On Pseudorandom Generators with Linear Stretch in NC^0 *

Benny Applebaum Yuval Ishai Eyal Kushilevitz

Computer Science Department, Technion
{abenny,yuvali,eyalk}@cs.technion.ac.il

April 16, 2007

Abstract

We consider the question of constructing cryptographic pseudorandom generators (PRGs) in NC^0, namely ones in which each bit of the output depends on just a constant number of input bits. Previous constructions of such PRGs were limited to stretching a seed of n bits to $n + o(n)$ bits. This leaves open the existence of a PRG with a linear (let alone superlinear) stretch in NC^0. In this work we study this question and obtain the following main results:

1. We show that the existence of a linear-stretch PRG in NC^0 implies non-trivial hardness of approximation results *without relying on PCP machinery*. In particular, it implies that Max3SAT is hard to approximate to within some multiplicative constant.
2. We construct a linear-stretch PRG in NC^0 under a specific intractability assumption related to the hardness of decoding "sparsely generated" linear codes. Such an assumption was previously conjectured by Alekhnovich (FOCS 2003).

*Preliminary version of this work appeared in the Proceedings of the 10th International Workshop on Randomization and Computation (RANDOM 2006). Research supported by grants 1310/06 and 36/03 from the Israel Science Foundation.

1 Introduction

A cryptographic pseudorandom generator (PRG) [10, 33] is a deterministic function that stretches a short random seed into a longer string that cannot be distinguished from random by any polynomial-time observer. In this work, we study the existence of PRGs that both (1) admit fast parallel computation and (2) stretch their seed by a significant amount.

Considering the first goal alone, it was recently shown in [4] that the ultimate level of parallelism can be achieved under most standard cryptographic assumptions. Specifically, any PRG in NC^1 (the existence of which follows, for example, from the intractability of factoring) can be efficiently "compiled" into a PRG in NC^0, namely one where each output bit depends on just a constant number of input bits. However, the PRGs produced by this compiler only stretch their seed by a sublinear amount: from n bits to $n + O(n^{\varepsilon})$ bits for some constant $\varepsilon < 1$. Thus, these PRGs do not meet our second goal.

Considering the second goal alone, even a PRG that stretches its seed by just one bit can be used to construct a PRG that stretches its seed by any polynomial number of bits [16, Sec. 3.3.2]. However, all known constructions of this type are inherently sequential. Thus, we cannot use known techniques for turning an NC^0 PRG with a sublinear stretch into one with a linear, let alone superlinear, stretch.

The above state of affairs leaves open the existence of a *linear-stretch* PRG (LPRG) in NC^0; namely, one that stretches a seed of n bits into $n + \Omega(n)$ output bits.[1] (In fact, there was no previous evidence for the existence of LPRGs even in the higher complexity class AC^0.) This question is the main focus of our work. The question has a very natural motivation from a cryptographic point of view. Indeed, most cryptographic applications of PRGs either require a linear stretch (for example Naor's bit commitment scheme [25]), or alternatively depend on a larger stretch for efficiency (this is the case for the standard construction of a stream cipher or stateful symmetric encryption from a PRG, see [18]). Thus, the existence of an LPRG in NC^0 would imply better parallel implementations of other cryptographic primitives.

1.1 Our Contribution

LPRG in NC^0 implies hardness of approximation. We give a very different, and somewhat unexpected, motivation for the foregoing question. We observe that the existence of an LPRG in NC^0 *directly* implies non-trivial and useful hardness

[1] Note that an NC^0 LPRG can be composed with itself a constant number of times to yield an NC^0 PRG with an arbitrary linear stretch.

of approximation results. Specifically, we show (via a simple argument) that an LPRG in NC^0 implies that Max3SAT (and hence all MaxSNP problems such as Max-Cut, Max2SAT and Vertex Cover [27]) cannot be efficiently approximated to within some multiplicative constant. This continues a recent line of work, initiated by Feige [14] and followed by Alekhnovich [1], that provides simpler alternatives to the traditional PCP-based approach by relying on stronger assumptions. Unlike these previous works, which rely on very specific assumptions, our assumption is of a more general flavor and may serve to further motivate the study of cryptography in NC^0. On the down side, the conclusions we get are weaker and in particular are implied by the PCP theorem. In contrast, some inapproximability results from [14, 1] could not be obtained using PCP machinery. It is instructive to note that by applying our general argument to the sublinear-stretch PRGs in NC^0 from [4] we only get "uninteresting" inapproximability results that follow from standard padding arguments (assuming P$\neq$NP). Furthermore, we do not know how to obtain stronger inapproximability results based on a superlinear-stretch PRG in NC^0. Thus, our main question of constructing LPRGs in NC^0 captures precisely what is needed for this application.

Constructing an LPRG in NC^0. We present a construction of an LPRG in NC^0 under a specific intractability assumption related to the hardness of decoding "sparsely generated" linear codes. Such an assumption was previously made by Alekhnovich in [1]. The starting point of our construction is a modified version of a PRG from [1] that has a large output locality (that is, each output bit depends on many input bits) but has a simple structure. We note that the output distribution of this generator can be sampled in NC^0; however the seed length of this NC^0 sampling procedure is too large to gain any stretch. To solve this problem we observe that the seed has large entropy even when the output of the generator is given. Hence, we can regain the stretch by employing a randomness extractor in NC^0 that uses a "sufficiently short" seed to extract randomness from sources with a "sufficiently high" entropy. We construct the latter by combining the known construction of randomness extractors from ε-biased generators [24, 8] with previous constructions of ε-biased generator in NC^0 [23]. Our LPRG can be implemented with locality 4; the stretch of this LPRG is essentially optimal, as it is known that no PRG with locality 4 can have a *superlinear* stretch [23]. However, the existence of superlinear-stretch PRG with possibly higher (but constant) locality remains open.

By combining the two main results described above, one gets non-trivial inapproximability results under the intractability assumption from [1]. These (and stronger) results were *directly* obtained in [1] from the same assumption *without* constructing an LPRG in NC^0. Our hope is that future work will yield constructions of LPRGs in NC^0 under different, perhaps more standard, assumptions, and

that the implications to hardness of approximation will be strengthened.

LPRG in NC^0 and Expanders. Finally, we observe that the input-output graph of any LPRG in NC^0 enjoys some non-trivial expansion property. This connection implies that a (deterministic) construction of an LPRG in NC^0 must use some non-trivial combinatorial objects. (In particular, one cannot hope that "simple" transformations, such as those given in [4], will yield LPRGs in NC^0.) The connection with expanders also allows to rule out the existence of *exponentially*-strong PRGs with *superlinear* stretch in NC^0.

1.2 Related Work

The existence of PRGs in NC^0 has been recently studied in [12, 23, 4]. Cryan and Miltersen [12] observe that there is no PRG in NC^0_2 (i.e., where each output bit depends on at most two input bits), and prove that there is no PRG in NC^0_3 achieving a superlinear stretch; namely, one that stretches n bits to $n + \omega(n)$ bits. Mossel et al. [23] extend this impossibility to NC^0_4. Viola [32] shows that an LPRG in AC^0 cannot be obtained from a OWF via non-adaptive black-box constructions. This result can be extended to rule out such a construction even if we start with a PRG whose stretch is sublinear.

On the positive side, Mossel et al. [23] constructed (non-cryptographic) ε-biased generators with linear stretch and exponentially small bias in NC^0_5. Applebaum et al. [4] subsequently showed that, under standard cryptographic assumptions, there are pseudorandom generators in NC^0_4. However, these PRGs have only *sublinear-stretch*. PRGs with linear stretch are known to exist (under plausible assumptions) in the class NC^1 and even in TC^0, e.g., [22, 26]. (Recall that TC^0 is the class of functions computable by constant depth circuits consisting of a polynomial number of threshold gates with unbounded fan-in; hence, $\mathrm{NC}^0 \subsetneq \mathrm{AC}^0 \subsetneq \mathrm{TC}^0 \subseteq \mathrm{NC}^1$.)

The first application of average-case complexity to inapproximability was suggested by Feige [14], who derived new inapproximability results under the assumption that refuting 3SAT is hard on average on some natural distribution. In [1] Alekhnovich continued this line of research. He considered the problem of determining the maximal number of satisfiable equations in a linear system chosen at random, and made several conjectures regarding the average case hardness of this problem. He showed that these conjectures imply Feige's assumption as well as several new inapproximability results. While the works of Feige and Alekhnovich derived *new* inapproximability results (that were not known to hold under the assumption that $\mathrm{P} \neq \mathrm{NP}$), they did not rely on the relation with a standard cryptographic assumption or primitive, but rather used specific average case hardness

assumptions tailored to their inapproximability applications. A relation between the security of a cryptographic primitive and approximation was implicitly used in [23], where an approximation algorithm for Max2LIN was used to derive an upper bound on the stretch of a PRG whose locality is 4.

Organization. The rest of this paper is structured as follows. We begin with a discussion of notation and preliminaries (Section 2). In Section 3 we prove that an LPRG in NC^0 implies that Max3SAT cannot be efficiently approximated to within some multiplicative constant. Then in Section 4 we extend these results and show how to derive the inapproximability of Max3SAT from NC^0 implementations of other cryptographic primitives. In Section 5 we present a construction of an LPRG in NC^0. This construction uses an NC^0 implementation of an ε-biased generator as an ingredient. A uniform construction of such an ε-biased generator is described in Section 5.4. Finally, in Section 6, we discuss the connection between LPRG in NC^0 to expander graphs.

2 Preliminaries

2.1 Basic Definitions

Probability notation. We use U_n to denote a random variable uniformly distributed over $\{0,1\}^n$. If X is a probability distribution, or a random variable, we write $x \leftarrow X$ to indicate that x is a sample taken from X. The *min-entropy* of a random variable X is defined as $\mathrm{H}_\infty(X) \stackrel{\text{def}}{=} \min_x \log(\frac{1}{\Pr[X=x]})$. The *statistical distance* between discrete probability distributions Y and Y', denoted $\mathrm{SD}(Y,Y')$, is defined as the maximum, over all functions A, of the *distinguishing advantage* $|\Pr[A(Y)=1] - \Pr[A(Y')=1]|$.

A function $\varepsilon(\cdot)$ is said to be *negligible* if $\varepsilon(n) < n^{-c}$ for any constant $c > 0$ and sufficiently large n. We will sometimes use $\mathrm{neg}(\cdot)$ to denote an unspecified negligible function. For two distribution ensembles $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$, we write $X_n \equiv Y_n$ if X_n and Y_n are identically distributed, and $X_n \stackrel{\text{s}}{\approx} Y_n$ if the two ensembles are *statistically indistinguishable*; namely, $\mathrm{SD}(X_n, Y_n)$ is negligible in n. A weaker notion of closeness between distributions is that of *computational* indistinguishability: We write $X_n \stackrel{\text{c}}{\approx}_{\delta(n)} Y_n$ if for every (non-uniform) polynomial-size circuit family $\{A_n\}$, the distinguishing advantage $|\Pr[A_n(X_n) = 1] - \Pr[A_n(Y_n) = 1]|$ is bounded by $\delta(n)$ for sufficiently large n. When the distinguishing advantage $\delta(n)$ is negligible, we simply write $X_n \stackrel{\text{c}}{\approx} Y_n$. By definition, $X_n \equiv Y_n$ implies that $X_n \stackrel{\text{s}}{\approx} Y_n$ which in turn implies that $X_n \stackrel{\text{c}}{\approx} Y_n$. A distribution ensemble $\{X_n\}_{n\in\mathbb{N}}$ is said to be *pseudorandom* if $X_n \stackrel{\text{c}}{\approx} U_n$.

We will use the following definition of a pseudorandom generator.

Definition 2.1 (Pseudorandom generator) *A pseudorandom generator (PRG) is a deterministic function $G : \{0,1\}^* \to \{0,1\}^*$ satisfying the following two conditions:*

- ***Expansion**: There exists a* stretch function $s : \mathbb{N} \to \mathbb{N}$ *such that $s(n) > n$, for all $n \in \mathbb{N}$, and $|G(x)| = s(|x|)$ for all $x \in \{0,1\}^*$.*
- ***Pseudorandomness**: The ensembles $\{G(U_n)\}_{n\in\mathbb{N}}$ and $\{U_{s(n)}\}_{n\in\mathbb{N}}$ are computationally indistinguishable.*

When $s(n) = n + \Omega(n)$ we say that G is a linear-stretch *pseudorandom generator (LPRG). By default, we require G to be polynomial time computable.*

It will sometimes be convenient to define a PRG by an infinite family of functions $\{G_n : \{0,1\}^{m(n)} \to \{0,1\}^{s(n)}\}_{n\in\mathbb{N}}$, where $m(\cdot)$ and $s(\cdot)$ are polynomials. Such a family can be transformed into a single function that satisfies Definition 2.1 via padding. We will abuse notation and write $G : \{0,1\}^{m(n)} \to \{0,1\}^{s(n)}$ to denote the family $\{G_n : \{0,1\}^{m(n)} \to \{0,1\}^{s(n)}\}_{n\in\mathbb{N}}$. We will also rely on *$\varepsilon$-biased generators*, defined similarly to PRGs except that the pseudorandomness holds only against linear functions over $\mathbb{F}_2$. Namely, for a bias function $\varepsilon : \mathbb{N} \to (0,1)$ we say that $G : \{0,1\}^n \to \{0,1\}^{s(n)}$ is an ε-biased generator if for every non-constant linear function $L : \mathbb{F}_2^n \to \mathbb{F}_2$ and all sufficiently large n's it holds that $|\Pr[L(G(U_n)) = 1] - \frac{1}{2}| < \varepsilon(n)$.

Locality. We say that $f : \{0,1\}^n \to \{0,1\}^s$ is *c-local* if each of its output bits depends on at most c input bits, and that $f : \{0,1\}^* \to \{0,1\}^*$ is c-local if for every n the restriction of f to n-bit inputs is c-local. For a constant c, the non-uniform class NC^0_c includes all c-local functions $f : \{0,1\}^* \to \{0,1\}^*$. The class NC^0 contains all functions with *some* constant locality, namely it is the union of all classes NC^0_c. The class *uniform*-NC^0 is the class of NC^0 functions that can be computed in polynomial time; i.e., *uniform*-$\mathrm{NC}^0 = \mathrm{NC}^0 \cap \mathrm{P}$.[2]

Expanders. In the followings think of m as larger than n. We say that a bipartite graph $G = ((L = [m], R = [n]), E)$ is (K, α) expanding if every set of left

[2]We can equivalently define the classes NC^0 and *uniform*-NC^0 in terms of circuits. In this case the class NC^0 is the class of functions which are computable by constant depth circuits with bounded fan-in, and the class *uniform*-NC^0 requires these circuits to be polynomial-time constructible. These definitions are trivially equivalent in the non-uniform case. This equivalence also holds in the uniform case since, given an oracle to a c-local function f, one can efficiently "learn" an NC^0 circuit that computes f. (Each of the output bits of f is a boolean function that depends only on c input bits, hence it can be learned in time $O(n^c)$ via a brute force search over all possible subsets of c relevant variables.)

vertices S of size smaller than K has at least $\alpha \cdot |S|$ right neighbors. A family of bipartite graphs $\{G_n\}_{n\in\mathbb{N}}$ where $G_n = ((L = [m(n)], R = [n]), E)$ is expanding if for some constants α and β and sufficiently large n the graph G_n is $(\beta \cdot m(n), \alpha)$ expanding. A family of $m(n) \times n$ binary matrices $\{M_n\}_{n\in\mathbb{N}}$ is expanding if the family of bipartite graphs $\{G_n\}_{n\in\mathbb{N}}$ represented by $\{M_n\}_{n\in\mathbb{N}}$ (i.e., M_n is the adjacency matrix of G_n) is expanding.

2.2 Some useful facts

We will rely on several standard facts. We begin with two facts regarding statistical distance whose proofs can be found in [29].

Fact 2.2 *For every distributions X and Y and every randomized process A, we have* $\mathrm{SD}(A(X), A(Y)) \leq \mathrm{SD}(X, Y)$.

For jointly distributed random variables A and B we write $B|_{A=a}$ to denote the conditional distribution of B given that $A = a$.

Fact 2.3 *Suppose that $X = (X_1, X_2)$ and $Y = (Y_1, Y_2)$ are probability distributions on a set $D \times E$ such that: (1) X_1 and Y_1 are identically distributed; and (2) with probability greater than $1-\varepsilon$ over $x \leftarrow X_1$, we have* $\mathrm{SD}(X_2|_{X_1=x}, Y_2|_{Y_1=x}) \leq \delta$. *Then* $\mathrm{SD}(X, Y) \leq \varepsilon + \delta$.

For a randomized algorithm A and an integer i we define A^i to be the randomized algorithm obtained by composing A exactly i times with itself; that is, $A^1(x) = A(x)$ and $A^i(x) = A(A^{i-1}(x))$, where in each invocation a fresh randomness is used. The following fact (which is implicit in [1]) can be proved via a hybrid argument.

Fact 2.4 *Let $\{X_n\}$ be a distribution ensemble, and let A be a randomized polynomial-time algorithm. Suppose that $\{X_n\} \stackrel{c}{\approx} \{A(X_n)\}$. Then for every polynomial $p(\cdot)$ we have $\{X_n\} \stackrel{c}{\approx} \{A^{p(n)}(X_n)\}$.*

We let $\mathrm{H}_2(\cdot)$ denote the binary entropy function, i.e., for $0 < p < 1$, $\mathrm{H}_2(p) \stackrel{\text{def}}{=} -p\log(p) - (1-p)\log(1-p)$. We will use the following well known bound on the sum of binomial coefficients.

Fact 2.5 *For $0 < p \leq 1/2$ we have $\sum_{i=0}^{pn} \binom{n}{i} \leq 2^{n\mathrm{H}_2(p)}$.*

The *bias* of a Bernoulli random variable X is defined to be $|\Pr[X = 1] - \frac{1}{2}|$. We will need the following fact which estimates the bias of sum of independent random coins (cf. [23, 30]).

Fact 2.6 *Let $X_1, \ldots, X_t$ be independent binary random variables. Suppose that for some $0 < \delta < \frac{1}{2}$ and every i it holds that $\mathrm{bias}(X_i) \leq \delta$. Then, $\mathrm{bias}(\bigoplus_{i=1}^{t} X_i) \leq \frac{1}{2}(2\delta)^t$.*

3 LPRG in NC^0 implies Hardness of Approximation

In the following we show that if there exists an LPRG in NC^0 then there is no polynomial-time approximation scheme (PTAS) for Max3SAT; that is, Max3SAT cannot be efficiently approximated within some multiplicative constant $r > 1$. Recall that in the Max3SAT problem we are given a 3CNF boolean formula with s clauses over n variables, and the goal is to find an assignment that satisfies the largest possible number of clauses. The Max ℓ-CSP problem is a generalization of Max3SAT in which instead of s clauses we get s boolean constraints $C = \{C_1, \ldots, C_s\}$ of arity ℓ. Again, our goal is to find an assignment that satisfies the largest possible number of constraints. (Recall that a constraint C of arity ℓ over n variables is an ℓ-local boolean function $f : \{0,1\}^n \to \{0,1\}$, and it is satisfied by an assignment $(\sigma_1, \ldots, \sigma_n)$ if $f(\sigma_1, \ldots, \sigma_n) = 1$.)

A simple and useful corollary of the PCP Theorem [5, 6] is the inapproximability of Max3SAT.

Theorem 3.1 *Assume that $\mathrm{P} \neq \mathrm{NP}$. Then, there is an $\varepsilon > 0$ such that there is no $(1+\varepsilon)$-approximatation algorithm for Max3SAT.*

We will prove a similar result under the (stronger) assumption that there exists an LPRG in NC^0. Our proof, however, does not rely on the PCP Theorem.

Theorem 3.2 *Assume that there exists an LPRG in NC^0. Then, there is an $\varepsilon > 0$ such that there is no $(1+\varepsilon)$-approximation algorithm for Max3SAT.*

The proof of Theorem 3.2 follows by combining the following Fact 3.3 and Lemma 3.4. The first fact shows that in order to prove that Max3SAT is hard to approximate, it suffices to prove that Max ℓ-CSP is hard to approximate. This standard result follows by applying Cook's reduction to transform every constraint into a 3CNF.

Fact 3.3 *Assume that, for some constants $\ell \in \mathbb{N}$ and $\varepsilon > 0$, there is no polynomial time $(1+\varepsilon)$-approximation algorithm for Max ℓ-CSP. Then there is an $\varepsilon' > 0$ such that there is no polynomial time $(1+\varepsilon')$-approximation algorithm for Max3SAT.*

Thus, the heart of the proof of Theorem 3.2 is showing that the existence of an LPRG in NC^0_ℓ implies that there is no PTAS for Max ℓ-CSP.

Lemma 3.4 *Let ℓ be a positive integer, and $c > 1$ be a constant such that $G : \{0,1\}^n \to \{0,1\}^{cn}$ is an LPRG which is computable in NC^0_ℓ. Then, there is no $1/(1-\varepsilon)$-approximation algorithm for Max ℓ-CSP, where $0 < \varepsilon < 1/2$ is a constant that satisfies $\mathrm{H}_2(\varepsilon) < 1 - 1/c$.*

For $\varepsilon = 1/10$ (i.e., ≈ 1.1-approximation) the constant $c = 2$ will do, whereas for $\varepsilon = 0.49$ (i.e., ≈ 2-approximation) $c = 3500$ will do.

Proof: Let $s = s(n) = cn$. Assume towards a contradiction that there exists an $1/(1-\varepsilon)$-approximation algorithm for Max ℓ-CSP where $\mathrm{H}_2(\varepsilon) < 1 - 1/c$. Then, there exists a polynomial-time algorithm A that given an ℓ-CSP instance ϕ outputs 1 if ϕ is satisfiable, and 0 if ϕ is ε-unsatisfiable (i.e., if every assignment fails to satisfy at least a fraction ε of the constraints). We show that, given such A, we can "break" the LPRG G; that is, we can construct an efficient (non-uniform) adversary that distinguishes between $G(U_n)$ and U_s. Our adversary B_n will (deterministically) translate a string $y \in \{0,1\}^s$ into an ℓ-CSP instance ϕ_y with s constraints such that the following holds:

1. If $y \leftarrow G(U_n)$ then ϕ_y is always satisfiable.
2. If $y \leftarrow U_s$ then, with probability $1 - \mathrm{neg}(n)$ over the choice of y, no assignment satisfies more than $(1-\varepsilon)s$ constraints of ϕ_y.

Then, B_n will run A on ϕ_y and will output $A(\phi_y)$. The distinguishing advantage of B is $1 - \mathrm{neg}(n)$ in contradiction to the pseudorandomness of G.

It is left to show how to translate $y \in \{0,1\}^s$ into an ℓ-CSP instance ϕ_y. We use n boolean variables $x_1, \ldots, x_n$ that represent the bits of an hypothetical pre-image of y under G. For every $1 \le i \le s$ we add a constraint $G_i(x) = y_i$ where G_i is the function that computes the i-th output bit of G. Since G_i is an ℓ-local function the arity of the constraint is at most ℓ.

Suppose first that $y \leftarrow G(U_n)$. Then, there exists a string $\sigma \in \{0,1\}^n$ such that $G(\sigma) = y$ and hence ϕ_y is satisfiable. We move on to the case where $y \leftarrow U_s$. Here, we rely on the fact that such a random y is very likely to be far from every element in the range of G. More formally, define a set $\mathrm{BAD}_n \subseteq \{0,1\}^s$ such that $y \in \mathrm{BAD}_n$ if ϕ_y is $(1-\varepsilon)$-satisfiable; that is, if there exists an assignment $\sigma \in \{0,1\}^n$ that satisfies at least $(1-\varepsilon)$ fraction of the constraints of ϕ_y. In other words, the Hamming distance between y and $G(\sigma)$ is at most εs. Hence, all the elements of BAD_n are εs-close (in Hamming distance) to some string in $\mathrm{Im}(G)$. Therefore, the size of BAD_n is bounded by

$$|\mathrm{Im}(G)| \cdot \sum_{i=0}^{\varepsilon s} \binom{s}{i} \le 2^n 2^{\mathrm{H}_2(\varepsilon)s} = 2^{(1+c\mathrm{H}_2(\varepsilon))n},$$

where the first inequality is due to Fact 2.5. Let $\alpha \stackrel{\text{def}}{=} c - (1 + c \cdot \mathrm{H}_2(\varepsilon))$ which is a positive constant since $\mathrm{H}_2(\varepsilon) < 1 - 1/c$. Hence, we have

$$\Pr_{y \leftarrow U_s}[\phi_y \text{ is } (1-\varepsilon) \text{ satisfiable}] = |\mathrm{BAD}_n| \cdot 2^{-s} \leq 2^{(1+c\mathrm{H}_2(\varepsilon))n - cn} = 2^{-\alpha n} = \mathrm{neg}(n),$$

which completes the proof. ■

Remark 3.5 Lemma 3.4 can tolerate some relaxations to the notion of LPRG. In particular, since the advantage of B_n is exponentially close to 1, we can consider an LPRG that satisfies a weaker notion of pseudorandomness in which the distinguisher's advantage is bounded by $1 - 1/p(n)$ for some polynomial $p(n)$. In Section 4 we consider additional cryptographic primitives that imply the inapproximability of Max3SAT.

Lemma 3.4 implies the following corollary.

Corollary 3.6 *Suppose there exists a PRG in* NC^0_ℓ *with* an arbitrary *linear stretch; i.e., for every* $c > 0$ *there exists a PRG* $G : \{0,1\}^n \to \{0,1\}^{c \cdot n} \in \mathrm{NC}^0_\ell$. *Then, Max* ℓ*-CSP cannot be approximated to within any constant* $\delta < 2$ *that is arbitrarily close to 2.*

Remark 3.7 Corollary 3.6 is tight, as any CSP problem of the form $G(x) = y$ (for any $y \in \{0,1\}^s$) can be easily approximated within a factor of 2. To see this, note that the function $G_i(x)$ which computes the i-th output bit of G must be balanced, i.e., $\Pr_x[G_i(x) = 1] = 1/2$. (Otherwise, since $G_i \in \mathrm{NC}^0$, the function G_i has a constant bias and so $G(U_n)$ cannot be pseudorandom.) Therefore, a random assignment is expected to satisfy 1/2 of the constraints of the instance $G(x) = y$. This algorithm can be derandomized by using the method of conditional expectations.

Papadimitriou and Yannakakis [27] defined a class MaxSNP, in which Max3SAT is complete in the sense that any problem in MaxSNP has a PTAS if and only if Max3SAT has a PTAS. Hence, we get the following corollary (again, without the PCP machinery):

Corollary 3.8 *Assume that there exists an LPRG in* NC^0. *Then, all Max SNP problems (e.g., Max-Cut, Max2SAT, Vertex Cover) do not have a PTAS.*

4 Using NC^0 Implementations of Other Cryptographic Primitives

In the following we extend the results of the Section 3, and show that the inapproximability of Max3SAT can be based on NC^0 implementations of the following primitives: (1) pseudoentropy generator that gains a linear amount of computational entropy; (2) string commitment of linear size; and (3) public-key encryption whose ciphertext length is linear in the message length. We start by abstracting the proof of Theorem 3.2. That is, we show that the following assumption imply the inapproximability of Max3SAT.

Consider a pair of distribution ensembles A and B, a parameter δ, and a constant ε. The assumption holds if (1) A is samplable by NC^0 circuits; (2) the computational distance between A and B is bounded by δ; and (3) the probability that the outcome of B will be ε-close to the support of A is smaller than $1-\delta$. More formally, we assume the following.

Assumption 4.1 *There exist two distribution ensembles $\{A_n\}_{n\in\mathbb{N}}$ and $\{B_n\}_{n\in\mathbb{N}}$ where A_n and B_n are distributed over $\{0,1\}^{s(n)}$, and the ensemble $\{A_n\}$ is samplable by an NC^0 circuit family. There exists a function $\delta(n):\mathbb{N}\to[0,1]$, and a constant $\varepsilon>0$ such that the following holds:*

1. *$\{A_n\} \stackrel{c}{\approx}_{\delta(n)} \{B_n\}$. That is, every polynomial-size circuit family distinguishes $\{A_n\}$ from $\{B_n\}$ with advantage at most $\delta(n)$ for sufficiently large n.*

2. *With probability smaller than $1-\delta(n)$ a string $b \leftarrow B_n$ is ε-close (in normalized Hamming distance) to some string in the support of A_n. That is, $\Pr_{b\leftarrow B_n}[\exists a \in \mathrm{support}(A_n)$ s.t. $\mathrm{dist}(a,b) \le \varepsilon\cdot s(n)] < 1-\delta(n)$, where $\mathrm{dist}(a,b)$ denotes the Hamming distance between the strings a and b.*

This assumption is implied by the existence of an LPRG in NC^0. Indeed, if $G:\{0,1\}^n\to\{0,1\}^{cn}$ is an LPRG in NC^0 then Assumption 4.1 holds with respect to $A_n = G(U_n)$, $B_n = U_{cn}$, $\delta(n) = 1/n$ and a constant $0<\varepsilon<1/2$ that satisfies $1 + c\cdot \mathrm{H}_2(\varepsilon) < c$. (This is implicitly shown in the proof of Lemma 3.4.)

Lemma 4.2 *Assumption 4.1 implies that there is no PTAS for Max3SAT.*

Proof sketch: The proof is very similar to the proof of Lemma 3.4. Let $G\in\mathrm{NC}^0_\ell$ be the circuit that samples the distribution A_n. Assume towards a contradiction that the claim does not hold. Then, there exists an algorithm D that given an ℓ-CSP instance ϕ outputs 1 if ϕ is satisfiable, and 0 if ϕ is ε-unsatisfiable.

We use this procedure to distinguish $\{A_n\}$ from $\{B_n\}$ with advantage greater than $\delta(n)$. Given a challenge $y \in \{0,1\}^{s(n)}$, we translate it into an ℓ-CSP instance ϕ_y of the form $G(x) = y$, and output $D(\phi_y)$. If $y \leftarrow A_n$ then ϕ_y is always satisfiable. On the other hand, if $y \leftarrow B_n$ then, with probability larger than $\delta(n)$, the formula ϕ_y is ε-unsatisfiable. ■

4.1 Pseudoentropy Generator

We now show that an NC^0 implementation of a relaxed notion of LPRG implies Assumption 4.1. In particular, instead of being pseudorandom, the distribution $G(U_n)$ is only required to be computationally close to some distribution whose min-entropy is (much) larger. Moreover, we allow a non-negligible distinguishing advantage. This relaxation can be considered as a weak pseudoentropy generator that gains a linear amount of computational entropy cf.[21, 7].

Lemma 4.3 (Weak LPRG in NC^0 $\Rightarrow$ inapproximability) *Suppose that there exist an NC^0 function $G : \{0,1\}^n \to \{0,1\}^{s(n)}$ and a distribution ensemble $\{B_n\}$, such that:*

- $\{G(U_n)\} \stackrel{c}{\approx}_{\delta(n)} \{B_n\}$ *for some $\delta(n)$ such that $\delta(n) \leq 1 - 2^{-o(s(n))}$.*
- $\mathrm{H}_\infty(B_n) - n = \Omega(s(n))$.

Then, there is no PTAS for Max3SAT.

Proof: Let $A_n \stackrel{\text{def}}{=} G(U_n)$. We show that $A_n, B_n, \delta(n)$ and some constant $\varepsilon < 1/2$ satisfy Assumption 4.1. Indeed, the only non-trivial part is item (2). Let $\mathrm{BAD}_{\varepsilon,n} \subseteq \{0,1\}^{s(n)}$ be the set of all strings which are ε-close to $\mathrm{Im}(G)$. Then,

$$\begin{aligned}
\Pr_{b \leftarrow B_n}[b \text{ is } \varepsilon\text{-close to } \mathrm{Im}(G)] &= \sum_{y \in \mathrm{BAD}_{\varepsilon,n}} \Pr_{b \leftarrow B_n}[b = y] \\
&\leq \left(|\mathrm{Im}(G)| \cdot \sum_{i=0}^{\varepsilon s(n)} \binom{s(n)}{i} \right) \cdot 2^{-\mathrm{H}_\infty(B_n)} \\
&\leq 2^{n + s(n)\cdot \mathrm{H}_2(\varepsilon) - \mathrm{H}_\infty(B_n)} \leq 2^{-\Omega(s(n))} < 1 - \delta(n),
\end{aligned}$$

where the second inequality is due to Fact 2.5, the third inequality holds for sufficiently small (constant) ε, and the last inequality holds for sufficiently large n . ■

4.2 String Commitment

Another sufficient assumption is an NC^0 implementation of a non-interactive string commitment with a constant information rate, namely one in which the length of the commitment is linear in that of the committed string. A non-interactive commitment scheme is defined by a function $\textsc{Com} : \{0,1\}^n \times \{0,1\}^{m(n)} \to \{0,1\}^{s(n)}$ such that:

1. (Binding) For every pair of different strings $x, y \in \{0,1\}^n$ the supports of $\textsc{Com}(x, U_{m(n)})$ and $\textsc{Com}(y, U_{m(n)})$ are disjoint.
2. (Hiding) For every pair of string families $\{x_n\}_{n\in\mathbb{N}}$ and $\{y_n\}_{n\in\mathbb{N}}$ where $x_n, y_n \in \{0,1\}^n$, we have $\textsc{Com}(x_n, U_{m(n)}) \stackrel{c}{\approx} \textsc{Com}(y_n, U_{m(n)})$.

In fact, for our purpose we can relax the hiding property to be $\textsc{Com}(x_n, U_{m(n)}) \stackrel{c}{\approx}_{\delta(n)} \textsc{Com}(y_n, U_{m(n)})$ where $\delta(n) = 1 - 2^{-o(n)}$.

Lemma 4.4 (Constant rate string commitment in NC^0 $\Rightarrow$ inapproximability) *Let $c > 1$ be a constant. Suppose that there exists a (non-interactive) commitment scheme $\textsc{Com} : \{0,1\}^n \times \{0,1\}^{m(n)} \to \{0,1\}^{c\cdot n}$ computable in NC^0. Then, there is no PTAS for Max3SAT.*

Proof: Let ε be a sufficiently small constant for which $\mathrm{H}_2(\varepsilon)\cdot c < 0.9$. Let $A_n \stackrel{\text{def}}{=} \textsc{Com}(U_n, U_{m(n)})$ and $B_n \stackrel{\text{def}}{=} \textsc{Com}(0^n, U_{m(n)})$. We show that A_n, B_n, $\delta(n) = 1 - 2^{-o(n)}$ and ε satisfy Assumption 4.1. Again, we focus on proving that the second item of the assumption holds.

Fix some $r \in \{0,1\}^{m(n)}$. There are at most $\sum_{i=0}^{\varepsilon cn} \binom{cn}{i} \leq 2^{\mathrm{H}_2(\varepsilon)cn} \leq 2^{0.9n}$ strings which are ε-close to $\textsc{Com}(0^n, r)$. Hence, by the binding property, we have

$$\Pr[\textsc{Com}(0^n, r) \text{ is } \varepsilon\text{-close to } \mathrm{support}(\textsc{Com}(U_n, U_{m(n)}))] \leq 2^{0.9n-n} = 2^{-0.1n}.$$

Thus,

$$\Pr_{r\leftarrow U_{m(n)}}[\textsc{Com}(0^n, r) \text{ is } \varepsilon\text{-close to } \mathrm{support}(\textsc{Com}(U_n, U_{m(n)}))] \leq 2^{-0.1n} < 1-\delta(n),$$

where the last inequality holds for sufficiently large n. ■

Public-Key Encryption. Suppose we have an error-free public-key encryption scheme whose encryption algorithm is in NC^0 and whose information rate is constant (i.e., the ciphertext length is linear in the message length). Then, we can construct a (collection of) constant-rate NC^0 non-interactive commitments. (Set $\textsc{Com}_e(x, r) \stackrel{\text{def}}{=} E_e(x, r)$ where $E_e(x, r)$ is the encryption function which encrypts the message x using the key e and randomness r.) Hence, such a scheme also implies the inapproximability of Max3SAT.

5 A Construction of LPRG in NC^0

5.1 Overview

We start with an informal description of our construction. Consider the following distribution: *fix* a sparse matrix $M \in \{0,1\}^{m \times n}$ in which every row contains a constant number of ones, multiply it with a random n-bit vector x, and add a noise vector $e \in \{0,1\}^m$ which is uniformly distributed over all m-bits vectors whose Hamming weight is $\lceil \mu \cdot m \rceil$. (For concreteness, think of $m = 5n$ and $\mu = 0.1$.) That is, we consider the distribution $\hat{D}_\mu(M) \stackrel{\text{def}}{=} M \cdot x + e$, where all arithmetic is over $\mathbb{F}_2$.

Consider the distribution $\hat{D}_{\mu+m^{-1}}(M)$ which is similar to the previous distribution except that this time the noise vector is uniformly distributed over m-bit vectors whose weight is $(\mu + 1/m) \cdot m = \mu m + 1$. Alekhnovich conjectured in [1, Conjecture 1] that for a proper choice of M these distributions are computationally indistinguishable. He also showed that if indeed this is the case, then $\hat{D}_\mu(M)$ is pseudorandom; that is, $\hat{D}_\mu(M)$ is computationally indistinguishable from U_m. Since the distribution $\hat{D}_\mu(M)$ can be sampled (efficiently) by using roughly $n + \log \binom{m}{\mu \cdot m} \leq n + m\mathrm{H}_2(\mu)$ random bits, it gives rise to a pseudorandom generator with linear stretch (when the parameters are chosen properly).

We would like to sample $\hat{D}_\mu(M)$ by an NC^0 function. Indeed, since the rows of M contains only a constant number of ones, we can easily compute the product Mx in NC^0 (recall that M itself is fixed). Unfortunately, we do not know how to sample the noise vector e by an NC^0 function. To solve this, we change the noise distribution. That is, we consider a slightly different distribution $D_\mu(M)$ in which each entry of the noise vector e is chosen to be 1 with probability μ (independently of other entries). We adopt Alekhnovich's conjecture to this setting; namely, we assume that $D_\mu(M)$ cannot be distinguished efficiently from $D_{\mu+m^{-1}}(M)$. (In fact, the new assumption is implied by the original one. See Appendix A.) Similarly to the previous case, we show that under this assumption $D_\mu(M)$ is pseudorandom.

Now, whenever $\mu = 2^{-t}$ for some integer t, we can sample each bit of the noise vector by taking the product of t random bits. Hence, in this case $D_\mu(M)$ is samplable in NC^0 (as we think of μ as a constant). The problem is that our NC^0 procedure which samples $D_\mu(M)$ consumes more bits than it produces (i.e., it consumes $n + t \cdot m$ bits and produces m bits). Hence, we lose the stretch. To solve this, we note that most of the entropy of the seed was not used. Thus, we can gain more output bits by applying a randomness extractor to the seed. To be useful, this randomness extractor should be computable in NC^0. We construct such an extractor by relying on the construction of ε-biased generator in NC^0 of [23].

For ease of presentation, we describe our construction in a non-uniform way.

We will later discuss a uniform variant of the construction.

5.2 The Assumption

Let $m = m(n)$ be an output length parameter where $m(n) > n$, let $\ell = \ell(n)$ be a locality parameter (typically a constant), and let $0 < \mu < 1/2$ be a noise parameter. Let $\mathcal{M}_{m,n,\ell}$ be the set of all $m \times n$ matrices over $\mathbb{F}_2$ in which each row contains at most ℓ ones. For a matrix $M \in \mathcal{M}_{m,n,\ell}$ denote by $D_\mu(M)$ the distribution of the random m-bit vector

$$Mx + e,$$

where $x \leftarrow U_n$ and $e \in \{0,1\}^m$ is a random error vector in which each entry is chosen to be 1 with probability μ (independently of other entries), and arithmetic is over $\mathbb{F}_2$. The following assumption is a close variant of a conjecture suggested by Alekhnovich in [1, Conjecture 1]. [3]

Assumption 5.1 *For any $m(n) = O(n)$, and any constant $0 < \mu < 1/2$, there exists a positive integer ℓ, and an infinite family of matrices $\{M_n\}_{n\in\mathbb{N}}$, $M_n \in \mathcal{M}_{m(n),n,\ell}$, such that*

$$D_\mu(M_n) \overset{c}{\approx} D_{\mu+m(n)^{-1}}(M_n)$$

(Note that since we consider non-uniform distinguishers, we can assume that M_n is public and is available to the distinguisher.)

Remark 5.2 Note that in Assumption 5.1 we do not require $\{M_n\}$ to be polynomial-time computable. We will later present a uniform construction based on the following version of Assumption 5.1. For any $m(n) = O(n)$, any constant $0 < \mu < 1/2$, and any infinite family of $m(n) \times n$ binary matrices $\{M_n\}_{n\in\mathbb{N}}$, if $\{M_n\}$ is expanding then $D_\mu(M_n) \overset{c}{\approx} D_{\mu+m(n)^{-1}}(M_n)$. This assumption seems likely as argued by Alekhnovich [1, Remark 1].

The following lemma shows that if the distribution $D_\mu(M_n)$ satisfies the above assumption then it is pseudorandom. (The proof is very similar to the proof of [1, Theorem 3.1], and it is given here for completeness.)

[3] Our assumption is essentially the same as Alekhnovich's. The main difference between the two assumptions is that the noise vector e in [1] is a random vector of weight exactly $\lceil \mu m \rceil$, as opposed to our noise vector whose entries are chosen to be 1 independently with probability μ. In Appendix A we show that our assumption is implied by Alekhnovich's assumption. Intuitively, the implication follows from the fact that our noise vectors can be viewed as a convex combination of noise vectors of fixed weight. We do not know whether the converse implication holds. Indeed, a distribution D which can be described as a convex combination of distributions $D_1, \ldots, D_n$ may be pseudorandom even if each of the distributions D_i is not pseudorandom.

Lemma 5.3 *For any polynomial $m(n)$ and constant $0 < \mu < 1/2$, and any infinite family, $\{M_n\}_{n\in\mathbb{N}}$, of $m(n) \times n$ matrices over $\mathbb{F}_2$, if $D_\mu(M_n) \stackrel{c}{\approx} D_{\mu+m(n)^{-1}}(M_n)$, then $D_\mu(M_n) \stackrel{c}{\approx} U_{m(n)}$.*

Proof: Let $m = m(n)$. Let r_n denote the distribution of an m-bit vector in which each entry is chosen to be 1 with probability c/m (independently of other entries) where c is the constant $1/(1-2\mu)$. As shown next, we can write

$$D_{\mu+m^{-1}}(M_n) \equiv D_\mu(M_n) + r_n. \tag{1}$$

To see this, let $e, e' \in \{0,1\}^m$ be noise vectors of rate $\mu, \mu + 1/m$ respectively. Then, to prove Eq. 1 it suffices to show that $e' \equiv e + r_n$. Indeed, the entries of $e + r_n$ are iid Bernoulli random variables whose success probability is

$$\mu \cdot (1 - (m(1-2\mu))^{-1}) + (1-\mu) \cdot (m(1-2\mu))^{-1} = \mu + m(n)^{-1}.$$

Now, by Eq. 1 and the lemma's hypothesis, we have

$$D_\mu(M_n) \stackrel{c}{\approx} D_\mu(M_n) + r_n. \tag{2}$$

Let r_n^i be the distribution resulting from summing (over $\mathbb{F}_2^m$) i independent samples from r_n. Let $p(\cdot)$ be a polynomial. Then, by Fact 2.4, we get that

$$D_\mu(M_n) \stackrel{c}{\approx} D_\mu(M_n) + r_n^{p(n)}. \tag{3}$$

Recall that r_n is a vector of iid Bernoulli random variables whose success probability is $\Theta(1/m)$. Hence, for some polynomial $p(\cdot)$ (e.g., $p(n) = nm$) it holds that

$$r_n^{p(n)} \stackrel{s}{\approx} U_{m(n)}. \tag{4}$$

(To see this, note that $r_n^{p(n)}$ is a vector of iid Bernoulli random variables whose success probability is, by Fact 2.6, $1/2 \pm (1/2 - \Theta(1/m))^{p(n)} = 1/2 \pm \mathrm{neg}(n)$.) By combining Eq. 3 and 4, we have

$$D_\mu(M_n) \stackrel{c}{\approx} D_\mu(M_n) + r_n^{p(n)} \stackrel{s}{\approx} D_\mu(M_n) + U_{m(n)} \equiv U_{m(n)},$$

and the lemma follows. ■

By combining Assumption 5.1 and Lemma 5.3, we get the following.

Proposition 5.4 *Suppose that Assumption 5.1 holds. Then, for any $m(n) = O(n)$, and any constant $0 < \mu < 1/2$, there exists a constant $\ell \in \mathbb{N}$, and an infinite family of matrices $\{M_n\}_{n\in\mathbb{N}}$ where $M_n \in \mathcal{M}_{m(n),n,\ell}$ such that $D_\mu(M_n) \stackrel{c}{\approx} U_{m(n)}$.*

Remark 5.5 If the restriction on the density of the matrices M_n is dropped, the above proposition can be based on the conjectured (average case) hardness of decoding a random linear code (cf., [9, 19]). In fact, under the latter assumption we have that $D_\mu(M_n) \stackrel{c}{\approx} U_{m(n)}$ for *most* choices of M_n's.

5.3 The Construction

From here on, we let $\mu = 2^{-t}$ for some $t \in \mathbb{N}$. Then, we can sample each bit of the error vector e by taking the product of t independent random bits. This naturally gives rise to an NC^0 function whose output distribution is pseudorandom, namely,

$$f_n(x, y) = M_n x + E(y)$$

where

$$x \in \{0,1\}^n, \qquad y \in \{0,1\}^{t \cdot m(n)}, \qquad E(y) = \left(\prod_{j=1}^{t} y_{t\cdot(i-1)+j} \right)_{i=1}^{m(n)} . \tag{5}$$

Since $f_n(U_n, U_{t\cdot m(n)}) \equiv D_\mu(M_n)$, the distribution $f_n(U_n, U_{t\cdot m(n)})$ is pseudorandom under Assumption 5.1 (when the parameters are chosen appropriately). Moreover, the locality of f_n is $\ell + t = O(1)$. However, f_n is not a pseudorandom generator as it uses $n + t \cdot m(n)$ input bits while it outputs only $m(n)$ bits. To overcome this obstacle, we note that most of the entropy of y was not "used". Indeed, we use the $t \cdot m(n)$ random bits of y to sample the distribution $E(y)$ whose entropy is only $m(n) \cdot \mathrm{H}_2(2^{-t}) < (t+2) \cdot 2^{-t} \cdot m(n)$. Hence, we can apply an *extractor* to regain the lost entropy. Of course, in order to get a PRG in NC^0 the extractor should also be computed in NC^0. Moreover, to get a linear stretch we should extract almost all of the $t \cdot m(n)$ random bits from y while investing less than m additional random bits. In the following, we show that such extractors can be implemented by using *ε-biased generators*.

First, we show that the distribution of y given $E(y)$ contains (with high probability) a lot of entropy. In the following we let $m = m(n)$.

Lemma 5.6 *Let $y \leftarrow U_{t\cdot m}$ and $E(y)$ be defined as in Eq. 5. Denote by $[y|E(y)]$ the distribution of y given the outcome of $E(y)$. Then, except with probability $e^{-(2^{-t}m)/3}$ over the choice of y, it holds that*

$$\mathrm{H}_\infty([y|E(y)]) \geq (1 - \delta(t)) \cdot tm, \tag{6}$$

where $\delta(t) = 2^{-\Omega(t)}$.

Proof: We view $E(y)$ as a sequence of m independent Bernoulli trials, each with a probability 2^{-t} of success. Recall that y is composed of m blocks of length t, and that the i-th bit of $E(y)$ equals the product of the bits in the i-th block of y. Hence, whenever $E(y)_i = 1$ all the bits of the i-th block of y equal to 1, and when $E(y)_i = 0$ the i-th block of y is uniformly distributed over $\{0,1\}^t \setminus \{1^t\}$.

Consider the case in which at most $2 \cdot 2^{-t}m$ components of $E(y)$ are ones. By a Chernoff bound, the probability of this event is at least $1 - e^{-(2^{-t}m)/3}$. In this case, y is uniformly distributed over a set of size at least $(2^t-1)^{(1-2^{-t+1})m}$. Hence, conditioning on the event that at most $2 \cdot 2^{-t}m$ components of $E(y)$ are ones, the min-entropy of $[y|E(y)]$ is at least $m(1-2^{-t+1})\log(2^t-1) \geq tm(1-\delta(t))$, for $\delta(t) = 2^{-\Omega(t)}$. ■

ε-biased generators can be used to extract random bits from distributions that contain sufficient randomness. Extractors based on ε-biased generators were previously used in [13].

Lemma 5.7 ([13, Lemma 4]) *Let $g : \{0,1\}^n \to \{0,1\}^s$ be an ε-biased generator, and let X_s be a random variable taking values in $\{0,1\}^s$ whose min-entropy is at least $(1-\delta) \cdot s$, for some $\delta \geq 0$. Then,*

$$\mathrm{SD}((g(U_n) + X_s), U_s) \leq \varepsilon \cdot 2^{\delta \cdot s/2 - 1/2} \ ,$$

where vector addition is taken over $\mathbb{F}_2$.

The above lemma follows directly by analyzing the affect of a random step over a Cayley graph whose generator set is an ε-biased set (cf. [20, Lemma 2.3] and [24, 3]).

Recently, Mossel et al. [23] constructed an ε-biased generator in NC^0_5 with an arbitrary linear stretch and exponentially small bias.

Lemma 5.8 ([23, Thm. 14]) *For every constant c, there exists a (non-explicit) ε-biased generator $g : \{0,1\}^n \to \{0,1\}^{cn}$ in NC^0_5 whose bias is at most $2^{-bn/c^4}$ (where $b > 0$ is some universal constant that does not depend on c).*

In Section 5.4 we provide an explicit version of the above lemma in which the bias is only $2^{-n/\mathrm{polylog}(c)}$. The price we pay is in the locality which grows polylogarithmically with the stretch constant c. (See Theorem 5.12.)

We can now describe our LPRG.

Construction 5.9 *Let t and ℓ be positive integers, and $c, k > 1$ be real numbers that will effect the stretch factor. Let $m = kn$ and let $\{M_n \in \mathcal{M}_{n,m,\ell}\}$ be an infinite family of matrices. Let $g : \{0,1\}^{tm/c} \to \{0,1\}^{tm}$ be the ε-biased generator promised by Lemma 5.8. We define the function*

$$G_n(x,y,r) = (M_n x + E(y), g(r) + y),$$

where $x \in \{0,1\}^n, y \in \{0,1\}^{t \cdot m}, r \in \{0,1\}^{t \cdot m/c}, E(y) = \left(\prod_{j=1}^{t} y_{t\cdot(i-1)+j}\right)_{i=1}^{m}$. Thus, $G_n : \{0,1\}^{n+tm+\frac{tm}{c}} \to \{0,1\}^{m+tm}$.

Observe that G_n is an NC^0 function. We show that if the parameters are chosen properly then G_n is an LPRG.

Lemma 5.10 *Under Assumption 5.1, there exist constants $t, \ell \in \mathbb{N}$, constants $c, k > 1$, and a family of matrices $\{M_n \in \mathcal{M}_{n,m,\ell}\}$ such that the function G_n defined in Construction 5.9 is an LPRG.*

Proof: Let $k > 1$ be some arbitrary constant and $m = m(n) = kn$. Let c and t be constants such that:

$$c = 2t/(1 - 1/k)$$

and

$$\Delta \stackrel{\text{def}}{=} t\left(\frac{b}{c^5} - \delta(t)\right) > 0, \tag{7}$$

where $\delta(\cdot)$ is the negligible function from Eq. 6 and b is the bias constant of Lemma 5.8. Such constants c and t do exist since $\delta(t) = 2^{-\Omega(t)}$ while $b/c^5 = \Theta(1/t^5)$. Let $\ell \in \mathbb{N}$ be a constant and $\{M_n \in \mathcal{M}_{n,m,\ell}\}$ be an infinite family of matrices satisfying Assumption 5.1.

First, we show that G_n has linear stretch. The input length of G_n is $n + tm + tm/c = (tk + k/2 + 1/2) \cdot n$. The output length is $(t + 1) \cdot m = (tk + k) \cdot n$. Hence, since $k > 1$, the function G_n has a linear stretch.

Let x, y and r be uniformly distributed over $\{0,1\}^n$, $\{0,1\}^{t \cdot m}$ and $\{0,1\}^{t \cdot m/c}$ respectively. We prove that the distribution $G_{M_n}(x, y, r)$ is pseudorandom. By Fact 2.3 and Lemmas 5.6, 5.7 and 5.8 it holds that

$$\begin{aligned}
\mathrm{SD}((E(y), y + g(r)), (E(y), U_{t \cdot m})) &\leq e^{-(2^{-t}m)/3} + 2^{-b \cdot (tm/c)/c^4} \cdot 2^{tm \cdot \delta(t)/2 - 1/2} \\
&\leq e^{-(2^{-t}m)/3} + 2^{(-b/c^5 + \delta(t)) \cdot tm} \\
&\leq e^{-(2^{-t}m)/3} + 2^{-\Delta m} = \mathrm{neg}(m) = \mathrm{neg}(n),
\end{aligned}$$

where the last inequality is due to Eq. 7. Therefore, by Fact 2.2 and Proposition 5.4, we get that

$$(M_n x + E(y), g(r) + y) \stackrel{\text{s}}{\approx} (M_n x + E(y), U_{t \cdot m}) \equiv (D_{2^{-t}}(M_n), U_{t \cdot m}) \stackrel{\text{c}}{\approx} (U_m, U_{t \cdot m}),$$

and the lemma follows. ■

By the above lemma we get a construction of LPRG in NC^0 from Assumption 5.1. In fact, in [4, Thm. 6.5] it is (implicitly) shown that such an LPRG can be transformed into an LPRG whose locality is 4. More precisely, [4] prove that for some (small) constant c, any PRG $G : \{0,1\}^n \to \{0,1\}^{n+s(n)}$ such that each of its output bits is computable by an NC^1 circuit of size $l(n)$ can be transformed into a PRG $\hat{G} : \{0,1\}^{n+s(n) \cdot l(n)^c} \to \{0,1\}^{n+s(n) \cdot l(n)^c + s(n)}$ in NC^0_4. Typically in [4],

$l(n)$ is superconstant and so the stretch $s(n)$ of the resulting generator is only *sub-linear* in its input length $n + s(n) \cdot l(n)^c$. However, when $G \in \mathrm{NC}^0$ each output bit is computable by a constant size circuit and so $l(n) = O(1)$. Therefore, if G is an LPRG in NC^0, i.e., $s(n) = \Theta(n)$ and $l(n) = O(1)$, then the stretch of the resulting PRG which is $s(n) = \Theta(n)$ is still linear in its input length $n + O(n) + O(n) \cdot O(1)$. Hence, we have:

Theorem 5.11 *Under Assumption 5.1, there exists an LPRG in* NC^0_4.

Mossel et al. [23] showed that a PRG in NC^0_4 cannot achieve a superlinear stretch. Hence, Theorem 5.11 is essentially optimal with respect to stretch.

Remarks on Theorem 5.11.

1. (Uniformity) Our construction uses two non-uniform advices: (1) a family of good ε-biased generators in NC^0 as in Lemma 5.8; and (2) a family of matrices $\{M_n\}$ satisfying Assumption 5.1. In Section 5.4 we eliminate the use of the first advice by proving a uniform version of Lemma 5.8. We can also eliminate the second advice and construct an LPRG in *uniform* NC^0_4 by using an explicit variant of Assumption 5.1. In particular, we follow Alekhnovich (cf. [1, Remark 1]) and conjecture that any family of matrices $\{M_n\}$ that represent graphs with good expansion satisfies Assumption 5.1. Hence, our construction can be implemented by using an explicit family of asymmetric constant-degree bipartite expanders such as the one given in [11, Theorem 7.1].

2. (The stretch of the construction) Our techniques do not yield a PRG with *superlinear* stretch in NC^0. To see this, consider a variant of Assumption 5.1 in which we allow $m(n)$ to be superlinear. If we let $\mu(n)$ to be a constant, then, by information-theoretic arguments, we need $\Omega(m(n))$ random bits to sample the noise vector (i.e., the entropy of the noise vector is $\Omega(m(n))$), and so we get only linear stretch. On the other hand, if we set $\mu(n)$ to be subconstant, then the noise distribution cannot be sampled in NC^0 (as any bit of an NC^0-samplable distribution depends on a constant number of random bits). This problem can be bypassed by extending Assumption 5.1 to alternative noise models in which the noise is not independently and identically distributed. However, it is not clear how such a modification affects the hardness assumption. (Also note that we do not know how to reduce the locality of a superlinear PRG in NC^0 while preserving its superlinear stretch. In particular, applying the transformations of [4] to such a PRG, will result in a *linear* PRG with locality 4.)

5.4 ε-Biased Generators in Uniform NC^0

In [23, Theorem 14], Mossel et al. constructed an ε-biased generator in *non-uniform* NC^0_5 with an arbitrary linear stretch cn and bias $\varepsilon = 2^{-\Omega(n/c^4)}$.[4] We generalize their construction and provide a complementary result which gives a better tradeoff between the bias and stretch and allows a uniform implementation. However, the locality of our construction grows with the stretch constant.

Theorem 5.12 *For every constant c, there exist an ε-biased generator $g : \{0,1\}^n \to \{0,1\}^{cn}$ in* uniform NC^0 *whose bias is $\varepsilon = 2^{-n/\mathrm{polylog}(c)}$ and its locality is $\ell = \mathrm{polylog}(c)$.*

As in [23], our generator is obtained by XORing the outputs of two functions: a generator $g^{(s)}$ which is robust against linear functions that involve small number of output bits ("small tests") and a generator $g^{(l)}$ which is robust against linear functions that involve large number of output bits ("large tests"). More precisely, for a random variable $X = (X_1, \ldots, X_n)$ ranging over $\{0,1\}^n$, a set $S \subseteq \{1, \ldots, n\}$, and an integer $0 < k \leq n$, we define

$$\begin{aligned}
\mathrm{bias}_S(X) &\stackrel{\text{def}}{=} \left| \Pr[\bigoplus_{i \in S} X_i = 0] - \frac{1}{2} \right|, \\
\mathrm{bias}_k(X) &\stackrel{\text{def}}{=} \max_{S \subseteq \{1,\ldots,n\}, |S|=k} \mathrm{bias}_S(X), \\
\mathrm{bias}(X) &\stackrel{\text{def}}{=} \max_{0<k\leq n} \mathrm{bias}_k(X) = \max_{S \subseteq \{1,\ldots,n\}, S \neq \emptyset} \mathrm{bias}_S(X) \ .
\end{aligned}$$

Then, we prove Theorem 5.12 by using the following two lemmas (whose proofs is postponed to Sections 5.4.1, 5.4.2):

Lemma 5.13 (Generator against small tests) *For every constant c, there exist a function $g^{(s)} : \{0,1\}^n \to \{0,1\}^{cn}$ in* uniform $\mathrm{NC}^0_{\mathrm{polylog}(c)}$ *such that for sufficiently large n's and every $0 < k \leq \Omega(n/\mathrm{polylog}(c))$, we have $\mathrm{bias}_k(g^{(s)}(U_n)) = 0$.*

Lemma 5.14 (Generator against large tests) *For every constant c, there exist a function $g^{(l)} : \{0,1\}^n \to \{0,1\}^{cn}$ in* uniform $\mathrm{NC}^0_{O(\log(c))}$ *such that for sufficiently large n's and every $k \in \{1, \ldots, cn\}$, we have $\mathrm{bias}_k(g^{(l)}(U_n)) \leq 2^{-k/5}$.*

Given these two lemmas we can prove Theorem 5.12.

Proof: (of Theorem 5.12) Let c be a constant. Let $g^{(s)} : \{0,1\}^n \to \{0,1\}^{2cn}$ and $g^{(l)} : \{0,1\}^n \to \{0,1\}^{2cn}$ be the generators promised by Lemmas 5.13, 5.14

[4] In fact, cn can be slightly super-linear.

(instantiate with the constant $2c$). Then, the function $g(x,y) = g^{(s)}(x) \oplus g^{(l)}(y)$ satisfies Theorem 5.12. To see this, observe that for any *independent* random variables X and Y and any non-uniform statistical test T, the success probability of T on the random variable $X \oplus Y$ is not larger than its success probability on X (or Y). ■

5.4.1 Proof of Lemma 5.13

Let M be an $m \times n$ matrix over $\mathbb{F}_2$ such that every subset of k rows of M are linearly independent. Then, it is well known that the function $f : \{0,1\}^n \to \{0,1\}^m$ that maps x into $M \cdot x$ is a k-wise independent generator (cf. [2]). That is, for every $0 < j \leq k$, we have $\mathrm{bias}_j(f(U_n)) = 0$. If each row of M contains at most ℓ ones then the function f is in NC^0_ℓ. It turns out that there exists a (uniform) family of such matrices whose parameters match the parameters of Lemma 5.13. Specifically, we use the following result which is a corollary of [11, Theorem 7.1].

Lemma 5.15 ([11]) *For every constant c there exists a family of matrices $\{M_n\}_{n\in\mathbb{N}}$ such that*

- *M_n is an $cn \times n$ matrix over $\mathbb{F}_2$.*
- *Every row of M_n has at most $\mathrm{polylog}(c)$ ones.*
- *Every subset of $k = \Omega(n/\mathrm{polylog}(c))$ rows of M_n are linearly independent.*
- *M_n can be constructed in time $\mathrm{poly}(n)$.*

Hence, the generator for small tests can be defined as $g^{(s)}(x) = M_n \cdot x$.

5.4.2 Proof of Lemma 5.14

We will need the following standard claim that can be proved via the probabilistic method (see [17, Lecture 8, Prop. 2.1]).

Claim 5.16 *For sufficiently large n, there exists an ε-biased generator $f : \{0,1\}^n \to \{0,1\}^{2^{n/2}}$ whose bias is $\varepsilon = 2^{-n/4}$.*

We can now prove Lemma 5.14. Let c be the desired stretch constant. Let $\ell = 4\log c$. Let $m = 2^{\ell/2}$ and $f : \{0,1\}^\ell \to \{0,1\}^m$ be an ε-biased generator whose bias is $\varepsilon = 2^{-\ell/4}$ as promised by Claim 5.16. (Since c is a constant, such f can be found by using exhaustive search.) Our generator will partition its n-bit input x into $b = \lfloor n/\ell \rfloor$ blocks $x^{(1)}, \ldots, x^{(b)}$ of length ℓ each. Then, the generator

will apply f to each block separately, and concatenate the result. Namely, $g^{(l)}(x) \stackrel{\text{def}}{=} (f(x^{(1)}), \ldots, f(x^{(b)}))$. The locality of $g^{(l)}$ is ℓ and its output length is $bm = \left\lfloor \frac{c^2 n}{4 \log c} \right\rfloor$ which is larger than cn for sufficiently large c.

We now analyze the bias of $g^{(l)}$. To simplify notation, we index the outputs of $g^{(l)}$ by pairs (j,i) and let $g^{(l)}_{j,i}(x) = f_i(x^{(j)})$ (where $1 \leq j \leq b$, $1 \leq i \leq m$ and $f_i(x)$ denotes the i-th output bit of $f(x)$). Let $S \subseteq \{1, \ldots b\} \times \{1, \ldots m\}$ be a linear test of cardinality k. Let S_j be the restriction of S to the indices of the j-th block, i.e., $S_j = \{i : (j,i) \in S\}$. Then, $S_1, \ldots, S_b$ is a partition of S. Let $T = \{i : S_i \neq \emptyset\} \subseteq \{1, \ldots, b\}$. Hence, for $x \leftarrow U_n$, we have

$$\mathrm{bias}_S(g^{(l)}(x)) \quad = \quad \mathrm{bias}\left(\bigoplus_{j \in T} \bigoplus_{i \in S_j} f_i(x^{(j)}) \right).$$

Since f is an ε-biased generator, for each $j \in T$ we have that $\mathrm{bias}(\bigoplus_{i \in S_j} f_i(x^{(j)})) \leq \varepsilon$. Since $g^{(l)}(x)$ is partitioned into blocks of length ℓ, the test S contains output bits coming from at least k/ℓ different blocks and so $|T| \geq k/\ell$. Thus we can use Fact 2.6 to upper bound $\mathrm{bias}_S(g^{(l)}(x))$ by

$$\frac{1}{2}(2\varepsilon)^{k/\ell} \leq \frac{1}{2}(2^{-\ell/4+1})^{k/\ell} \leq \frac{1}{2}(2^{-\ell/5})^{k/\ell} \leq 2^{-k/5},$$

as required. ■

6 The Necessity of Expansion

As pointed out in Section 5, our construction of LPRG makes use of expander graphs. This is also the case in several constructions of "hard functions" with low locality (e.g., [15, 23, 1]). We argue that this is not coincidental, at least in the case of PRGs. Namely, we show that the input-output graph of any LPRG in NC^0 enjoys some expansion property. (In fact, this holds even in the case of ε-biased generators.) Then, we use known lower bounds for expander graphs to rule out the possibility of exponentially strong PRG with superlinear stretch in NC^0. These results are discussed from a wider perspective in Section 6.2. We start with the technical results.

6.1 Actual Results

For a function $G : \{0,1\}^n \to \{0,1\}^s$, we define the input-output graph $H_G = ((\mathrm{Out} = [s], \mathrm{In} = [n]), E)$ to be the bipartite graph whose edges correspond to

the input-output dependencies in G; that is, (i,j) is an edge if and only if the i-th output bit of G depends on the j-th input bit. When G is a function family, H_G denotes a graph family.

Proposition 6.1 *Let $G : \{0,1\}^n \to \{0,1\}^{s(n)}$ be a PRG. Then, the graph (family) $H_G = ((\mathrm{Out} = [s(n)], \mathrm{In} = [n]), E)$ enjoys the following output expansion property: for every constant c and sufficiently large n, every set of output vertices $T \subseteq \mathrm{Out}$ whose size is at most $c \log n$ touches at least $|T|$ input vertices.*

Proof: Assume towards a contradiction that there exists a small set T of output vertices that touches less than $|T|$ input vertices. Let $G_T(\cdot)$ be the restriction of G to the output bits of T. Then, the function $G_T(\cdot)$ cannot be onto as it depends on less than $|T|$ input bits. Therefore, there exists a string $z \in \{0,1\}^{|T|}$ such that $\Pr[G_T(U_n) = z] = 0$. Hence, a (non-uniform) distinguisher which given $y \in \{0,1\}^{s(n)}$ checks whether $y_T = z$, distinguishes between $G(U_n)$ and $U_{s(n)}$ with advantage $2^{-c \log n} = 1/n^c$, in contradiction to the pseudorandomness of G. ■

More generally, if G is ε-hard (i.e., cannot be broken by any efficient adversary with advantage ε), then every set of $t \leq \log(1/\varepsilon)$ output vertices touches at least t input vertices. This claim also extends to the case of ε-biased generators.

Proposition 6.2 *Let $G : \{0,1\}^n \to \{0,1\}^s$ be an ε-biased generator. Then, every set of $t \leq \log(1/\varepsilon)$ output vertices in H_G touches at least t input vertices.*

Proof: Assume towards a contradiction that there exists a set T of output vertices of size $t \leq \log(1/\varepsilon)$ that touches less than t input vertices. Then $G_T(U_n) \not\equiv U_t$. Therefore, there exists a linear function $L : \mathbb{F}_2^t \to \mathbb{F}_2$ that distinguishes between $G_T(U_n)$ and U_t. Namely, $|\Pr[L(G_T(U_n)) = 1] - \Pr[L(U_t) = 1]| \neq 0$. Since the distribution $G_T(U_n)$ is sampled by less than t random bits, the distinguishing advantage of L is larger than $2^{-t} \geq \varepsilon$, and so G is not ε-biased in contradiction to the hypothesis. ■

The above propositions show that when G is an ε-hard PRG (or even ε-biased generator), the bipartite graph $H_G = ((\mathrm{Out} = [s(n)], \mathrm{In} = [n]), E)$ enjoys some output expansion property. Radhakrishnan and Ta-Shma [28] obtained some lower bounds for such graphs.

Proposition 6.3 ([28], Theorem 1.5) *Let $H = ((V_1 = [s], V_2 = [n]), E)$ be a bipartite graph in which every set $S \subseteq V_1$ of cardinality k touches at least m vertices from V_2. Then, the average degree of V_1 is at least $\Omega\left(\frac{\log(s/k)}{\log(m/n)}\right)$*

By combining this lower bound with the previous propositions we derive the following limitation on the strength of PRGs with superlinear stretch in NC^0.

Corollary 6.4 *Let $G : \{0,1\}^n \to \{0,1\}^s$ be a 2^{-t}-hard PRG (or 2^{-t}-biased generator). Then, the locality of G is at least $\Omega\left(\frac{\log(s/t)}{\log(n/t)}\right)$. In particular, there is no $2^{-\Omega(n)}$-hard PRG, or even a $2^{-\Omega(n)}$-biased generator, with superlinear stretch in NC^0.*

6.2 Discussion

To put the above results in context, some background on unbalanced bipartite expanders is needed. Consider a bipartite graph $H = ((\mathrm{Out} = [s], \mathrm{In} = [n]), E)$ in which each of the output vertices is connected to at most d inputs. Recall that H is a (K, α)-expander if every set of output vertices S of size smaller than K has at least $\alpha \cdot |S|$ input neighbors. We say that the expander is unbalanced if $s > n$. Unbalanced expanders have had numerous applications in computer science (see details and references in [11]). Today, there are only two such constructions [31, 11]. Ta-Shma et al. [31] considered the highly unbalanced case in which $n < o(s)$. They constructed a (K, α)-expander with degree $d = \mathrm{polylog}(s)$, expansion threshold $K < s^{\varepsilon}$ and almost optimal expansion factor $\alpha = (1 - \delta)d$, where $\delta > 0$ is an arbitrary constant. Capalbo et al. [11] present a construction for the setting in which n is an (arbitrary) constant fraction of s (i.e., $s = n + \Theta(n)$). They construct a (K, α)-expander with (nearly) optimal parameters; Namely, the degree d of the graph is constant, and its expansion parameters are $K = \Omega(s)$ and $\alpha = (1 - \delta)d$, where $\delta > 0$ is an arbitrary constant.

In Section 6.1 we showed that if $G : \{0,1\}^n \to \{0,1\}^s$ is a PRG then its input-output graph $H_G = ((\mathrm{Out} = [s], \mathrm{In} = [n]), E)$ is an $(\omega(\log n), 1)$-expander. This property is trivial to satisfy when the output degree of H_G is unbounded (as in standard constructions of PRGs in which every output bit depends on all the input bits). It is also easy to construct such a graph with constant output degree when $s(n)$ is not much larger than n (as in the NC^0 constructions of [4]).

To see this, consider the following bipartite graph. First, let $C = ((O, I), D)$ be a bipartite graph over $[2n + 1]$ whose output vertices are the odd integers, its input vertices are the even integers, and its edges correspond to pairs of consecutive integers, i.e., $O = \{1, 3, \ldots, 2n + 1\}$, $I = \{2, 4, \ldots, 2n\}$, and $D = \{(1, 2), (2, 3), \ldots, (2n, 2n + 1)\}$. That is, C is a chain of length $2n + 1$. Let $m > n$. Take m disjoint copies of C, and let O_i (resp. I_i) be the set of output (resp. input) vertices of the i-th copy. In addition, add n input vertices $I_0 = [n]$ and match them to the first n vertices of each of the output clusters (i.e., connect the j-th vertex of I_0 to the vertex $2j - 1$ of each O_i). Let $H = ((\mathrm{Out} = O_1 \cup \cdots \cup O_m, \mathrm{In} = I_1 \cup \cdots \cup I_m \cup I_0), E)$. (See Figure 1.) Clearly, H has $m(n + 1)$ output vertices, $mn + n$ input vertices, and each output vertex is connected to at most 3 inputs. It

is not hard to verify that H is $(n^2, 1)$-expanding. However, the number of outputs is only slightly larger than the number of inputs; i.e., $|\mathrm{Out}| - |\mathrm{In}| = m - n < m$ which is sublinear in $|\mathrm{In}|$ when n is non-constant.

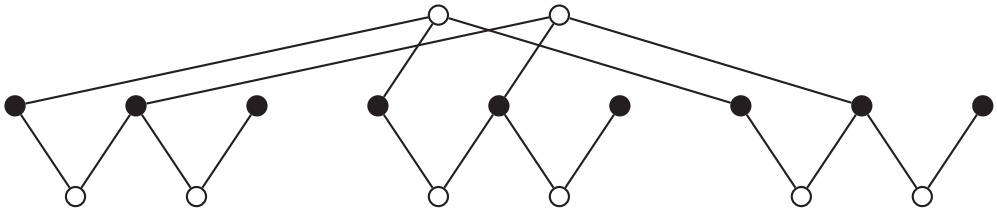

Figure 1: The graph H with $n = 2$ and $m = 3$. Black circles denote output vertices while empty circles denote input vertices.

However, when the locality of the pseudorandom generator G is constant and the stretch is linear, H_G is a sparse bipartite graph having n input vertices, $s(n) = n + \Omega(n)$ output vertices, and a constant output degree. It seems that it is not trivial to explicitly construct such a graph that achieves $(\omega(\log n), 1)$-expansion. (Indeed, the construction of [11] gives similar graphs whose expansion is even stronger, but this construction is highly non-trivial.) Hence, any construction of LPRG in NC^0 defines a non-trivial combinatorial structure. In particular, one cannot hope that "simple" *deterministic* transformations, such as those given in [4], will yield LPRGs in NC^0.

Note that an exponentially strong PRG (or exponentially strong ε-biased generator) with linear stretch gives an $(\Omega(n), 1)$-expander graph whose output size grows linearly with its input size. Indeed, the exponentially strong ε-biased generator of [23] is based on a similar (but slightly stronger) unbalanced expander. The above argument shows that such an ingredient is necessary.

Acknowledgments. We thank Eli Ben-Sasson, Amir Shpilka and Amnon Ta-Shma for helpful discussions. We also thank Oded Goldreich for many useful suggestions which improved the presentation of this paper.

References

[1] M. Alekhnovich. More on average case vs approximation complexity. In *Proc. 44th FOCS*, pages 298–307, 2003.

[2] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. of Algorithms*, 7(4):567–583, 1986.

[3] N. Alon and Y. Roichman. Random cayley graphs and expanders. *Random Struct. Algorithms*, 5(2):271–285, 1994.

[4] B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006. Preliminary version in Proc. 45th FOCS, 2004.

[5] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. *J. of the ACM*, 45(3):501–555, 1998. Preliminary version in Proc. 33rd FOCS, 1992.

[6] S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of np. *J. of the ACM*, 45(1):70–122, 1998. Preliminary version in Proc. 33rd FOCS, 1992.

[7] B. Barak, R. Shaltiel, and A. Wigderson. Computational analogues of entropy. In *Proc. 7th Conference on Randomization and Computation, (RANDOM)*, 2003.

[8] E. Ben-Sasson, M. Sudan, S. Vadhan, and A. Wigderson. Randomness-efficient low-degree tests and short pcps via epsilon-biased sets. In *Proc. 35th STOC*, pages 612–621, 2003.

[9] A. Blum, M. Furst, M. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *Advances in Cryptology: Proc. of CRYPTO '93*, volume 773 of *LNCS*, pages 278–291, 1994.

[10] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM J. Comput.*, 13:850–864, 1984. Preliminary version in Proc. 23rd FOCS, 1982.

[11] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree lossless expanders. In *Proc. 34th STOC*, pages 659–668, 2002.

[12] M. Cryan and P. B. Miltersen. On pseudorandom generators in NC^0. In *Proc. 26th MFCS*, pages 272–284, 2001.

[13] Y. Dodis and A. Smith. Correcting errors without leaking partial information. In *Proc. 37th STOC*, pages 654–663, 2005.

[14] U. Feige. Relations between average case complexity and approximation complexity. In *Proc. of 34th STOC*, pages 534–543, 2002.

[15] O. Goldreich. Candidate one-way functions based on expander graphs. *Electronic Colloquium on Computational Complexity (ECCC)*, 7(090), 2000.

[16] O. Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.

[17] O. Goldreich. Randomized methods in computation - lecture notes, 2001. http://www.wisdom.weizmann.ac.il/∼oded/rnd.html.

[18] O. Goldreich. *Foundations of Cryptography: Basic Applications*. Cambridge University Press, 2004.

[19] O. Goldreich, H. Krawczyk, and M. Luby. On the existence of pseudorandom generators. *SIAM J. Comput.*, 22(6):1163–1175, 1993. Preliminary version in Proc. 29th FOCS, 1988.

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

[21] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[22] J. Håstad, A. W. Schrift, and A. Shamir. The discrete logarithm modulo a composite hides $\mathrm{O}(\mathrm{n})$ bits. *JCSS*, 47(3):376–404, 1993. Preliminary version in Proc. 22nd STOC, 1990.

[23] E. Mossel, A. Shpilka, and L. Trevisan. On ϵ-biased generators in NC^0. In *Proc. 44th FOCS*, pages 136–145, 2003.

[24] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. Preliminary version in Proc. 22th STOC, 1990.

[25] M. Naor. Bit commitment using pseudorandomness. *J. of Cryptology*, 4:151–158, 1991.

[26] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. Preliminary version in Proc. 38th FOCS, 1997.

[27] C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *J. of Computer and Systems Sciences*, 43:425–440, 1991. Preliminary version in Proc. 20th STOC, 1988.

[28] J. Radhakrishnan and A. Ta-Shma. Tight bounds for depth-two superconcentrators. *SIAM J. Discrete Math.*, 13(1):2–24, 2000. Preliminary version in Proc. 38th FOCS, 1997.

[29] A. Sahai and S. Vadhan. A complete problem for statistical zero knowledge. *J. of the ACM*, 50(2):1–54, 2003.

[30] A. Shpilka. Constructions of low-degree and error-correcting e-biased generators. In *Proc. 21st Conference on Computational Complexity (CCC)*, pages 33–45, 2006.

[31] A. Ta-Shma, C. Umans, and D. Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proc. 33rd STOC*, pages 143–152, 2001.

[32] E. Viola. On constructing parallel pseudorandom generators from one-way functions. In *Proc. 20th Conference on Computational Complexity (CCC)*, pages 183– 197, 2005.

[33] A. C. Yao. Theory and application of trapdoor functions. In *Proc. 23rd FOCS*, pages 80–91, 1982.

A Alekhnovich's Assumption Implies Assumption 5.1

We show that Alekhnovich's Assumption [1, Conjecture 2, Remark 1] implies Assumption 5.1. The main difference between the two assumptions is that the noise vector e in [1] is a random vector of weight exactly $\lceil \mu m \rceil$, as opposed to our noise vector whose entries are chosen to be 1 independently with probability μ. The implication follows from the fact that our noise vectors can be viewed as a convex combination of noise vectors of fixed weight. We give the details below.

Recall that for an $m \times n$ matrix M we let $\hat{D}_\mu(M)$ denote the distribution of $M \cdot x + e$, where x is a random n-bit vector and e is a noise vector which is uniformly distributed over all m-bits vectors whose Hamming weight is $\mu \cdot m$. The distribution $D_\mu(M) \stackrel{\text{def}}{=} M \cdot x + e$ is analogous to $\hat{D}_\mu(M)$, except that each entry of the noise vector e is chosen to be 1 with probability μ (independently of other entries).

Assumption A.1 ***(Alekhnovich's Assumption)*** *For any $m(n) = O(n)$, there exists an infinite family of matrices $\{M_n\}_{n \in \mathbb{N}}$, $M_n \in \mathcal{M}_{m(n),n,3}$, such that for any*

constant $0 < \mu_0 < 1/2$*, and function* $\mu(n)$ *that satisfies* $\mu_0 < \mu(n) < 1/2$ *for every* n*, it holds that*

$$\hat{D}_{\mu(n)}(M_n) \stackrel{c}{\approx} \hat{D}_{\mu(n)+m(n)^{-1}}(M_n).$$

Fix a matrix family $\{M_n\}_{n\in\mathbb{N}}$ of size $m(n) \times n$ where $m(n)$ is an integer valued function. We will prove that Assumption A.1 instantiated with the family $\{M_n\}_{n\in\mathbb{N}}$ implies Assumption 5.1 instantiated with the same family of matrices. To do this we use the following two intermediate assumptions.

Assumption A.2 *For any constant* $0 < \mu_0 < 1/2$*, and function* $\mu(n)$ *that satisfies* $\mu_0 < \mu(n) < 1/2$ *for all* n*'s,* $\hat{D}_{\mu(n)}(M_n) \stackrel{c}{\approx} U_{m(n)}$.

Assumption A.3 *For any constant* $0 < \mu < 1/2$*, we have* $D_\mu(M_n) \stackrel{c}{\approx} U_{m(n)}$.

In [1, Thm. 3.1] it is shown that Assumption A.1 implies Assumption A.2. Hence to prove that Assumption A.1 implies Assumption 5.1 it suffices to show that: (1) Assumption A.2 implies Assumption A.3; and (2) Assumption A.3 implies Assumption 5.1.

Lemma A.4 *Assumption A.2 implies Assumption A.3.*

Proof: Suppose that Assumption A.3 does not hold. Then, for some constant $0 < \mu < 1/2$, the distribution $D_\mu(M_n)$ is not pseudorandom. That is, there exists a polynomial-size circuit family $\{A_n\}$ and a polynomial $q(\cdot)$ such that

$$\Pr[A_n(D_\mu(M_n)) = 1] - \Pr[A_n(U_{m(n)}) = 1] > 1/q(n), \tag{8}$$

for infinitely many n's. We will show that, for some constant $0 < \hat{\mu}_0 < 1/2$, and function $\hat{\mu}(n)$ that satisfies $\hat{\mu}_0 < \hat{\mu}(n) < 1/2$, Assumption A.2 is violated. Namely, $\Pr[A_n(\hat{D}_{\hat{\mu}(n)}(M_n)) = 1] - \Pr[A_n(U_{m(n)}) = 1] > 1/q'(n)$ for some polynomial $q'(\cdot)$ and infinitely many n's.

Fix some n for which Eq. 8 holds, and let $m = m(n)$. Let $p \stackrel{\text{def}}{=} \Pr[A_n(D_\mu(M_n)) = 1]$ and $p(k) \stackrel{\text{def}}{=} \Pr[A_n(\hat{D}_{k/m}(M_n)) = 1]$ for $0 \le k \le m$. Let $e \in \{0,1\}^m$ be a random error vector in which each entry is chosen to be 1 with probability μ (independently of other entries) and let $t(k)$ be the probability that e contains exactly k ones. We can think of the distribution of e as the outcome of the following process: first choose $0 \le k \le m$ with probability $t(k)$, then choose a random noise vector of weight k. Hence, we can write,

$$p = \sum_{k=0}^{m} p(k) \cdot t(k).$$

Let $\varepsilon > 0$ be a constant for which $\mu \cdot \varepsilon < 1/2$. Then, by a Chernoff bound, it holds that

$$\sum_{k<(1-\varepsilon)\cdot\mu m} t(k) + \sum_{k>(1+\varepsilon)\cdot\mu m} t(k) = \Pr\left[\left|\sum_{i=1}^{m} e_i - \mu m\right| > \varepsilon \cdot \mu m\right] < 2e^{-\varepsilon^2\mu m/3}.$$

Hence,

$$\sum_{(1-\varepsilon)\cdot\mu m\leq k\leq(1+\varepsilon)\cdot\mu m} p(k)\cdot t(k) > p - 2e^{-\varepsilon^2\mu m/3}.$$

Thus, by an averaging argument, there exists some $(1-\varepsilon)\cdot\mu m \leq k \leq (1-\varepsilon)\cdot\mu m$ for which

$$p(k) > p - 2e^{-\varepsilon^2\mu m/3}.$$

Let $\hat{\mu}(n)$ be k/m and let $\hat{\mu}_0$ be the constant $(1-\varepsilon)\cdot\mu m/2$. Then, by Eq. 8, we have

$$\Pr[A_n(\hat{D}_{\hat{\mu}(n)}(M_n)) = 1] - \Pr[A_n(U_{m(n)}) = 1] > 1/q(n) - 2e^{-\varepsilon^2\mu m/3} > 1/q'(n),$$

where $q'(\cdot)$ is a polynomial. This completes the proof since $\hat{\mu}_0 < \hat{\mu}(n) < 1/2$ for every n. ■

It is left to prove the following lemma.

Lemma A.5 *If Assumption A.3 holds then Assumption 5.1 also holds with respect to* $\{M_n\}_{n\in\mathbb{N}}$.

Proof: As shown in the proof of Lemma 5.3 we can write $D_{\mu+m^{-1}}(M_n) \equiv D_\mu(M_n) + r_n$, where r_n denotes the distribution of an m-bit vector in which each entry is chosen to be 1 with probability c/m (independently of other entries) for some constant c. Hence, by two invocations of Assumption A.3, we have

$$D_{\mu+m^{-1}}(M_n) \equiv D_\mu(M_n) + r_n \stackrel{c}{\approx} U_{m(n)} + r_n \equiv U_{m(n)} \stackrel{c}{\approx} D_\mu(M_n).$$

■