2012 IEEE 53rd Annual Symposium on Foundations of Computer Science

# Better Pseudorandom Generators from Milder Pseudorandom Restrictions.

Parikshit Gopalan
*MSR-SVC*

Raghu Meka*
*IAS Princeton*

Omer Reingold
*MSR-SVC*

Luca Trevisan
*Stanford University*

Salil Vadhan†
*Harvard University*

***Abstract*—We present an iterative approach to constructing pseudorandom generators, based on the repeated application of mild pseudorandom restrictions. We use this template to construct pseudorandom generators for combinatorial rectangles and read-once CNFs and a hitting set generator for width-3 branching programs, all of which achieve near-optimal seed-length even in the low-error regime: We get seed-length $\tilde{O}(\log(n/\varepsilon))$ for error $\varepsilon$. Previously, only constructions with seed-length $O(\log^{3/2} n)$ or $O(\log^2 n)$ were known for these classes with error $\varepsilon = 1/\mathrm{poly}(n)$.**

**The (pseudo)random restrictions we use are milder than those typically used for proving circuit lower bounds in that we only set a constant fraction of the bits at a time. While such restrictions do not simplify the functions drastically, we show that they can be derandomized using small-bias spaces.**

***Keywords*-Pseudorandom generators, random restrictions, DNF formulas, combinatorial rectangles, branching programs.**

## I. Introduction

### A. Pseudorandom Generators

The theory of pseudorandomness has given compelling evidence that very strong pseudorandom generators exist. For example, assuming that there are computational problems solvable in exponential time that require exponential-sized circuits, Impagliazzo and Wigderson [1] have shown that for every $n$, $c$ and $\varepsilon > 0$, there exist efficient pseudorandom generators (PRGs) mapping a random seed of length $O(\log(n^c/\varepsilon))$ to $n$ pseudorandom bits that cannot be distinguished from $n$ uniformly random bits with probability more than $\varepsilon$, by any Boolean circuit of size $n^c$. These PRGs, which fool arbitrary efficient computations (represented by polynomial-sized Boolean circuits), have remarkable consequences for derandomization: every randomized algorithm can be made deterministic with only a polynomial slowdown, and thus $\mathrm{P} = \mathrm{BPP}$.

These results, however, remain conditional on a circuit complexity assumption whose proof seems far off at present. Since PRGs that fool a class of Boolean circuits also imply lower bounds for that class, we cannot hope to remove the assumption. Thus unconditional generators are only possible for restricted models of computation for which we have lower bounds.

*Bounded-depth circuits* and *bounded-space algorithms* are two models of computations for which we know how to construct PRGs with $O(\log^{O(1)}(n/\varepsilon))$ seed length [2], [3]. Known PRG constructions for these classes have found several striking applications including the design of streaming algorithms [4], algorithmic derandomization [5], randomness extractors [6], hashing [7], hardness amplification [8], almost $k$-wise independent permutations [9], and cryptographic PRGs [10]. Arguably, constructing PRGs with the optimal $O(\log(n/\varepsilon))$ seed length for these classes are two of the outstanding open problems in derandomization.

Nisan [3] devised a PRG of seed length $O(\log^2 n)$ that fools polynomial-width branching programs, the non-uniform model of computation that captures logspace randomized algorithms: a space-$s$ algorithm is modeled by a branching program[1] of width $2^s$. Nisan's generator has been used by Saks and Zhou [11] to prove that every randomized logspace algorithms can be simulated in space $O(\log^{3/2} n)$, Nisan's generator remains the best known generator for polynomial-width branching programs (and logspace randomized algorithms) and, despite much progress in this area [12], [13], [14], [15], [16], [17], [18], [19], [20], there are very few cases where we can improve on Nisan's twenty year old bound of $O(\log^2 n)$ [3]. For constant-width *regular* branching programs, Braverman et al. [17] have given a pseudorandom generator with seed length $\tilde{O}((\log n)\cdot(\log(1/\varepsilon)))$, which is $\tilde{O}(\log n)$ for $\varepsilon = 1/\mathrm{polylog}(n)$, but is no better than Nisan's generator when $\varepsilon = 1/\mathrm{poly}(n)$. Only for constant-width *permutation* branching programs and for width-2 branching programs has seed length $O(\log(n/\varepsilon))$ been achieved, by Koucký, Nimbhorkar, Pudlák [19] and Saks and Zuckerman [21], respectively. Remarkably, even for width-3 branching programs we do not know of any efficiently computable PRG with seed length $o(\log^2 n)$. Recently, Sima and Zak [22] have constructed *hitting set generators* (HSGs, which are a weaker form of pseudorandom generators) for width-3 branching programs with optimal seed length $O(\log n)$, for a large error parameter $\varepsilon > 5/6$.

*Raghu Meka is supported in part by NSF grant DMS-0835373. Work done in part while the author was an intern at Microsoft Research Silicon Valley.

†Salil Vadhan was supported in part by NSF grant CCF-1116616. Work done in part while on leave as a Visiting Researcher at Microsoft Research Silicon Valley and a Visiting Scholar at Stanford University.

[1]Space-bounded randomized algorithms are modeled by *oblivious, read-once* branching programs, which read the input bits in a specified order and read each input bit only once. In this paper, all the references to "branching programs" refer to "oblivious read-once branching programs."

0272-5428/12 $26.00 © 2012 IEEE
DOI 10.1109/FOCS.2012.77

In a different work, Nisan [2] also gave a gives a PRG that $\varepsilon$-fools $AC_0$ circuits of depth $d$ and size $s$ using seed length $O(\log^{2d+6}(s/\varepsilon))$. For the special case of depth-2 circuits, that is, CNFs and DNFs, the work of Bazzi [23], simplified by Razborov [24], provides a PRG of seed length $O(\log n \cdot \log^2(s/\varepsilon))$, which has been improved to $\tilde{O}(\log^2(s/\varepsilon))$ by De et al. [25]. For the restricted case of *read-$k$* DNFs and CNFs, De et al. (for $k$ =1), and Klivans et al. [26] (for $k$ constant) improve the seed length to $O(\log \varepsilon^{-1} \cdot \log s)$, which is optimal for constant $\varepsilon$, but it is essentially no better than the bound for general CNFs and DNFs when $\varepsilon$ is polynomial in $1/n$.

The model of *combinatorial rectangles* is closely related to both bounded-width branching programs and read-once CNFs and are interesting combinatorial objects with a variety of applications of their own [27]. The problem of constructing PRGs for combinatorial rectangles is closely related to the construction of small sample spaces that approximate the uniform distribution on many multivalued random variables [28]: they can be seen as an alternate generalization of the versatile notion of almost $k$-wise independent distributions on $\{0,1\}^n$ to larger domains $[m]^n$. Versions of this problem where each coordinate is a real interval were first studied in number theory and analysis [27]. Subsequently there has been much work on this problem [28], [29], [27], [30], [31]. A PRG with seed length $O(\log n + \log^{3/2}(1/\varepsilon))$ [30] is known for combinatorial rectangles; such a generator achieves the optimal seed length $O(\log n)$ when $\varepsilon \geq 2^{-O(\log^{2/3} n)}$, but not for $\varepsilon = 1/\mathrm{poly}(n)$. It is known how to construct HSGs (which are a weakening of PRGs) with seed length $O(\log(n/\varepsilon))$ [29].

Indeed, there are few models of computations for which we know how to construct PRGs with the optimal seed length $O(\log(n/\varepsilon))$ or even $\log^{1+o(1)}(n/\varepsilon)$. The most prominent examples are bounded-degree polynomials over finite fields [32], [33], [34], [35], [31], with parities (which are fooled by small-bias distributions [32]) as a special case, and models that can be reduced to these cases, such as width-2 branching programs [21], [36].

In summary, there are several interesting models of computation for which a *polylogarithmic* dependence on $n$ and $1/\varepsilon$ is known, and the dependence on one parameter is logarithmic on its own (e.g. seed length $O(\log n \log(1/\varepsilon))$), but a logarithmic bound in both parameters together has been elusive. Finally, we remark that not having a logarithmic dependence on the error $\varepsilon$ is often a symptom of a more fundamental bottleneck. For instance, HSGs with constant error for width 4 branching programs imply HSGs with polynomially small error for width 3 branching programs, so achieving the latter is a natural first step towards the former. A polynomial-time computable PRG for CNFs with seed length $O(\log n/\varepsilon)$ would imply the existence of a problem in exponential time that requires depth-3 circuits of size $2^{\Omega(n)}$ and that cannot be solved by general circuits of size $O(n)$ and depth $O(\log n)$, which is a long-standing open problem in circuit complexity [37].

### B. Our Results

In this paper, we construct the first generators with seed length $\tilde{O}(\log(n/\varepsilon))$ (where $\tilde{O}(\ )$ hides polylogarithmic factors in its argument) for several well-studied classes of functions mentioned above.

- PRGs for combinatorial rectangles. Previously, it was known how to construct HSGs with seed length $O(\log(n/\varepsilon))$ [29], but the best seed length for PRGs was $O(\log n + \log^{3/2}(1/\varepsilon))$ [30].
- PRGs for read-once CNF and DNF formulas. Previously, De, Etesami, Trevisan, and Tulsiani [25] and Klivans, Lee and Wan [26] had constructed PRGs with seed length $O(\log n \cdot \log(1/\varepsilon))$.
- HSGs for width 3 branching programs. Previously, Sima and Zak [22] had constructed hitting set generators for width 3 branching programs with seed length $O(\log n)$ in case the error parameter $\varepsilon$ is very large (greater than 5/6).

As a corollary of our PRG for combinatorial rectangles we get improved hardness amplification in NP by combining our results with those of Lu, Tsai and Wu [38]. We defer the details to the full version.

### C. Techniques

Our generators are all based on a general new technique — the iterative application of "mild" (pseudo)random restrictions.

To motivate our technique, we first recall Håstad's switching lemma [39], [40], [41]: if we randomly assign a $1 - 1/O(k)$ fraction of the variables of a $k$-CNF, then the residual formula on the $n/O(k)$ unassigned variables is likely to become a constant. Ajtai and Wigderson [42] proposed the following natural approach to constructing PRGs for CNFs: construct a small pseudorandom family of restrictions that: 1) makes any given CNF collapse to a constant function with high probability; and 2) ensures that the CNF collapses to each constant function with the *right* probability as determined by the bias of the formula. Known derandomizations of the switching lemma are far from optimal in terms of the number of random bits needed [42], [43], [44]. We will show that, for read-once CNFs, such a pseudorandom restriction can be generated using $\tilde{O}(\log(m/\varepsilon))$ random bits.

We apply restrictions that only set a constant fraction of the variables at a time. The novel insight in our construction is that although we cannot set all the bits at one go from a small-bias distribution, we can set a constant fraction of bits from such a distribution and prove that the bias of the formula is preserved (on average). Hence we use only $\tilde{O}(\log(m/\varepsilon))$ truly random bits per phase. While such mild random restrictions do not drastically simplify the formulas,

we show that in each phase a suitable measure of progress improves (e.g. most clauses will either be satisfied or will have reduced width), implying that the formula collapses to a constant after $O(\log\log(m/\varepsilon))$ steps; and so the total randomness will be $\tilde{O}(\log(m/\varepsilon))$. The idea of setting a few variables at a time is inspired by a recent PRG for hashing balls into bins due to Celis, Reingold, Segev, and Wieder [7].

We illustrate our technique below with a toy example.

*A Toy Example:* Consider a read-once CNF formula $f$ of width $w$ with $m = 2^{w+1}$ clauses in which the variables appear in order (aka the Tribes function of [45]). That is,

$$f(x) = f_1(x_1, \ldots, x_w) \wedge \cdots \wedge f_m(x_{(m-1)w+1}, \ldots, x_{mw})$$

where each $f_i$ is the OR function. $f$ has constant bias and can be computed both by a combinatorial rectangle and a width-3 branching program. De et al. showed that fooling this function with error $\varepsilon$ using small-bias spaces requires seed-length $\Omega(w\log(1/\varepsilon)/\log\log(1/\varepsilon))$.

Assume we partition the input bits into two parts: $x$ which contains the first $w/2$ variables of each clause and $y$ which contains the rest. Let $x \circ y$ denote the concatenation of the two strings. We would like to show that for $\mathcal{D}$ a small-bias distribution and $\mathcal{U}$ the uniform distribution,

$$\left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}, y\sim\mathcal{U}} [f(x\circ y)] - \mathop{\mathbb{E}}_{x\sim\mathcal{U}, y\sim\mathcal{U}} [f(x\circ y)] \right| \leq \varepsilon \qquad \text{(I.1)}$$

A naive approach might be to view setting $y \sim \mathcal{U}$ as applying a random restriction with probability $1/2$. If this simplified the function $f$ to the extent that it can be fooled by small-bias spaces, we would be done. Unfortunately, this is too much to hope for; it is not hard to see that such a random restriction is very likely to give another Tribes-like function with width $w/2$, which is not much easier to fool using small bias than $f$ itself.

Rather, we need to shift our attention to the *bias function* of $f$. For each partial assignment $x$, we define the bias function $F(x)$ as

$$F(x) = \mathop{\mathbb{E}}_{y\sim\mathcal{U}} [f(x\circ y)]. \qquad \text{(I.2)}$$

We can now rewrite Equation (I.1) as

$$\left| \mathop{\mathbb{E}}_{x\sim\mathcal{D}} [F(x)] - \mathop{\mathbb{E}}_{x\sim\mathcal{U}} [F(x)] \right| \leq \varepsilon \qquad \text{(I.3)}$$

Our key insight is that for restrictions as above, the function $F$ is in fact easy to fool using a small-biased space. This is despite the fact that $F(x)$ is an average of functions $f(x\circ y)$ (by Equation (I.2)), most of which are Tribes-like and hence are not easy to fool.

Let us give some intuition for why this happens. Since $f(x\circ y) = \prod_{i=1}^m f_i(x\circ y)$,

$$F(x) = \mathop{\mathbb{E}}_{y\sim\mathcal{U}} [f(x\circ y)] = \prod_{i=1}^m \mathop{\mathbb{E}}_{y\sim\mathcal{U}} [f_i(x\circ y)] = \prod_{i=1}^m F_i(x),$$

where $F_i(x)$ is the *bias function* of the $i^{th}$ clause. But note that over a random choice of $y$, $f_i(x)$ is set to 1 with probability $1-2^{-w/2}$ and is a clause of width $w/2$ otherwise. Hence

$$F_i(x) = \mathop{\mathbb{E}}_{y\sim\mathcal{U}} [f_i(x\circ y)] = 1 - \frac{1}{2^{w/2}} + \frac{\bigvee_{j=1}^{w/2} x_{w(i-1)+j}}{2^{w/2}}.$$

As a consequence, over a random choice of $x$, we now have

$$F_i(x) = \begin{cases} 1 & \text{w.p. } 1-2^{-w/2} \\ 1-2^{-w/2} & \text{w.p. } 2^{-w/2} \end{cases}$$

Thus each $F_i(x)$ is a random variable with $\mathbb{E}_x[F_i(x)] = 1-2^{-w}$ and $\mathsf{Var}_x[F_i(x)] \approx 2^{-3w/2}$. In contrast, when we assign all the variables in the clauses at once, each $f_i(x)$ behaves like a Bernoulli random variable with bias $1-2^{-w}$. While it also has $\mathbb{E}_x[f_i(x)] = 1-2^{-w}$, the variance is much larger: $\mathsf{Var}_x[f_i(x)] \approx 2^{-w}$. The qualitative difference between $2^{-3w/2}$ and $2^{-w}$ is that in the former case, the sum of the variances over all $2^{w+1}$ clauses is small ($2^{-w/2}$), but in the latter it is more than 1. We leverage the small total variance to show that small-bias fools $F$, even though it does not fool $f$ itself. Indeed, setting any constant fraction $\alpha < 1$ of variables in each clause would work.

We now sketch our proof that small-bias spaces fool $F$. Let $g_i(x) = F_i(x) - (1-2^{-w})$ be $F_i$ shifted to have mean 0, so that $\mathbb{E}_x[g_i(x)^2] = \mathsf{Var}[F_i(x)]$. We can write

$$F(x) = \prod_{i=1}^m \left(1 - 2^{-w} + g_i(x)\right) \qquad \text{(I.4)}$$

$$= \sum_{k=1}^m c_k S_k(g_1(x), \ldots, g_m(x)) \qquad \text{(I.5)}$$

where $S_k$ denotes the $k^{th}$ elementary symmetric polynomial and $c_k \in [0,1]$.[2]

Under the uniform distribution, one can show that

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}} [\,|S_k(g_1(x), \ldots, g_m(x))|\,] \leq \left( \sum_{i=1}^m \mathop{\mathbb{E}}_{x\sim\mathcal{U}} \left[g_i(x)^2\right] \right)^{k/2} \leq 2^{-wk/4}.$$

Thus for $k \geq O((\log n)/w)$, we expect each term in the summation in Equation (I.4) to be $1/\mathrm{poly}(n)$. So we can truncate at $d = O((\log n)/w)$ terms and retain a good approximation under the uniform distribution.

Our analysis of the small-bias case is inspired by the *gradually increasing independence* paradigm of Celis et al. [7],

[2] In the toy example we are currently studying, an alternative and simpler approach is to write $F_i(x) = (1-2^{-w/2})^{1-h_i(x)}$, where $h_i(x) = \bigvee_{j=1}^{w/2} x_{w(i-1)+j}$ is the indicator for whether $x$ already satisfies the $i$'th clause on its own. Then $F(x) = \prod_i F_i(x)$ expands as a power series in $\sum_i (1-h_i(x)-2^{-w/2})$, and higher moment bounds can be used to analyze what happens when we truncate this expansion. However, this expansion is rather specific to the highly symmetric Tribes function, whereas the expansion in terms of symmetric polynomials is much more general.

developed in the context of hashing. Every monomial in the $g_i$'s of degree at most $d$ depends on at most $wd = O(\log n)$ variables. A small-bias space provides an almost $O(\log n)$-wise independent distribution on the variables of $x$, so the $g_i(x)$'s will be almost $d$-wise independent. This ensures that polynomials in $g_1(x), \ldots, g_m(x)$ of degree at most $d$ (such as $S_1, \ldots, S_d$) will behave like they do under the uniform distribution. But we also need to argue that the $S_k$'s for $k > d$ have a small contribution to $\mathbb{E}_{x \sim \mathcal{D}}[F(x)]$.

Towards this end, we prove the following inequality for any real numbers $z_1, \ldots, z_m$: If $|S_1(z_1, \ldots, z_m)| \leq \frac{\mu}{2}, |S_2(z_1, \ldots, z_m)| \leq \frac{\mu^2}{2}$, then $|S_k(z_1, \ldots, z_m)| \leq \mu^k$. The proof uses the Newton–Girard formulas (see [46]) which relate the symmetric polynomials and power sums. This lets us repeat the same truncation argument, provided that $S_1(g_1(x), \ldots, g_m(x))$ and $S_2(g_1(x), \ldots, g_m(x))$ are tightly concentrated even under small-bias distributions. We prove this concentration holds via suitable higher moment inequalities.[3]

This lets us show that small bias fools $F(x)$. By iterating this argument $\log w$ times, we get a PRG for $f$ with polynomially small error and seed-length $O((\log n)(\log w)) = O((\log n)(\log \log n))$.

*Read-Once* CNF*s:* The case of general read-once CNFs presents several additional challenges. Since we no longer know how the variables are grouped into clauses, we (pseudo)randomly choose a subset of variables to assign using $\varepsilon$-biased spaces, and argue that for most clauses, we will not assign few variables. Clauses could now have very different sizes, and our approximation argument relied on tuning the amount of independence (or where we truncate) to the width of the clause. We handle this via an XOR lemma for $\varepsilon$-biased spaces, which lets us break the formula into $O(\log \log n)$ formulae, each having clauses of nearly equal size and argue about them separately.

*Combinatorial Rectangles:* A combinatorial rectangle $f : [W]^m \to \{0,1\}$ is a function of the form $f(x_1, \ldots, x_m) = \wedge_{i=1}^m f_i(x_i)$ for some Boolean functions $f_1, \ldots, f_m$. Thus, here we know which parts of the input correspond to which clauses (like the toy example above), but our clauses are arbitrary functions rather than ORs. To handle this, we use a more powerful family of gradual restrictions. Rather than setting $w/2$ bits of each coordinate, we instead (pseudo)randomly restrict the domain of each $x_i$ to a set of size $W^{1/2}$. More precisely, we use a small-bias space to pseudorandomly choose hash functions $h_1, \ldots, h_m : [W^{1/2}] \to [W]$ and replace $f$ with the restricted function $f'(z_1, \ldots, z_m) = \wedge_{i=1}^m (f_i \circ h_i)(z_i)$.

*Width* 3 *Branching Programs:* For width 3 branching programs, inspired by Sima and Zak [22] we reduce the task of constructing HSGs for width 3 to that of constructing HSGs for read-once CNF formulas where we also allow some clauses to be parities. Our PRG construction for read-once CNFs directly extends to also handle such formulas with parities (intuitively because small-bias spaces treat parities just like individual variables). The first step of our reduction actually works for any width $d$, and shows how to reduce the the task of constructing HSGs for width $d$ to constructing hitting set generators for width $d$ branching programs with sudden death, where the states in the *bottom level* are all assumed to be Reject states.

[3]These inequalities actually require higher moment bounds for the $g_i$'s. We ignore this issue in this description for clarity, and because we suspect that this requirement should not be necessary.

## II. Preliminaries

We briefly review some notation and definitions. We use $x \sim \mathcal{D}$ to denote sampling $x$ from a distribution $\mathcal{D}$. For a set $S$, $x \sim S$ denotes sampling uniformly from $S$. By abuse of notation, for a function $G : \{0,1\}^s \to \{0,1\}^n$ we let $G$ denote the distribution over $\{0,1\}^n$ of $G(y)$ when $y \sim \{0,1\}^s$. For a function $f : \{0,1\}^n \to \mathbb{R}$, we denote $\mathbb{E}[f] = \mathbb{E}_{x \sim \{0,1\}^n}[f(x)]$.

*Sandwiching Approximators.:* One of the central tools we use is to construct *sandwiching polynomial approximations* for various classes of functions. The approximating polynomials $(P_\ell, P_u)$ we construct for a function $f$ will have two properties: 1) low-complexity as measured by the "$\mathsf{L}_1$-norm" of $P_\ell, P_u$ and 2) they "sandwich" $f$, $P_u \leq f \leq P_u$. The first property will be important to argue that small-bias spaces *fool* the approximating polynomials and the second property will allow us to lift this property to the function being approximated. We formalize these notions below. For notational convenience, we shall view functions and polynomials as defined over $\{\pm 1\}^n$.

**Definition II.1.** *Let $P : \{\pm 1\}^n \to \mathbb{R}$ be a polynomial defined as $P(x) = \sum_{I \subseteq [n]} c_I \prod_{i \in I} x_i$. Then, the $\mathsf{L}_1$-norm of $P$ is defined by $\mathsf{L}_1[P] = \sum_{I \subseteq [n]} |c_I|$. We say $f : \{\pm 1\}^n \to \mathbb{R}$ has* $\delta$-sandwiching approximations *of $\mathsf{L}_1$ norm $t$ if there exist functions $f_u, f_\ell : \{\pm 1\}^n \to \mathbb{R}$ such that*

$$f_\ell(x) \leq f(x) \leq f_u(x) \ \forall x,$$
$$\mathbb{E}[f_u(x)] - \mathbb{E}[f_\ell(x)] \leq \delta$$
$$\mathsf{L}_1(f_\ell), \mathsf{L}_1(f_u) \leq t.$$

*We refer to $f_\ell$ and $f_u$ as the lower and upper sandwiching approximations to $f$ respectively.*

It is easy to see that the existence of such approximations implies that $f$ is $\delta + t\varepsilon$ fooled by any $\varepsilon$-biased distribution. In fact, as was implicit in the work of Bazzi [23] and formalized in the work of De et. al. [25], being fooled by small-bias spaces is essentially equivalent to the existence of good sandwiching approximators. We defer the detailed statements to the full version.

## III. Tools for Constructing Sandwiching Approximators

*Sandwiching Symmetric Functions:* We state our main result on sandwiching approximators for symmetric functions.

For $k \geq 1$, let $S_k : \mathbb{R}^m \to \mathbb{R}$ denote the $k^{th}$ elementary symmetric polynomial defined by

$$S_k(z_1, \ldots, z_m) = \sum_{I \subseteq [m], |I|=k} \prod_{i \in I} z_i.$$

**Theorem III.1.** *Let $g_1, \ldots, g_m : \{\pm 1\}^n \to \mathbb{R}$ be functions on disjoint sets of input variables and $\sigma_1, \sigma_2, \ldots, \sigma_m$ be positive numbers such that for all $i \in [m]$, $\mathbb{E}[g_i] = 0, \mathsf{L}_1[g_i] \leq t$, and for $k \geq 1$,*

$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[(g_i)^{2k}] \leq (2k)^{2k} \sigma_i^{2k}.$$

*Let $\sigma^2 = (\sum_i \sigma_i^2)/m$ and $\delta \in (0,1)$ and $\varepsilon, k > 0$ be such that*

$$\begin{aligned} m\sigma^2 &\leq \frac{1}{\log(1/\delta)^{25}}, \\ k &= \left\lceil \frac{5\log(1/\delta)}{\log(1/m\sigma^2)} \right\rceil, \\ \varepsilon &= \frac{\delta^4}{(mt+1)^{2k}}. \end{aligned} \tag{III.1}$$

*Let $P(x) = \sum_{i=0}^m c_i S_i(g_1(x), \ldots, g_m(x))$ be a symmetric multilinear function of the $g_i$s that computes a bounded function $P : \{\pm 1\}^n \to [-B, B]$, with $|c_i| \leq C$ for all $i \in [m]$. Then,*

1) *For every $\varepsilon$-biased distribution $\mathcal{D}$, we have*

$$\left| \mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[P(x)] - \mathop{\mathbb{E}}_{x \sim \mathcal{D}}[P(x)] \right| \leq O(B+C)\delta.$$

2) *$P$ has $O(B+C)\delta$ sandwiching approximations of $\mathsf{L}_1$ norm $O((B+C)(mt+1)^{2k}\delta^{-3})$.*

As an illustration of this theorem, we state the following immediate corollary which formalizes the intuition behind our arguments for the toy example in the introduction.

**Theorem III.2.** *Let $\kappa > 0$ be a constant. Let $g_1, \ldots, g_m : \{\pm 1\}^n \to [-\sigma, \sigma]$ be functions on disjoint sets of input variables with $\mathbb{E}[g_i] = 0$, $\mathsf{L}_1[g_i] = O(1)$ and $\sigma \leq 1/m^{-1/2-\kappa}$. Let $P : \{\pm 1\}^n \to [-1, 1]$ be a symmetric polynomial in $g_i$'s of the form $P(x) = \sum_{i=0}^m c_i S_i(g_1, \ldots, g_m)$, with $|c_i| \leq 1$. Then, for every $\delta \in (0,1)$, with $\log(1/\sigma) \geq \Omega_\kappa(\log(1/\delta))$, $P$ has $\delta$-sandwiching polynomials of $\mathsf{L}_1$-norm at most $\mathrm{poly}(1/\delta)$.*

In the notation from Section I-C, $m = 2^{w+1}$, $\sigma^2 \approx 2^{-3w/2}$ and all the other conditions hold.

A key insight in the proof of Theorem III.1 is that $|\sum_i g_i|$, $\sum_i g_i^2$ being small implies the smallness in absolute value of $S_k(g_1, \ldots, g_m)$ for every $k \geq 2$.[4] This is done using the following inequality, which might be of independent interest.

**Lemma III.3.** *Let $z_1, \ldots, z_m$ be real numbers that satisfy*

$$\left| \sum_{i=1}^m z_i \right| \leq \mu, \quad \sum_{i=1}^m z_i^2 \leq \mu^2.$$

*Then for every $k \geq 2$ we have $|S_k(z_1, \ldots, z_m)| \leq \mu^k$.*

*Proof:* To prove this lemma, we first bound the power sums $E_k(z_1, \ldots, z_m)$ which are defined as

$$E_k(z_1, \ldots, z_m) = \sum_{i=1}^m z_i^k.$$

Note that $E_1 = S_1$. We start by bounding $E_k$ for $k \geq 2$ using the $\mathbb{L}_k$ norm inequalities

$$\begin{aligned} |E_k(z_1, \ldots, z_m)|^{\frac{1}{k}} &\leq \left( \sum_{i=1}^m |z_i|^k \right)^{\frac{1}{k}} \\ &\leq \left( \sum_{i=1}^m z_i^2 \right)^{\frac{1}{2}} \\ &= E_2(z_1, \ldots, z_m)^{\frac{1}{2}} \end{aligned}$$

Hence we have $|E_k(z_1, \ldots, z_m)| \leq \mu^k$.

The relation between the power sums and elementary symmetric polynomials is given by the Newton-Girard identities (see [46], Chapter 7.1 for instance) discovered in the 17th century.

$$\begin{aligned} &S_k(z_1, \ldots, z_m) = \\ &\frac{1}{k} \sum_{i=1}^k (-1)^{i-1} S_{k-i}(z_1, \ldots, z_m) E_i(z_1, \ldots, z_m). \end{aligned} \tag{III.2}$$

We use these to show by induction on $k$ that $|S_k| \leq \mu^k$. For $k = 2$, we have

$$\begin{aligned} S_2(z_1, \ldots, z_m) &= \frac{1}{2}(S_1(z_1, \ldots, z_m)^2 - E_2(z_1, \ldots, z_m)) \\ &\leq \frac{1}{2}(\mu^2 + \mu^2) \leq \mu^2. \end{aligned}$$

Assume we have proved the bound up to $k-1$. Using the Newton-Girard formula,

$$\begin{aligned} |S_k(z_1, \ldots, z_m)| &\leq \frac{1}{k} \sum_{i=1}^k |S_{k-i}(z_1, \ldots, z_m)||E_i(z_1, \ldots, z_m)| \\ &\leq \frac{1}{k} \sum_{i=1}^k \mu^{k-i} \mu^i \leq \mu^k. \end{aligned}$$

■

[4] Up to constants, the hypothesis is equivalent to assuming bounds on $|S_1|$ and $|S_2|$

*XOR Lemma for $\varepsilon$-biased spaces:* We now state an *XOR Lemma* that helps us show the existence of good sandwiching approximators for the composition of a function on few variables with functions on disjoint sets of variables, each of which have good sandwiching approximators. We call it an XOR lemma, since one can view it as a generalization of Vazirani's XOR lemma.

**Theorem III.4.** *Let $f^1, \ldots, f^k : \{\pm 1\}^n \to [0,1]$ be functions on disjoint input variables such that each $f^i$ has $\varepsilon$-sandwiching approximations of $\mathsf{L}_1$ norm $t$. Let $H : [0,1]^k \to [0,1]$ be a multilinear function in its inputs. Let $h : \{\pm 1\}^n \to [0,1]$ be defined as $h(x) = H(f^1(x), \ldots, f^k(x))$. Then $h$ has $(16^k\varepsilon)$-sandwiching approximations of $\mathsf{L}_1$ norm $4^k(t+1)^k$.*

*Proof:* For $S \subseteq [k]$ define the monomial

$$M^S(x) = \prod_{i \in S} f^i(x) \prod_{j \notin S} (1 - f^j(x)).$$

Let $f_u^i$ and $f_\ell^i$ denote the upper and lower sandwiching approximations to $f^i$. Then we have

$$f_u^i(x) \geq f^i(x), \quad \mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[f_u^i(x) - f^i(x)] \leq \varepsilon.$$
$$1 - f_\ell^j(x) \geq 1 - f_\ell^j(x),$$
$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[(1 - f_\ell^j(x)) - (1 - f^j(x))] \leq \varepsilon.$$

Hence, if we define

$$M_u^S(x) = \prod_{i \in S} f_u^i(x) \prod_{j \notin S} (1 - f_\ell^j(x)),$$

then we have

$$M_u^S(x) \geq M^S(x) \ \forall \ x \in \{\pm 1\}^n,$$
$$\mathsf{L}_1[M_u^S] = \prod_{i \in S} \mathsf{L}_1[f_u^i] \prod_{j \notin S} \mathsf{L}_1[1 - f_\ell^j] \leq (t+1)^k.$$

One can show using a hybrid argument, that

$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[M_u^S(x) - M^S(x)] \leq 2^k \varepsilon.$$

We omit the proof for lack of space.

To construct a lower-sandwiching approximator, we observe that

$$\sum_{S \subseteq [k]} M^S(x) = \prod_{i \in [k]} (f^i(x) + 1 - f^i(x)) = 1.$$

Hence if we define

$$M_\ell^S(x) = 1 - \sum_{T \neq S} M_u^T(x)$$

then

$$M_\ell^S(x) \leq 1 - \sum_{T \neq S} M^T(x) = M^S(x),$$
$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[M^S(x) - M_\ell^S(x)] = \sum_{T \neq S} M_u^T(x) - M^T(x) \leq 4^k \varepsilon,$$
$$\mathsf{L}_1[M_\ell^S] \leq 2^k (t+1)^k.$$

Finally, let $\mathbf{1}_S \in \{0,1\}^k$ denote the indicator vector of the set $S$. Since $H$ is multilinear, we can write

$$H(y) = \sum_{S \subseteq [k]} H(\mathbf{1}_S) \prod_{i \in S} y_i \prod_{j \notin S} (1 - y_j)$$

where $H(\mathbf{1}_S) \in [0,1]$. Hence

$$h(x) = \sum_{S \subseteq [k]} H(\mathbf{1}_S) \prod_{i \in S} f_i(x) \prod_{j \notin S} (1 - f_j(x))$$
$$= \sum_{S \subseteq [k]} H(\mathbf{1}_S) M^S(x)$$

We define the polynomials

$$h_u(x) = \sum_{S \subseteq [k]} H(\mathbf{1}_S) M_u^S(x),$$
$$h_\ell(x) = \sum_{S \subseteq [k]} H(\mathbf{1}_S) M_\ell^S(x).$$

It follows that $h_u(x) \geq h(x) \geq h_\ell(x)$. Further

$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[h_u(x) - h_\ell(x)] \leq$$
$$\sum_{S \subseteq [k]} H(\mathbf{1}_S) \mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n}[M_u^S(x) - M_\ell^S(x)] \leq 16^k \varepsilon,$$
$$\mathsf{L}_1[h_u] \leq 2^k (t+1)^k, \ \mathsf{L}_1[h_\ell] \leq 4^k (t+1)^k.$$

■

## IV. PRG for Combinatorial Rectangles

We start by defining combinatorial rectangles (CRs).

**Definition IV.1.** *A* combinatorial rectangle *is a function $f : (\{\pm 1\}^w)^m \to \{0,1\}$ of the form $f(x_1, \ldots, x_m) = \bigwedge_{i=1}^m f_i(x_i)$, where $f_i : \{\pm 1\}^w \to \{0,1\}$, and each $x_i \in \{\pm 1\}^w$. We refer to the $f_i$s as the* co-ordinate functions *of $f$. We refer to $m$ as the* size[5] *of $f$ and $w$ as the* width.

We construct an explicit PRG for CRs with seed-length $\tilde{O}(\log m + w + \log(1/\delta))$. The previous best construction by Lu had a seed-length of $O(\log m + w + \log^{3/2}(1/\delta))$ [30].

**Theorem IV.2.** *There is an explicit pseudorandom generator for the class of combinatorial rectangles of width $w$ and size $m$ with error at most $\delta$ and seed-length $O((\log w)(\log(m) + w + \log(1/\delta)) + \log(1/\delta) \log\log(1/\delta) \log\log\log(1/\delta))$.*

Our generator uses a recursive sampling technique. We describe a single step of this procedure. For this informal description suppose that $\delta = 1/\mathrm{poly}(m)$, $w = O(\log m)$ and let $v = 3w/4$. Fix a CR $f : (\{\pm 1\}^w)^m \to \{0,1\}$.

Consider the following two-step process for generating a uniformly element $x$ from $(\{\pm 1\}^w)^m$.

- Choose a sequence of multi-sets $S_1, \ldots, S_m \subseteq \{\pm 1\}^w$ each of size $2^v$ by picking $2^v$ elements of $\{\pm 1\}^w$ independently and uniformly at random.

[5]This is usually referred to as the *dimension* in the literature; we use this terminology for the CNF analogy.

- Sample $x_i \sim S_i$ and set $x = (x_1, \ldots, x_m)$.

This results in an $x$ that is uniformly distributed over $(\{\pm 1\}^w)^m$. We will show that the $\mathbb{E}_x[f(x)]$ will not change much, even if the sampling in the first step can is done pseudorandomly using a small-bias space for suitably small $\varepsilon$.

Our final generator is obtained by iterating the one-step procedure for $T = O(\log\log m))$ steps: At step $t$ we choose multi-sets $S_1^t \subseteq S_1^{t-1}, \ldots, S_m^t \subseteq S_m^{t-1}$ each of cardinality exactly $2^{(3/4)^t w}$ using small-bias. After $T$ steps, we are left with a rectangle of width $w = O(\log\log m)$. Such rectangles can be fooled by $\varepsilon$-bias spaces where $\varepsilon = 1/m^{O(\log\log m)}$. The total randomness used over all the steps is $O((\log m) \cdot (\log\log m))$.

### A. Sandwiching Approximations for Bias Functions

In the following, let $f$ be a CR of width $w$ and coordinate functions $f_1, \ldots, f_m : \{\pm 1\}^w \to \{0,1\}$. We describe a restriction of $f$ which reduces the width from $w$ to $v = 3w/4$.

- For every $a \in \{\pm 1\}^v$, we sample string $x_a = (x_{a,1}, \ldots, x_{a,m}) \sim \{\{\pm 1\}^w\}^m$.
- For $i \in [m]$, we define restricted co-ordinate functions $f_i^v$ on inputs $y_i$ by $f_i^v(y_i) = f(x_{y_i,i})$.
- Define the restricted rectangle $f^v : (\{\pm 1\}^v)^m \to \{0,1\}$ on $y_1, \ldots, y_m$ by $f^v(y_1, \ldots, y_m) = \bigwedge_{i=1}^m f_i^v(y_i)$.

Let $\bar{x} \in \{\{\pm 1\}^w\}^{2^v \times m}$ denote the matrix whose rows are indexed by $a \in \{\pm 1\}^v$, the columns by $i \in [m]$ and $(a,i)$'th entry is given by $\bar{x}[a,i] = x_{a,i} \in \{\pm 1\}^w$. Every such matrix defines a *restriction* of $f$. We will show that if choosing $\bar{x}$ from an $\varepsilon$-biased space for $\varepsilon = 1/\mathrm{poly}(m)$ suitably small, and from the uniform distribution have almost the same effect on $f$. For $i \in [m]$, let $\bar{x}[i]$ denote the $i$'th column of $\bar{x}$. For each coordinate function $f_i$, define the *sample average* function

$$\bar{f}_i(\bar{x}) = \frac{1}{2^v} \sum_{a \in \{\pm 1\}^v} f_i(x_{a,i}) = \mathop{\mathbb{E}}_{a \sim \{\pm 1\}^v} \left[ f_i^v(a) \right]. \quad \text{(IV.1)}$$

Note that each $\bar{f}_i$ only depends on column $i$ of $\bar{x}$. Define the *bias* function of $\bar{x}$ as

$$F(\bar{x}) = \prod_{i=1}^m \bar{f}_i(\bar{x}) = \mathop{\mathbb{E}}_{y \sim (\{\pm 1\}^v)^m} \left[ f^v(y) \right]. \quad \text{(IV.2)}$$

The main lemma of this section shows that this bias function can be fooled by small-bias spaces.

**Lemma IV.3** (Main)**.** *Let $F$ be as defined in Equation IV.2. Assume that $\delta < 1/4$ and $w \leq \log(1/\delta)$, $v = 3w/4 \geq 50 \log\log(1/\delta)$. Then $F(x)$ has $\delta$-sandwiching approximations of $\mathsf{L}_1$ norm $\mathrm{poly}(1/\delta)$.*

We start by stating two simple claims, the second one justifies the name 'bias function' for $F$.

**Claim IV.4.** *For the sample average functions $\bar{f}_i$ defined as in Equation IV.1, we have*

$$\mathsf{L}_1(\bar{f}_i) \leq \mathsf{L}_1(f_i) \leq 2^{w/2}, \quad \mathop{\mathbb{E}}_{\bar{x} \sim \mathcal{U}} [\bar{f}_i(\bar{x})] = \mathop{\mathbb{E}}_{a \sim \{\pm 1\}^w} [f_i(a)].$$

**Claim IV.5.** *For $f^v$ and $F$ as defined in Equation IV.2, $\mathbb{E}_{y \sim (\{\pm 1\}^v)^m}[f^v(y)] = F(\bar{x})$.*

We will prove Lemma IV.3 by applying Theorem III.1 to the functions $g_i : \{\{\pm 1\}^w\}^{2^v} \to \mathbb{R}$ defined as follows: $g_i(\bar{x}) = (\bar{f}_i(\bar{x}) - p_i)/p_i$,, where $p_i = \mathbb{E}_{a \sim \{\pm 1\}^w}[f_i(x)]$. (We assume $p_i \neq 0$.)

We will need the following technical lemma, which helps us show that the functions $g_i$ satisfy the moment conditions needed to apply Theorem III.1. For brevity, let $\mathcal{U}$ denote $(\{\pm 1\}^v)^{2^v \times m}$ in the remainder of this section. We defer the proof to the full version.

**Lemma IV.6.** *Let $p_i, g_i, \mathcal{U}$ be defined as above. We have $\mathbb{E}_{\bar{x} \sim \mathcal{U}}[g_i(\bar{x})^{2k}] \leq (2k)^{2k} \sigma_i^{2k}$ where*

$$\sigma_i^2 = \begin{cases} \frac{(1-p_i)}{2^v p_i} & \text{for } p_i \in [2^{-v/10}, 1/2], \\ \frac{2(1-p_i)}{2^v} & \text{for } p_i \in [1/2, 1-2^{-v}], \\ \frac{2}{2^{2v}} & \text{for } p_i \in [1-2^{-v}, 1]. \end{cases}$$

*Proof of Lemma IV.3:* We first show the claim under the assumption that $\mathbb{E}[f] = p \geq \delta$ and later show how to get around this assumption. Define the sets

$$S_1 = \{i : p_i \in (0, 2^{-v/10}]\},$$
$$S_2 = \{i : p_i \in (2^{-v/10}, 1-2^{-v}],$$
$$S_3 = \{i : p_i \in (1-2^{-v}, 1]\}$$

For $j \in [3]$, let $F_j(\bar{x}) = \prod_{i \in S_j} \bar{f}_i(\bar{x})$ so that $F(\bar{x}) = \prod_{j=1}^3 F_j(\bar{x})$. We will construct sandwiching approximations for each $F_j$ and then combine them via Theorem III.4. We assume without loss of generality that $p_i \leq 1 - 2^{-w}$. Else, the $i$'th coordinate has bias 1 and can be ignored without changing the rest of the proof.

*Sandwiching $F_1$.:* We show that $\mathsf{L}_1[F_1]$ is itself small. Observe that

$$\delta \leq p = \prod_{i=1}^m p_i \leq \prod_{i \in S_1} p_i \leq 2^{-v|S_1|/10}$$

which implies that $|S_1| \leq 10\log(1/\delta)/v$. Thus, by Claim IV.4,

$$\mathsf{L}_1[F_1] \leq \prod_{i \in S_1} \mathsf{L}_1[\bar{f}_i] \leq 2^{\frac{w}{2}|S_1|} \leq \left(\frac{1}{\delta}\right)^{5w/v} \leq \frac{1}{\delta^{20/3}}.$$

*Sandwiching $F_2$.:* Note that

$$F_2(\bar{x}) = \prod_{i \in S_2} \bar{f}_i(\bar{x}) = \prod_{i \in S_2} p_i \cdot (1 + g_i(\bar{x})).$$

Notice that $F_2$ is a symmetric polynomial in the $g_i$'s, so we will obtain sandwiching polynomials for $F_2$ by applying Theorem III.1 to $g_i$'s. As before,

$$\delta \leq p \leq \prod_{i \in S_2} p_i \leq (1 - 2^{-v})^{|S_2|},$$

so we have $|S_2| \leq 2^v \log(1/\delta)$. Further we can write

$$\delta \leq p \leq \prod_{i \in S_2} (1 - (1 - p_i)) \leq e^{-\sum_{i \in S_2}(1-p_i)},$$

$$\text{Hence} \quad \sum_{i \in S_2} (1 - p_i) \leq 2 \log(1/\delta).$$

By Lemma IV.6, we have

$$\mathop{\mathbb{E}}_{x \sim \{\pm 1\}^n} [g_i(\bar{x})^{2k}] \leq (2k)^{2k} \sigma_i^{2k},$$

where $2/2^v \leq \sigma_i^2 = (1 - p_i)/2^{9v/10}$ for every $i \in S_2$. Hence,

$$\sum_{i \in S_2} \sigma_i^2 = \sum_{i \in S_2} \frac{1 - p_i}{2^{9v/10}} \leq \frac{2 \log(1/\delta)}{2^{9v/10}} \leq \frac{1}{\log(1/\delta)^{25}}.$$

Hence Theorem III.1 implies the existence of $O(\delta)$ ($B = 1$ and $C = \prod_{i \in S_2} p_i \leq 1$) sandwiching approximations with $\mathsf{L}_1$ norm bounded by $(mt + 1)^{2k}$ where

$$\begin{aligned} m = |S_2| &\leq 2^v \log(1/\delta) \leq 2^{5v/4}, \\ t &\leq 2^{w/2} \leq 2^v, \\ k &\leq \frac{5 \log(1/\delta)}{\log(1/\sum_i \sigma_i^2)} \leq \frac{25 \log(1/\delta)}{4v}. \end{aligned}$$

which implies the $\mathsf{L}_1$ norm is bounded by $\mathrm{poly}(1/\delta)$.

The arguments for getting sandwiching approximators for $F_3$ and from there to $F$ are similar. For lack of space, we defer the details to the full version. ∎

### *B. A Recursive Sampler for Combinatorial Rectangles*

We now use Lemma IV.3 recursively to prove Theorem IV.2. Our generator is based on a derandomized recursive sampling procedure which we describe below. The inputs are the width $w$ and the size $m$ of the rectangles we wish to fool and an error parameter $\delta \leq 1/2^w$.

1) Let $v_0 = w$, $v_j = \left(\frac{3}{4}\right)^j w$.
2) While $v_j \geq 50 \log\log(1/\delta)$ we sample $\bar{x}_j \in \{\{\pm 1\}^{v_{j-1}}\}^{2^{v_j} \times m}$ according to an $\varepsilon_1$-biased distribution for $\varepsilon \leq (1/\delta)^{c_1}$ for some large constant $c_1$.
3) Assume that at step $t$ (where $t = O(\log w)$), $v_t \leq 50 \log\log(1/\delta)$. Sample an input $\bar{x}_t \in (\{\pm 1\}^{v_{t-1}})^m$ from an $\varepsilon_2$-biased distribution where, for some large constant $c_2$,

$$\varepsilon_2 \leq (1/\delta)^{c_2 (\log\log(1/\delta) \log\log\log(1/\delta))}.$$

We next describe how we use $\mathbf{x} = (\bar{x}_1, \ldots, \bar{x}_t)$ to output an element of $(\{\pm 1\}^w)^m$. For $k \in \{1, \ldots, t-1\}$ we denote by $s_k$ the recursive sampling function which takes strings $\bar{x}_j \in \{\{\pm 1\}^{v_{j-1}}\}^{2^{v_j} \times m}$ for $j \in \{k+1, \ldots, t-1\}$ and $\bar{x}_t \in (\{\pm 1\}^{v_t})^m$ and produces an output string $s_k(\bar{x}_{k+1}, \ldots, \bar{x}_t) \in (\{\pm 1\}^{v_k})^m$. Set $s_{t-1}(\bar{x}_t) \equiv \bar{x}_t$. Fix $k < t-1$ and let $z = s_{k+1}(\bar{x}_{k+2}, \ldots, \bar{x}_t)$ be already defined. To define $s_k$, we will use $z$ to look up entries from the matrix $\bar{x}_{k+1}$, so that the $i$'th coordinate of $s_k$ will be the entry of $\bar{x}_{k+1}$ in the $z_i$'th row and $i$'th column:

$$\begin{aligned} s_k(\mathbf{x}) &\equiv s_k(\bar{x}_{k+1}, \ldots, \bar{x}_t) \\ &= ((\bar{x}_{k+1})_{z_1,1}, (\bar{x}_{k+1})_{z_2,2}, \ldots, (\bar{x}_{k+1})_{z_m,m}) \\ &\in (\{\pm 1\}^{v_k})^m. \end{aligned}$$

The above definition, though intuitive is a bit cumbersome to work with. It will be far easier for analysis to fix the input combinatorial rectangle $f : (\{\pm 1\}^w)^m \to \{0, 1\}$ and study the effect of the samplers $s_k$ on $f$. Let $f^0 = f$. Each matrix $\bar{x}_j$ gives a restriction of $f^{j-1}$: it defines restricted co-ordinate functions $f_i^j : \{\pm 1\}^{v_j} \to \{0, 1\}$ and a corresponding restricted rectangle $f^j : \{\{\pm 1\}^{v_j}\}^m \to \{0, 1\}$. We only use the following property of the $s_j$s:

$$f(s_0(\mathbf{x})) = f^1(s_1(\mathbf{x})) \cdots f^{t-1}(s_{t-1}(\mathbf{x})). \tag{IV.3}$$

To analyze the last step, we shall use the following simple corollary that follows from [25]. We defer the proof of the corollary to the full version.

**Corollary IV.7.** *Every combinatorial rectangle $f : \{\{\pm 1\}^v\}^m \to \{\pm 1\}$ is $\delta$-fooled by $\varepsilon$-bias spaces for $\varepsilon = (m 2^v/\delta)^{-O(v \log v)}$.*

Let $\mathcal{U}$ equal the domain of $\mathbf{x}$:

$$(\{\pm 1\}^{v_0})^{2^{v_1} \times m} \cdots \times (\{\pm 1\}^{v_{t-2}})^{2^{v_{t-1}} \times m} \times (\{\pm 1\}^{v_{t-1}})^m.$$

Let $\mathcal{D}^j$ denote the distribution on $\mathcal{U}$ where $\bar{x}_i$ are sampled from an $\varepsilon$-biased distribution for $i < j$ and uniformly for $i \geq j$. Then, $s_0(\mathcal{D}^0)$ is the uniform distribution on $\{\{\pm 1\}^w\}^m$ whereas $s_0(\mathcal{D}^t)$ is the output of our Recursive Sampler.

**Lemma IV.8.** *Let $f : \{\{\pm 1\}^w\}^m \to \{0, 1\}$ be a combinatorial rectangle with width $w$ and size $m$. For distributions $\mathcal{D}^0$ and $\mathcal{D}^t$ defined above, we have*

$$\left| \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}^0} [f(s_0(\mathbf{x}))] - \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}^t} [f(s_0(\mathbf{x}))] \right| \leq \delta.$$

*Proof:* Let $\delta' = \delta/t$. We will show by a hybrid argument that for all $j \in \{1, \ldots, t\}$

$$\left| \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}^{j-1}} [f(s_0(\mathbf{x}))] - \mathop{\mathbb{E}}_{\mathbf{x} \sim \mathcal{D}^j} [f(s_0(\mathbf{x}))] \right| \leq \delta'. \tag{IV.4}$$

In both $\mathcal{D}^{j-1}$ and $\mathcal{D}^j$, $\bar{x}_i$ is drawn from an $\varepsilon$-biased distribution for $i < j$, and from the uniform distribution for $i > j$. The only difference is $\bar{x}_j$ which is sampled uniformly in $\mathcal{D}^{j-1}$ and from an $\varepsilon$-biased distribution in $\mathcal{D}^j$.

We couple the two distributions by drawing $\bar{x}_i$ for $i < j$ according to an $\varepsilon$-biased distribution. By Equation (IV.3),

$$\begin{aligned}\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j-1}}[f(s_0(\mathbf{x}))] &= \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j-1}}[f^{j-1}(s_{j-1}(\mathbf{x}))],\\ \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j}}[f(s_0(\mathbf{x}))] &= \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j}}[f^{j-1}(s_{j-1}(\mathbf{x}))]\end{aligned}$$

and our goal is now to show that

$$\left|\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j-1}}[f^{j-1}(s_{j-1}(\mathbf{x}))] - \mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}^{j}}[f^{j-1}(s_{j-1}(\mathbf{x}))]\right| \le \delta'. \tag{IV.5}$$

Define the bias function $F^{j-1}$ of the rectangle $f^{j-1}$ as in Equation (IV.2). The string $\bar{x}_j$ defines a restricted rectangle $f^j : \{\{\pm1\}^{v_j}\}^m \to \{0,1\}$. Applying Claim IV.5 we get

$$\mathop{\mathbb{E}}_{z\sim(\{\pm1\}^{v_j})^m}[f^j(z)] = F^{j-1}(\bar{x}_j).$$

In both distributions $\mathcal{D}^{j-1}$ and $\mathcal{D}^j$, $\bar{x}_{j+1},\ldots,\bar{x}_t$ are distributed uniformly at random, hence $s_j(\mathcal{D}^{j-1}) = s_j(\mathcal{D}^j) \sim (\{\pm1\}^{v_j})^m$ are uniformly distributed, and this variable is independent of $\bar{x}_j$. So we have

$$\begin{aligned}&\mathop{\mathbb{E}}_{x\sim\mathcal{D}^{j-1}}[f^{j-1}(s_{j-1}(x))] =\\ &\mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j-1}}\left[\mathop{\mathbb{E}}_{(\bar{x}_{j+1}\ldots,\bar{x}_t)\sim\mathcal{D}^{j-1}}[f^j(s_j(\bar{x}_{j+1},\ldots,\bar{x}_t))]\right] =\\ &\mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j-1}}[F^{j-1}(\bar{x}_j)]\\ &\mathop{\mathbb{E}}_{x\sim\mathcal{D}^{j}}[f^{j-1}(s_{j-1}(x))] =\\ &\mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j}}\left[\mathop{\mathbb{E}}_{(\bar{x}_{j+1}\ldots,\bar{x}_t)\sim\mathcal{D}^{j}}[f^j(s_j(\bar{x}_{j+1},\ldots,\bar{x}_t))]\right] =\\ &\mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j}}[F^{j-1}(\bar{x}_j)]\end{aligned},$$

Thus it suffices to show that

$$\left|\mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j-1}}[F^{j-1}(\bar{x}_j)] - \mathop{\mathbb{E}}_{\bar{x}_j\sim\mathcal{D}^{j}}[F^{j-1}(\bar{x}_j)]\right| \le \delta'$$

By Lemma IV.3, this holds true for $j \le t-1$ provided that $\varepsilon_1 \le \mathrm{poly}(1/\delta')$.

For $j = t$, note that this is equivalent to showing that $\varepsilon_2$-bias fools the rectangle $f^t$. By Corollary IV.7, $f^t$ is $\delta'$ fooled by $\varepsilon_2$-biased spaces where

$$\begin{aligned}\varepsilon_2 &= \left(\frac{m2^{v_t}}{\delta'}\right)^{-O(v_t\log v_t)}\\ &= \left(\frac{1}{\delta'}\right)^{O(\log\log(1/\delta')\log\log\log(1/\delta'))}.\end{aligned}$$

Plugging these back into Equation (IV.4), the error is bounded by $t\cdot\delta' \le \delta$. ∎

To complete the proof of Theorem IV.2, we observe that the total seed-length is

$$\begin{aligned}s &= O\left((\log w)(\log(m2^w/\varepsilon_1) + \log(m2^w/\varepsilon_2)\right)\\ &= O(\log w\,(\log m + w + \log(1/\delta)) +\\ &\log(1/\delta)\log\log(1/\delta)\log\log\log(1/\delta)).\end{aligned}$$

## References


[1] R. Impagliazzo and A. Wigderson, "*P = BPP* if *E* requires exponential circuits: Derandomizing the XOR lemma," in *STOC*, 1997, pp. 220–229.

[2] N. Nisan, "Pseudorandom bits for constant depth circuits," *Combinatorica*, vol. 11, no. 1, pp. 63–70, 1991.

[3] ——, "Pseudorandom generators for space-bounded computation," *Combinatorica*, vol. 12, no. 4, pp. 449–461, 1992.

[4] P. Indyk, "Stable distributions, pseudorandom generators, embeddings, and data stream computation," *J. ACM*, vol. 53, no. 3, pp. 307–323, 2006.

[5] D. Sivakumar, "Algorithmic derandomization via complexity theory," in *IEEE Conference on Computational Complexity*, 2002, p. 10.

[6] L. Trevisan, "Extractors and pseudorandom generators," *J. ACM*, vol. 48, no. 4, pp. 860–879, 2001.

[7] L. E. Celis, O. Reingold, G. Segev, and U. Wieder, "Balls and bins: Smaller hash families and faster evaluation," in *FOCS*, 2011, pp. 599–608.

[8] A. Healy, S. P. Vadhan, and E. Viola, "Using nondeterminism to amplify hardness," *SIAM J. Comput.*, vol. 35, no. 4, pp. 903–931, 2006.

[9] E. Kaplan, M. Naor, and O. Reingold, "Derandomized constructions of k-wise (almost) independent permutations," in *APPROX-RANDOM*, 2005, pp. 354–365.

[10] I. Haitner, D. Harnik, and O. Reingold, "On the power of the randomized iterate," in *Advances in Cryptology—CRYPTO '06*, ser. Lecture Notes in Computer Science, C. Dwork, Ed. Springer-Verlag, 2006.

[11] M. E. Saks and S. Zhou, "$\mathrm{BP}_H\mathrm{SPACE(S)} \subseteq \mathrm{DSPACE}(S^{3/2})$," *J. Comput. Syst. Sci.*, vol. 58, no. 2, pp. 376–403, 1999.

[12] R. Impagliazzo, N. Nisan, and A. Wigderson, "Pseudorandomness for network algorithms," in *STOC*, 1994, pp. 356–364.

[13] N. Nisan and D. Zuckerman, "Randomness is linear in space," *J. Comput. Syst. Sci.*, vol. 52, no. 1, pp. 43–52, 1996.

[14] R. Raz and O. Reingold, "On recycling the randomness of states in space bounded computation," in *STOC*, 1999, pp. 159–168.

[15] O. Reingold, "Undirected connectivity in log-space," *J. ACM*, vol. 55, no. 4, pp. Art. 17, 24, 2008.

[16] O. Reingold, L. Trevisan, and S. Vadhan, "Pseudorandom walks in regular digraphs and the RL vs. L problem," in *STOC*, 2006, pp. 457–466.

[17] M. Braverman, A. Rao, R. Raz, and A. Yehudayoff, "Pseudorandom generators for regular branching programs," in *FOCS*, 2010, pp. 40–47.

[18] J. Brody and E. Verbin, "The coin problem and pseudorandomness for branching programs," in *FOCS*, 2010, pp. 30–39.

[19] M. Koucký, P. Nimbhorkar, and P. Pudlák, "Pseudorandom generators for group products: extended abstract," in *STOC*. ACM, 2011, pp. 263–272.

[20] A. De, "Pseudorandomness for permutation and regular branching programs," in *IEEE Conference on Computational Complexity*. IEEE Computer Society, 2011, pp. 221–231.

[21] M. Saks and D. Zuckerman, 1995, unpublished manuscript.

[22] J. Síma and S. Zák, "Almost $k$-wise independent sets establish hitting sets for width-3 1-branching programs," in *CSR*, 2011, pp. 120–133.

[23] L. M. J. Bazzi, "Polylogarithmic independence can fool DNF formulas," *SIAM J. Comput.*, vol. 38, no. 6, pp. 2220–2272, 2009.

[24] A. Razborov, "A simple proof of Bazzis theorem," *ACM Trans. Comput. Theory*, vol. 1, no. 1, pp. 3:1–3:5, Feb. 2009.

[25] A. De, O. Etesami, L. Trevisan, and M. Tulsiani, "Improved pseudorandom generators for depth 2 circuits," in *APPROX-RANDOM*, 2010, pp. 504–517.

[26] A. R. Klivans, H. K. Lee, and A. Wan, "Mansour's conjecture is true for random DNF formulas," in *COLT*, 2010, pp. 368–380.

[27] R. Armoni, M. E. Saks, A. Wigderson, and S. Zhou, "Discrepancy sets and pseudorandom generators for combinatorial rectangles," in *FOCS*, 1996, pp. 412–421.

[28] G. Even, O. Goldreich, M. Luby, N. Nisan, and B. Velickovic, "Efficient approximation of product distributions," *Random Struct. Algorithms*, vol. 13, no. 1, pp. 1–16, 1998.

[29] N. Linial, M. Luby, M. E. Saks, and D. Zuckerman, "Efficient construction of a small hitting set for combinatorial rectangles in high dimension," *Combinatorica*, vol. 17, no. 2, pp. 215–234, 1997.

[30] C.-J. Lu, "Improved pseudorandom generators for combinatorial rectangles," *Combinatorica*, vol. 22, no. 3, pp. 417–434, 2002.

[31] E. Viola, "The sum of d small-bias generators fools polynomials of degree d," in *IEEE Conference on Computational Complexity*, 2008, pp. 124–127.

[32] J. Naor and M. Naor, "Small-bias probability spaces: Efficient constructions and applications," *SIAM J. Comput.*, vol. 22, no. 4, pp. 838–856, 1993.

[33] N. Alon, O. Goldreich, J. Håstad, and R. Peralta, "Simple construction of almost k-wise independent random variables," *Random Struct. Algorithms*, vol. 3, no. 3, pp. 289–304, 1992.

[34] A. Bogdanov and E. Viola, "Pseudorandom bits for polynomials," *SIAM J. Comput.*, vol. 39, no. 6, pp. 2464–2486, 2010.

[35] S. Lovett, "Unconditional pseudorandom generators for low degree polynomials," in *STOC*, 2008, pp. 557–562.

[36] A. Bogdanov, Z. Dvir, E. Verbin, and A. Yehudayoff, "Pseudorandomness for width 2 branching programs," *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 16, p. 70, 2009.

[37] L. G. Valiant, "Graph-theoretic arguments in low-level complexity," in *Mathematical foundations of computer science (Proc. Sixth Sympos., Tatranská Lomnica, 1977)*. Berlin: Springer, 1977, pp. 162–176. Lecture Notes in Comput. Sci., Vol. 53.

[38] C.-J. Lu, S.-C. Tsai, and H.-L. Wu, "Improved hardness amplification in np," *Theor. Comput. Sci.*, vol. 370, no. 1-3, pp. 293–298, 2007.

[39] M. Ajtai, "$\Sigma_1^2$-formula on finite structures," *Ann. Pure. Appl. Logic*, vol. 24, pp. 1–48, 1983.

[40] M. L. Furst, J. B. Saxe, and M. Sipser, "Parity, circuits, and the polynomial-time hierarchy," *Mathematical Systems Theory*, vol. 17, no. 1, pp. 13–27, 1984.

[41] J. Håstad, "Almost optimal lower bounds for small depth circuits," in *STOC*, 1986, pp. 6–20.

[42] M. Ajtai and A. Wigderson, "Deterministic simulation of probabilistic constant depth circuits," in *FOCS*, 1985, pp. 11–19.

[43] M. Agrawal, E. Allender, R. Impagliazzo, T. Pitassi, and S. Rudich, "Reducing the complexity of reductions," *IEEE Conference on Computational Complexity*, vol. 10, no. 2, pp. 117–138, 2001.

[44] P. Gopalan, R. Meka, and O. Reingold, "DNF sparsification and a faster deterministic counting algorithm," in *IEEE Conference on Computational Complexity*, 2012.

[45] M. Ben-Or and N. Linial, "Collective coin flipping, robust voting schemes and minima of banzhaf values," in *FOCS*, 1985, pp. 408–416.

[46] D. Cox, J. Little, and D. O'Shea, *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer, 2007.