On ε-Biased Generators in NC^0

Elchanan Mossel[*]
Statistics
U.C. Berkeley
Berkeley, CA 94720-1776
mossel@stat.berkeley.edu

Amir Shpilka [†]
DEAS , LCS
Harvard M.I.T.
Cambridge, MA 02138
amirs@deas.harvard.edu

Luca Trevisan
Computer Science
U.C. Berkeley
Berkeley, CA 94720-1776
luca@cs.berkeley.edu

Abstract

Cryan and Miltersen [7] recently considered the question of whether there can be a pseudorandom generator in NC^0, that is, a pseudorandom generator that maps n bits strings to m bits strings and such that every bit of the output depends on a constant number k of bits of the seed.

They show that for $k = 3$, if $m \geq 4n + 1$, there is a distinguisher; in fact,they show that in this case it is possible to break the generator with a linear test*, that is, there is a subset of bits of the output whose XOR has a noticeable bias.*

They leave the question open for $k \geq 4$. In fact they ask whether every NC^0 generator can be broken by a statistical test that simply XORs some bits of the input. Equivalently, is it the case that no NC^0 generator can sample an ε-biased space with negligible ε?

We give a generator for $k = 5$ that maps n bits into cn bits, so that every bit of the output depends on 5 bits of the seed, and the XOR of every subset of the bits of the output has bias $2^{-\Omega(n/c^4)}$. For large values of k, we construct generators that map n bits to $n^{\Omega(\sqrt{k})}$ bits and such that every XOR of outputs has bias $2^{-n^{\frac{1}{2\sqrt{k}}}}$.

We also present a polynomial-time distinguisher for $k = 4, m \geq 24n$ having constant distinguishing probability. For large values of k we show that a linear distinguisher with a constant distinguishing probability exists once $m \geq \Omega(2^k n^{\lceil k/2 \rceil})$.

Finally, we consider a variant of the problem where each of the output bits is a degree k polynomial in the inputs. We show there exists a degree $k = 2$ pseudo random generator for which the XOR of every subset of the outputs has bias $2^{-\Omega(n)}$ and which map n bits to $\Omega(n^2)$ bits.

1 Introduction

A pseudorandom generator is an efficient deterministic procedure that maps a shorter random input into a longer output that is indistinguishable from the uniform distribution by resource-bounded observers.

A standard formalization of the above informal definition is to consider polynomial-time procedures G mapping n bits into $m(n) > n$ bits such that for every property P computable by a family of polynomial-size circuits we have that the quantity

$$\left| \mathop{\mathbf{Pr}}_{z \in \{0,1\}^{l(n)}} [P(z) = 1] - \mathop{\mathbf{Pr}}_{x \in \{0,1\}^n} [P(G(x))] \right|$$

goes to zero faster than any inverse polynomial in n. The existence of such a procedure G is equivalent to the existence of one-way functions [13], pseudorandom functions [9] and pseudorandom permutations [20].

What are the minimal computational requirements needed to compute a pseudorandom generator? Linial et al. [17] prove that pseudorandom functions cannot be computed in AC^0 (constant-depth circuits with NOT gates and unbounded fan-in AND and OR gates),[1] but their result does not rule out the possibility that pseudorandom generators could be computed in AC^0, since the transformation of pseudorandom generators into pseudorandom functions does not preserve bounded-depth.

Impagliazzo and Naor [15], in fact, present a candidate pseudorandom generator in AC^0. Goldreich [10] suggests a candidate one-way function in NC^0. Recall that NC^0 is the class of functions computed by bounded-depth circuits with NOT gates and bounded fan-in AND and OR gates. In an NC^0 function, every bit of the output depends on a constant number of bits of the inputs. While it is easy to see that there can be no one-way function such that every bit of the output depends on only two bits of the input,[2] it still remains open

[*] Supported by a Miller fellowship

[†] Supported by National Security Agency (NSA) and Advanced Research and Development Activity (ARDA) under Research Office (ARO) contract no. DAAD19-01-1-0506.

[1] To be precise, the results in [17] only rule out security against adversaries running in time $O(n^{(\log n)^{O(1)}})$.

[2] Finding an inverse can be formulated as a 2SAT problem.

whether there can be a one-way function such that every bit of the output depends on only three bits of the input.

Cryan and Miltersen [7] consider the question of whether there can be pseudorandom generators in NC^0, that is, whether there can be a pseudorandom generator such that every bit of the output depends only on some a constant k number of bits of the input.

They present a distinguisher in the case $k = 3, m > 4n$, and they observe that their distinguisher is a *linear* distinguisher, that is, it simply XORs a subset of the bits of the output. Cryan and Miltersen ask if there is no pseudorandom generator in NC^{0}when m is superlinear in n. Specifically, they ask if it is the case that for every constant k if m is super-linear in n then for every generator such that every bit of the output depends on k bits of the input, a linear distinguisher exist.

In order to formulate an equivalent version of this problem, we introduce the notion of a ε*-biased* distribution. For $\varepsilon > 0$, we say that a random variable $X = (X_1, \ldots, X_m)$ ranging over $\{0,1\}^m$ is ε-biased if for every subset $S \subseteq [m]$ we have $1/2 - \varepsilon \leq \mathbf{Pr}[\bigoplus_{i \in S} X_i = 0] \leq 1/2 + \varepsilon$. It is known [23, 2] that an ε-biased distribution can be sampled by using only $O(\log(m/\varepsilon))$ random bits, which is tight up to the constant in the big-Oh.

The problem of [7] can be therefore formulated as asking if there is no ε-biased generator in NC^0 that samples an m-bit ε-biased distribution starting from, say, $o(m)$ random bits and with a negligible ε.

Our Results

We first extend the result of Cryan and Miltersen by giving a (non linear) distinguisher for the case $k = 4, m \geq 24n$. Our distinguisher has a constant distinguishing probability, which we show to be impossible to achieve with linear distinguishers. Our distinguisher uses semidefinite programming and uses an idea similar to the "correlation attacks" used in practice against stream cyphers.

For all k, it is trivial that a distinguisher exists for $m \geq 2^{2^k}\binom{n}{k}$, and it easy to see that a distinguisher exist when $m \geq k\binom{n}{k}$. We show using a duality lemma proven in [22] that in fact, a distinguisher with a constant distinguishing probability exists once $m \geq \Omega(2^k n^{\lceil k/2 \rceil})$.

Then we present an ε-biased generator mapping n bits into cn bits such that $\varepsilon = 1/2^{\Omega(n/c^4)}$ and every bit of the output depends only on $k = 5$ bits of the seed. The parameter c can be chosen arbitrarily, and may depend on n. The constant in the $\Omega()$ notation does not depend on c.

The main idea in the construction is to develop a generator with $k = 3$ that handles well linear tests that XOR a *small* number of bits, and then develop a generator with $k = 2$ that handles well linear tests that XOR a *large* number of bits. The final generator outputs the bitwise XOR of the outputs of the two generators, on two independent seeds.

The generator uses a kind of unique-neighbor expander graphs that are shown to exist using the probabilistic method, but that are not known to be efficiently constructable, so the generator is in NC^0 but not in *uniform* NC^0.

Later we present similar constructions for large values of k which output $n^{\lfloor \sqrt{k} \rfloor \cdot (\frac{1}{2} - o(1))}$ bits whose bias is at most $\exp\left(-|\mathrm{n}|^{\frac{1-\mathrm{o}(1)}{2\lfloor\sqrt{\mathrm{k}}\rfloor}}\right)$.

Note the gap for large values of k between our constructions that output $n^{(\sqrt{k}/2)(1-o(1))}$ bits, and the bounds showing a distinguisher exists for generators that output $n^{(k/2)(1+o(1))}$ bits.

Finally, we begin a study of the question of whether there are pseudorandom generators with superlinear stretch such that each bit of the output is a function of the seed expressible as a degree-k polynomial over $GF(2)$, where k is a constant. This is a generalization of the main question addressed in this paper, since a function depending on only k inputs can always be expressed as a degree-k polynomial. Furthermore, low-degree polynomials are a standard class of "low complexity" functions from an algebraic perspective. In our NC^0_5 construction of an ε-biased generator with exponentially small ε and superlinear stretch, every bit of the output is a degree-2 polynomial. We show that, for degree-2 polynomials, the stretch can be improved to quadratic, which is optimal up to a constant factor.

Organization

In section 2 we review the analysis for the case $k = 3$ of [7]. In section 3 we give a distinguisher for the case $k = 4$. In section 4 we prove an upper bound on the length of the output of an ε-bias generator in NC^0_k.

In section 5 we construct ε-bias generator for the cases $k = 4, 5$. The results for larger k are discussed in section 6. In section 7 we explicitly construct an ε-bias generator such that every bit of the output is a polynomial of degree 2. Finally we give some open problems in section 8.

2 Review of the Case $k = 3$

In this section we summarize the main result of [7]. We also generalize some of the arguments of [7] that are needed for our results.

2.1 Preliminaries

We say that a function $g : \{0,1\}^n \to \{0,1\}$ is *balanced* if $\mathbf{Pr}_x[g(x) = 1] = 1/2$. We say that a function $g : \{0,1\}^n \to \{0,1\}$ is *unbiased* towards a function $f : \{0,1\}^n \to \{0,1\}$ if $\mathbf{Pr}_x[g(x) = f(x)] = 1/2$.

A function $g : \{0,1\}^n \to \{0,1\}$ is *affine* if there are values $a_0, \ldots, a_n \in \{0,1\}$ such that $g(x_1, \ldots, x_n) = a_0 \oplus a_1 x_1 \oplus \ldots \oplus a_n x_n$.

The following lemma was proved by case analysis for $k = 3$ in [7], and the case $k = 4$ could also be derived from a case analysis appearing in [7] (but it is not explicitly stated). The proof of the general case follows using the Fourier representation of boolean functions and is omitted here.

Lemma 1 *Let $g : \{0,1\}^n \to \{0,1\}$ be a non-affine function that depends on only k variables. Then*

- *There exist an affine function on at most $k-2$ variables that is correlated with g.*
- *Let l be the affine function that is biased towards g and that depends on a minimal number of variables. That is, for some d, l depends on d variables, $\Pr_x[g(x) = l(x)] > 1/2$, and g is unbiased towards affine functions that depend on less than d variables.*

  Then $\Pr_x[g(x) = l(x)] \geq 1/2 + 2^{d-k}$.

For example, for $k = 3$, a non-affine function g is either unbalanced, or it is biased towards one of its inputs; in the latter case it agrees with an input bit (or with its complement) with probability at least $3/4$.

For $k = 4$, a function g either is affine, or it is unbalanced, or it has agreement at least $5/8$ with an affine function that depends on only one input bit, or it has agreement at least $3/4$ with an affine functions that depends on only two input bits.

2.2 The Case $k = 3$

Let $G : \{0,1\}^n \to \{0,1\}^m$ be a generator and let $g_i : \{0,1\}^n \to \{0,1\}$ be the i-th bit of the output of the generator. Suppose each g_i depends on only three bits of the input.

Suppose that one of the g_i is not a balanced function. Then we immediately have a distinguisher.

Suppose that more than n of the g_i are affine. Then one of them is linearly dependent of the others, and we also have a distinguisher.

It remains to consider the case where at least $m-n$ of the functions g_i are balanced and not affine. Let I be the set of i for which g_i is as above. Then, by lemma 1, for each such g_i there is a affine function l_i that depends on only *one* bit, such that g_i agrees with l_i on a $3/4$ fraction of the inputs. By replacing g_i with $g_i \oplus 1$ when needed, we may assume that each such g_i has high correlation with one of the bits of its input.

By the pigeonhole principle, there is a bit x_j of the seed, and a set C, $|C| \geq 1 + (m-n-1)/n$, such that the output of $g_i(x_1, \ldots, x_n)$ is correlated to x_j for every $i \in C$.

Lemma 2 *For every $\delta > 0$ there are constant $c_\delta = O(1/\delta^2)$ and $\varepsilon_\delta = O(1/\delta^2)$ such that the following holds. Let $G : \{0,1\}^n \to \{0,1\}^m$, and let $G(x) = (g_1(x), \ldots, g_m(x))$. Let L be a set of functions and suppose that each function $g_i(x)$ agrees with an element of L or with its complement with probability at least $1/2+\delta$, and that $m \geq 1 + c_\delta |L|$; then there are $i \neq j$ such that $g_i \oplus g_j$ has bias at least ε_δ.*

In order to prove the lemma let $g_1, \ldots, g_c$ have correlation at least $1/2 + \delta$ with the same bit x_i. Note that the avergae of $Z(x) = |\{\#i \in C : g_i(x) = 0\} - \{\#i \in C : g_i(x) = 1\}|$ is at least $2c\delta$. For $c = O(\delta^{-2})$ is a sufficiently large constant, then the restriction of the generator to C has constant statistical distance from the uniform distribution over c bits, for which that average value of Z is $O(\sqrt{c})$. By the Vazirani XOR lemma [27], it also follows that the XOR of some subset of the bits of C has bias $\Omega(2^{-|C|}) = \Omega(2^{-\delta^{-2}})$.

Alternitively, we note that $Z^2 = \sum_{i,j} Z_{i,j}$, where $Z_{i,j} = 1_{g_i = g_j} - 1_{g_i \neq g_j}$. Therefor truly random bits, $\mathbf{E}[Z^2] = c$, while for the pseudo random generator, $\mathbf{E}[Z^2] \geq \mathbf{E}[|Z|]^2 \geq 4c^2\delta^2$. So for $c = O(\delta^{-2})$ sufficiently large constant, there must be $i \neq j$ such that $g_i \oplus g_j$ has a $O(\delta^2)$ bias.

While the above analysis uses the same ideas as in [7], it is slightly better because we achieve constant bias instead of inverse polynomial bias.

In particular, we can compute that when we flip 4 random coins, the average of the maximum between the number of zeroes and ones is $2.75 < \frac{3}{4} \cdot 4$, so we can set $c_{1/4} = 3$. In particular, we obtain a constant distinguishing probability once $m \geq 4n + 1$.

For the next section, it is useful to note that when we flip 10 random coins, the average of the maximum between the number of zeroes and ones is $6.23 < \frac{5}{8} \cdot 10$, so we can set $c_{1/8} = 9$.

3 Distinguisher for the Case $k = 4$

In this section we construct a distinguisher for $k = 4$.

Theorem 3 *Let $G = (g_1, \ldots, g_m) : \{0,1\}^n \to \{0,1\}^m$ be a map such that each g_i depends on at most 4 coordinates of the input and $m \geq 24n$. Then there exists a polynomial time algorithm which distinguish between G and a random string with constant distinguishing probability. More precisely, the algorithm will output "yes" for the output of the generator G with probability $\Omega(1)$, and for a random string with probability $e^{-\Omega(m)}$.*

Note that it is easy to construct a distinguisher if any of the g_i is unbalanced, or if more than n of the g_i are linear.

If one of the g_i is biased towards one of the bits of its input, then it follows from lemma 1 that it must agree with that bit or its complement with probability at least $5/8$.

Thus, if more than $c_{1/8}n = 9n$ of the functions g_i have bias towards one bit, then we can obtain a distinguisher from lemma 2.

It remains to consider the case where at least $m - 10n$ of the functions are balanced, non-linear, and unbiased towards single bits. Following [7], we call such functions *problematic*. It follows from lemma 1 that for each problematic g there is an affine function l of two variables that agrees with g on a $3/4$ fraction of the inputs. Again, by replacing g_i by $g_i \oplus 1$, when needed, we may assume that all the $g_i's$ in P have $3/4$ agreement probability with some linear function.

Let P be the set of i such that g_i is problematic. For each such i we denote by l_i the linear function of two inputs that agrees with g_i on a $3/4$ fraction of the inputs. In the next section we show how if $p = |P| \geq 14n$, one can "break" the generator using correlation attack. Correlation attacks are often used in practice to break pseudo random generators. The distinguisher below is a an interesting example where one can actually prove that correlation attack results in a polynomial time distinguisher.

3.1 The Distinguisher Based on Semidefinite Programming

Given a string $r_1, \ldots, r_p \in \{0,1\}^p$, consider the following linear system over $GF(2)$ with two variables per equation.

$$\forall i \in P \quad l_i(x) = r_i \tag{1}$$

We will argue that the largest fraction of satisfying assignments in the system (1) is distributed differently if $r_1, \ldots, r_p$ is uniform or if it is the output of G. By Markov inequality it follows that,

Lemma 4 *If $r_1, \ldots, r_p$ are the output of $g_1, \ldots, g_p$, respectively, then, for every $\varepsilon > 0$, there is a probability at least ε that at least $3/4 - \varepsilon$ fraction of the equations in (1) are satisfiable. More formally*

$$\Pr_{z \in \{0,1\}^p}\left[\#\{\, i \mid g_i(z) = \ell_i(z)\} \geq \frac{3}{4} - \varepsilon\right] \geq \varepsilon.$$

Lemma 5 *If $r_1, \ldots, r_p$ is chosen uniformly at random from $\{0,1\}^p$, and $|P| > (1/2\delta^2)(\ln 2)(n + c)$, then the probability that there is an assignment that satisfies more than a $1/2 + \delta$ fraction of the equations of (1) is at most 2^{-c}.*

PROOF: Fix an assignment z; then the probability that a fraction at least $1/2 + \delta$ of the r_i agree with $l_i(z)$ is at most $e^{-2\delta^2 p} \leq 2^{-c-n}$. By a union bound, there is at most a probability 2^{-c} that such a z exists. □

Given a system of linear equations over $GF(2)$ with two variables per equation, it is NP-hard to determine the largest number of equations that can be satisfied, but the problem can be approximated to within a $.878$ factor using semidefinite programming [11]. We now prove theorem 3

Proof of Theorem 3: Fix ε and δ small enough so that $.878(3/4 - \varepsilon) > 1/2 + \delta$. Using semidefinite programming [11] we get a polynomial time algorithm that is successful if a fraction $3/4 - \varepsilon$ of the equations is holds, and fails if no more than $0.878(3/4 - \varepsilon)$ of the equations hold. Fixing $\delta = .158$ and $\varepsilon = 10^{-4}$, we obtain the statement of theorem, where $p = 14n$. □

3.2 Correlation Attacks

In this section we discuss how our distinguisher for the case $k = 4$ can be seen as a "correlation attack."

Correlation attacks are a class of attacks that are often attempted in practice against candidate pseudorandom generators,[3] see e.g. the introduction of [16] for an overview.

The basic idea is as follows. Given a candidate generator $G : \{0,1\}^n \to \{0,1\}^m$, where $G(x) = g_1(x), \ldots, g_m(x)$, we first try and find linear relations between input bits and output bits that are satisfied with non-trivial probability. For example, suppose we find coefficients $a_{i,j}$, $b_{i,j}$ and c_j such that each of the equations

$$\begin{array}{l} \sum_{i=1}^n a_{i,1}x_i + \sum_{i=1}^m b_{i,1}g_i(x) = c_1 \pmod 2 \\ \sum_{i=1}^n a_{i,2}x_i + \sum_{i=1}^m b_{i,2}g_i(x) = c_2 \pmod 2 \\ \ldots \\ \sum_{i=1}^n a_{i,t}x_i + \sum_{i=1}^m b_{i,t}g_i(x) = c_t \pmod 2 \end{array} \tag{2}$$

is satisfied with probability bounded away from 1/2.

Now we want to use this system of equations in order to build a distinguisher. The distinguisher is given a sample $\mathbf{z} = (z_1, \ldots, z_m)$ and has to decide whether $\mathbf{z}$ is uniform or is the output of G. The distinguisher substitutes z_i in place of $g_i(x)$ in (2) and then tries to find an $\mathbf{x}$ that maximizes the number of satisfied equations. The hope is that, if $\mathbf{z} = G(\mathbf{x})$, then we will find $\mathbf{x}$ as a solution of the optimization problem.

Unfortunately, maximizing the number of satisfied equations in a linear system over $GF(2)$ is an NP-hard problem, and, in fact, it is NP-hard to achieve an approximation factor better than 1/2 [12]. In practice, one uses belief-propagation algorithms that often work, although the method is typically not amenable to a formal analysis.

In Section 3, we were able to derive a formal analysis of a related method because we ended up with a system of equations having only two variables per equation, a class of instances for which good approximation algorithms are known. Furthermore, we did not try to argue that, when

[3] Pseudorandom generators are called "stream ciphers" in the applied cryptography literature.

the method is applied to the output of the generator, we are likely to recover the seed; instead, we argued that just being able to approximate the largest fraction of satisfiable equations gives a way to distinguish samples of the generators from random strings.

4 $O(n^{k/2})$ upper bound

In this section we state the following theorem which gives an upper bound on the maximal stretch of an ε-bias generator in NC^0_k.

Theorem 6 *There exists a constant c such that for every integer $0 < k$ and any $0 < \varepsilon < 2^{-2k}$, if $G = (g_1, \ldots, g_m)$ is an ε biased pseudo random generator, where each of the g_i's depend on at most k bits, then $m \le c2^k n^{\lceil k/2 \rceil}$.*

The proof uses the following lemma from [22].

Lemma 7 ([22]) *Let $f : \{0,1\}^k \to \{0,1\}$ then for all r*

- *Either f is a polynomial of degree at most r over F_2, or*
- *f is biased towards an affine function of at most $k - r$ variables.*

Proof of Theorem 6: Set $r = \lfloor k/2 \rfloor, s = k - r$ and for $0 \le t \le n$, $B(t) = \sum_{i=0}^{t} \binom{n}{i}$. Note that there exists a constant $\tilde{c}$ such that $B(r) \le B(s) \le \tilde{c} n^{\lceil k/2 \rceil}$, and $B(s - 1) \le \tilde{c} n^{k/2-1}$. By lemma 7 every g_i is either a degree $\le r$ polynomial, or is biased towards an affine function of at most s variables. Let p be the number of degree $\le r$ polynomials among the g_i's, b_s be the number of g_i's biased towards an affine function of exactly s variables (but not towards less than s variables), and $b_{<s}$ be the number of g_i's biased towards an affine function of at most $s-1$ variables. Clearly, $m \le p + b_s + b_{<s}$.

Note that the $B(r)$ monomials of degree $\le r$ on the variables $x_1, \ldots, x_n$ form a basis to the vector space of all degree $\le r$ polynomials in $x_1, \ldots, x_n$. Therefore if $p > B(r)$, there is a linear dependency between the $g_i^{\prime}s$. We therefore conclude that

$$p \le B(r) \le \tilde{c} n^{\lceil k/2 \rceil}. \qquad (3)$$

On the other hand, note that by lemma 1, if g is biased towards an affine function of $d \le s$ variables, then there exist an affine function ℓ of at most d variables such that $\mathbf{Pr}[f = \ell] \ge 1/2 + 2^{d-k}$. Moreover, there are exactly $B(s-1)$ linear functions on at most $s-1$ variables, and $\binom{n}{s}$ linear functions on exactly s variables.

Now lemma 2 implies that there exists a constant c' such that if $b_s \ge c' \binom{n}{s} 2^k$, or $b_{<s} \ge c' B(s-1) 4^k$ then there is a $\oplus$ of two of the g_i's that has an $O(2^{-k})$ bias or $O(2^{-2k})$ bias respectively. It therefore follows that

$$b_s + b_{<s} \le c'(2^k \binom{n}{s} + 4^k B(s-1)) \le \hat{c} 2^k n^{\lceil k/2 \rceil} \qquad (4)$$

where $\hat{c}$ is some constant, and n is large enough.

Combining (4) and (3) we obtain that

$$m \le p + b_s + b_{<s} \le c 2^k n^{\lceil k/2 \rceil},$$

for some constant c as needed. □

5 Constructions for $k = 5$ and $k = 4$

5.1 Preliminaries

We will construct a generator mapping $2n$ bits into cn bits. It is helpful to think of c as a large constant, although the results hold also if c is a function of n.

We will construct two generators: one will be good against linear tests that involve a small number of output bits (we call them *small tests*), and another is good against linear tests that involve a large number of output bits (we call them *large tests*). The final generator will be obtained by computing the two generators on independent seeds, and then XOR-ing their output bit by bit. In this way, we fool every possible test.

The generator that is good against large tests is such that every bit of the output is just the product of two bits of the seed. We argue that the sum (modulo 2) of t output bits of the generator has bias exponentially small in t/c^2, where c, as above, is the stretch of the generator.

Then we describe a generator that completely fools linear tests of size up to about n/c^2, and such that every bit of the output is the sum of three bits of the seed. Combined with the generator for large tests, we get a generator in NC^0_5 such that every linear test has bias $2^{-O(n/c^4)}$.

5.2 The Generator for Large Tests

Let us call the bits of the seed $y_1, \ldots, y_n$.

Let K be an undirected graph formed by $n/(2c+1)$ disjoint cliques each with $2c+1$ vertices. K has n vertices that we identify with the elements of $[n]$. K has and $cn = m$ edges. Fix some ordering of the edges of K, and let (a_j, b_j) be the j-th edge of K. Define the functions $q_1, \ldots, q_m$ as $q_j(y_1, \ldots, y_n) = y_{a_j} y_{b_j}$.

Claim 8 *For every subset $S \subset [m]$, the function $q_S(\mathbf{y}) = \sum_{j \in S} q_j(\mathbf{y})$ is such that*

$$|\mathbf{Pr}_{\mathbf{y}}[q_S(\mathbf{y}) = 0] - \frac{1}{2}| \le \left(\frac{1}{2}\right)^{1+|S|/(2c^2+c)}.$$

The proof relies on the following two standard lemmas. The first one from [7] is a special case of the Schwartz-Zippel lemma [25, 28].

Lemma 9 ([7]) *Let p be a non-constant degree-2 multilinear polynomial over $GF(2)$. Then $1/4 \leq \mathbf{Pr}[p(x) = 0] \leq 3/4$.*

Lemma 10 *Let $X_1, \ldots, X_t$ be independent 0/1 random variables, and suppose that for every i we have $\delta \leq \mathbf{Pr}[X_i = 0] \leq 1 - \delta$. Then*

$$\frac{1}{2} + \frac{1}{2}(1-2\delta)^t \leq \mathbf{Pr}\left[\bigoplus_i X_i = 0\right] \leq \frac{1}{2} + \frac{1}{2}(1-2\delta)^t.$$

We can now prove claim 8.

PROOF OF CLAIM 8. We can see S as a subset of the edges of K. Each connected component of K has $2c^2 + c$ edges, so S contains edges coming from at least $|S|/(2c^2 + c)$ different connected components. Let t be the number of connected components. If we decompose the summation $\sum_{j \in S} q_j(y_1, \ldots, y_n)$ into terms depending on each of the connected components, then each term is a non-trivial degree-2 polynomial, and the t terms are independent random variables when $y_1, \ldots, y_n$ are picked at random. We can then apply lemma 10, where the X_i are the values taken by each of the t terms in the summation, $\delta = 1/4$, and $t \geq |S|/(2c^2 + c)$. □

5.3 The Generator for Small Tests

Let $A \in \{0,1\}^{n \times m}$ be a matrix such that every row is a vector in $\{0,1\}^n$ with exactly three non-zero entries, and let also A be such that every subset of σ rows are linearly independent. Let $A_1, \ldots, A_m$ be the rows of A. We define the linear functions $l_1, \ldots, l_m$ as $l_i(\mathbf{x}) = A_i \cdot \mathbf{x}$. Note that each of these linear functions depends on only three bits of the input.

Claim 11 *For every subset $S \subseteq [m]$, $|S| < \sigma$, the function $l_S(\mathbf{x}) = \sum_{j \in S} l_j(\mathbf{x})$ is balanced.*

PROOF: We have $l_S(\mathbf{x}) = (\sum_{j \in S} A_j) \cdot \mathbf{x}$, and since $\sum_{j \in S} A_j$ is a non-zero element of $\{0,1\}^n$, it follows that $l_S()$ is a non-trivial linear function, and therefore it is balanced. □

Lemma 12 *For every $c = c(n) = o(\sqrt{n}/(\log n)^{3/4})$ and for sufficiently large n there is a 0/1 matrix A with cn rows and n columns such that every row has exactly three non-zero entries and such that every subset of $\sigma = n/(4e^2c^2(n))$ rows are linearly independent.*

This is a standard probabilistic construction similar to [3, 5, 4]. The proof is omitted.

5.4 Putting Everything Together

In order to obtain the generator, we take $G_1 : \{0,1\}^n \to \{0,1\}^m$ to be a generator satisfying claim 8, and $G_2 : \{0,1\}^n \to \{0,1\}^m$ to satisfy lemma 12. Then we take $G : \{0,1\}^{2n} \to \{0,1\}^m$ defined by $G(x,y) = G_1(x) \oplus G_2(y)$ to fool both small tests and large tests. We thus obtain

Theorem 13 *For every c and sufficiently large n, there is a generator in NC^0_5 mapping n bits into cn bits and sampling an ε-biased distribution, where $\varepsilon = 2^{-n/O(c^4)}$.*

5.5 Generator for $k = 4$

When $k = 4$ we want to replace the generator for small sets by a generator which depends only on two bits. The construction is essentially the one in [7].

The generator is obtained by taking a graph H on cn edges, with girth $\Omega(\log n / \log c)$ and letting $x_i \oplus x_j$ be an output bit, if (i,j) is an edge of the graph.

Let H be an undirected graph with n vertices, that we identify with $[n]$, having cn edges and girth γ. Fix some ordering of the edges of H, and let (a_j, b_j) be the j-th edge of H. We define the linear functions $l_1, \ldots, l_m$ as $l_i(x_1, \ldots, x_n) = x_{a_j} + x_{b_j}$.

Claim 14 *For every subset $S \subseteq [m]$, $|S| < \gamma$, the function $l_S(\mathbf{x}) = \sum_{j \in S} l_j(\mathbf{x})$ is balanced.*

PROOF: Since $|S| < g$, the subgraph of H induced by the edges of S is a forest. Therefore $l_S(\mathbf{x})$ is non-zero linear function. □

Lemma 15 ([18]) *For every c and for sufficiently large n there are explicitly constructible graphs H with n vertices, cn edges, and girth $\Omega((\log n)/(\log c))$.*

We thus obtain.

Theorem 16 *For every c and sufficiently large n, there is a generator in uniform NC^0_4 mapping n bits into cn bits and sampling an ε-biased distribution, where $\varepsilon = n^{-1/O(c^2 \log c)}$.*

6 ε-biased generator for large k

In this section we construct an ε-biased generator in NC^0_k, for large k, which outputs $n^{\Omega(\sqrt{k})}$ bits. More precisely,

Theorem 17 *Let k be a positive integer. There exist an ε-bias generator in NC^0_k from n bits to $n^{\lfloor\sqrt{k}\rfloor \cdot (\frac{1}{2} - o(1))}$ bits whose bias ε is at most*

$$\varepsilon = \exp\left(-|\mathrm{n}|^{\frac{1-\mathrm{o}(1)}{2\lfloor\sqrt{\mathrm{k}}\rfloor}}\right)$$

.

6.1 The Generator for Large Tests

We will assume through this sub-section that $n = p^2$.

Consider the following bi-partite graph $G = (L, R, E)$ where $|L| = p$, $|R| = \binom{p}{d}$. Identify the vertices of L with the numbers $1, ..., p$ and the vertices of R with $\binom{[p]}{d}$, the set of all subsets of $[p]$ of size d. The edges of G are all pairs (i, S) such that $i \in [p]$, $S \in \binom{[p]}{d}$ and $i \in S$.

For a set of vertices, V, we denote with $N(V)$ the set of neighbors of V. For a vertex i let $\mathrm{deg(i)} = \mathrm{|N(\{i\})|}$.

Claim 18 *For any set of right vertices $V \subset R$ we have that $|N(V)| \geq \frac{d|V|^{\frac{1}{d}}}{e}$.*

PROOF: Any set of t left vertices has $\binom{t}{d}$ right neighbors. The result follows from the inequality

$$|V| \leq \binom{|N(V)|}{d} \leq \left(\frac{e|N(V)|}{d}\right)^d$$

□

Our construction will assign a monomial of degree d, in the input variables, to each edge. We think about the vertices of L as representing disjoint subsets of the input variables and each edge leaving such input set corresponds to a monomial in its variables. The right vertices, R, correspond to the output bits. Each output is the sum of monomials that label the edges that fan into it. We now give the formal construction.

Let $X = \bigsqcup_{i=1}^{p} X_i$ be a partition of $X = \{x_1, ..., x_n\}$ to p disjoint sets each of size p.

We assign the set X_i to the i'th vertex of L. Let M_i be the set of all multilinear monomials of degree d in the variables of X_i. We have that

$$|M_i| = \binom{p}{d} > \binom{p-1}{d-1} = \mathrm{deg(i)}$$

Therefor we can assign to each edge leaving i a different monomial from M_i.

Each right vertex corresponds to an output bit. For a right vertex j the j'th output is the sum of all monomials that were assigned to the edges adjacent to j. Thus each output is the sum of d monomials each of degree d. Hence each output depends on d^2 input variables. Denote with f_j the j'th output. We now show that any large linear combination has a small bias.

Lemma 19 *In the notations above any linear combination (over $GF(2)$) $f = \sum_{j \in J} f_j$ has bias at most*

$$\exp\left(\frac{-|\mathrm{J}|^{\frac{1}{\mathrm{d}}}}{2^{\mathrm{d}}}\right)$$

PROOF: The proof is essentially the same as the proof of claim 8 and follows from the following easy claims.

Claim 20 *f can be written as the sum of at least $N(J)$ polynomials of degree d, each in a different set of variables.*

PROOF: The set of outputs J, has $N(J)$ left neighbors. The edges connecting the set J to a neighbor $i \in N(J)$ are labeled with polynomials of degree d in X_i. □

From the Schwartz-Zippel lemma [25, 28] we get

Claim 21 *The bias of any polynomial of degree d is bounded above by $\frac{1}{2^d}$.*

Thus according to lemma 10 we get that the bias of f is at most

$$\frac{1}{2}\left(1 - \frac{2}{2^d}\right)^{N(J)} \leq \frac{1}{2} \cdot \exp\left(\frac{-2\mathrm{N(J)}}{2^{\mathrm{d}}}\right) \leq \exp\left(\frac{-|\mathrm{J}|^{\frac{1}{\mathrm{d}}}}{2^{\mathrm{d}}}\right)$$

This finishes the proof of lemma 19 □

This finishes the construction of the generator for large tests. We now describe the generator for small tests.

6.2 The Generator for Small Tests

Similar to the $k = 4, 5$ cases this generator will output only linear functions. We will have the property that any small set of these linear functions is linearly independent. This is now a standard construction that follows from unique neighbor property of expanding graphs. We omit the proof of the following lemma.

Lemma 22 *Let t be positive integer t and $\Delta = 10t$. There exist a mapping from n bits to n^t bits such that every output depends on Δ input variables, and such that any linear combination of at most $\sqrt{n}$ outputs is linearly independent.*

6.3 Putting things together

We now prove theorem 17.

PROOF: Let $k' = (\lfloor\sqrt{k}\rfloor - 5)^2$, $n' = \lfloor\sqrt{\frac{n}{2}}\rfloor^2$. We have that

$$k > k' + 10\sqrt{k'}, \quad k' > k - 12\sqrt{k}, \quad \tfrac{n}{2} \geq n' > \tfrac{n}{2} - \sqrt{2n}.$$

Let $X = \{x_1, ..., x_{n'}\}$, $Y = \{y_1, ..., y_n'\}$. Let $f_1(X), \ldots, f_{\binom{p}{d}}(X)$ be the outputs of the generator against long tests with the parameters $p = \sqrt{n'}$, $d = \sqrt{k'}$. Let $h_1(Y), \ldots, h_{n'^{k'}}(Y)$ be the outputs of the generator for small tests on Y, given the parameter $t = \sqrt{k'}$. Note that

$$n'^{k'} > \binom{\sqrt{n'}}{\sqrt{k'}} = \binom{p}{d}.$$

Our generator G will output the functions

$$\forall 1 \leq i \leq \binom{p}{d} \quad g_i(X,Y) = f_i(X) + h_i(Y).$$

Notice that as we have more h_i's than f_i's we don't use most of the h_i's. Clearly, each output of the generator depends on $k' + 10\sqrt{k'} < k$ input variables.

From lemma 19,22 we get that the bias of any non trivial linear combination of the outputs is at most

$$\exp\left(\frac{-|\mathrm{n}'|^{\frac{1}{2\mathrm{d}}}}{2^{\mathrm{d}}}\right)$$

Thus our generator takes $2n' \leq n$ inputs and outputs

$$\binom{p}{d} \geq \left(\frac{e^2 n'}{k'}\right)^{\frac{\sqrt{k'}}{2}} = n^{\lfloor\sqrt{k}\rfloor \cdot (\frac{1}{2} - o(1))}$$

and has an exponentially small bias. □

7 A degree 2 generator

In this section we consider a variant of the problem presented in the paper. Suppose that we require that every output bit is a degree k polynomial in the input bits. It is clear that if we want the output to be ε-biased, then the number of output bits m is at most the dimension of degree k polynomials in n variables $\sum_{i=k}^{s} \binom{n}{i} = O(n^k)$.

Clearly this is a relaxation of the problem described above. In particular any upper bound here will imply an upper bound for NC^0_k. The problem is also of independent interest, as low degree generators are "simple" in an intuitive sense.

In this section we construct a generator of ε-biased set such that every output is a polynomial of degree 2 in the input variables. We show that unlike the $k = 2$ case we can output $\Omega(n^2)$ bits. In particular we prove

Theorem 23 *$\forall 1 \leq m \leq n$ there exists an ε-bias generator $G = (g_1, ..., g_t) : \{0,1\}^n \mapsto \{0,1\}^t$, $t = \lfloor \frac{n}{2} \rfloor \cdot m$, such that g_i is a degree 2 polynomial, and the bias of any non trivial linear combination of the g_i's is at most $2^{\frac{n-2m}{4}}$.*

We begin by studying the bias of a degree 2 polynomial, over $GF(2)$.

7.1 The Bias of Degree 2 polynomials

Let $P(x_1, ..., x_n)$ be a degree 2 polynomial. P is also called a quadratic form over $GF(2)$. We say that a matrix A represents P with respect to a basis of $GF(2)^n$, $\{v_i\}_{i=1}^n$, if for every vector $v = \sum_{i=1}^n x_i \cdot v_i$ we have that $P(v) = x^t A x$ ($x = (x_1, ..., x_n)$). Notice that we can always find an upper triangular matrix that represents P; let

$$P(a_1, ..., a_n) = \sum_{1 \leq i \leq j \leq n} \alpha_{i,j} a_i a_j$$

Define

$$A(P)_{i,j} = \begin{cases} \alpha_{i,j} & i \leq j \\ 0 & i > j \end{cases}$$

Clearly $P(\sum_{i=1}^n e_i \cdot x_i) = x^t A(P) x$ and $A(P)$ represents P with respect to the standard basis.

The bias of a quadratic form is bounded by the rank of the matrix representing it as follows.

Theorem 24 *The bias of a degree 2 polynomial P is at most*

$$2^{-\left(1 + \frac{\mathrm{rank}(\mathrm{A}+\mathrm{A}^{\mathrm{t}})}{4}\right)}$$

for any matrix A that represents P.

Theorem 24 shows that in order to output m polynomials of degree 2, such that any non trivial linear combination of them is almost unbiased it suffices to find matrices $A_1, ..., A_m$ such that for any non trivial combination of them, $B = \sum_{i=1}^m \alpha_i A_i$ ($\alpha_i \in GF(2)$), we have that $\mathrm{rank}(\mathrm{B} + \mathrm{B}^{\mathrm{t}})$ is high.

7.2 Proof of theorem 24

The following claim is trivial.

Claim 25 $P \equiv 0$ iff *there exist a symmetric matrix that represents P w.r.t. some basis* iff *any matrix that represents P is symmetric.*

The proof of theorem 24 will follow from the following claims.

Claim 26 *For any quadratic form P on n variables, there exist a basis of $GF(2)^n$ e_i, f_i $i = 1, ..., r$ and g_j $j = 1, ..., s$ such that $2r + s = n$ and n elements in $GF(2)$, a_i, b_i $i = 1, ..., r$, c_j $j = 1, ..., s$, such that for*

$$v = \sum_{i=1}^{r} x_i e_i + \sum_{i=1}^{r} x_{r+i} f_i + \sum_{j=1}^{s} x_{2r+j} g_j$$

we have

$$P(v) = \sum_{i=1}^{r} (a_i {x_i}^2 + x_i x_{r+i} + b_i {x_{r+i}}^2) + \sum_{j=1}^{s} c_j {x_{2r+j}}^2$$

Such a basis is called "a canonical basis for P".

PROOF: See the proof of theorem 5.1.7 in [14]. □

Claim 27 *Let P be a quadratic form on n variables. Let A represent P with respect to the standard basis and D represent P with respect to the canonical basis. Then*

$$\mathrm{rank(D)} \geq \frac{\mathrm{rank(A + A^t)}}{2}$$

PROOF: Let B be the matrix whose columns are $e_1, ..., e_r, f_1, ..., f_r, g_1, ..., g_s$ written w.r.t. the standard basis. We have that

$$\forall x \in GF(2)^n \;\; x^t D x = x^t B^t A B x.$$

In other words

$$\forall x \in GF(2)^n \;\; x^t (D - B^t A B) x = 0.$$

Therefor there exist a symmetric matrix S such that

$$D - B^t A B = S,$$

or

$$D = B^t (A + (B^{-1})^t S (B^{-1})) B.$$

As $(B^{-1})^t S (B^{-1})$ is a symmetric matrix we get by the next claim (claim 28) that

$$\mathrm{rank(D) = rank(A + (B^{-1})^t S (B^{-1}))} \geq \frac{\mathrm{rank(A + A^t)}}{2}.$$

□

Claim 28 *For upper diagonal matrix A with zeros on the diagonal, and any symmetric matrix S we have that*

$$\mathrm{rank(A + S)} \geq \frac{\mathrm{rank(A + A^t)}}{2}$$

where A^t is the transpose of A.

PROOF: Let $r = \mathrm{rank(A + S) = rank(A^t + S)}$. Then

$$\mathrm{rank(A + A^t) \leq rank(A + S) + rank(A^t + S) = 2r}$$

□

PROOF OF THEOREM 24. Clearly the bias of P does not change if we calculate it w.r.t. to a canonical basis, $\{v_i\}_{i=1}^n$, for P. In such a basis, for $v = \sum_{i=1}^n x_i \cdot v_i$, we have that

$$P(v) = \sum_{i=1}^r (a_i {x_i}^2 + x_i x_{r+i} + b_i {x_{r+i}}^2) + \sum_{j=1}^s c_j {x_{2r+j}}^2$$

First notice that if for some $1 \leq j \leq s$ $c_j \neq 0$ then P is unbiased. Otherwise, we note that for every i the bias of $(a_i {x_i}^2 + x_i x_{r+i} + b_i {x_{r+i}}^2)$ is at most $\frac{1}{4}$. Therefore according to lemma 10 we get the bias of P is at most $\left(\frac{1}{2}\right)^{r+1}$. As we assume that $\forall j \;\; c_j = 0$ we see that

$$r \geq \frac{\mathrm{rank(D)}}{2}$$

The theorem now follows from claim 27. □

7.3 The generator

In this subsection we give a construction of a linear space of matrices with the property that for every non zero matrix in the space, A, we have that $\mathrm{rank(A + A^t)}$ is high.

Such a construction was first given by Roth [24], and later simplified by Meshulam [21] (see also [26]).

Theorem 29 *For any positive natural numbers $n \geq m$ there exist $t = \lfloor \frac{n}{2} \rfloor \cdot m$ matrices $A_1, ..., A_t \in M_n(GF(2)$ such that for every non trivial combination of them $B = \sum_{i=1}^t \alpha_i A_i$ we have that*

$$\mathrm{rank(B + B^t) \geq n - 2m}$$

We now prove theorem 23.

PROOF: Let $A_1, ..., A_t$ be the matrices guaranteed by theorem 29. Define $g_i(x) = x^t A_i x$. Consider any non trivial linear combination

$$g(x) = \sum_{i=1}^t \alpha_i g_i(x) = x^t \left(\sum_{i=1}^n \alpha_i A_i \right) x$$

According to theorem 29, we have that $\mathrm{rank}(g) \geq n - 2m$. Theorem 24 shows that the bias of g is at most $2^{\frac{n-2m}{4}}$. □

8 Conclusions

Several questions remain open.

Even for the case $k = 3$, we only know how to break the generator assuming that the output length is a sufficiently large constant multiple than the seed length. It is not clear whether there is a linear test, or even a polynomial time algorithm, that breaks the case $k = 3$ when, say, $m = n + 1$.

It is still open whether there can be an ε-biased generator with negligible ε in the case $k = 4$. We conjecture that this is not the case for sufficiently large linear stretch, but we do not have a strong feeling about what happens for very small stretch.

The main open question is whether our generator for the case $k = 5$ can be broken by a polynomial time algorithm and, in general, whether polynomial time algorithms can break all NC^0 generators.

Another important open problem which may be more accesible it to understand the right asymptotics for ε-biased generators for large k. It is tempting to conjecture that either the upper bound $n^{O(k)}$ or the lower bound $n^{\Omega(\sqrt{k})}$ are actually tight.

Acknowledgements

We wish to thank David Wagner for suggesting the relevance of correlation attacks. A.S. would also like to thank Avi Wigderson for helpful discussions.

References

[1] N. Alon, M. Capalbo. Explicit Unique-Neighbor Expanders. Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science, pages 73-79, 2000.

[2] N. Alon, O. Goldreich, J. Håstad, and R. Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures and Algorithms*, 3(3):289–304, 1992.

[3] P. Beame, R. Karp, T. Pitassi, and M. Saks. On the complexity of unsatisfiability proofs for random k-cnf formulas. In *Proceedings of the 30th ACM Symposium on Theory of Computing*, 1998.

[4] A. Bogdanov, K. Obata, and L. Trevisan. A lower bound for testing 3-colorability in bounded degree graphs. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pages 93–102, 2002.

[5] E. Ben-Sasson and A. Wigderson. Short proofs are narrow: Resolution made simple. *Journal of the ACM*, 48(2), 2001.

[6] M. Capalbo. Explicit Constant-Degree Unique-Neighbor Expanders, 2001.

[7] M. Cryan and P. B. Miltersen. On pseudorandom generators in NC0. In *Proceedings of MFCS'01*, 2001.

[8] M. Capalbo, O. Reingold, S. Vadhan and A. Wigderson. Randomness Conductors and Constant-Degree Expansion Beyond the Degree/2 Barrier. Proceedings of the 34th Symposium on the Theory of Computing, 659-668, 2000.

[9] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.

[10] O. Goldreich. Candidate one-way functions based on expander graphs. Technical Report TR00-090, ECCC, 2000.

[11] M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[12] J. Håstad. Some optimal inapproximability results. In *Proceedings of the 29th ACM Symposium on Theory of Computing*, pages 1–10, 1997.

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

[14] J. W. P. Hirschfeld, Projective Geometries over Finite Fields, Oxford University Press, 1979.

[15] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *Journal of Cryptology*, 9(4):199–216, 1996.

[16] T. Johansson and F. Jonsson. Improved fast correlation attacks on stream ciphers via convolutional codes. In *Proceedings of EUROCRYPT'99*, 1999.

[17] N. Linial, Y. Mansour, and N. Nisan. Constant depth circuits, fourier transform and learnability. *Journal of the ACM*, 40(3):607–620, 1993.

[18] A. Lubotzky, R. Phillips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8:261–277, 1988.

[19] C. J. Lu and O. Reingold and S. Vadhan and A. Wigderson Extractors: Optimal Up to Constant Factors. To appear in proceedings of the 35th Annual symposium on the theory of computing (STOC).

[20] M. Luby and C. Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 2(17):373–386, 1988.

[21] . R. Meshulam. Spaces of Hankel matrices over finite fields, *Linear Algebra Appl.* **218**, 73–76, 1995.

[22] E. Mossel, R. O'Donnell and R. Servedio (2003) Learning Juntas. To appear in proceedings of the 35th Annual symposium on the theory of computing (STOC).

[23] J. Naor and M. Naor. Small-bias probability spaces: efficient constructions and applications, *SIAM Journal on Computing*, 22(4):838–856, 1993.

[24] R. Roth. Maximum rank array codes and their application to crisscross error correction, *IEEE Trans. on Info. Th.* **37,** 328–336, 1991.

[25] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[26] A. Shpilka. On the rigidity of matrices. Manuscript, 2002.

[27] U. Vazirani. *Randomness, Adversaries and Computation*. PhD thesis, University of California, Berkeley, 1986.

[28] R. Zippel. Probabilistic algorithms for sparse polynomials. In *Symbolic and algebraic computation (EUROSAM '79, Internat. Sympos., Marseille, 1979)*, pages 216–226. Springer, Berlin, 1979.