On Constructing Locally Computable Extractors and Cryptosystems in the Bounded Storage Model

Salil P. Vadhan[*]
Division of Engineering & Applied Sciences
Harvard University
Cambridge, MA 02138
`salil@eecs.harvard.edu`
`http://eecs.harvard.edu/~salil/`

November 1, 2002

Abstract

We consider the problem of constructing randomness extractors which are *locally computable*, i.e. only read a small number of bits from their input. As recently shown by Lu (*CRYPTO '02*), locally computable extractors directly yield secure private-key cryptosystems in Maurer's bounded storage model (*J. Cryptology*, 1992).

In this note, we observe that a fundamental lemma of Nisan and Zuckerman (*J. Computer and System Sciences,* 1996) yields a general technique for constructing locally computable extractors. Specifically, we obtain a locally computable extractor by combining any extractor with any randomness-efficient (averaging) sampler. Plugging in known extractor and sampler constructions, we obtain locally computable extractors, and hence cryptosystems in the bounded storage model, whose parameters improve upon previous constructions and come quite close to the lower bounds.

Along the way, we also present a refinement of the Nisan–Zuckerman lemma, showing that random sampling bits from a weak random source preserves the min-entropy rate up to an arbitrarily small additive loss (whereas the original lemma loses a logarithmic factor).

Keywords: extractors, bounded storage model, everlasting security, space-bounded adversaries, unconditional security, averaging samplers, expander graphs

[*]Supported by NSF grants CCR-0205423 and CCR-0133096.

1 Introduction

Maurer's *bounded storage model* for private-key cryptography [Mau92] has been the subject of much recent activity. In this model, one assumes that there is public, high-rate source of randomness and that all parties have limited storage so that they cannot record all the randomness from the source. Remarkably, this quite plausible model makes it possible to construct private-key cryptosystems which are information-theoretically secure and require no unproven complexity assumptions (in contrast to most of modern cryptography). Intuitively, a shared secret key can be used by legitimate parties to randomly select bits from the random source about which the adversary has little information (due to the bound on its storage). With some further processing, the legitimate parties can convert these unpredictable bits into ones which the adversary cannot distinguish from truly random (in an information-theoretic sense), and hence they can safely be used for cryptographic purposes, e.g. as a one-time pad for encryption.

A sequence of works [Mau92, CM97, AR99, ADR02, DR02, DM02, Lu02] has given increasingly secure and efficient protocols in this model. In particular, the works of Aumann, Ding, and Rabin [ADR02, DR02] showed that protocols in this model have the novel property of "everlasting security" — the security is preserved even if the key is reused an exponential number of times and is subsequently revealed to the adversary.

Recently, Lu [Lu02] showed that work in this model can be cast nicely in the framework of *randomness extractors*. Extractors, introduced by Nisan and Zuckerman [NZ96], are procedures for extracting almost-uniform bits from sources of biased and correlated bits. These powerful tools have been the subject of intense study, and have found many applications to a wide variety of topics in the theory of computation. (See the surveys [NT99, Sha02].) One of the first applications, in the original paper of Nisan and Zuckerman, was to construct pseudorandom generators for space-bounded computation. Thus, they seem a natural tool to use in the bounded storage model, and indeed Lu [Lu02] showed that any extractor yields secure private-key cryptosystems in the bounded-storage model. However, the efficiency considerations of the bounded-storage model require a nonstandard property from extractors — namely that they are *locally computable*,[1] i.e. they can be computed by reading only a few bits from the random source. Lu constructed locally computable extractors by first constructing locally computable error-correcting codes, and then plugging them into the specific extractor construction of Trevisan [Tre01].

In this note, we observe that a fundamental lemma of Nisan and Zuckerman [NZ96] yields a general technique for constructing locally computable extractors. Roughly speaking, the lemma states that a random of sample of bits from a string of high min-entropy[2] also has high min-entropy. This allows us to obtain a locally computable extractor by combining any extractor with any randomness-efficient "sampler" (a procedure for sampling a subset of a universe so that the average value of any $[0, 1]$-valued function is approximately preserved with high probability). By plugging in known extractor and sampler constructions, we obtain locally computable extractors, and hence cryptosystems in the bounded storage model, whose parameters are better than the previous constructions and come quite close to the lower bounds. Along the way, we also present a refinement of the Nisan–Zuckerman lemma, showing that random sampling preserves the min-

[1]This terminology was suggested by Yan Zong Ding, and we prefer it to the terminology "on-line extractors," which was used with different meanings in [BRST02, Lu02]. The issue of "local computation" versus "on-line computation" is discussed in more detail in Section 2.

[2]Like Shannon entropy, the min-entropy of a probability distribution X is a measure of the number of bits of "randomness" in X. A formal definition is given in Section 2.

entropy rate up to an arbitrarily small additive loss (whereas the original lemma loses a logarithmic factor).

In retrospect, several previous cryptosystems in the bounded-storage model, such as [CM97] and [ADR02], can be viewed as special cases of our approach, with particular choices for the extractor and sampler. By abstracting the properties needed from the underlying tools, we are able to use state-of-the-art extractors and samplers, and thereby obtain our improvements.

2 The Bounded-Storage Model

The Random Source. The original model of Maurer [Mau92] envisioned the random source as a high-rate stream of perfectly random bits being broadcast from some natural or artificial source of randomness. However, since it may difficult to obtain perfectly random bits from a physical source, particularly at a high rate, we feel it is important to investigate the minimal conditions on the random source under which this type of cryptography can be performed. As noted in [Lu02, DM02], the existing constructions still work even if we only assume that the source has sufficient “entropy”. Below we formalize this observation, taking particular note of the kind of independence that is needed when the cryptosystem is used many times.

We model the random source as a sequence of random variables $X_1, X_2, \ldots$, each distributed over $\{0,1\}^n$, where X_t is the state of the source at time period t. To model a random source which is a high-rate “stream” of bits, the X_t's can be thought of as a partition of the stream into contiguous substrings of length n. However, one may also consider random sources that are not a stream, but rather a (natural or artificial) “oracle” of length n, which changes over time and can be probed at positions of one's choice. In both cases, n should be thought of as very large, greater than the storage capacity of the adversary (and the legitimate parties).

To obtain the original model of a perfectly random stream, each the X_t's can be taken to be uniform on $\{0,1\}^t$ and independent of each other. Here we wish to allow biases and correlations in the source, only assuming that each X_t has sufficient randomness, as measured by the following variant of entropy (advocated in [Zuc96]). The *min-entropy* of a random variable X is defined to be $\mathrm{H}_\infty(X) \overset{\text{def}}{=} \min_x \log(1/\Pr[X = x])$. (All logarithms in this paper are base 2.) Intuitively, min-entropy measures randomness in the “worst case,” whereas standard (Shannon) entropy measures the randomness in X “on average.” Adopting a worst-case measure is important here since we want security to hold with high probability and not just “on average”. (The results will also apply for random sources that are statistically close to having high min-entropy, such as those of high Renyi entropy.) X is called a *k-source* if $\mathrm{H}_\infty(X) \geq k$, i.e. for all x, $\Pr[X = x] \leq 2^{-k}$.

For our cryptosystems, we will need to require that the random source has high min-entropy conditioned on the future. For a random variable A and an event E we write $A|_E$ for the distribution of A conditioned on E. With this notation, our model of a random source is:

Definition 1 *A sequence of random variables $X_1, X_2, \ldots$, each distributed over $\{0,1\}^n$ is a* reverse block source *of* entropy rate α *if for every $t \in \mathbb{N}$ and every $x_{t+1}, x_{t+2}, \ldots$, the random variable $X_t|_{X_{t+1}=x_{t+1}, X_{t+2}=x_{t+2}, \ldots}$ is an αn-source.*

As the terminology suggests, this is the same as the Chor-Goldreich [CG88] notion of a *block source*, but “backwards” in time. Intuitively, it means that X_t possesses αn bits of randomness that will be “forgotten” at the next time step. This is somewhat less natural than the standard notion of a (forward) block source, but it still may be a reasonable model for some physical sources of randomness which are not perfectly random.[3] Below we will see why some condition of this form (high entropy conditioned on the future) is necessary for the cryptography. In any case, in the special case $\alpha = 1$, Definition 1 is equivalent to requiring that X_t's are uniform and independent, so the issue of reversal is moot.

Cryptosystems. Here, as in previous works, we focus on the task of using a shared key to extract pseudorandom bits from the source. These pseudorandom bits can then be used to perform private-key encryption or message authentication. A *pseudorandom extraction scheme in the bounded storage model* is a function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. Such a scheme is to be used as follows. Two parties share a key $K \in \{0,1\}^d$. At time t, they apply $\mathrm{Ext}(\cdot, K)$ to the random source X_t to obtain m pseudorandom bits, given by $Y_t = \mathrm{Ext}(X_t, K)$. At time $t+1$ (or later), Y_t will be pseudorandom to the adversary (if the scheme is secure), and hence can be used by the legitimate parties as a shared random string for any purpose (e.g. as a one-time pad for encryption). The pseudorandomness of Y_t will rely on the fact that, at time $t+1$, X_t is no longer accessible to the adversary. More generally, we need X_t to be unpredictable from future states of the random source, as captured by our notion of a reverse block source. Note that even if Y_t will only be used exactly at time $t+1$, we still need X_t to have high min-entropy given the entire future, as the adversary can store Y_t.

We now formally define security for a pseudorandom extraction scheme. Let βn be the bound on the storage of the adversary A, and we'll denote by $S_t \in \{0,1\}^{\beta n}$ the state of the adversary at time t. Following the usual paradigm for pseudorandomness, we consider the adversary's ability to distinguish two experiments — the "real" one, in which the extraction scheme is used, and an "ideal" one, in which truly random bits are used.

Real Experiment: Let $X_1, X_2, \ldots$ be the random source, and let $K \overset{\mathrm{R}}{\leftarrow} \{0,1\}^d$. For all t, let $Y_t = \mathrm{Ext}(X_t, K)$ be the extracted bits. Let $S_0 = 0^d$, and for $t = 1, \ldots, T$, let $S_t = A(S_{t-1}, X_t, Y_1, \ldots, Y_{t-1})$. Output $A(S_{T-1}, X_T, Y_1, \ldots, Y_{T-1}, K) \in \{0,1\}$.

Ideal Experiment: Let $X_1, X_2, \ldots$ be the random source, and let $K \overset{\mathrm{R}}{\leftarrow} \{0,1\}^d$. For all t, let $Y_t \overset{\mathrm{R}}{\leftarrow} \{0,1\}^m$. Let $S_0 = 0^d$, and for $t = 1, \ldots, T$, let $S_t = A(S_{t-1}, X_t, Y_1, \ldots, Y_{t-1})$. Output $A(S_{T-1}, X_T, Y_1, \ldots, Y_{T-1}, K) \in \{0,1\}$.

Definition 2 *We call* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ ε-secure *for storage rate β and entropy rate α if for every reverse block source (X_t) of entropy rate α, every adversary A with storage bound βn, and every $T > 0$, A distinguishes between the real and ideal experiments with advantage at most $T \cdot \varepsilon$.*

Remarks:

- In the real experiment, we give the extracted strings Y_t explicitly to the adversary, as is typical in definitions of pseudorandomness. However, when they are used in a cryptographic application (e.g. as one-time pads), they of course will not be given explicitly to the adversary.
- The string Y_t extracted at time t is not given to the adversary (i.e. is not "used") until time $t+1$. As mentioned above, this is crucial for security.

[3]The consideration of such sources raise interesting philosophical questions: does the universe keep a perfect record of the past? If not, then reverse block sources seem plausible.

- The definition would be interesting even if $Y_1, \ldots, Y_{t-2}$ were not given to the adversary at time t, (i.e. $S_t = A(S_{t-1}, X_t, Y_{t-1})$), and K were not given to the adversary at the end. Giving all the previous Y_i's implies that it is "safe" to use Y_i at any time period after i (rather than exactly at time $i+1$). Giving the adversary the key at time T means that even if the secret key is compromised, earlier transactions remain secure. This is the remarkable property of "everlasting security" noticed by [ADR02].

- We require that the security degrade only linearly with the number of times the same key is reused.

- No constraint is put on the computational power of the adversary except for the storage bound of βn (as captured by $S_t \in \{0,1\}^{\beta n}$). This means that the distributions of $(S_{T-1}, X_T, Y_1, \ldots, Y_{T-1}, K)$ in the real and ideal experiments are actually close in a statistical sense (namely, with respect to statistical difference, as defined in the next section).

- The definition is impossible to meet unless $\alpha > \beta$: If $\alpha \leq \beta$, we can take each X_t to have its first αn bits uniform and the rest fixed to zero. Then an adversary with βn storage can entirely record X_t, and thus can distinguish $Y_t = \mathrm{Ext}(X_t, K)$ from uniform at time $t+1$ (unless $|K| \geq |Y_t|$, in which case each Y_t reveals information about the key and security is also lost).

As usual, the above definition implies that to design a cryptosystem (e.g. private-key encryption or message authentication) one need only prove its security in the ideal experiment, where the two parties effectively share an infinite sequence of random strings $Y_1, Y_2, \ldots$. Security in the bounded storage model immediately follows if the Y_i's are then replaced with a secure pseudorandom extraction scheme.

Efficiency Considerations. In addition to security, it is important for the extraction scheme to be efficient. In the usual spirit of cryptography, we of course would like the computation of Ext to require much less space than the adversary's storage bound of βn. However, note that the honest parties will have to store the entire extracted key $Y_t \in \{0,1\}^m$ during time t (when it is not yet safe to use), so reducing the space to m is the best one can hope for (but we envision $m \ll n$ so this is still useful). However, since n is typically envisioned to be huge, it is preferable not just to reduce the space for Ext to much less than n, but also the time spent.[4] Thus, we adopt as our efficiency measure the *number of bits read from the source*. Of course, once these bits are read, it is important that the actual computation of Ext is efficient with respect to both time and space. In our constructions (and all previous constructions), Ext can be computed in polynomial time and polylogarithmic work space (indeed even in NC). Thus the total storage required by the legitimate parties equals the number of bits read from the source up to a constant factor.

The following shows that the number of bits read from the source must be linear in m, and grow as the difference between the entropy rate and storage rate goes to zero.

Proposition 3 *If* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is an ε-secure pseudorandom extraction scheme for storage rate β and entropy rate α, then* $\mathrm{Ext}(\cdot, K)$ *depends on at least* $(1-\varepsilon) \cdot (1/(\alpha - \beta)) \cdot m - 1$ *bits of its input (on average, taken over K).*

[4]If having Ext computable in small space with one pass through the random source is considered sufficiently efficient, then the work of Bar-Yossef et al. [BRST02] is also applicable here. See Section 3.

Proof Sketch: For this proof sketch, we ignore the parameter ε (i.e. pretend it is zero). For a subset $T_X \subset [n]$ of size $|T_X| = \alpha n$, consider a source (X_t) where the X_t's are independent and each X_t is uniform on the bits in T_X and 0 elsewhere. For a subset $T_A \subset T_X$ of size $|T_A| = \beta n$, consider an adversary which records the bits of X_t in T_A. For any T_A and T_X, the this source has entropy rate α and the adversary has storage rate β. Now, if $\mathrm{Ext}(\cdot, K)$ reads only a set T_K of bits from the source, it only gets $|T_K \cap (T_X \setminus T_A)|$ bits of entropy beyond what the adversary sees, so intuitively m must be at most $\mathrm{E}_K[|T_K \cap (T_X \setminus T_A)|])$.

Now consider randomly chosen subsets $T_A \subset T_X \subset [n]$. Then $T_X \setminus T_A$ is a random subset of density $(\alpha-\beta)$, so for a fixed K the expected intersection size $|T_K \cap (T_X \setminus T_A)|$ is at most $(\alpha-\beta)\cdot|T_K|$. By averaging, there exists T_A and T_X such that $\mathrm{E}_K[|T_K \cap (T_X \setminus T_A)|] \leq (\alpha-\beta)\cdot \mathrm{E}_K[|T_K|]$. We conclude that m is at most $(\alpha-\beta)\cdot \mathrm{E}_K[|T_K|]$. This proof can be formalized via standard entropy arguments. □

Another common complexity measure in cryptography is the key length, which should be minimized. Figure 1 describes the performance of previous schemes and our new constructions with respect to these two complexity measures.[5]

reference	key length	# bits read	restrictions
[CM97]	$O(\log n)$	$O(m/\varepsilon^2)$	interactive
[AR99]	$O(\log n \cdot \log(1/\varepsilon))$	$O(m \cdot \log(1/\varepsilon))$	$\alpha = 1, \beta < 1/m$
[ADR02]	$O(m \cdot \log n \cdot \log(1/\varepsilon))$	$O(m \cdot \log(1/\varepsilon))$	
[DR02]	$O(\log n \cdot \log(1/\varepsilon))$	$O(m \cdot \log(1/\varepsilon))$	$\alpha = 1,\ \beta < 1/\log m$
[DM02]	$O(\log n \cdot \log(1/\varepsilon))$	$O(m \cdot \log(1/\varepsilon))$	
[Lu02]	$O(m \cdot (\log n + \log(1/\varepsilon)))$	$O(m \cdot \log(1/\varepsilon))$	
[Lu02]	$O((\log^2(n/\varepsilon)/\log n))$	$O(m \cdot \log(1/\varepsilon))$	$m \leq n^{1-\Omega(1)}$
here	$O(\log n + \log(1/\varepsilon))$	$O(m + \log(1/\varepsilon))$	$\varepsilon > \exp(-n/2^{O(\log^* n)})$

Figure 1: Comparison of pseudorandom extraction schemes in the bounded-storage model. Parameters are for ε-secure schemes $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, for constant storage rate β and entropy rate α, where $\alpha > \beta$. We only list the parameters for the case that the number of bits read from the source is $o(n)$, as n is assumed to be huge and infeasible.

We now touch upon a couple of additional efficiency considerations. First, if the random source is indeed a high-rate stream (as opposed to an "oracle source"), it is important that the positions to be read from the source can be computed offline (from just the key) and sorted so that they can be quickly read in order as the stream goes by. This is the case for our scheme and previous ones.

Second, one can hope to reduce the space of the legitimate parties to exactly $m + d$ (i.e., the length of the extracted string plus the key). That is, even though the schemes read more than m bits from the source, the actual computation of the m-bit extracting string can be done "in place" as the bits from the source are read. This property holds for most of the previous constructions, as each bit of the the extracted string is a parity of $O(\log(1/\varepsilon))$ bits of the source. Our construction

[5]Most of the previous schemes were explicitly analyzed only for the case of a perfectly random source, i.e. $\alpha = 1$, but the proofs actually also work for weak random sources provided $\alpha > \beta$ (except where otherwise noted) [Lu02, Mau02].

does not seem to have this property in general (though specific instantiations may); each bit of the output can be a function of the entire $O(m + \log(1/\varepsilon))$ bits read from the source. Still the space used by our scheme is $O(m + \log(1/\varepsilon))$, only a constant factor larger than optimal.

The fact that each bit of the extracted string depends on only $O(\log(1/\varepsilon))$ bits of the source in previous constructions has one other potential benefit: it may be useful for incorporating error-correction in case the random source cannot be accessed perfectly [Rab01]. Assume the extracted strings are used as one-time pads for encryption. Then, in the previous schemes, a bit-error probability of $O(1/\log(1/\varepsilon))$ in accessing the source can be viewed as a constant bit-error probability in the plaintext, which can be handled by using a standard, asymptotically good error-correcting code.

3 Locally Computable Extractors

In this section, we define extractors and locally computable extractors, and recall Lu's result [Lu02] about their applicability to the bounded storage model. Then we discuss averaging samplers and describe how using them to sample bits from a random source preserves the min-entropy rate, via a lemma of Nisan and Zuckerman [NZ96] which we refine. Finally, we give our general construction which combines any extractor and any averaging sampler to yield a locally computable extractor.

The *statistical difference* (or variation distance) between two random variables X, Y taking values in a universe $\mathcal{U}$ is defined to be

$$\Delta(X,Y) \overset{\text{def}}{=} \max_{S \subset \mathcal{U}} \left| \Pr[X \in S] - \Pr[Y \in S] \right| = \frac{1}{2} \sum_{x \in \mathcal{U}} \left| \Pr[X = x] - \Pr[Y = x] \right|.$$

We say X and Y are *ε-close* if $\Delta(X,Y) \le \varepsilon$.

An extractor is a procedure for extracting almost-uniform bits from any random source of sufficient min-entropy. This is not possible to do deterministically, but it is possible using a short *seed* of truly random bits, as captured in the following definition of Nisan and Zuckerman.

Definition 4 ([NZ96]) $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* strong[6] (k,ε)-extractor *if for every k-source X, $U_d \circ \mathrm{Ext}(X, U_d)$ is ε-close to $U_d \times U_m$.*

The goal in constructing extractors is to minimize the seed length d and maximize the output length m. We will be precise about the parameters in later sections, but, for reference, an "optimal" extractor has a seed length of $d = O(\log n + \log(1/\varepsilon))$ and an output length of $m = k - O(\log(1/\varepsilon))$, i.e. almost all of the min-entropy is extracted using a seed of logarithmic length.

Recently, Lu proved that any extractor yields secure cryptosystems in the bounded storage model:

Theorem 5 ([Lu02]) *If* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a strong $(\delta n - \log(1/\varepsilon), \varepsilon)$-extractor, then for any $\beta > 0$,* Ext *is an 2ε-secure pseudorandom extraction scheme for storage rate β and entropy rate $\beta + \delta$.*

However, as noted by Lu, for a satisfactory solution to cryptography in the bounded-storage model, the extractor should only read only a few bits from the source.

[6] A standard (i.e. non-strong) extractor requires only that $\mathrm{Ext}(X, U_d)$ is ε-close to uniform.

Definition 6 Ext : $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is* t-locally computable *(or* t-local*) if for every* $r \in \{0,1\}^d$*,* $\mathrm{Ext}(x,r)$ *depends on only* t *bits of* x*.*

Thus, in addition to the usual goals of minimizing d and maximizing m, we also wish to minimize t. Bar-Yossef, Reingold, Shaltiel, and Trevisan [BRST02] studied a related notion of *on-line extractors*, which are required to be computable in small space in one pass. They show that space $\approx m$ is necessary and sufficient to evaluate extractors with output length m. Since the small-space requirement is weaker than being locally computable, their lower bound also applies here.[7] But a stronger lower bound for locally computable extractors can be obtained by combining Proposition 3 and Theorem 5, or by mimicking the proof of Proposition 3 directly for t-local extractors:

Proposition 7 *If* Ext : $\{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a t-local strong $(\delta n, \varepsilon)$-extractor, then* $t \geq (1-\varepsilon)\cdot(1/\delta)\cdot m - 1$*.*

Lu [Lu02] observed that the encryption schemes of Aumann, Ding, and Rabin [AR99, ADR02, DR02] can be viewed as locally computable extractors, albeit with long seeds. He constructed locally computable extractors with shorter seeds based on Trevisan's extractor [Tre01]. The construction of Dziembowski and Maurer [DM02] can also be viewed as a locally computable extractor [Mau02]. The parameters of these constructions can be deduced from Figure 1.

Our construction of locally computable extractors is based on a fundamental lemma of Nisan and Zuckerman [NZ96], which says that if one samples a random subset of bits from a weak random source, the min-entropy rate of the source is (nearly) preserved. More precisely, if $X \in \{0,1\}^n$ is a δn-source, and $X_S \in \{0,1\}^t$ is the projection of X onto a random set $S \subset [n]$ of t positions, then w.h.p. X_S is ε-close to a $\delta' t$-source, for some δ' depending on δ. Thus, to obtain a locally computable extractor, we can simply apply a (standard) extractor to X_S, and thereby output $\delta' t$ almost-uniform bits. That is, part of the seed of the locally computable extractor will be used to select S, and the remainder as the seed for applying the extractor to X_S.

However, choosing a completely random set S of positions is expensive in the seed length, requiring $\approx |S| \cdot \log n$ random bits. (This gives a result analogous to [ADR02], because $|S| \geq m$.) To save on randomness, Nisan and Zuckerman [NZ96] showed that S can be sampled k-wise independently. More generally, their proof only requires that S has large intersection with any subset of $[n]$ of a certain density with high probability [RSW00]. We will impose a slightly stronger requirement on the sampling method: for any $[0,1]$-valued function, its average on S should approximate its average on $[n]$ with high probability. Such sampling procedures are known as *averaging* (or *oblivious*) *samplers*, and have been studied extensively [BR94, CEG95, Zuc97, Gol97]. Our definition differs slightly from the standard definition, to allow us to obtain some savings in parameters (discussed later).

Definition 8 *A function* Samp : $\{0,1\}^r \to [n]^t$ *is a* (μ, θ, γ) averaging sampler *if for every function* $f : [n] \to [0,1]$ *with average value* $\frac{1}{n}\sum_i f(i) \geq \mu$,

$$\Pr_{(i_1,\ldots,i_t) \overset{R}{\leftarrow} \mathrm{Samp}(U_r)} \left[\frac{1}{t} \sum_{j=1}^{t} f(i_j) < \mu - \theta \right] \leq \gamma.$$

[7]This is because their space lower bounds apply also to a nonuniform branching program model of computation, where the space is always at most the number of bits read from the input.

Samp *has* distinct samples *if for every* $x \in \{0,1\}^r$*, the samples produced by* Samp(x) *are all distinct.*

That is, for any function f whose average value is at least μ, with high probability the sampler selects a sample of positions on which the the average value of f is not much smaller than μ. The goal in constructing averaging samplers is usually to simultaneously minimize the randomness r and sample complexity t. We will be precise about the parameters in later sections, but, for reference, an "optimal" averaging sampler uses only $t = O(\log(1/\gamma))$ samples and $r = O(\log n + \log(1/\gamma))$ random bits (for constant μ, θ).

In contrast to most applications of samplers, we will not necessarily be interested in minimizing the sample complexity. Ideally, we prefer samplers where the number of distinct samples can be chosen anywhere in the interval $[t_0, n]$, where t_0 is the minimum possible sample complexity. (Note that without the requirement of distinct samples, the number of samples can be trivially increased by repeating each sample several times.) Another atypical aspect of our definition is that we make the parameter μ explicit. Averaging samplers are usually required to give an approximation within additive error θ regardless of the average value of f, but being explicit about μ will allow us to obtain some savings in the parameters.

Using averaging samplers (rather than just samplers which intersect large sets) allows us to obtain a slight improvement to the Nisan–Zuckerman lemma. Specifically, Nisan and Zuckerman show that sampling bits from a source of min-entropy rate δ yields a source of min-entropy rate $\Omega(\delta/\log(1/\delta))$; our method can yield min-entropy rate $\delta - \tau$ for any desired τ.

For a string $x \in \{0,1\}^n$ and a sequence $s = (i_1, \ldots, i_t) \in [n]^t$, define $x_s \in \{0,1\}^t$ to be the string $x_{i_1} x_{i_2} \cdots x_{i_t}$. Recall that for a pair of jointly distributed random variables (A, B), we write $B|_{A=a}$ for B conditioned on the event $A = a$.

Lemma 9 (refining [NZ96]) *Suppose that* Samp : $\{0,1\}^r \to [n]^t$ *is an* (μ, θ, γ) *averaging sampler with distinct samples for* $\mu = (\delta - 2\tau)/\log(1/\tau)$ *and* $\theta = \tau/\log(1/\tau)$*. Then for every* δn*-source* X *on* $\{0,1\}^n$*, the random variable* $(U_r, X_{\mathrm{Samp}(U_r)})$ *is* $(\gamma + 2^{-\Omega(\tau n)})$*-close to* (A, B) *where* A *is uniform on* $\{0,1\}^r$ *and for every* $a \in \{0,1\}^r$*,*[8] *the random variable* $B|_{A=a}$ *is* $(\delta - 3\tau)t$*-source.*

The above lemma is where we use the fact that the sampler has distinct samples. Clearly, sampling the same bits of X many times cannot increase the min-entropy of the output, whereas the above lemma guarantees that the min-entropy grows linearly with t, the number of samples.

An alternative method to extract a shorter string from a weak random source while preserving the min-entropy rate up to a constant factor was given by Reingold, Shaltiel, and Wigderson [RSW00], as a subroutine in their improved extractor construction. However, the string produced by their method consists of bits of an encoding of the source in an error-correcting code rather than bits of the source itself, and hence is not good for constructing locally computable extractors (which were not their goal). As pointed out to us by Chi-Jen Lu and Omer Reingold, Lemma 9 eliminates the need for error-correcting codes in [RSW00].

The proof of Lemma 9 is deferred to Section 4. Given the lemma, it follows that combining an averaging sampler and an extractor yields a locally computable extractor.

[8]Intuitively, the reason we can guarantee that B has high min-entropy conditioned on *every* value of A, is that the "bad" values of A are absorbed in the $\gamma + 2^{-\Omega(\tau n)}$ statistical difference.

Theorem 10 *Suppose that* $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ *is an* (μ, θ, γ) *averaging sampler with distinct samples for* $\mu = (\delta - 2\tau)/\log(1/\tau)$ *and* $\theta = \tau/\log(1/\tau)$. *and* $\mathrm{Ext} : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}^m$ *is a strong* $((\delta - 3\tau)t, \varepsilon)$ *extractor. Define* $\mathrm{Ext}' : \{0,1\}^n \times \{0,1\}^{r+d} \to \{0,1\}^m$ *by*

$$\mathrm{Ext}'(x, (y_1, y_2)) = \mathrm{Ext}(x_{\mathrm{Samp}(y_1)}, y_2).$$

Then Ext' *is a t-local strong* $(\delta n, \varepsilon + \gamma + 2^{-\Omega(\tau n)})$ *extractor.*

Proof: For every (y_1, y_2), $\mathrm{Ext}'(x, (y_1, y_2))$ only reads the t bits of x selected by $\mathrm{Samp}(y_1)$, so Ext' is indeed t-local. We now argue that it is a $(\delta n, \varepsilon + \gamma + 2^{-\Omega(\tau n)})$ extractor. Let X be any δn-source. We need to prove that the random variable $Z = (U_r, U_d, \mathrm{Ext}'(X, (U_r, U_d))) = (U_r, U_d, \mathrm{Ext}(X_{\mathrm{Samp}(U_r)}, U_d))$ is close to uniform. By Lemma 9, $(U_r, X_{\mathrm{Samp}(U_r)})$ is $(\gamma + 2^{-\Omega(\tau n)})$-close to (A, B) where A is uniform on $\{0,1\}^r$ and $B|_{A=a}$ is a $(\delta - 3\tau)t$-source for every a. This implies that Z is $(\gamma + 2^{-\Omega(\tau n)})$-close to $(A, U_d, \mathrm{Ext}(B, U_d))$. Since Ext is a strong $((\delta - 3\tau)t, \varepsilon)$ extractor, $(U_d, \mathrm{Ext}(B|_{A=a}, U_d))$ is ε-close to $U_d \times U_m$ for all a. This implies that $(A, U_d, \mathrm{Ext}(B, U_d))$ is ε-close to $A \times U_d \times U_m = U_r \times U_d \times U_m$. By the triangle inequality, Z is $(\varepsilon + \gamma + 2^{-\Omega(\tau n)})$-close to $U_r \times U_d \times U_m$. ■

For intuition about the parameters, consider the case when $\delta > 0$ is an arbitrary constant, $\tau = \delta/6$, and $\gamma = \varepsilon$. Then using "optimal" averaging samplers and extractors will give a locally computable extractor with seed length $r + d = O(\log n + \log(1/\varepsilon))$ and output length $m = \Omega(\delta t) - O(\log(1/\varepsilon))$. This matches, up to constant factors, the seed length of an optimal extractor (local or not) and the optimal relationship between the output length and the number of bits read from the source.

4 Proof of Lemma 9

In this section, we prove Lemma 9. For readers willing to assume the lemma, this section can be skipped. Our proof has the same general structure as the Nisan–Zuckerman proof; the main point of departure will be pointed out below.

We use the convention that capital letters denote random variables, and lower-case letters denote specific values for them.

Let X be a δn-source X and let $S = \mathrm{Samp}(U_d)$. For every $x \in \mathrm{Supp}(X)$, define $p_i(x) = \Pr[X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}]$. Let $h_i(x) = \log(1/p_i(x))$. Intuitively, $h_i(x)$ is the min-entropy contributed by the i'th bit of x. Note that for any $x \in \mathrm{Supp}(X)$, $\Pr[X = x] = \prod p_i(x) = 2^{-\sum_i h_i(x)}$. Since X is a δn-source, $2^{-\sum_i h_i(x)} \leq 2^{-\delta n}$, i.e. the average of $h_i(x)$ is at least δ for every $x \in \mathrm{Supp}(X)$. The intuition for the lemma is that, with high probability, the sampler will select a sequence of positions i on which this average is preserved, and hence the entropy rate will be preserved on the sample. One problem is that the sampler is guaranteed to work for $[0,1]$-valued functions, but $h_i(x)$ may be greater than 1. Thus, we truncate large values and argue that we do not lose much by doing this. Specifically, let $h'_i(x) = \min\{h_i(x), \log(1/\tau)\}$.

Here we see the main difference between our proof and the Nisan–Zuckerman proof. In their proof, they consider the set of positions i where $p_i(x) \leq 1/2$, and argue that there are usually many such positions so the sampler will hit this set many times. This can be seen as counting either 1 or 0 bits of min-entropy per position (define $h'_i(x)$ to be 1 if $h_i(x) \geq 1$ and 0 otherwise). Here, the bits of min-entropy we count per position can be fractional or greater than 1 (up to $\log(1/\tau)$). This allows us to lose less min-entropy, but forces us to use samplers for functions (rather than sets).

We argue now that truncating the h_i's does not cost us too much min-entropy. Call x *well-spread* if $\sum_i h'_i(x) \geq (\delta - 2\tau)n$ (i.e. truncating cost us at most 2τ in the entropy rate).

Claim 11 $\Pr[X$ *is not well-spread*$] \leq 2^{-\Omega(\tau n)}$.

> **Proof of claim:** Consider the random variables $\Delta_1, \ldots, \Delta_n$ defined by $\Delta_i = h_i(X) - h'_i(X)$. We need to show that $\Pr[\sum_i \Delta_i > 2\tau n] \leq 2^{-\Omega(\tau n)}$. We first bound the tails of the individual Δ_i's, conditioned on any prefix. For any $q > 0$, we have
>
> $$\begin{aligned} &\Pr[\Delta_i = q | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \\ &= \Pr[h_i(X) = q + \log(1/\tau) | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \\ &= \Pr\left[p_i(X) = 2^{-q+\log(1/\tau)} | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}\right] \\ &= \begin{cases} 2^{-q+\log(1/\tau)} & \text{if } \exists x_i \in \{0,1\} \text{ s.t. } \Pr[X_i = x_i | X_1 = x_1 \cdots X_{i-1} = x_{i-1}] = 2^{-q+\log(1/\tau)} \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$
>
> In particular, the above conditional probability is nonzero for at most one value of q, and thus, for any $q > 0$,
>
> $$\Pr[\Delta_i \geq q | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \leq \tau \cdot 2^{-q}. \tag{1}$$
>
> Now we apply the following Chernoff-type bound for random variables with exponentially vanishing tails (proven in Appendix A).
>
> **Chernoff-type Bound:** *Suppose $Z_1, \ldots, Z_n$ are independent, nonnegative random variables with probability distribution function given by* $\Pr[Z_i \geq q] = \tau \cdot 2^{-q}$ *for all* $q > 0$. *Then* $\Pr[\sum_i Z_i > 2\tau n] < 2^{-\Omega(\tau n)}$.
>
> Actually, we cannot apply this immediately because the Δ_i's are not independent. However, because Bound (1) on the tail of Δ_i holds even conditioned on $\Delta_1, \ldots, \Delta_{i-1}$, they are dominated by independent $Z_1, \ldots, Z_n$ as in the lemma. That is, we can extend the probability space to include independent $Z_1, \ldots, Z_n$ s.t. $\Delta_i \leq Z_i$ always holds. Thus $\Pr[\sum_i \Delta_i > 2\tau n] \leq \Pr[\sum_i Z_i > 2\tau n] \leq 2^{-\Omega(\tau n)}$, as desired. $\square$

Let's call a sequence $s = (i_1, \ldots, i_t) \in [n]^t$ *good* for x if

$$\frac{1}{t}\sum_j h'_{i_j}(x) \geq \delta - 3\tau,$$

and otherwise call s *bad* for x. Let $b(s) = \Pr[s \text{ is bad for } X]$. First we argue that the sampler rarely produces bad sequences.

Claim 12 $\mathrm{E}[b(S)] \leq \gamma + 2^{-\Omega(\tau n)}$.

Proof of claim: Consider any well-spread element $x \in \mathrm{Supp}(X)$. Consider the function $f : [n] \to \{0,1\}$ defined by $f(i) = h'_i(x)/\log(1/\tau)$. Because x is well-spread, f has average value at least $\mu = (\delta - 2\tau)/\log(1/\tau)$. A sequence s is good for x iff the average of f on s is at least $(\delta - 3\tau)/\log(1/\tau) = \mu - \theta$. Thus, since S comes from an (μ, θ, γ) averaging sampler, we have $\Pr[S \text{ is not good for } x] \leq \gamma$ (for any well-spread x). Now, averaging over $x \overset{\mathrm{R}}{\leftarrow} X$, we get:

$$\begin{aligned} \mathrm{E}[b(S)] &= \Pr[S \text{ is not good for } X] \\ &\leq \Pr[S \text{ is not good for } X | X \text{ well-spread}] + \Pr[X \text{ not well-spread}] \\ &\leq \gamma + 2^{-\Omega(\tau n)}. \end{aligned}$$

□

Next, we show that good sequences yield high min-entropy on the sampled bits.

Claim 13 *For every s, the random variable X_s is $b(s)$-close to a $(\delta - 3\tau)t$-source.*

Proof of claim: Fix $s = (i_1, \ldots, i_t)$. The basic idea is to "fix" the bits of X in positions outside of s, and then argue that whenever s is good for x, x_s has low probability mass. Then averaging over the fixed bits can only increase the min-entropy. However, as in the proof of [NZ96], fixing the bits of X is somewhat delicate because of the correlations between the bits of X.

We can envision X being generated by a process of the following form. The probability space consists of independent random variables $F_1, \ldots, F_n$, where F_i is distributed (arbitrarily) over functions from $\{0,1\}^{i-1} \to \{0,1\}$, and, given these functions, X is deterministically generated by setting $X_i = F_i(X_1 X_2 \cdots X_{i-1})$.

We will fix F_i for every $i \notin s$. Consider any $\overline{f} = (f_i)_{i \notin s}$, and let $X^{\overline{f}}$ denote X conditioned on $F_i = f_i$ for all $i \notin s$. Then, for any string $z \in \{0,1\}^t$ in the support of $X_s^{\overline{f}}$, there is a unique $x \in \{0,1\}^n$ in the support of $X^{\overline{f}}$ such that $x_s = z$. We denote this x by $x^{\overline{f}}(z)$. Then

$$\begin{aligned} \Pr\left[X_s^{\overline{f}} = z\right] &= \Pr\left[X^{\overline{f}} = x^{\overline{f}}(z)\right] \\ &= \prod_{i=1}^{n} \Pr\left[X_i^{\overline{f}} = x^{\overline{f}}(z)_i | X_1^{\overline{f}} = x^{\overline{f}}(z)_1, \ldots, X_{i-1}^{\overline{f}} = x^{\overline{f}}(z)_{i-1}\right] \end{aligned}$$

The conditional probability in the i'th factor above is 1 when $i \notin s$ (because then $X_i^{\overline{f}} = f_i(X_1^{\overline{f}} \cdots X_{i-1}^{\overline{f}})$ always) and equals $p_i(x^{\overline{f}}(z))$ otherwise. Thus, if s is good for $x^{\overline{f}}(z)$ (and consists of distinct samples), we have:

$$\Pr\left[X_s^{\overline{f}} = z\right] = \prod_{i \in s} p_i\left(x^{\overline{f}}(z)\right) \leq \prod_{i \in s} 2^{-h'_i(x^{\overline{f}}(z))} \leq 2^{-(\delta - 3\tau)t}.$$

Let $b(s, \overline{f})$ be the probability that s is bad for $X^{\overline{f}}$. Then, by the above, $X_s^{\overline{f}}$ is $b(s, \overline{f})$-close to some $(\delta - 3\tau)t$-source, call it $Z^{\overline{f}}$. Now consider the random variable $\overline{F} = (F_i)_{i \notin s}$.

Then $X_s = X_s^{\overline{F}}$ and $Z^{\overline{F}}$ is a convex combination of $(\delta - 3\tau)t$-sources and hence is a $(\delta - 3\tau)t$-source itself. The statistical difference between X_s and $Z^{\overline{F}}$ is at most $\mathrm{E}[b(s, \overline{F})] = b(s)$, as desired. □

Now we deduce Lemma 9 from Claims 12 and 13. By Claim 13, X_s is $b(s)$-close to some $(\delta - 3\tau)t$-source Z_s. Consider the random variable $(U_r, Z_{\mathrm{Samp}(U_r)})$ (i.e. first sample $y \overset{\mathrm{R}}{\leftarrow} U_r$, then sample $z \overset{\mathrm{R}}{\leftarrow} Z_{\mathrm{Samp}(y)}$, and output (y, z)). The statistical difference between $(U_r, Z_{\mathrm{Samp}(U_r)})$ and $(U_r, X_{\mathrm{Samp}(U_r)})$ is exactly the expected statistical difference between X_s and Z_s over $s \overset{\mathrm{R}}{\leftarrow} S$, which is at most $\mathrm{E}[b(S)] \le \gamma + 2^{-\Omega(\tau n)}$ by Claim 12. Thus, $(A, B) = (U_r, Z_{\mathrm{Samp}(U_r)})$ satisfies the requirements of Lemma 9.

5 Non-Explicit Constructions

In this section, we describe the locally computable extractors obtained by using really optimal extractors and samplers in Theorem 10. This does not give efficient constructions of locally computable extractors, because optimal extractors and samplers are only known by nonconstructive applications of the Probabilistic Method. However, it shows what Theorem 10 will yield as one discovers constructions which approach the optimal bounds. In fact, the explicit constructions known are already quite close, and (as we will see in Section 6) match the optimal bounds within constant factors for the range of parameters most relevant to the bounded-storage model.

5.1 The Extractor

The Probabilistic Method yields extractors with the following expressions for the seed length d and output length m, both of which are tight up to additive constants [RT97].

Lemma 14 (nonconstructive extractors (cf., [RT97])) *For every n, $k \le n$, $\varepsilon > 0$, there exists a strong (k, ε)-extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = \log(n-k) + 2\log(1/\varepsilon) + O(1)$, $m = k - 2\log(1/\varepsilon) - O(1)$.*

5.2 The Sampler

Similarly, the following lemma states the averaging samplers implied by the Probabilistic Method. There are matching lower bounds for both the randomness complexity and the sample complexity [CEG95] (except for the dependence on μ, which was not considered there). The proof of the lemma follows the argument implicit in [Zuc97], with the modifications that it makes the dependence on μ explicit and guarantees distinct samples.

Lemma 15 (nonconstructive samplers) *For every $n \in \mathbb{N}$, $1 > \mu > \theta > 0$, $\gamma > 0$, there is a (μ, θ, γ) averaging sampler $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ that uses*

- *t distinct samples for any $t \in \left[O\left(\frac{\mu}{\theta^2} \cdot \log \frac{1}{\gamma}\right), n\right]$*
- *$r = \log(n/t) + \log(1/\gamma) + 2\log(\mu/\theta) + \mathrm{loglog}(1/\mu) + O(1)$ random bits.*

Proof: Let $R = 2^r$ so we can associate $[R] = \{0,1\}^r$, and for simplicity assume $n = t \cdot n_0$ for an integer n_0 so we can associate $[n] = [t] \times [n_0]$. We will consider a randomly chosen function $\mathrm{Samp}_0 : [R] \to [m]^t$ and set $\mathrm{Samp}(x)_j = (j, \mathrm{Samp}_0(x))$. (This guarantees distinct samples.)

Instead of directly proving that Samp is a sampler, we will first prove that it has the following property with high probability.

Claim 16 Samp *has the following property with probability at least* $1/2$*. For every subset* $S \subset [R]$ *of size* γR*, and every* boolean $f : [n] \to \{0,1\}$ *with average value* μ*,*

$$\Pr_{x \overset{R}{\leftarrow} S, j \overset{R}{\leftarrow} [t]} [f(\mathrm{Samp}(x)_j) = 1] \geq \mu - \theta$$

Proof of claim: Consider a fixed S and boolean f as in the claim. For every $x \in S$ and $j \in [t]$, define the random variable $X_{x,j} = f(\mathrm{Samp}(x)_j)$ (over the choice of Samp). These are $|S| \cdot t$ independent random variables. Let X be the average of the $X_{x,j}$'s. The expectation of X is the average value of f, which equals μ. We wish to show that $X \geq \mu - \theta$ with very high probability.

By a Chernoff Bound (e.g. [ASE92, Thm A.13]),

$$\Pr_{\mathrm{Samp}} [X < \mu - \theta] \leq \exp\left(-\Omega(|S| \cdot t \cdot \theta^2/\mu)\right)$$

Thus, taking a union bound over f and S, the probability that Samp fails to satisfy the claim is at most

$$\begin{aligned} & \exp\left(-\Omega(|S| \cdot t \cdot \theta^2/\mu)\right) \cdot \binom{R}{\gamma R} \cdot \binom{n}{\mu n} \\ \leq \;& \exp\left(-\Omega(|S| \cdot t \cdot \theta^2/\mu)\right) \cdot \left(\frac{eR}{\gamma R}\right)^{\gamma R} \cdot \left(\frac{en}{\mu n}\right)^{\mu n} \\ \leq \;& \exp\left(-\Omega(\gamma R \cdot t \cdot \theta^2/\mu)\right) \cdot \exp(\log(1/\gamma) \cdot \gamma R) \cdot \exp(\log(1/\mu) \cdot \mu n) \end{aligned}$$

This probability is at most 1/2 if the following two conditions hold for a sufficiently large constant c:

$$\gamma \cdot R \cdot t \cdot \theta^2/\mu \geq c \cdot \gamma \cdot \log(1/\gamma) \cdot R,$$

and

$$\gamma \cdot R \cdot t \cdot \theta^2/\mu \geq c \cdot \mu \cdot \log(1/\mu) \cdot n.$$

The first condition is $t \geq c \cdot (\mu/\theta^2) \cdot \log(1/\gamma)$, which is guaranteed by the hypothesis of the lemma. The second condition says $R \geq c \cdot (n/t) \cdot (1/\gamma) \cdot (\mu/\theta)^2 \cdot \log(1/\mu)$. Taking logs, we obtain exactly the condition on r in the hypothesis of the lemma. $\square$

We now argue that any Samp satisfying Claim 16 is a (μ, θ, γ) averaging sampler. Suppose not. Then there is a (not necessarily boolean) function $f : [n] \to [0,1]$ with average value μ such that

$$\Pr_{x \overset{\mathrm{R}}{\leftarrow} U_r} \left[\frac{1}{t} \sum_{j=1}^{t} f(\mathrm{Samp}(x)_j) < \mu - \theta \right] > \gamma$$

That is, there is a set S of γR seeds $x \in [R]$ such that $\frac{1}{t}\sum_{j=1}^{t} f(\mathrm{Samp}(x)_j) < \mu - \theta$. In particular, we have:

$$\Pr_{x \overset{\mathrm{R}}{\leftarrow} S, j \overset{\mathrm{R}}{\leftarrow} [t]} [f(\mathrm{Samp}(x)_j) = 1] < \mu - \theta \tag{2}$$

This would contradict Claim 16, except that f is not necessarily boolean. However, every $[0,1]$-valued function with average value μ is a convex combination of boolean functions with average value μ.[9] Thus, the function f in Inequality (2) can be viewed as a distribution over boolean functions. By averaging, there is a boolean function violating Inequality (2), in contradiction to Claim 16. ■

5.3 The Local Extractor

Plugging the above two lemmas into Theorem 10, we obtain the following local extractors.

Theorem 17 (nonconstructive local extractors) *For every $n \in \mathbb{N}$, $\delta > 0$, $\varepsilon > 0$, and $m \leq \delta n/2 - 2\log(1/\varepsilon) - O(1)$, there is a t-local strong $(\delta n, \varepsilon)$ extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with*

- $d = \log n + 3\log(1/\varepsilon) + \mathrm{loglog}(1/\delta) + O(1)$.
- $t = O\left(\frac{m + \log(1/\delta)\cdot\log(1/\varepsilon)}{\delta}\right)$.

Proof: Set $\tau = \delta/6$, $\mu = \delta/\log(1/\tau)$, and $\theta = \tau/\log(1/\tau)$. By Lemma 15, there is a $(\mu, \theta, \varepsilon/2)$ averaging sampler $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ with distinct samples, using

$$r = \log(n/t) + \log(1/\gamma) + 2\log(\mu/\theta) + \mathrm{loglog}(1/\mu) + O(1) = \log n - \log t + \log(1/\varepsilon) + \mathrm{loglog}(1/\delta) + O(1)$$

random bits, provided $t \geq t_0$ for $t_0 = O\left((\theta^2/\mu)\cdot\log(1/\gamma)\right) = O\left(\log(1/\delta)\cdot\log(1/\varepsilon)/\delta\right)$, which is satisfied by the above setting for t.

By Lemma 14, there is a strong $(\delta t/2, \varepsilon/2)$ extractor $\mathrm{Ext} : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}^m$ with $d = \log t + 2\log(1/\varepsilon) + O(1)$, provided $m \leq \delta t/2 - 2\log(1/\varepsilon) - O(1)$, which is satisfied by the above setting for t. Noting that $(\delta - 3\tau)t = \delta t/2$, we combine these via Theorem 10 to obtain a t-local extractor with seed length $r + d = \log n + 3\log(1/\varepsilon) + \mathrm{loglog}(1/\delta) + O(1)$. ■

6 Explicit Constructions

In the previous section, we showed that very good locally computable extractors exist, but for applications we need *explicit* constructions. For an extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ or a sampler $\mathrm{Samp} : \{0,1\}^r \to [n]^t$, explicit means that it is computable in polynomial time and polylogarithmic work-space with respect to their input+output lengths (i.e., $n + d + m$ for an extractor and $r + t\log n$ for a sampler). For a t-local extractor $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, we give it oracle access to its first input and view the input length as $\log n + t + d$ ($\log n$ to specify the length of the oracle, and t as the number of bits actually read from it).

[9] This is equivalent to the fact that every distribution of min-entropy k is a convex combination of flat distributions of min-entropy k. Alternatively, it can be proven directly by calculating the vertices of the parallelopiped consisting of all $[0,1]$-valued functions with average value μ.

There are a many explicit constructions of averaging samplers and extractors in the literature and thus a variety of local extractors can be obtained by plugging these into Theorem 10. We do not attempt to describe all possible combinations here, but rather describe a few that seem particularly relevant to cryptography in the bounded storage model. We recall the following features of this application:

- The local extractor should work for sources of min-entropy $(\alpha-\beta)n-\log(1/\varepsilon)$, which is $\Omega(n)$ for most natural settings of the parameters. (Recall that α is the entropy rate of the random source and β is the storage rate of the adversary.) That is, we can concentrate on constant min-entropy rate.
- Optimizing the number of bits read from the source (to within, say, a constant factor) seems to be at least as important as the seed-length of the extractor.
- The error ε of the extractor will typically be very small, as this corresponds to the "security" of the scheme.
- We are not concerned with extracting all of the entropy from the source, since we anyhow will only be reading a small fraction of the source.

6.1 The Extractor

With the above criteria in mind, the most natural extractor to use (in Theorem 10) is Zuckerman's extractor for constant entropy rate [Zuc97]:

Lemma 18 ([Zuc97]) *For every constant $\delta > 0$, every $n \in \mathbb{N}$, and every $\varepsilon > \exp(-n/2^{O(\log^* n)})$, there is an explicit strong $(\delta n, \varepsilon)$-extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with $d = O(\log n + \log(1/\varepsilon))$ and $m = \delta n/2$.*

6.2 The Sampler

For the averaging sampler, the well-known sampler based on random walks on expander graphs provides good parameters for this application. Indeed, its randomness and sample complexities are both optimal within a constant factor when μ and θ are constant (and the minimal sample complexity is used). However, we cannot apply it directly because it does not guarantee distinct samples, and we do not necessarily want to minimize the number of samples. Thus we introduce some new techniques to deal with these issues.

The following gives a modification of the expander sampler which guarantees distinct samples.

Lemma 19 (modified expander sampler) *For every $0 < \theta < \mu < 1$, $\gamma > 0$, and $n \in \mathbb{N}$, there is an explicit (μ, θ, γ) averaging sampler* $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ *that uses*

- *t distinct samples for any $t \in \left[O\left(\frac{1}{\theta^2} \cdot \log(1/\gamma)\right), n\right]$, and*
- *$r = \log n + O(t \cdot \log(1/\theta))$ random bits.*

Proof: Consider an explicit d-regular expander graph G on n vertices.[10] Let M be the adjacency matrix of G divided by d, i.e. the stochastic matrix which describes the random walk on G. M

[10] Actually, we do not have explicit constructions of expander graphs for every n. Instead, we use an expander with $n' \in [n, (1+\theta)n]$ vertices, and extend f by setting $f(i) = 0$ for all $i \in [n'] \setminus [n]$.

has an eigenvalue of 1, and we require that all its other eigenvalues have absolute value less than $\lambda = \theta/16$. This is possible to achieve with degree $d = \mathrm{poly}(1/\theta)$ (by taking an appropriate power of any constant-degree expander with bounded second eigenvalue).

The standard expander sampler outputs a random walk $w = (w_1, \ldots, w_t)$ on G of length t started at a random vertex w_1. For any function $f : [n] \to [0,1]$, Gillman's Chernoff bound for random walks on expanders [Gil98] says:

$$\Pr_w \left[\left| \frac{1}{t} \sum_{i=1}^{t} f(w_i) - \mu \right| > \theta/2 \right] < \exp\left(-\Omega(\theta^2 t)\right) < \gamma/4,$$

for $t \geq O(\log(1/\gamma)/\theta^2)$.

This sampler uses $r_0 = \log n + t \cdot \log d = \log n + O(t \cdot \log(1/\theta))$ random bits, as desired. However, it does not guarantee distinct samples. We now show that discarding the repeated vertices does not affect the sampler too much. For a walk $(w_1, \ldots, w_t)$, call w_i a *repeat* if there exists a $j < i$ such that $w_j = w_i$, and *new* otherwise. We modify our sampler to use the first $(1 - (\theta/2))t$ new vertices in the walk, and output `fail` if there are not this many new vertices. If failure does not occur, this sampler uses distinct samples and the approximation is only affected by an additive error of at most $\theta/2$. We will describe what to do in case of failures later.

We now bound the failure probability. For a fixed $j < i$, the probability over a random walk w that $w_i = w_j$ is exactly the trace of M^{i-j} divided by n, which equals the sum of the eigenvalues of M^{i-j}. We use this to bound the probability that w_i is a repeat as follows.

$$\Pr_w[\exists j < i, w_j = w_j] \leq \sum_{k=1}^{i-1} \frac{1}{n} \mathrm{Tr}(M^k) \leq \sum_{k=1}^{i-1} \frac{(1 + (n-1)\lambda^k)}{n} \leq \frac{t}{n} + 2\lambda \leq \theta/4,$$

where the last inequality assumes $t \leq \theta n/8$ (otherwise $\log \binom{n}{t} = O(t \cdot \log(n/t)) = O(t \cdot \log(1/\theta)) < r$, so we have enough randomness to sample a uniformly random subset of $[n]$ of size t). Thus, the expected fraction of vertices in a random walk that are repeats is at most $\theta/4$. By Markov's inequality, the probability that there are more than $\theta/2$ repeats is at most $1/2$.

So our modified sampler outputs `fail` with probability at most $1/2$, and conditioned on non-failure, outputs a sample with error greater than θ with probability at most $(\gamma/4)/(1/2) = \gamma/2$. Thus our task is reduced to sampling from the non-failing subset of our sample space with high probability $(1 - \gamma/2)$. This can be done by using a random walk of length $O(\log(1/\gamma))$ on an expander on 2^{r_0} vertices, for a total of $r = r_0 + O(\log(1/\gamma))$ random bits. (When this sampler doesn't succeed in finding a non-failing expander walk, it outputs an arbitrary t-subset, e.g. $(1, \ldots, t)$.) ■

A drawback of the expander sampler is that the randomness increases with the number of samples, whereas in the optimal sampler of Lemma 15 the randomness actually decreases with the number of samples. To fix this, we use the following lemma, which shows that the number of (distinct) samples can be increased at no cost.

Lemma 20 *Suppose there is an explicit (μ, θ, γ) averaging sampler* Samp : $\{0,1\}^r \to [n]^t$ *with distinct samples. Then for every $m \in \mathbb{N}$, there is an explicit (μ, θ, γ) averaging sampler* Samp$'$: $\{0,1\}^r \to [m \cdot n]^{m \cdot t}$.

In fact, there is a gain as the sample complexity increases, because the randomness complexity depends only on the original value of n, rather than $n' = m \cdot n$.

Proof: We associate $[m \cdot n]$ and $[m \cdot t]$ with $[m] \times [n]$ and $[m] \times [t]$, respectively, and define $\mathrm{Samp}'(x)_{(i,j)} = (i, \mathrm{Samp}(x)_j)$. For any function $f' : [m] \times [n] \to [0,1]$, define $f : [n] \to [0,1]$ by setting $f(z)$ to be the average of $f'(i,z)$ over $i \in [m]$. The average value of f is the same as the average value of f', and for any coin tosses x, if $\mathrm{Samp}(x)$ successfully approximates the average of f, then $\mathrm{Samp}'(x)$ also succeeds for f'. ■

To apply this lemma to construct a sampler with given values of n, t, μ, θ, and γ, it is best to start with a sampler $\mathrm{Samp}_0 : \{0,1\}^{r_0} \to [n_0]^{t_0}$ using the minimal sample complexity $t_0 = t_0(\theta, \mu, \gamma) < t$ and domain size $n_0 = n \cdot (t_0/t)$. For example, for constant μ and θ, the sampler of Lemma 19 will give $t_0 = O(\log(1/\gamma))$ and $r_0 = \log n_0 + O(\log(1/\gamma)) = \log(n/t) + O(\log(1/\gamma))$. Then setting $m = t/t_0$, Lemma 20 gives a sampler for domain size $n_0 \cdot m = n$, using $t_0 \cdot m = t$ distinct samples, and r_0 random bits. This is how we obtain our final sampler, stated in the next lemma.

Lemma 21 *For every $n \in \mathbb{N}$, $1 > \mu > \theta > 0$, $\gamma > 0$, there is a (μ, θ, γ) averaging sampler* $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ *that uses*

- *t distinct samples for any $t \in \left[O\left(\frac{1}{\theta^2} \cdot \log \frac{1}{\gamma}\right), n\right]$*
- *$r = \log(n/t) + \log(1/\gamma) \cdot \mathrm{poly}(1/\theta)$ random bits.*

Unfortunately, when t/t_0 and $n \cdot (t_0/t)$ are not integers, some care is needed to deal with the rounding issues in the argument given above. The tedious details follow.

Proof: Let $h = h(\theta, \gamma) = O(\log(1/\gamma)/\theta^2)$ be the lower bound on t in Lemma 19. Given $n \geq t \geq h$, we can find $t_0 \in [h, 2h]$ and m such that $m \cdot t_0 \leq t \leq m \cdot t_0 + \min\{m, t_0\}$. In particular, this implies that $m \cdot t_0 \in [t - \sqrt{t}, t] \subseteq [(1-\theta)t, t]$. Now, we can find n_0 such that $m \cdot n_0 \geq n \geq m \cdot n_0 - m$, so $m \cdot n_0 \in [n, n+m] \subseteq [n, (1+\theta)n]$. By Lemma 19, there is a (μ, θ, γ) sampler $\mathrm{Samp}_0 : \{0,1\}^r \to [n_0]^{t_0}$ with

$$r = \log n_0 + O(t_0 \cdot \log(1/\theta)) = \log n + \log(1/\gamma) \cdot \mathrm{poly}(1/\theta) - \log t$$

(using the bound $n_0 \leq nt_0/t + 1$).

Applying Lemma 20, we get a sampler $\mathrm{Samp} : \{0,1\}^r \to [m \cdot n_0]^{m \cdot t_0}$. Since both $m \cdot t_0$ approximates t and $m \cdot n_0$ approximates n within relative errors of θ, we can view this sampler as a $(\mu + \theta, 2\theta, \gamma)$ sampler $\mathrm{Samp}' : \{0,1\}^r \to [n]^t$. ■

6.3 The Local Extractor

Analogously to Theorem 17, we plug Lemmas 18 and 21 into Theorem 10 to obtain:

Theorem 22 (explicit local extractors) *For every constant $\delta > 0$, $n \in \mathbb{N}$, $\varepsilon > \exp(-n/2^{O(\log^* n)})$, $m \leq \delta n/4$, there is an explicit t-local strong $(\delta n, \varepsilon)$ extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with*

- *$d = O(\log n + \log(1/\varepsilon))$.*
- *$t = O(m + \log(1/\varepsilon))$.*

Proof: Set $\tau = \delta/6$, $\mu = \delta/\log(1/\tau)$, and $\theta = \tau/\log(1/\tau)$. When δ is constant, so are τ, μ, and θ. By Lemma 21, there is a $(\mu, \theta, \varepsilon/2)$ averaging sampler $\mathrm{Samp} : \{0,1\}^r \to [n]^t$ with distinct samples, using $r = \log(n/t) + O(\log(1/\varepsilon))$ random bits, provided $t \geq t_0$ for $t_0 = O(\log(1/\varepsilon))$, which is satisfied by the above setting.

By Lemma 14, there is a strong $(\delta t/2, \varepsilon/2)$ extractor $\mathrm{Ext} : \{0,1\}^t \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log t + \log(1/\varepsilon)) \leq O(\log n + \log(1/\varepsilon))$, provided $m \leq \delta t/2 - 2\log(1/\varepsilon) - O(1)$, which is satisfied by the above setting. Noting that $(\delta - 3\tau)t = \delta t/2$, we combine these via Theorem 10 to obtain a t-local extractor with seed length $r + d = \log n + 3\log(1/\varepsilon) + \mathrm{loglog}(1/\delta) + O(1)$. ■

The above construction does generalize to the case of subconstant δ. The expression for t becomes actually $t = O(m/\delta + \log(1/\varepsilon)/\delta^2)$, which is not too bad compared with non-explicit construction of Theorem 17. The seed length d becomes $d = O((\log n + \log(1/\varepsilon)) \cdot \mathrm{poly}(1/\delta))$, but here the multiplicative dependence on $\mathrm{poly}(1/\delta)$ is much worse than the additive dependence on $\mathrm{loglog}(1/\delta)$ in Theorem 17. This is due to both underlying components — the extractor (Lemma 18) and the averaging sampler (Lemma 21). The dependence on δ in the extractor component can be made logarithmic by using one of the many known explicit extractors for subconstant min-entropy rate. For the averaging sampler, too, there are constructions whose randomness complexity is within a constant factor of optimal [Zuc97].[11] However, these constructions have a sample complexity that is only polynomial in the optimal bound, resulting in a t-local extractor with $t = \mathrm{poly}(\log(1/\gamma), 1/\delta)$.[12] It is an interesting open problem, posed in [Gol97], to construct averaging samplers whose sample and randomness complexities are both within a constant factor of optimal. (Without the "averaging" constraint, there are samplers which achieve this [BGG93, GW97, Gol97].)

6.4 Previous Constructions.

Some previous constructions of cryptosystems in the bounded storage model can be understood using our approach, namely Theorem 10 together with Theorem 5 (of [Lu02]). For example, the cryptosystem of Cachin and Maurer [CM97] amounts to using pairwise independence for both the averaging sampler and the extractor. (The fact that pairwise independence yields a sampler follows from Chebychev's Inequality [CG89], and that it yields an extractor is the Leftover Hash Lemma of [HILL99].) Actually, in the description in [CM97], the seed for the extractor is chosen at the time of encryption and sent in an additional interactive step. But it follows from this analysis that it actually can be incorporated in the secret key, so interaction is not necessary.

Our approach also yields an alternative proof of security for the ADR cryptosystem [AR99, ADR02]. Consider the sampler which simply chooses a random t-subset of $[n]$ for $t = O(\log(1/\varepsilon))$ and the extractor $\mathrm{Ext} : \{0,1\}^t \times \{0,1\}^t \to \{0,1\}$ defined by $\mathrm{Ext}(x, r) = x \cdot r \bmod 2$. The correctness of the sampler follows from Chernoff-type bounds, and the correctness of the extractor from the Leftover Hash Lemma [HILL99]. Combining these via Theorem 10 yields a locally computable extractor which simply outputs the parity of a random subset of $O(\log(1/\varepsilon))$ bits from the source. This is essentially the same as the ADR cryptosystem, except that the size of the subset is chosen

[11] The averaging samplers of [Zuc97] can be made to have distinct samples: he shows that taking any (not necessarily strong) extractor Ext, defining $\mathrm{Samp}(x)_y = \mathrm{Ext}(x, y)$ yields an averaging sampler whose parameters are related to those of the extractor. Thus, if Ext is a strong extractor, then $\mathrm{Samp}(x)_y = y \circ \mathrm{Ext}(x, y)$ is an averaging sampler with distinct samples.

[12] To obtain this bound rather than $t = \mathrm{poly}(\log(1/\gamma), 1/\delta, \log n)$ as claimed in [Zuc97], the averaging samplers of [RVW00] should be used; these are also obtained by applying Zuckerman's transformation to appropriate extractors.

according to a binomial distribution rather than fixed. However, the security of the original ADR cryptosystem follows, because subsets of size exactly $t/2$ are a nonnegligible fraction ($\Omega(1/\sqrt{t})$) of the binomial distribution. To extract m bits, one can apply this extractor m times with independent seeds, as done in [ADR02].

Acknowledgments

Noga Alon, Omer Reingold, and Amnon Ta-Shma also thought of similar approaches to this problem, and I am grateful to them for providing me with several helpful suggestions. I am also grateful to Yan Zong Ding and Chi-Jen Lu for pointing out errors in an earlier version of this paper. I thank Oded Goldreich for his encouragement and many helpful comments on the presentation. I thank Yan Zong Ding, Dick Lipton, and Michael Rabin for a number of illuminating discussions about the bounded-storage model.

References

[ASE92] Noga Alon, Joel H. Spencer, and Paul Erdös. *The Probabilistic Method.* Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc., 1992.

[ADR02] Yonatan Aumann, Yan Zong Ding, and Michael O. Rabin. Everlasting security in the bounded storage model. *IEEE Transactions on Information Theory*, 48(6):1668–1680, June 2002.

[AR99] Yonatan Aumann and Michael O. Rabin. Information theoretically secure communication in the limited storage space model. In *Advances in Cryptology—CRYPTO '99*, Lecture Notes in Computer Science, pages 65–79. Springer-Verlag, 1999, 15–19 August 1999.

[BRST02] Ziv Bar-Yossef, Omer Reingold, Ronen Shaltiel, and Luca Trevisan. Streaming computation of combinatorial objects. In *Proceedings of the Seventeenth Annual IEEE Conference on Computational Complexity*, pages 165–174, Montreal, Canada, 21–24 May 2002. IEEE Computer Society Press.

[BGG93] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. Randomness in interactive proofs. *Computational Complexity*, 3(4):319–354, 1993.

[BR94] Mihir Bellare and John Rompel. Randomness-efficient oblivious sampling. In *35th Annual Symposium on Foundations of Computer Science*, pages 276–287, Santa Fe, New Mexico, 20–22 November 1994. IEEE.

[CM97] Christian Cachin and Ueli Maurer. Unconditional security against memory-bounded adversaries. In Burton S. Kaliski Jr., editor, *Advances in Cryptology — CRYPTO '97*, volume 1294 of *Lecture Notes in Computer Science*, pages 292–306. Springer-Verlag, 1997.

[CEG95] Ran Canetti, Guy Even, and Oded Goldreich. Lower bounds for sampling algorithms for estimating the average. *Information Processing Letters*, 53(1):17–25, 13 January 1995.

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

[CG89] Benny Chor and Oded Goldreich. On the power of two-point based sampling. *Journal of Complexity*, 5(1):96–106, 1989.

[DR02] Yan Zong Ding and Michael O. Rabin. Hyper-encryption and everlasting security (extended abstract). In *STACS 2002 — 19th Annual Symposium on Theoretical Aspects of Computer Science*, Lecture Notes in Computer Science, pages 1–26. Springer-Verlag, 14–16 March 2002.

[DM02] Stefan Dziembowski and Ueli Maurer. Tight security proofs for the bounded-storage model. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 341–350, Montreal, 19–21 May 2002.

[Gil98] David Gillman. A Chernoff bound for random walks on expander graphs. *SIAM Journal on Computing*, 27(4):1203–1220 (electronic), 1998.

[Gol97] Oded Goldreich. A sample of samplers: A computational perspective on sampling. Technical Report TR97-020, Electronic Colloquium on Computational Complexity, May 1997.

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

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396 (electronic), 1999.

[Lu02] Chi-Jen Lu. Hyper-encryption against space-bounded adversaries from on-line strong extractors. In *Advances in Cryptology—CRYPTO '02*, Lecture Notes in Computer Science. Springer-Verlag, 2002, 18–22 August 2002. To appear.

[Mau92] Ueli Maurer. Conditionally-perfect secrecy and a provably-secure randomized cipher. *Journal of Cryptology*, 5(1):53–66, 1992.

[Mau02] Ueli Maurer. Personal communication, August 2002.

[NT99] Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 58(1):148–173, 1999.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, February 1996.

[Rab01] Michael O. Rabin. Personal communication, December 2001.

[RT97] Jaikumar Radhakrishnan and Amnon Ta-Shma. Tight bounds for depth-two superconcentrators. In *38th Annual Symposium on Foundations of Computer Science*, pages 585–594, Miami Beach, Florida, 20–22 October 1997. IEEE.

[RVW00] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *Proceedings of 41st Annual Symposium on Foundations of Computer Science*, pages 3–13, 2000.

[RSW00] Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. In *41st Annual Symposium on Foundations of Computer Science*, Redondo Beach, California, 12–14 November 2000.

[Sha02] Ronen Shaltiel. Recent developments in explicit constructions of extractors. *Bulletin of the European Association for Theoretical Computer Science*, 77:67–95, June 2002.

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

[Zuc96] David Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4/5):367–391, October/November 1996.

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

A A Chernoff Bound

Here we proof the Chernoff-type bound needed in the proof of Lemma 9.

Lemma 23 *Suppose $Z_1, \ldots, Z_n$ are independent, nonnegative random variables with probability distribution function given by* $\Pr[Z_i \geq q] = \tau \cdot 2^{-q}$ *for all* $q > 0$.[13] *Then* $\Pr\left[\sum_i Z_i > 2\tau n\right] < 2^{-\Omega(\tau n)}$.

Proof: Let $f(q) = \tau \cdot 2^{-q}$ be the distribution function of the Z_i's (for $q > 0$). Note that $\Pr[Z_i > 0] = \lim_{q \to 0} f(q) = \tau$. As usual in the proofs of Chernoff bounds, we consider the expectation of the exponential generating function $2^{tZ} = \prod_{i=1}^{n} 2^{tZ_i}$. For every $t \in (0,1)$, we have:

$$\begin{aligned}
\mathrm{E}\left[2^{tZ_i}\right] &= \Pr[Z_i = 0] \cdot 2^0 + \int_{q=0}^{\infty} \left(-\frac{d}{dq} f(q)\right) \cdot 2^{tq} dq \\
&= (1-\tau) + \int_{q=0}^{\infty} (\tau \ln 2 \cdot 2^{-q}) \cdot 2^{tq}) dq \\
&= (1-\tau) + \left[\frac{-\tau}{1-t} \cdot 2^{-(1-t)q}\right]_{q=0}^{\infty} \\
&= 1 + \frac{\tau t}{1-t}. \\
&\leq e^{\tau t/(1-t)}
\end{aligned}$$

Thus,

$$\mathrm{E}[2^{tZ}] = \prod_i \mathrm{E}[2^{tZ_i}] \leq \left(e^{\tau t/(1-t)}\right)^n.$$

[13]Note that the distribution function is discontinuous at $q = 0$ because $\Pr[Z_i \geq 0] = 1$.

By Markov's Inequality,

$$\Pr\left[Z \geq 2\tau n\right] = \Pr\left[2^{tZ} \geq 2^{2\tau t n}\right] \leq \frac{e^{\tau t n/(1-t)}}{2^{2\tau t n}} = \left(\frac{e^{t/(1-t)}}{2^{2t}}\right)^{\tau n},$$

for every $t \in (0,1)$. For $t = 1/4$, $e^{t/(1-t)} < 2^{2t}$, so this probability is indeed $2^{-\Omega(\tau n)}$, as desired. ■