. For every $S \subset \{0,1\}^n$, with high probability, a function h selected uniformly in H_n^m maps S to $\{0,1\}^m$ in an "almost uniform" manner. For example, for any $|S| = 2^m$ and each point y, with high probability over the choice of h, it holds that $|\{x \in S : h(x) = y\}| \leq \text{poly}(n)$.
2. The functions in H_n^m have succinct representation. For example, we may require that $H_n^m \equiv \{0,1\}^{\ell(n,m)}$, for some polynomial ℓ.
3. The functions in H_n^m can be efficiently evaluated. That is, there exists a polynomial-time algorithm that, on input a representation of a function, h (in H_n^m), and a string $x \in \{0,1\}^n$, returns $h(x)$. In some cases we make even more stringent requirements regarding the the algorithm (e.g., that it runs in linear space).

Condition 1 was left vague on purpose. At the very least, we require that the expected size of $\{x \in S : h(x) = y\}$ equals $|S|/2^m$. We shall see (in Section 2.3) that different (stronger) interpretations of Condition 1 are satisfied by different types of hashing functions. We focus on t-wise independent hashing functions, defined next.

Definition 1 (t-wise independent hashing functions): *A family H_n^m of functions from n-bit strings to m-bit strings is called* t-wise independent *if for every t distinct domain elements $x_1, ..., x_t \in \{0,1\}^n$ and every $y_1, ..., y_t \in \{0,1\}^m$ it holds that*

$$\mathsf{Pr}_{h \in H_n^m}[\wedge_{i=1}^t h(x_i) = y_1] \; = \; 2^{-t \cdot m}$$

That is, every t domain elements are mapped by a uniformly chosen $h \in H_n^m$ in a totally uniform manner. Note that for $t \geq 2$, it follows that the probability that a random $h \in H_n^m$ maps two distinct domain elements to the same image is 2^{-m}. Such (families of) functions are called universal (cf. [2]), but we will focus on the stronger condition of t-wise independence.

2.2 Constructions

The following constructions are merely a re-interpretation of the constructions presented in the context of (special purpose) pseudorandom generators. (Alternatively, one may view the latter constructions as a re-interpretation of the following two constructions.)

Construction 2 (t-wise independent hashing): *For $t, m, n \in \mathbb{N}$ such that $m \leq n$, consider the following family of hashing functions mapping n-bit strings to m-bit strings. Each t-sequence $\overline{s} = (s_0, s_1, ..., s_{t-1}) \in \{0,1\}^{t \cdot n}$ describes a function $h_{\overline{s}} : \{0,1\}^n \to \{0,1\}^m$ such that $h_{\overline{s}}(x)$ equals the m-bit prefix of the binary representation of $\sum_{j=0}^{t-1} s_j x^j$, where the arithmetic is that of* GF(2^n), *the finite field of 2^n elements.*

Construction 2 constitutes a family of t-wise independent hash functions. Typically, we will use either $t = 2$ or $t = \Theta(n)$. To make the construction totally explicit, we need an explicit representation of GF(2^n). An alternative construction for the case of $t = 2$ may be obtained as follows. Recall that a Toeplitz matrix is a matrix with all diagonals being homogeneous; that is, $T = (t_{i,j})$ is a Toeplitz matrix if $t_{i,j} = t_{i+1,j+1}$, for all i, j.

Construction 3 (Alternative pairwise independent hashing): *For $m \leq n$, consider the family of hashing functions in which each n-by-m Toeplitz matrix T and an m-dimensional vector b describes a function $h_{T,b} : \{0,1\}^n \to \{0,1\}^m$ such that $h_{T,b}(x) = Tx + b$.*

Construction 3 constitutes a family of pairwise independent hash functions. Note that a n-by-m Toeplitz matrix can be specified by $n + m - 1$ bits, yielding description length $n + 2m - 1$. An alternative construction (using $m \cdot n + m$ bits of representation) uses arbitrary n-by-m matrices rather than Toeplitz matrices.

2.3 The Leftover Hash Lemma

We now turn to the "almost uniform" cover condition (i.e., Condition 1) mentioned in Section 2.1. One concrete interpretation of this condition is implied by the following lemma.

Lemma 4 *Let $m < n$ be integers, H_n^m be a family of pairwise independent hash functions, and $S \subseteq \{0,1\}^n$. Then, for every $y \in \{0,1\}^m$ and every $\varepsilon > 0$, for all but at most an $\frac{2^m}{\varepsilon^2 |S|}$ fraction of $h \in H_n^m$ it holds that*

$$|\{x \in S : h(x) = y\}| = (1 \pm \varepsilon) \cdot \frac{|S|}{2^m}. \tag{6}$$

By pairwise independence (or rather even by "1-wise independence"), the expected size of $\{x \in S : h(x) = y\}$ is $|S|/2^m$, where the expectation is taken uniformly over all $h \in H_n^m$. The lemma upper bounds the fraction of h's that deviate from the expected value. Needless to say, the bound is meaningful only in case $|S| > 2^m$ (or alternatively for $\varepsilon > 1$). Setting $\varepsilon = \sqrt[3]{2^m/|S|}$ (and focusing on the case that $|S| > 2^m$), we infer that *for all but at most an ε fraction of $h \in H_n^m$ it holds that* $|\{x \in S : h(x) = y\}| = (1 \pm \varepsilon) \cdot |S|/2^m$. Thus, each range element has approximately the right number of h-preimages in the set S under almost all $h \in H_n^m$.

Proof: Fixing an arbitrary set $S \subseteq \{0,1\}^n$ and an arbitrary $y \in \{0,1\}^m$, we estimate the probability that a uniformly selected $h \in H_n^m$ violates Eq. (6). We define random variables ζ_x, over the aforementioned probability space, such that $\zeta_x = \zeta_x(h)$ equal 1 if $h(x) = y$ and 0 otherwise. The expected value of $\sum_{x \in S} \zeta_x$ is $\mu \stackrel{\text{def}}{=} |S| \cdot 2^{-m}$, and we are interested in the probability that this sum deviates from the expectation. Applying Chebyshev's Inequality, we get

$$\mathsf{Pr}\left[\left|\mu - \sum_{x \in S} \zeta_x\right| > \varepsilon \cdot \mu\right] \; < \; \frac{\mu}{\varepsilon^2 \mu^2}$$

because $\mathsf{Var}(\sum_{x \in S} \zeta_x) < |S| \cdot 2^{-m}$ by the pairwise independence of the ζ_x's and the fact that $\mathsf{E}[\zeta_x] = 2^{-m}$. The lemma follows. ■

A generalization (called mixing). The proof of Lemma 4 can be easily extended to show that *for every set* $T \subset \{0,1\}^m$ *and every* $\varepsilon > 0$, *for all but at most an* $\frac{2^m}{|T|\cdot|S|\varepsilon^2}$ *fraction of* $h \in H_n^m$ *it holds that* $|\{x \in S : h(x) = y\}| = (1 \pm \varepsilon) \cdot |T| \cdot |S|/2^m$. (Hint: just define $\zeta_x = \zeta(h) = 1$ if $h(x) \in T$ and 0 otherwise.) In the case that $m = n$, this is called a mixing property, and is meaningfull provided $|T| \cdot |S| > 2^m/\varepsilon$.

An extremely useful corollary. The aforementioned generalization of Lemma 4 asserts that most functions behave well with respect to any fixed sets of preimages $S \subset \{0,1\}^n$ and images $T \subset \{0,1\}^m$. A seemingly stronger statement, which is (non-trivially) implied by Lemma 4, is that for all adequate sets S most functions $h \in H_n^m$ map S to $\{0,1\}^m$ in an almost uniform manner.[3] This is a consequence of the following theorem.

Theorem 5 (a.k.a Leftover Hash Lemma): *Let* H_n^m *and* $S \subseteq \{0,1\}^n$ *be as in Lemma 4, and define* $\varepsilon = \sqrt[3]{2^m/|S|}$. *Consider random variable* X *and* H *that are uniformly distributed on* S *and* H_n^m, *respectively. Then, the statistical distance between* $(H, H(X))$ *and* (H, U_m) *is at most* 2ε.

Using the terminology of Section 4, we say that H_n^m yields a strong extractor (with parameters to be spelled out there).

Proof: Let V denote the set of pairs (h, y) that violate Eq. (6), and $\overline{V} \stackrel{\text{def}}{=} (H_n^m \times \{0,1\}^m) \setminus V$. Then for every $(h, y) \in \overline{V}$ it holds that

$$\begin{aligned}
\mathsf{Pr}[(H, H(X)) = (h, y)] &= \mathsf{Pr}[H = h] \cdot \mathsf{Pr}[h(X) = y] \\
&= (1 \pm \varepsilon) \cdot \mathsf{Pr}[(H, U_m) = (h, y)].
\end{aligned}$$

On the other hand, by Lemma 4 (which asserts $\mathsf{Pr}[(H, y) \in V] \leq \varepsilon$ for every $y \in \{0,1\}^m$), we have $\mathsf{Pr}[(H, U_m) \in V] \leq \varepsilon$. Using

$$\begin{aligned}
\mathsf{Pr}[(H, H(X)) \in V] &= 1 - \mathsf{Pr}[(H, H(X)) \in \overline{V}] \\
&\leq 1 - \mathsf{Pr}[(H, U_n)) \in \overline{V}] + \varepsilon \;\leq\; 2\varepsilon
\end{aligned}$$

we upper-bounded the statistical difference between $(H, H(X))$ and (H, U_m) by

$$\begin{aligned}
&\frac{1}{2} \cdot \sum_{(h,y) \in H_n^m \times \{0,1\}^m} |\mathsf{Pr}[(H, H(X)) = (h, y)] - \mathsf{Pr}[(H, U_m) = (h, y)]| \\
&\leq \frac{\varepsilon}{2} + \frac{1}{2} \cdot \sum_{(h,y) \in V} |\mathsf{Pr}[(H, H(X)) = (h, y)] - \mathsf{Pr}[(H, U_m) = (h, y)]| \\
&\leq \frac{\varepsilon}{2} + \frac{1}{2} \cdot \sum_{(h,y) \in V} (\mathsf{Pr}[(H, H(X)) = (h, y)] + \mathsf{Pr}[(H, U_m) = (h, y)]) \\
&\leq \frac{\varepsilon}{2} + \frac{1}{2} \cdot (2\varepsilon + \varepsilon)
\end{aligned}$$

and the claim follows. ■

[3] That is, for X as in Theorem 5 and any $\alpha > 0$, for all but at most an α fraction of the functions $h \in H_n^m$ it holds that $h(X)$ is $(2\varepsilon/\alpha)$-close to U_m.

An alternative proof of Theorem 5. Define the collision probability of a random variable Z, denote $\mathsf{cp}(Z)$, as the probability that two independent samples of Z yield the same result. Alternatively, $\mathsf{cp}(Z) \stackrel{\text{def}}{=} \sum_z \Pr[Z = z]^2$. Theorem 5 follows by combining the following two facts:

1. A general fact: *If $Z \in [N]$ and $\mathsf{cp}(Z) \leq (1+4\epsilon^2)/N$ then Z is ϵ-close to the uniform distribution on $[N]$.*

   We prove the contra-positive: Assuming that the statistical distance between Z and the uniform distribution on $[N]$ equals δ, we show that $\mathsf{cp}(Z) \geq (1 + 4\delta^2)/N$. This is done by defining $L \stackrel{\text{def}}{=} \{z : \Pr[Z = z] < 1/N\}$, and lower-bounding $\mathsf{cp}(Z)$ by using the fact that the collision probability minimizes on uniform distributions. Specifically,

   $$\mathsf{cp}(Z) \geq |L| \cdot \left(\frac{\Pr[Z \in L]}{|L|}\right)^2 + (N - |L|) \cdot \left(\frac{\Pr[Z \in [N] \setminus L]}{N - |L|}\right)^2,$$

   which equals $1 + (\delta^2/(1-\rho)\rho) \geq 1 + 4\delta^2$, where $\rho = |L|/N$.

2. The collision probability of $(H, H(X))$ is at most $(1 + (2^m/|S|))/(|H_n^m| \cdot 2^m)$. (Furthermore, this holds even if H_n^m is only universal.)

   The proof is by a straightforward calculation. Specifically, note that $\mathsf{cp}(H, H(X)) = |H_n^m|^{-1} \cdot \mathsf{E}_{h \in H_n^m}[\mathsf{cp}(h(X))]$, whereas $\mathsf{E}_{h \in H_n^m}[\mathsf{cp}(h(X)] = |S|^{-2} \sum_{x_1, x_2 \in S} \Pr[H(x_1) = H(x_2)]$. The sum equals $|S| + (|S|^2 - |S|) \cdot 2^{-m}$, and so $\mathsf{cp}(H, H(X)) < |H_n^m|^{-1} \cdot (2^{-m} + |S|^{-1})$.

Note that it follows that $(H, H(X))$ is $\sqrt{2^m/4|S|}$-close to (H, U_m), which is a stronger bound than the one provided in Theorem 5.

Stronger uniformity via higher independence. Recall that Lemma 4 asserts that for each point in the range of the hash function, with high probability over the choice of the hash function, *this fixed point* has approximately the expected number of preimages in S. A stronger condition asserts that, with high probability over the choice of the hash function, *every point in its range* has approximately the expected number of preimages in S. Such a guarantee can be obtained when using n-wise independent hashing functions.

Lemma 6 *Let $m < n$ be integers, H_n^m be a family of n-wise independent hash functions, and $S \subseteq \{0,1\}^n$. Then, for every $\varepsilon \in (0,1)$, for all but at most an $2^m \cdot (n \cdot 2^m/\varepsilon^2|S|)^{n/2}$ fraction of $h \in H_n^m$, it holds that $|\{x \in S : h(x) = y\}| = (1 \pm \varepsilon) \cdot |S|/2^m$ for every $y \in \{0,1\}^m$.*

Indeed, the lemma should be used with $2^m < \varepsilon^2|S|/4n$. In particular, using $m = \log_2|S| - \log_2(5n/\varepsilon^2)$ guarantees that with high probability each range elements has $(1\pm\varepsilon)\cdot|S|/2^m$ preimages in S. Under this setting of parameters $|S|/2^m = 5n/\varepsilon^2$, which is poly$(n)$ whenever $\varepsilon = 1/\text{poly}(n)$. Needless to say, this guarantee is stronger than the conclusion of Theorem 5.

Proof: The proof follows the footsteps of the proof of Lemma 4, taking advantage of the fact that the random variables (i.e., the ζ_x's) are now $2t$-wise independent, where $t = n/2$. This allows for the use of a so-called $2t^{\text{th}}$ moment analysis, which generalizes the analysis of pairwise independent samplying (presented in Section 1.2). As in the proof of Lemma 4, we fix any S and y, and define $\zeta_x = \zeta_x(h) = 1$ if and only if $h(x) = y$. Letting $\mu = \mathsf{E}[\sum_{x \in S} \zeta_x] = |S|/2^m$ and $\overline{\zeta}_x = \zeta_x - \mathsf{E}(\zeta_x)$, we start with Markov inequality:

$$\begin{aligned}\mathsf{Pr}\left[\left|\mu - \sum_{x\in S}\zeta_x\right| > \varepsilon\cdot\mu\right] &< \frac{\mathsf{E}[(\sum_{x\in S}\overline{\zeta}_x)^{2t}]}{\varepsilon^{2t}\mu^{2t}} \\ &= \frac{\sum_{x_1,\ldots,x_{2t}\in S}\mathsf{E}[\prod_{i=1}^{2t}\overline{\zeta}_{x_i}]}{\varepsilon^{2t}\cdot(|S|/2^m)^{2t}} \qquad (7)\end{aligned}$$

Using $2t$-wise independence, we note that only the terms in Eq. (7) that do not vanish are those in which each variable appears with multiplicity. This mean that only terms having less than t distinct variables contribute to Eq. (7). Now, for every $j \leq t$, we have less than $\binom{|S|}{j}\cdot(2t!) < (2t!/j!)\cdot|S|^j$ terms with j distinct variables, and each contributes less than $(2^{-m})^j$ to the sum. Thus, Eq. (7) is upper-bounded by

$$\frac{2t!}{(\varepsilon^{2t}|S|/2^m)^{2t}}\cdot\sum_{j=1}^{t}\frac{(|S|/2^m)^j}{j!} < \frac{2t!/t!}{(\varepsilon^2|S|/2^m)^t} < \left(\frac{2t\cdot 2^m}{\varepsilon^2|S|}\right)^t$$

where the first inequality assumes $|S| > n2^m$ (since the claim hold vacuously otherwise). This upper-bounds the probability that a random $h \in H_n^m$ violates the mapping condition regarding a fixed y. Using a union bound on all $y \in \{0,1\}^m$, the lemma follows. ■

3 Sampling

In many settings repeated sampling is used to estimate the average of a huge set of values. Namely, a "value" function $\nu : \{0,1\}^n \to \mathbb{R}$ is defined over a huge domain, and one wishes to approximate $\bar{\nu} \stackrel{\text{def}}{=} \frac{1}{2^n}\sum_{x\in\{0,1\}^n}\nu(x)$ without having to inspect the value of ν at each point in the domain. The obvious thing to do is to sample the domain at random, and obtain an approximation to $\bar{\nu}$ by taking the average of the values of ν on the sample points. It turns out that certain "pseudorandom" sequences of sample points may serve almost as well as truly random sequences of sample points.

Formal Setting. It is essential to have the range of ν be bounded (or else no reasonable approximation is possible). For simplicity, we adopt the convention of having $[0,1]$ be the range of ν, and the problem for other (predetermined) ranges can be treated analogously. Our notion of approximation depends on two parameters: *accuracy* (denoted ε) and *error probability* (denoted δ). We wish to have an algorithm that, with probability at least $1-\delta$, gets within ε of the correct value. This leads to the following definition.

Definition 7 (sampler): *A* sampler *is a randomized algorithm that on input parameters n* (length), ε (accuracy) *and* δ (error), *and oracle access to* any *function $\nu : \{0,1\}^n \to [0,1]$, outputs, with probability at least $1-\delta$, a value that is at most ε away from $\bar{\nu} \stackrel{\text{def}}{=} \frac{1}{2^n}\sum_{x\in\{0,1\}^n}\nu(x)$. Namely,*

$$\mathsf{Pr}[|\text{sampler}^{\nu}(n,\varepsilon,\delta) - \bar{\nu}| > \varepsilon] < \delta$$

where the probability is taken over the internal coin tosses of the sampler, also called its random seed.

A non-adaptive sampler *is a sampler that consists of two deterministic algorithms: a* sample generating *algorithm, G, and a* evaluation algorithm, *V. On input n,ε,δ and a random* seed, *algorithm G generates a sequence of queries, denoted $s_1,\ldots,s_m \in \{0,1\}^n$. Algorithm V is given the corresponding ν-values (i.e., $\nu(s_1),\ldots,\nu(s_m)$) and outputs an estimate to $\bar{\nu}$.*

We are interested in "the complexity of sampling" quantified as a function of the parameters n, ε and δ. Specifically, we will consider three complexity measures: The sample complexity (i.e., the number of oracle queries made by the sampler); the randomness complexity (i.e., the length of the random seed used by the sampler); and the computational complexity (i.e., the running-time of the sampler). We say that a sampler is efficient if its running-time is polynomial in the total length of its queries (i.e., polynomial in both its sample complexity and in n). We will focus on efficient samplers. Furthermore, we will focus on efficient samplers that have optimal (up-to a constant factor) sample complexity, and will wish the randomness complexity to be as low as possible.

We note that all positive results to be reviewed refer to non-adaptive samplers, whereas the lower bound hold also for general samplers. For more details see [5, Sec. 3.6.4].

The naive sampler. The straightforward method (or the naive sampler) consists of *uniformly and independently* selecting sufficiently many sample points (queries), and outputting the average value of the function on these points. Using Chernoff Bound it follows that $O(\frac{\log(1/\delta)}{\varepsilon^2})$ sample points suffice. The naive sampler is optimal (up-to a constant factor) in its sample complexity, but is quite wasteful in randomness.

It is known that $\Omega(\frac{\log(1/\delta)}{\varepsilon^2})$ samples are needed in any sampler, and that that samplers that make $s(n, \varepsilon, \delta)$ queries require randomness at least $n+\log_2(1/\delta)-\log_2 s(n, \varepsilon, \delta)-O(1)$. These lower bounds are tight (as demonstrated by non-explicit and inefficient samplers). These facts guide our quest for improvements, which is aimed at finding more randomness-efficient ways of *efficiently* generating sample sequences that can be used in conjunction with an appropriate evaluation algorithm V. (We stress that V need not necessarily take the average of the values of the sampled points.)

The pairwise-independent sampler. Using a pairwise-independence generator for generating sample points, along with the natural evaluation algorithm (which outputs the average of the values of these points), we obtain a great saving in the randomness complexity: pairwise-independent sampling uses $2n$ random bits rather than the $\Omega((\log(1/\delta))\varepsilon^{-2} \cdot n)$ coins used by the naive sampler. Using Eq. (3) it follows that $O(1/\delta\varepsilon^2)$ samples are sufficient to get accuracy ε with error δ. Thus, for constant $\delta > 0$, the Pairwise-Independent Sampler is optimal up-to a constant factor in both its sample and randomness complexities. However, for small δ (i.e., $\delta = o(1)$), this sampler is wasteful in sample complexity.

The Median-of-Averages sampler. A new idea is required for going further, and a relevant tool – random walks on expander graphs – is needed too. Specifically, we combine the Pairwise-Independent Sampler with the Expander Random Walk Generator to obtain a new sampler. The new sampler uses a random walk on an expander with vertex set $\{0,1\}^{2n}$ to generate a sequence of $t \stackrel{\text{def}}{=} O(\log(1/\delta))$ related seeds for t invocations of the Pairwise-Independent Sampler, where each of these invocations uses the corresponding $2n$ bits to generate a sequence of $O(1/\varepsilon^2)$ samples in $\{0,1\}^n$. Furthermore, each of these invocations returns a value that, with probability at least 0.9, is ε-close to $\bar{\nu}$. The Expander Random Walk Theorem is used to show that, with probability at least $1-\exp(-t) = 1-\delta$, most of these t invocations return an ε-close approximation. Hence, the median among these t values is an (ε, δ)-approximation to the correct value. The resulting sampler, called the Median-of-Averages Sampler, has sample complexity $O(\frac{\log(1/\delta)}{\varepsilon^2})$ and randomness complexity $2n + O(\log(1/\delta))$, which is optimal up-to a constant factor in both complexities.

Further improvements. The randomness complexity of the Median-of-Averages Sampler can be improved from $2n + O(\log(1/\delta))$ to $n + O(\log(1/\delta\varepsilon))$, while maintaining its (optimal) sample complexity (of $O(\frac{\log(1/\delta)}{\varepsilon^2})$). This is done by replacing the Pairwise Independent Sampler by a sampler that picks a random vertex in a suitable expander and samples all its neighbors.

Averaging Samplers. Averaging (a.k.a. Oblivious) samplers are non-adaptive samplers in which the evaluation algorithm is the natural one: that is, it merely outputs the average of the values of the sampled points. Indeed, the Pairwise-Independent Sampler is an averaging sampler, whereas the Median-of-Averages Sampler is not. Interestingly, averaging samplers have applications for which ordinary non-adaptive samplers do not suffice. Averaging samplers are closely related to randomness extractors, defined and discussed in Section 4.

An odd perspective. Recall that a non-adaptive sampler consists of a sample generator G and an evaluator V such that for every $\nu:\{0,1\}^n \to [0,1]$ it holds that

$$\mathsf{Pr}_{(s_1,...,s_m)\leftarrow G(U_k)}[|V(\nu(s_1),...,\nu(s_m)) - \bar{\nu}| > \varepsilon] \ < \ \delta.$$

Thus, we may view G as a pseudorandom generator that is subjected to a distinguishability text that is determined by a fixed algorithm V and an arbitrary function $\nu:\{0,1\}^n \to [0,1]$, where we assume that $\mathsf{Pr}[|V(\nu(U_n^{(1)}),...,\nu(U_n^{(m)})) - \bar{\nu}| > \varepsilon] \ < \ \delta$. What is a bit odd here is that, except for the case of averaging samplers, the distinguishability test contains a central component (i.e., the evaluator V) that is potentially custom-made to help the generator G pass the test.[4]

4 Randomness Extractors

Extracting almost-perfect randomness from sources of weak (i.e., defected) randomness is crucial for the actual use of randomized algorithms, procedures and protocols. The latter are analyzed assuming that they are given access to a perfect random source, while in reality one typically has access only to sources of weak randomness. Randomness extractors are efficient procedures that (possibly with the help of little extra randomness) enhance the quality of random sources, converting any source of weak randomness to an almost perfect one. In addition, randomness extractors are related to several other fundamental problems, to be further discuss later.

One key parameter, which was avoided in the foregoing discussion, is the class of weak random sources from which we need to extract almost perfect randomness. It is preferable to make as little assumptions as possible regarding the weak random source. In other words, we wish to consider a wide class of such sources, and require the randomness extractor (often referred to as the extractor) to "work well" for any source in this class. A general class of such sources is defined in §4.1.1, but first we wish to mention that even for very restricted classes of sources no deterministic extractor can work. To overcome this impossibility result, two approach are used:

Seeded extractors: The first approach consists of considering randomized extractors that use a relatively small amount of randomness (in addition to the weak random source). That is, these extractors obtain two input: a short truly random seed and a relatively long sequence generated by an arbitrary source that belong to the specified class of sources. This suggestion is motivated in two different ways:

[4]Another aspect in which samplers differ from the various pseudorandom generators is in the aim to minimize, rather than maximize, the number of blocks (denoted here by m) in the output sequence. However, also in case of samplers the aim is to maximize the block-length (denoted here by n).

1. The application may have access to an almost-perfect random source, but bits from this source are much more expensive than bits from the weak/defected random source. Thus, it makes sense to obtain few high-quality bits from the almost-perfect source and use them to "purify" the cheap bits obtained from the defected source.
2. In some applications (e.g., when using randomized algorithms), it may be possible to scan over all possible values of the seed and run the algorithm using the corresponding extracted randomness. That is, we obtain a sample r from the weak random source, and invoke the algorithm on $\texttt{extract}(s,r)$, for every possible seed s, ruling by majority. (This alternative is typically not applicable to distributed settings.)

Few independent sources: The second approach consists of considering deterministic extractors that obtain samples from a few (say two) *independent* sources of weak randomness. Such extractors are applicable in any setting (including in cryptography), provided that the application has access to the required number of independent weak random sources.

In this section we focus on the first type of extractors (i.e., the *seeded extractors*). This choice is motivated both by the relatively more mature state of the research in that direction and the closer connection between this direction and other topics in complexity.

4.1 Definitions and various perspectives

We first present a definition that corresponds to the foregoing motivational discussion, and later discuss its relation to other topics in complexity.

4.1.1 The Main Definition

A very wide class of weak random sources corresponds to sources for which no specific output is too probable (cf. [3]). That is, the class is parameterized by a bound β and consists of all sources X such that for every x it holds that $\mathsf{Pr}[X=x]\leq\beta$. In such a case, we say that X has min-entropy[5] at least $\log_2(1/\beta)$. Indeed, we represent sources as random variables, and assume that they are distributed over strings of a fixed length, denoted n. A (n,k)-source is a source that is distributed over $\{0,1\}^n$ and has min-entropy at least k.

An interesting special case of (n,k)-sources is that of sources that are uniform over a subset of 2^k strings. Such sources are called (n,k)-flat. A simple but useful observation is that each (n,k)-source is a convex combination of (n,k)-flat sources.

Definition 8 (extractor for (n,k)-sources):

1. *An algorithm* $\mathrm{Ext}:\{0,1\}^d\times\{0,1\}^n\to\{0,1\}^m$ *is called an* extractor with error ε for the class $\mathcal{C}$ *if for every source* X *in* $\mathcal{C}$ *it holds that* $\mathrm{Ext}(U_d,X)$ *is* ε*-close to* U_m.

   An algorithm Ext *is called a* (k,ε)-extractor *if it is an extractor with error* ε *for the class of* (n,k)*-sources.*

2. *An algorithm* Ext *is called a* strong extractor with error ε for $\mathcal{C}$ *if for every source* X *in* $\mathcal{C}$ *it holds that* $(U_d,\mathrm{Ext}(U_d,X))$ *is* ε*-close to* (U_d,U_m)*. A* strong (k,ε)-extractor *is defined analogously.*

[5] Recall that the entropy of a random variable X is defined as $\sum_x \mathsf{Pr}[X=x]\log_2(1/\mathsf{Pr}[X=x])$. Indeed the min-entropy of X equals $\min_x\{\log_2(1/\mathsf{Pr}[X=x])\}$, and is always upper-bounded by its entropy.

Using the "decomposition" of (n,k)-sources to (n,k)-flat sources, it follows that Ext *is a* (k,ε)-*extractor if and only if it is an extractor with error* ε *for the class of* (n,k)-*flat sources.* (A similar claim holds for strong extractors.) Thus, much of the technical analysis is conducted with respect to the class of (n,k)-flat sources. For example, it is easy to see that, for $d = \log_2(n/\varepsilon^2) + O(1)$, there exists a (k,ε)-extractor $\text{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^k$. (The proof is by the Probabilistic Method and uses a union bound on the set of all (n,k)-flat sources.)[6]

We seek, however, explicit extractors; that is, extractors that are implementable by polynomial-time algorithms. We note that the evaluation algorithm of any family of pairwise independent hash functions mapping n-bit strings to m-bit strings constitutes a (strong) (k,ε)-extractor for $\varepsilon = 2^{-(k-m)/2}$ (see the alternative proof of Theorem 5). However, these extractors necessarily use a long seed (i.e., $d \geq 2m$ must hold (and in fact $d = n + 2m - 1$ holds in Construction 3)). In Section 4.2 we survey constructions of efficient (k,ε)-extractors that obtain logarithmic seed length (i.e., $d = O(\log(n/\varepsilon))$). But before doing so, we provide a few alternative perspectives on extractors.

An important note on logarithmic seed length. The case of logarithmic seed length is of particular importance for a variety of reasons. Firstly, when emulating a randomized algorithm using a defected random source (as in Item 2 of the motivational discussion of seeded extractors), the overhead is exponential in the length of the seed. Thus, the emulation of a generic probabilistic polynomial-time algorithm can be done in polynomial time only if the seed length is logarithmic. Similarly, the applications discussed in §4.1.2 and §4.1.3 are feasible only if the seed length is logarithmic. Lastly, we note that logarithmic seed length is an absolute lower-bound for (k,ε)-extractors, whenever $n > k + k^{\Omega(1)}$ (and $m \geq 1$ and $\varepsilon < 1/2$).

4.1.2 Extractors as averaging samplers

There is a close relationship between extractors and averaging samplers (discussed towards the end of Section 3). We first show that any averaging sampler gives rise to an extractor. Let $G : \{0,1\}^n \to (\{0,1\}^m)^t$ be the sample generating algorithm of an averaging sampler having accuracy ε and error probability δ. That is, G uses n bits of randomness and generates t sample points in $\{0,1\}^m$ such that for every $f : \{0,1\}^m \to [0,1]$ with probability at least $1-\delta$ the average f-values of these points is in the interval $[\overline{f} \pm \varepsilon]$, where $\overline{f} \stackrel{\text{def}}{=} \mathsf{E}[f(U_m)]$. Define $\text{Ext} : [t] \times \{0,1\}^n \to \{0,1\}^m$ such that $\text{Ext}(i,r)$ is the i^{th} sample generated by $G(r)$. We shall prove that Ext is a $(k,2\varepsilon)$-extractor, for $k = n - \log_2(\varepsilon/\delta)$.

Suppose towards the contradiction that there exists a $(k,2\varepsilon)$-flat source X such that for some $S \subset \{0,1\}^m$ it is the case that $\mathsf{Pr}[\text{Ext}(U_d,X) \in S] > \mathsf{Pr}[U_m \in S] + 2\varepsilon$, where $d = \log_2 t$ and $[t] \equiv \{0,1\}^d$. Define

$$B = \{x \in \{0,1\}^n : \mathsf{Pr}[\text{Ext}(U_d,x) \in S] > (|S|/2^m) + \varepsilon\}.$$

Then, $|B| > \varepsilon \cdot 2^k = \delta \cdot 2^n$. Defining $f(z) = 1$ if $z \in S$ and $f(z) = 0$ otherwise, we have $\overline{f} \stackrel{\text{def}}{=} \mathsf{E}[f(U_m)] = |S|/2^m$. But, for every $r \in B$ the f-average of the sample $G(r)$ is greater than $\overline{f} + \varepsilon$, in contradiction to the hypothesis that the sampler has error probability δ (with respect to accuracy ε).

[6]The probability that a random function $\text{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^k$ is not an extractor with error ε for a fixed (n,k)-flat source is upper-bounded by $2^{2^k} \cdot \exp(-\Omega(2^{d+k}\varepsilon^2))$, which is smaller than $1/\binom{2^n}{2^k}$.

We now turn to show that extractors give rise to averaging samplers. Let $\mathrm{Ext}:\{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ be a (k,ε)-extractor. Consider the sample generation algorithm $G:\{0,1\}^n \to (\{0,1\}^m)^{2^d}$ define by $G(r) = (\mathrm{Ext}(s,r))_{s\in\{0,1\}^d}$. We prove that it corresponds to an averaging sampler with accuracy ε and error probability $\delta = 2^{-(n-k-1)}$.

Suppose towards the contradiction that there exists a function $f:\{0,1\}^m \to [0,1]$ such that for $\delta 2^n = 2^{k+1}$ strings $r \in \{0,1\}^n$ the average f-value of the sample $G(r)$ deviates from $\overline{f} \stackrel{\mathrm{def}}{=} \mathsf{E}[f(U_m)]$ by more than ε. Suppose, without loss of generality, that for at least half of these r's the average is greater than $\overline{f}+\varepsilon$, and let B denote the set of these r's. Then, for X that is uniformly distributed on B and is thus a (n,k)-source, we have

$$\mathsf{E}[f(\mathrm{Ext}(U_d,X))] > \mathsf{E}[f(U_m)] + \varepsilon,$$

which (using $|f(z)| \leq 1$ for every z) contradicts the hypothesis that $\mathrm{Ext}(U_d,X)$ is ε-close to U_m.

4.1.3 Extractors as randomness-efficient error-reductions

As may be clear from the foregoing discussion, extractors yield randomness-efficient methods for error-reduction. Indeed, error-reduction is a special case of the sampling problem, obtained by considering Boolean functions. Specifically, for a two-sided error decision procedure A, consider the function $f_x:\{0,1\}^{\ell(|x|)} \to \{0,1\}$ such that $f_x(r)=1$ if $A(x,r)=1$ and $f_x(r)=0$ otherwise. Assuming that the probability that A is correct is at least $0.5+\varepsilon$ (say $\varepsilon = 1/6$), error reduction amounts to providing a sampler with accuracy ε and any desired error probability δ for the Boolean function f_x. In particular, any (k,ε)-extractor $\mathrm{Ext}:\{0,1\}^d \times \{0,1\}^n \to \{0,1\}^{\ell(|x|)}$ with $k = n - \log(1/\delta) - 1$ will do, provided 2^d is feasible (e.g., $2^d = \mathrm{poly}(\ell(|x|))$, where $\ell(\cdot)$ represents the randomness complexity of the original algorithm A). The question of interest here is how does n (which represents the randomness complexity of the corresponding sampler) grow as a function of $\ell(|x|)$ and δ.

Error-reduction using the extractor $\mathrm{Ext}:[\mathrm{poly}(\ell(|x|))] \times \{0,1\}^n \to \{0,1\}^{\ell(|x|)}$

	error probability	randomness complexity
original algorithm	$1/3$	$\ell(\lvert x\rvert)$
resulting algorithm	δ (may depend on $\lvert x\rvert$)	n (function of $\ell(\lvert x\rvert)$ and δ)

Jumping ahead, we note that for every $\alpha > 1$, one can obtain $n = O(\ell(|x|)) + \alpha\log_2(1/\delta)$ (cf. [10, 11]).[7] Note that this bound on the randomness-complexity of error-reduction is better than the one provided (for the reduction of one-sided error) by the Expander Random Walk Generator, albeit the number of samples here is larger (but still polynomial in n).

Mentioning the reduction of *one-sided* error probability, brings us to a corresponding relaxation of the notion of an extractor, which is called a disperser. Loosely speaking, a (k,ε)-disperser is only required to hit (with positive probability) any set of density greater than ε in its image, rather than produce a distribution that is ε-close to uniform.

Definition 9 (dispersers): *An algorithm* $\mathrm{Dsp}:\{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ *is called a* (k,ε)-disperser *if for every* (n,k)*-source* X *the support of* $\mathrm{Dsp}(U_d,X)$ *covers at least* $(1-\varepsilon)\cdot 2^m$ *points. Alternatively, for every set* $S \subset \{0,1\}^m$ *of size greater than* $\varepsilon 2^m$ *it holds that* $\Pr[\mathrm{Dsp}(U_d,X) \in S] > 0$.

[7] In general, if $n = \mathrm{poly}(\ell(|x|)) + \alpha\log_2(1/\delta)$ then for sufficiently small $\delta > \exp(-\mathrm{poly}\,\ell(|x|)))$, we have $n = (\alpha + o(1))\log_2(1/\delta) = \mathrm{poly}(\ell(|x|))$, which means $\delta < 2^{-n/\alpha'}$ for every $\alpha' > \alpha$.

Dispersers can be used for the reduction of one-sided error analogously to the use of extractors for the reduction of two-sided error. Specifically, regarding the aforementioned function f_x (and assuming that either $\Pr[f_x(U_{\ell(|x|)}) = 1] > \varepsilon$ or $f_x(U_{\ell(|x|)}) = 0$), we may use any (k,ε)-disperser $\mathrm{Dsp} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^{\ell(|x|)}$ in attempt to find a point z such that $f_x(z) = 1$. Indeed, if $\Pr[f_x(U_{\ell(|x|)}) = 1] > \varepsilon$ then $|\{z : (\forall s \in \{0,1\}^d)\, f_x(\mathrm{Dsp}(s,z)) = 0\}| < 2^k$, and thus the one-sided error can be reduced from $1-\varepsilon$ to $2^{-(n-k)}$ while using n random bits.

4.1.4 Other perspectives

Extractors and dispersers have an appealing interpretation in terms of bipartite graphs. Starting with dispersers, we view a disperser $\mathrm{Dsp} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ as a bipartite graph $G = ((\{0,1\}^n, \{0,1\}^m), E)$ such that $E = \{(x, \mathrm{Dsp}(s,x)) : x \in \{0,1\}^n, s \in \{0,1\}^d\}$. This graph has the property that *any* subset of 2^k vertices on the left (i.e., in $\{0,1\}^n$) has a neighborhood that contains more than a $1-\varepsilon$ fraction of the vertices of the right, which is remarkable in the typical case where d is small (e.g., $d = O(\log n/\varepsilon)$) and $n \gg k \geq m$ whereas $m = \Omega(k)$ (or at least $m = k^{\Omega(1)}$). Furthermore, if Dsp is efficiently computable then this bipartite graph is strongly constructible in the sense that, given a vertex on the left, one can efficiently find all its neighbors. An extractor $\mathrm{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ yields an analogous graph with a even stronger property: the neighborhood multi-set of *any* subset of 2^k vertices on the left covers the vertices on the right in an almost uniform manner.

An odd perspective. In addition to viewing extractors as averaging samplers, which in turn may be viewed within the scope of the pseudorandomness paradigm, we mention here an even more odd perspective. Specifically, randomness extractors may be viewed as randomized (by the seed) algorithms designed on purpose such that to be fooled by any weak random source (but not by an even worse source). Consider an (k,ε)-extractor $\mathrm{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$, for say $\varepsilon \leq 1/100$, $m = k = \omega(\log n/\varepsilon)$ and $d = O(\log n/\varepsilon)$, and a potential test T_S, parameterized by a set $S \subset \{0,1\}^m$, such that $\Pr[T_S(x) = 1] = \Pr[\mathrm{Ext}(U_d, x) \in S]$ (i.e., on input $x \in \{0,1\}^n$, the test uniformly selects $s \in \{0,1\}^d$ and outputs 1 if and only if $\mathrm{Ext}(s,x) \in S$). Then, for every (n,k)-source X the test T_S does not distinguish X from U_n (i.e., $\Pr[T_S(X)] = \Pr[T_S(U_n)] \pm 2\varepsilon$, because $\mathrm{Ext}(U_d, X)$ is 2ε-close to $\mathrm{Ext}(U_d, U_n)$). On the other hand, for every $(n, k-d-4)$-flat source Y there exists a set S such that T_S distinguish Y from U_n with gap 0.9 (e.g., for S that equals the support of $\mathrm{Ext}(U_d, Y)$, it holds that $\Pr[T_S(Y)] = 1$ and $\Pr[T_S(U_n)] \leq |S| \cdot 2^{-m} + \varepsilon = 2^{-4} + \varepsilon < 0.1$). Furthermore, this class of tests detects as defected, with probability $2/3$, any source that has entropy below $(k/4) - d$.[8] Thus, these weird class of tests views each (n,k)-source as "pseudorandom" while detecting sources of lower entropy (e.g., entropy lower than $(k/4) - d$) as non-pseudorandom. Indeed, this perspective stretches the pseudorandomness paradigm quite far.

4.2 Constructions

Recall that we seek explicit constructions of extractors; that is, functions $\mathrm{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ that can be computed in polynomial-time. The question, of course, is of parameters; that is, having (k,ε)-extractors with m as large as possible and d as small as possible. We first note

[8] For any such source Y, the distribution $Z = \mathrm{Ext}(U_d, Y)$ has entropy at most $k/4 = m/4$, and thus is 0.7-far from U_m (and 2/3-far from $\mathrm{Ext}(U_d, U_n)$). The lower-bound on the statistical distance of Z to U_m can be proven by the contra-positive: if Z is δ-close to U_m then its entropy is at least $(1-\delta) \cdot m - 1$ (e.g., by using Fano's inequality, see [4, Thm. 2.11.1]).

that $m \leq k + d - (2\log_2(1/\varepsilon) - O(1))$ and $d \geq \log_2((n-k)/\varepsilon^2) - O(1)$, regardless of explicitness. The aforementioned bounds are in fact tight; that is, there exists (non-explicit) (k, ε)-extractors with $m = k + d - 2\log_2(1/\varepsilon) - O(1)$ and $d = \log_2((n-k)/\varepsilon^2) + O(1)$. The obvious goal is to meet these bounds via explicit constructions.

Despite tremendous progress on this problem (and occasional claims regarding "optimal" explicit constructions), the ultimate goal was not reached yet. Yet, we are pretty close. In particular, we have the following.

Theorem 10 (explicit constructions of extractors): *Explicit (k, ε)-extractors of the form* $\mathrm{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ *exist in the following cases:*

- *For any constants $\varepsilon, \alpha > 0$, with $d = O(\log n)$ and $m = (1-\alpha) \cdot k$.*
- *For any constants $\varepsilon, \alpha > 0$, with $d = (1-\alpha) \cdot \log_2 n$ and $m = k/\mathrm{poly}(\log n)$.*
- *For any $\varepsilon > \exp(-k/\log k)$, with $d = O(\log n/\varepsilon)$ and $m = \Omega(k/\log k)$.*

The second item is due to [8], and the other two are due to [6], where these works build on previous ones (which are not cited here). We note that, for sake of simplicity, we did not quote the best possible bounds. Furthermore, we did not mention additional incomparable results (which are relevant for different ranges of parameters). In general, it seems that the "last word" has not been said yet: indeed the current results are close to optimal, but this cannot be said about the way that they are achieved. In view of the foregoing, we refrain from trying to provide an overview of the proof of Theorem 10, and review instead a conceptual insight that opened the door to much of the recent developments in the area.

The pseudorandomness connection

We conclude this section with an overview of a fruitful connection between extractors and certain pseudorandom generators. The connection, discovered by Trevisan [9], is surprising in the sense that it goes in a non-standard direction: it transforms certain pseudorandom generators into extractors. As argued throughout this course, computational objects are typically more complex than the corresponding information theoretical objects. Thus, if pseudorandom generators and extractors are at all related (which was not suspected before [9]) then this relation should not be expected to help in the construction of extractors, which seem an information theoretic object. Nevertheless, the discovery of this relation did yield a breakthrough in the study of extractors.[9]

But before describing the connection, let us wonder for a moment. Just looking at the syntax, we note that pseudorandom generators have a single input (i.e., the seed), while extractors have two inputs (i.e., the n-bit long source and the d-bit long seed). But taking a second look at the Nisan–Wigderson Generator, we note that this construction can be viewed as taking two inputs: a d-bit long seed and a "hard" predicate on d'-bit long strings (where $d' = \Omega(d)$). Now, an appealing idea is to use the n-bit long source as a description of a (worse-case) hard predicate (which indeed means setting $n = 2^{d'}$). The key observation is that even if the source is only weakly random we expect it to represent a function that is hard on the worst-case.

Recall that the aforementioned construction is supposed to yield a pseudorandom generator whenever it starts with a hard predicate. In the current context, where there are no computational

[9] We note that once the connection became better understood, influence started going in the "right" direction: from extractors to pseudorandom generators.

restrictions, pseudorandomness is supposed to hold against any (computationally unbounded) distinguisher, and thus here pseudorandomness means being statistically close to the uniform distribution (on strings of the adequate length, denoted ℓ). Intuitively, this makes sense only if the observed sequence is shorter that the amount of randomness in the source (and seed), which is indeed the case (i.e., $\ell < k + d$, where k denotes the min-entropy of the source). Hence, there is hope to obtain a good extractor this way.

To turn the hope into a reality, we need a proof. Looking again at the Nisan–Wigderson Generator, we note that the proof of indistinguishability of the generator provides a black-box procedure for computing the underlying predicate when given oracle access to any potential distinguisher. This black-box procedure was implemented by a relatively small circuit, so it is clear that this procedure contains relatively little information on top of the observed ℓ-bit long output of the extractor/generator. Specifically, a crude estimate yields an upper-bound of $b \stackrel{\text{def}}{=} \text{poly}(\ell)$ on the amount of information available to the procedure, which is suppose to compute the predicate correctly on every input. That is, this amount of information is supposed to fully determine the predicate in use, which in turn is identical to the n-bit long source. Thus, if the source has min-entropy exceeding b, then it cannot be fully determine using only b bits of information. It follows that the foregoing construction constitutes a $(b + O(1), 1/6)$-extractor (outputting ℓ bits), where the constant $1/6$ is the one used in the standard analysis of the Nisan–Wigderson Generator. Note that this was obtained with a seed of length $d = O(d') = O(\log n)$. The argument can be extended to obtain $(k, \text{poly}(1/k))$-extractors that output $m = k^{\Omega(1)}$ bits using a seed of length $d = O(\log n)$, provided that $k = n^{\Omega(1)}$. (For further details, the interested reader is referred to either [9] or [7].)

We note that the foregoing description has only referred to two abstract properties of the Nisan–Wigderson Generator: (1) the fact that it uses any worst-case hard predicate as a black-box, and (2) the fact that its analysis uses any distinguisher as a black-box. In particular, we viewed the amplification of worst-case hardness to inapproximability as part of the construction of the pseudorandom generator. An alternative presentation, which is more self-contained, replaces the amplification step by a direct argument in the current (information theoretic) context and plugs the resulting predicate directly into the Nisan–Wigderson Generator.

The alternative presentation. Recall that we viewed the violation of worst-case hardness as an ability to fully determine the predicate, and used the upper-bound on the information available to the "universal" evaluation procedure to upper-bound the number of sources that may be bad. From this perspective, the violation of inapproximability may be viewed as ability to generate a function that is close to the predicate. If the predicates of interests are far apart, then this directly yields the desired bound (on the number of bad sources that are encoded in such predicates). Thus, the idea is to encode the n-bit long source by an error correcting code of length $n' = \text{poly}(n)$ and relative distance $0.5 - (1/n)^2$, and use the resulting codeword as a truth-table of a predicate for the Nisan–Wigderson Generator. Such codes (coupled with efficient encoding algorithms) do exist, and the benefit in using them is that each n'-bit long string (determined by the information available to the aforementioned "universal" evaluation procedure) may be $(0.5 - (1/n))$-close to at most $O(n^2)$ codewords.

Recommended reading

The interested reader is referred to a truly excellent survey of Shaltiel [7]. This survey contains a comprehensive introduction to the area, including an overview of the ideas that underly the various constructions. In particular, the survey describes the approaches used before the discovery of the pseudorandomness connection, the connection itself (and the constructions that arise from it), and the "third generation" of constructions that followed.

References

[1] N. Alon and J.H. Spencer. *The Probabilistic Method.* John Wiley & Sons, Inc., 1992.

[2] L. Carter and M. Wegman. Universal Hash Functions. *Journal of Computer and System Science*, Vol. 18, 1979, pages 143–154.

[3] B. Chor and O. Goldreich. Unbiased Bits from Sources of Weak Randomness and Probabilistic Communication Complexity. *SIAM Journal on Computing*, Vol. 17, No. 2, pages 230–261, 1988.

[4] T.M. Cover and G.A. Thomas. *Elements of Information Theory.* John Wiley & Sons, Inc., New-York, 1991.

[5] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness.* Algorithms and Combinatorics series (Vol. 17), Springer, 1999.

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

[7] R. Shaltiel. Recent Developments in Explicit Constructions of Extractors. *Bulletin of the EATCS 77*, pages 67–95, 2002.

[8] R. Shaltiel and C. Umans. Simple Extractors for All Min-Entropies and a New Pseudo-Random Generator. In *42nd IEEE Symposium on Foundations of Computer Science*, pages 648–657, 2001.

[9] L. Trevisan. Constructions of Near-Optimal Extractors Using Pseudo-Random Generators. In *31st ACM Symposium on the Theory of Computing*, pages 141–148, 1998.

[10] D. Zuckerman. Simulating BPP Using a General Weak Random Source. *Algorithmica*, Vol. 16, pages 367–391, 1996.

[11] D. Zuckerman. Randomness-Optimal Oblivious Sampling. *Journal of Random Structures and Algorithms*, Vol. 11, Nr. 4, December 1997, pages 345–367.