Quantified Derandomization: How to Find Water in the Ocean

Other titles in Foundations and Trends® in Theoretical Computer Science

Complexity Theory, Game Theory, and Economics: The Barbados Lectures
Tim Roughgarden
ISBN: 978-1-68083-654-7

Semialgebraic Proofs and Efficient Algorithm Design
Noah Fleming, Pravesh Kothari and Toniann Pitassi
ISBN: 978-1-68083-636-3

Higher-order Fourier Analysis and Applications
Hamed Hatami, Pooya Hatami and Shachar Lovett
ISBN: 978-1-68083-592-2

On Doubly-Efficient Interactive Proof Systems
Oded Goldreich
ISBN: 978-1-68083-424-6

Coding for Interactive Communication: A Survey
Ran Gelles
ISBN: 978-1-68083-346-1

Quantified Derandomization: How to Find Water in the Ocean

Roei Tell
Institute for Advanced Study and DIMACS
roeitell@gmail.com

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

R. Tell. *Quantified Derandomization: How to Find Water in the Ocean*. Foundations and Trends® in Theoretical Computer Science, vol. 15, no. 1, pp. 1–125, 2022.

ISBN: 978-1-63828-093-4

Foundations and Trends® in Theoretical Computer Science
Volume 15, Issue 1, 2022

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Theoretical Computer Science publishes survey and tutorial articles in the following topics:

- Algorithmic game theory
- Computational algebra
- Computational aspects of combinatorics and graph theory
- Computational aspects of communication
- Computational biology
- Computational complexity
- Computational geometry
- Computational learning
- Computational Models and Complexity
- Computational Number Theory
- Cryptography and information security
- Data structures
- Database theory
- Design and analysis of algorithms
- Distributed computing
- Information retrieval
- Operations Research
- Parallel algorithms
- Quantum Computation
- Randomness in Computation

Information for Librarians

Foundations and Trends® in Theoretical Computer Science, 2022, Volume 15, 4 issues. ISSN paper version 1551-305X. ISSN online version 1551-3068. Also available as a combined paper and online subscription.

Contents

Quantified Derandomization: How to Find Water in the Ocean

Roei Tell

Institute for Advanced Study and DIMACS, New Jersey, USA; roeitell@gmail.com

ABSTRACT

The focus of this survey is the question of quantified derandomization, which was introduced by Goldreich and Wigderson [44]: Does derandomization of probabilistic algorithms become easier if we only want to derandomize algorithms that err with extremely small probability? How small does this probability need to be in order for the problem's complexity to be affected?

This question opens the door to studying natural relaxed versions of the derandomization problem, and allows us to construct algorithms that are more efficient than in the general case as well as to make gradual progress towards solving the general case. In the survey I describe the body of knowledge accumulated since the question's introduction, focusing on the following directions and results:

1. **Hardness vs "quantified" randomness:** Assuming sufficiently strong circuit lower bounds, we can derandomize probabilistic algorithms that err extremely rarely while incurring *essentially no time overhead*.
2. For general probabilistic polynomial-time algorithms, **improving on the brute-force algorithm for quan-**

Roei Tell (2022), "Quantified Derandomization: How to Find Water in the Ocean", Foundations and Trends® in Theoretical Computer Science: Vol. 15, No. 1, pp 1–125.
DOI: 10.1561/0400000108.

tified derandomization implies breakthrough circuit lower bounds, and this statement holds for *any given probability of error*.

3. Unconditional **algorithms for quantified derandomization of low-depth circuits and formulas,** as well as *near-matching reductions* of the general derandomization problem to quantified derandomization for such models.
4. **Arithmetic quantified derandomization**, and in particular constructions of hitting-set generators for polynomials that vanish extremely rarely.
5. **Limitations of certain black-box techniques** in quantified derandomization, as well as a tight connection between black-box quantified derandomization and the classic notion of **pseudoentropy**.

Most of the results in the survey are from known works, but several results are either new or are strengthenings of known results. The survey also offers a host of concrete challenges and open questions surrounding quantified derandomization.

1

Introduction

> *Does derandomization of probabilistic algorithms become easier when the number of "bad" random inputs is extremely small? (Goldreich and Wigderson [44])*

The context for this survey is the question of derandomization: Can we simulate randomness in a deterministic and efficient way? More accurately, we ask *which types* of randomized algorithms can be simulated in a deterministic way, and what is the precise cost of simulation. The main focus in this study is on simulating randomized algorithms that solve *decision problems*, which is the $\mathcal{BPP}$ vs $\mathcal{P}$ question.[1] As we can expect of one the main questions in complexity theory, progress on it has been challenging, and we know that essentially any progress on this question is closely related to progress on other central questions in complexity theory.

The textbook definition of *probabilistically solving a decision problem* $L \subseteq \{0,1\}^*$, which underlies the definition of $\mathcal{BPP}$, considers a

[1] As usual, this focus is taken merely for simplicity, and there is an efficient search-to-decision reduction in this setting (i.e., search problems that can be efficiently solved by probabilistic algorithms, and for which solutions can be efficiently verified, reduce to *promise-*$\mathcal{BPP}$; see [42, Theorem 3.5]).

randomized algorithm to be successful if it errs with probability at most 1/3 on every fixed input; that is, the fraction of random strings that cause the algorithm to err is at most 1/3.

This survey is concerned with the seemingly innocent choice of error bound 1/3. Going back to the original definition of $\mathcal{BPP}$ in [38], the class was defined with an unspecified error bound that can be any constant smaller than 1/2, such as .49. On the other hand, when we present this topic to non-expert audiences, we sometimes choose a miniscule constant such as 10^{-10} for dramatic effect. Of course, both formulations are essentially equivalent, since we can apply *error reduction* to efficiently reduce the error from $1/2 - n^{-O(1)}$ to $2^{-\mathrm{poly}(n)}$ with only a polynomial runtime overhead.

Therefore, a common sentiment is that the precise choice of error bound doesn't really matter, as long as it is noticeably smaller than 1/2.[2] But is this sentiment accurate even when we take a sub-constant error bound very close to zero, focusing on algorithms that only err extremely rarely? It turns out that in this setting, *the precise choice of error bound matters a lot*. In fact, the problem is so sensitive to this choice that even tiny changes in the error bound mark the difference between settings in which efficient derandomization is known, and settings in which showing even mild derandomization would yield dramatic consequences in complexity theory.

1.1 The general question

Let's start with a trivial extreme point: If we define a probabilistic algorithm to be successful only if it never errs – that is, we set the error bound in the definition of $\mathcal{BPP}$ to be zero – then we just defined *deterministic* computation in a cumbersome way; needless to say, derandomization becomes trivial in this case. But what if we allow the randomized algorithm to err on *just a single random string*, out of the exponentially many possible choices for random strings? What if we

[2] Allowing error that is arbitrarily close to 1/2 is a different story. Such a choice is less natural (since we are defining a negligible improvement over a random coin toss as "successfully solving a problem") and yields the very large complexity class $\mathcal{PP}$ (recall that $\mathcal{P}^{\mathcal{PP}} = \mathcal{P}^{\#\mathcal{P}} \supseteq \mathcal{PH}$, using [104]).

allow it to err on polynomially many strings? Where is the threshold at which the derandomization problem stops being trivial, and what happens beyond this threshold?

Several years ago Goldreich and Wigderson [44] asked these questions in a broad and methodical way, leading to a fruitful study of what they called quantified derandomization: This is the question of *derandomizing algorithms that err extremely rarely*, where "extremely rarely" here refers to the number of random strings that cause the probabilistic algorithm to err. As they mention in their work, an early form of this question was already considered long ago by Sipser [93], who considered the class "strong $\mathcal{R}$" of problems solvable with extremely small one-sided error.

Let us define the notion of probabilistically solving a decision problem with error bound B, where the parameter B will quantify the number of exceptional strings of random bits (i.e., the number of strings that, when used by the algorithm as a sequence of coin tosses, cause the algorithm to err). We will measure B as a function of the number of random coins (rather than of the input length), since we are interested in comparing the number of exceptional random strings to the total number of choices for a random string. For simplicity of presentation, let us assume for the moment that the number of random coins equals the running time. (We will get rid of this simplifying assumption later on in Section 3.3.)

Definition 1.1 (probabilistically solving a decision problem with error bound B)**.** For $B\colon \mathbb{N} \to \mathbb{N}$, we say that $\Pi = (\mathsf{Y}, \mathsf{N}) \subseteq \{0,1\}^* \times \{0,1\}^*$ is in $pr\mathcal{BPTIME}_B[T]$ if there exists a randomized algorithm that gets input $x \in \{0,1\}^*$, runs in time $T = T(|x|)$, and:

1. If $x \in \mathsf{Y}$, the algorithm *accepts* given all but at most $B(T)$ choices of random strings.

2. If $x \in \mathsf{N}$, the algorithm *rejects* given all but at most $B(T)$ choices of random strings.

I stress again that that $B(T)$ is the *absolute number* of exceptional random strings in Definition 1.1, rather than their fraction. Thus, and since we assumed (for now) that the number of random coins equals the running time T, the error probability of the algorithm in Definition 1.1

is $B(T)/2^T$. Indeed, the standard definition of $pr\mathcal{BPTIME}[T]$ is the special case obtained by using $B(T) = 2^T/3$.

Trying to derandomize only algorithms that err extremely rarely makes the challenge potentially easier; that is, Definition 1.1 opens the door for a *relaxation of the classical derandomization problem*. However, this relaxation entirely hinges on the choice of function B: For small values of B (e.g., for $B(T)) = O(1)$) the corresponding derandomization problem is easy, since we can just use the brute-force deterministic simulation that runs the original algorithm using $2B(T) + 1$ fixed choices of a random string; whereas for larger values of B (e.g., for $B(T) = \Omega(2^T)$) the derandomization problem is as challenging as the original and general derandomization problem.

1.2 The role of error-reduction

As mentioned above, we can efficiently *reduce the error* of a probabilistic algorithm. The naive way to do so is to repeat an algorithm that has error $1/3$ for k times and output the majority decision, which reduces its error to $2^{-\Omega(k)}$. This naive method reduces B only mildly as a function of the number of random coins, and using more sophisticated tools we can reduce B to be (say) subexponential in the number of random coins at a relatively low computational cost (see Section 4.3 for details).[3] This means that, in high-level, *general derandomization reduces to quantified derandomization* with relatively small values of B and with a corresponding computational overhead.

The point is that, in contrast to a common mistaken intuition, this does not trivialize the question of quantified derandomization, but rather (to the contrary) highlights its importance. Specifically, this suggests a natural approach to solve the general derandomization problem: First reduce general derandomization to quantified derandomization (e.g., by error-reduction), and then solve the corresponding quantified derandomization problem. Indeed, when taking this approach what we

[3] To be more precise, let us measure B as a function of the number of random coins R. Naive error-reduction only yields $B(R) = 2^{(1-o(1))\cdot R}$, since repeating an algorithm with $r = \omega(1)$ coins for k times yields an algorithm with $R = k \cdot r$ coins and error probability $2^{-\Omega(k)} = 2^{-\Omega(R/r)}$.

are actually asking is whether we can *reduce general derandomization to a target setting of quantified derandomization that we can efficiently solve*. This calls for developing efficient algorithms for quantified derandomization, as well as efficient approaches for error-reduction.[4] We will see both types of results in this survey.

1.3 Additional motivation

Derandomizing algorithms that err extremely rarely is, in my view, a natural problem that is inherently interesting, and therefore it does not need additional external motivations. (Indeed, recall that the problem was considered as early as 1986 [93].) For example, one may ask what is the precise time complexity of derandomizing algorithms that err extremely rarely, or which assumptions are sufficient and necessary in order to do so (as we will see, both questions have recently been studied).

Nevertheless, let me mention two additional motivations for studying quantified derandomization, where both of them view this question as a *stepping-stone towards solving the general derandomization problem*. The first additional motivation is that, as explained in Section 1.2, a natural approach to solve the general derandomization problem is to reduce it to quantified derandomization and then solve the latter.

The second additional motivation is more generic: Studying a potentially easy special case (i.e., quantified derandomization) may shed light on the general case (i.e., general derandomization), and pave the way for gradual progress towards solving the latter. It turns out that this generic motivation materialized in a fruitful way in the case of quantified derandomization: The *results* that we will see are surprising, rely on new techniques, and point both at specific technical challenges that create bottlenecks and at connections between quantified derandomization and well-known questions in complexity theory (e.g., circuit lower bounds and pseudoentropy).

[4] In general, applying standard black-box techniques for error-reduction and then the brute-force algorithm for quantified derandomization does not yield a non-trivial algorithm for general derandomization (see Appendix A). Thus, when using this approach, we need either a better-than-brute-force algorithm for quantified derandomization, or a non-standard technique for error-reduction.

Lastly, as pointed out by Avi Wigderson, the study of quantified derandomization led to constructions of important pseudorandom objects. For example, Sipser's [93] original work was one of the driving forces behind the study of explicit randomness extractors (see, e.g., [28, Acknowledgements]). Analogously, the recent introduction of quantified derandomization in [44] led to constructions of pseudorandom restriction algorithms for weak circuit classes, and to constructions of extractors that are computable in weak circuit classes (see, e.g., Appendices B and C, respectively).

1.4 Organization

An overview of the results that are included in this survey is presented in Section 3. After stating preliminary definitions in Section 4, the subsequent Sections 5, 6, 7, 8, 9, and 10 expand on each of the subsections of Section 3, respectively, elaborating on the high-level results with more technical details and explanations. A reader interested in open problems in quantified derandomization will find numerous ones in Section 11.

Appendix A expands on Footnote (4) above. Appendices B and C describe technical constructions that underlie some of the results described in Section 3. Finally, Appendix D surveys two additional settings for quantified derandomization that have been explored relatively less so far.

Acknowledgements

I'm grateful to Oded Goldreich for many valuable comments on a draft of the survey, for elaborate discussions about the conceptual perspective, and for pointing out the elementary proof of Theorem 3.1 (replacing an original complicated proof). I thank Ryan Williams for encouraging me to write the survey, for pointing out that Theorem 3.5 is a strict generalization of his result [114], and for providing good writing advice. I'm grateful to Avi Wigderson for several useful comments and additions, and in particular for pointing out a flaw that existed in the definitions in an early draft. I thank Lijie Chen for very useful comments, and for suggesting the same elementary proof of Theorem 3.1 suggested by Oded. And I'm grateful to William Hoza for sharing his proof of Theorem D.1 and for his permission to include it in the survey.

I thank two anonymous reviewers for their useful suggestions about the presentation and organization of the survey, as well as for pointing out many minor inaccuracies.

Part of this work was conducted while I was supported by the National Science Foundation under grant number CCF-1445755 and under grant number CCF-1900460.

Appendices

A

Error-reduction by Itself Is Not Enough

Can we construct a better-than-brute-force algorithm for CAPP via the naive approach of first reducing CAPP to QD_B using a standard sampler-based error-reduction, and then using a brute-force algorithm for QD_B (i.e., solving quantified derandomization by evaluating the given circuit over some fixed $O(B(n))$ inputs)?

The following result shows a negative answer to this question: Any such algorithm will be slower than the brute-force algorithm that simply evaluates the original circuit on all of its inputs. The meaning of this result is that when constructing CAPP algorithms that are based on an initial step of sampler-based error-reduction, a *non-trivial algorithm for quantified derandomization is necessary*. The statement below shows that this is the case even for derandomization with one-sided error (i.e., for $\mathsf{CAPP}_{\frac{1}{2},0}$) and even when using dispersers rather than samplers.

Definition A.1 (disperser)**.** A function $\mathsf{Disp}\colon \{0,1\}^{\bar{n}} \times \{0,1\}^{\ell} \to \{0,1\}^{n}$ is a (k,ϵ)-disperser if for every $T \subseteq \{0,1\}^{n}$ of density $|T|/2^{n} \geq \epsilon$, for all but at most 2^{k} strings $z \in \{0,1\}^{\bar{n}}$ there exists $s \in \{0,1\}^{\ell}$ such that $\mathsf{Disp}(z,s) \in T$.

Theorem A.1 (disperser-based error-reduction should be coupled with non-trivial algorithms for quantified derandomization)**.** Consider the follow-

ing algorithm for $\mathsf{CAPP}_{\frac{1}{2},0}$. Given an n-bit circuit C, let $\mathsf{Disp}\colon \{0,1\}^{\bar{n}} \times \{0,1\}^{\ell} \to \{0,1\}^{n}$ be an arbitrary $(k,.01)$-disperser for some value of $k \leq n$. The algorithm:

1. Constructs the circuit $C'\colon \{0,1\}^{\bar{n}} \to \{0,1\}$ such that $C'(z) = \bigvee_{s\in\{0,1\}^{\ell}} C(\mathsf{Disp}(z,s))$.[1]
2. Evaluates C' over (arbitrary) fixed 2^k+1 inputs.
3. Outputs "yes" if and only if C' accepted one of the inputs.

Then, the running time of this algorithm is larger than $2^n \cdot \tilde{O}(|C|)$.

Proof. Radhakrishnan and Ta-Shma [83] proved that for any $(k,.01)$-disperser $\mathsf{Disp}\colon \{0,1\}^{\bar{n}} \times \{0,1\}^{\ell} \to \{0,1\}^{n}$ it holds that $n \leq k+\ell-O(1)$ (i.e., an entropy loss is inherent). Also note that the size of C' is more than $2^{\ell} \cdot |C|$, even without taking into account the complexity of Disp. Thus, the running time of the algorithm is $\left(2^k+1\right) \cdot \tilde{O}(|C'|) > 2^{k+\ell} \cdot \tilde{O}(|C|) \geq 2^n \cdot \tilde{O}(|C|)$. ■

[1] That is, the circuit C' gets input $z \in \{0,1\}^{\bar{n}}$, computes the 2^{ℓ} values $\{\mathsf{Disp}(z,s)\}_{s\in\{0,1\}^{\ell}}$, evaluates C on each of these values, and outputs 1 iff there is $s \in \{0,1\}^{\ell}$ such that $C(\mathsf{Disp}(z,s)) = 1$.

B

Pseudorandom Restrictions for Low-depth Circuits and Formulas

In this section I describe the technical results underlying the algorithms for quantified derandomization that were presented in Section 7. These technical results assert the existence of *efficient pseudorandom restriction procedures* that yield simplifier sets, in the sense of Definition 8.1.

B.1 Width-dependent derandomization of Håstad's switching lemma

Let me start with the class $\mathcal{AC}^0$. Using standard techniques following [47], the problem of constructing a pseudorandom restriction procedure reduces to the problem of derandomizing Håstad's switching lemma [47]; that is, to the problem of constructing a pseudorandom distribution of restrictions that simplifies every depth-2 formula into a decision tree of bounded depth, with high probability (see, e.g., [101, Proof of Theorem 5.16] for an explanation).[1]

[1] We will focus on pseudorandom distributions that achieve the same bound on the decision tree depth, and approximately the same error probability, as in Håstad's original result [47]. Pseudorandom restriction procedures that achieve worse parameters but are more efficient are known (these date back to [5], with a recent construction presented in [45]).

Note that for our application (i.e., to construct an algorithm for QD_B) we want to pseudorandomly choose *both the variables to fix and the values for fixed variables*. This should be distinguished from applications for which we only need to pseudorandomly choose which variables to fix, while leaving the choice of values to be completely uniform. (A very recent result of Kelley [62] showed that the latter task can be solved in polynomial time with seed length $O(\log(n))$.)

To optimize the trade-off between $B(n)$ and the seed length, we will be interested in derandomization of the switching lemma for depth-two formulas of *bounded width* (see [101] for an explanation of why this is the case). We denote the formula size by $m \geq n$ and its width by w, and for our application we can assume wlog that $w \leq O(\log(m))$ and we fix the error probability to be $1/\mathrm{poly}(m)$ for a sufficiently large polynomial.

For such parameters, Trevisan and Xue [106] constructed a pseudorandom restriction algorithm with seed length $\tilde{O}(w) \cdot \log^2(m)$, and Goldreich and Wigderson [44] constructed such an algorithm with seed length $\tilde{O}(2^w) \cdot \log(m)$. The following result from [101] improved on both these results by constructing a pseudorandom restriction algorithm with seed length $\tilde{O}(w^2 \cdot \log(m))$:[2]

Proposition B.1 (width-dependent derandomization of Håstad's switching lemma; see [101, Theorem 1.4])**.** Let $m, n \in \mathbb{N}$, let $w \leq O(\log(m))$, and let $\delta = \delta(n) > 0$. Then, there exists an algorithm that gets as input a random seed of length $\tilde{O}(w^2 \cdot \log(mn/\delta))$, runs in time $\mathrm{poly}(n)$, and outputs a restriction $\rho \in \{0, 1, \star\}^n$ such that for every n-bit depth-2 formula F of size m and width w, with probability $1 - O(\delta)$ the following holds:

1. The number of variables kept alive by ρ is $\Omega(n/w)$.
2. There exist "lower-sandwiching" and "upper-sandwiching" formulas $F^{\mathtt{low}}$ and $F^{\mathtt{high}}$ for F[3] such that both $F^{\mathtt{low}}{\restriction_\rho}$ and $F^{\mathtt{high}}{\restriction_\rho}$ can be computed by decision trees of depth $O(\log(1/\delta))$, and each of the two formulas agrees with $F{\restriction_\rho}$ on $1 - \delta$ of the inputs.

[2] Strictly speaking, the result of [44] is still better in the case of $w = O(1)$, since it yields seed length $O(\log(n))$ rather than $\tilde{O}(\log(n))$.

[3] That is, for every $x \in \{0,1\}^n$ it holds that $F^{\mathtt{low}}(x) \leq F(x) \leq F^{\mathtt{high}}(x)$.

Proposition B.1 is the main technical result underlying the algorithm for QD_B in Theorem 3.6. Observe that, crucially, both $F^{\mathtt{low}}$ and $F^{\mathtt{high}}$ agree with F on $1-\delta$ of the inputs *in the subcube that corresponds to* ρ; that is, they approximate F *after* the restriction. Also, we can take δ to be an arbitrarily large polynomial in m without noticeably affecting the seed length.

B.2 Pseudorandom restrictions for threshold circuits

For constant-depth linear threshold circuits (LTF circuits), even *random* restriction procedures (let alone pseudorandom procedures) are relatively new. Impagliazzo, Paturi, and Saks [57] showed a random restriction procedure in which neither the fixed variables nor their values are chosen uniformly; this procedure sufficed to show worst-case lower bounds, but does not suffice for many applications, such as proving average-case lower bounds or constructing quantified derandomization algorithms.

Several years ago Chen, Santhanam, and Srinivasan [26] (relying on results developed in [30], [88] and other works) showed a random restriction procedure for LTF circuits in which the variables are chosen in an adaptive way that depends on the given circuit, but values for fixed variables are chosen uniformly; they used this procedure to deduce average-case lower bounds for LTF circuits. This restriction procedure was subsequently derandomized and refined in [100], yielding the following result, which is the main technical result underlying Theorem 3.7:

Proposition B.2 (pseudorandom restrictions for LTF circuits; see [100, Proposition 3.1])**.** Let $c, d \geq 1$, let $\epsilon > 0$ be a sufficiently small constant, and let $\delta = d \cdot 30^{d-1} \cdot \epsilon$. Then, there exists a polynomial-time algorithm that for every $n \in \mathbb{N}$, when given as input an LTF circuit over n input bits of depth d with at most $n^{1+\epsilon}$ wires, and a random seed of length $O(\log(n) \cdot \mathrm{loglog}(n)^{)}$, with probability at least $1 - n^{-\epsilon/2}$ outputs the following:

1. A restriction ρ that keeps at least $n^{1-\delta}$ variables alive.
2. An LTF that is $(1 - n^{-c})$-close to $C\restriction_{\rho}$.

Note that the original statement in [100] only asserts that Φ is $(9/10)$-close to $C\restriction_\rho$, but the proof already shows that the closeness is $1-n^{-c}$ for every desired constant $c \in \mathbb{N}$. (To see this, note that in Claim 5.11.1 of the full version, the bound on the closeness of each biased gate to the corresponding constant after all the restriction is stated to be $\delta_t = 1 - n^{-c}$ for an arbitrary constant $c \in \mathbb{N}$.)

Let me also note that another pseudorandom restriction procedure for LTF circuits was very recently shown by Hatami *et al.* [51]. In this procedure the failure probaiblity is $\exp(-n^{\Omega(1)})$ instead of $n^{-\Omega(1)}$, but only the variables to be fixed are chosen pseudorandomly, whereas values for fixed variables are chosen uniformly.

Kabanets and Lu [60] showed a result analogous to Proposition B.2 that holds for the stronger class of PTF circuits of low degree; this is the main technical result underlying the algorithm for QD_B of PTF circuits in Theorem 7.12. They also showed a similar result for PTF circuits in which each gate computes a sparse polynomial (i.e., a polynomial with n^Δ monomials for a small constant Δ).

Proposition B.3 ($\mathsf{pseudorandom\ restrictions\ for\ low\text{-}degree\ PTF\ circuits}$; see [60, Proof of Theorem 4.4])**.** Let $c, d \geq 1$, let $E \geq 11$, and let $\Delta\colon \mathbb{N} \to \mathbb{N}$ such that $\Delta \ll \sqrt{\epsilon_d \cdot \log(n)/\mathrm{loglog}(n)}$, where $\epsilon_d = E^{-2(d-1)}$. Let $\mathcal{C}_n$ be the class of PTF circuits over n input bits of depth d with $n^{1+\epsilon_d}$ wires in which each gate computes a PTF with degree at most $\Delta(n)$. Then, there exists an algorithm that gets as input $C \in \mathcal{C}_n$ and a random seed of length $\log(n)^{O(\Delta(n)^2)}$, and with probability at least $1 - n^{\Omega(1)}$ outputs the following:

1. A restriction ρ that keeps at least $n^{1-6/E}$ variables alive.
2. A PTF with at most $n^{\epsilon_d \cdot \Delta(n)}$ monomials that is $(1-n^{-c})$-close to $C\restriction_\rho$.

Proposition B.3 is not explicitly stated in [60] (which is a conference version), but as explained there after the statement of Theorem 4.7, this result follows immediately by mimicking the proof of Theorem 4.4 (which is an analogous result for PTF circuits in which each gate computes a sparse polynomial). Also, similarly to Proposition B.2, in [60]

the closeness parameter is taken to be $9/10$ rather than $1 - n^{-c}$, but the latter value is immediate from their proof. (To see this, in the proof of Theorem 4.4, instantiate Lemma 4.5 with an arbitrarily large constant $c \geq 1$ instead of with $c = 1$.)

The restriction procedures are non-black-box. The algorithms in Propositions B.2 and B.3 both work in a non-black-box fashion: They get as input a circuit C, and tailor a restriction that is specifically designed to simplify C. However, as mentioned in Section 8, a key component in these procedures is already "somewhat black-box" (i.e., going layer-by-layer, these restrictions are pseudorandom distributions that simplify each of the gates in the layer with high marginal probability). Moreover, both procedures can be made fully black-box at the expense of simplifying the circuit not to a single LTF or PTF, but rather to the more complicated model of a relatively shallow decision tree with LTFs or PTFs at its leaves; see [51] for an explanation.

B.3 Pseudorandom restrictions for formulas

Random restrictions for De Morgan formulas have been extensively studied since the 1960's, focusing on the well-known implication that a formula is expected to shrink (in size) under such restrictions (see, e.g., [48], [56], [81], [94], [96]). However, only in the last decade have *pseudorandom versions* been constructed.

Impagliazzo, Meka, and Zuckerman [55] constructed a pseudorandom restriction procedure that shrinks any formula of size s to be of size $p^2 \cdot s^{1+o(1)}$, with probability $1 - n^{-O(1)}$; this procedure has seed length $2^{O(\log^{2/3}(s))} = s^{o(1)}$. Hatami *et al.* [51] showed a pseudorandom retriction procedure that supports a much smaller failure probability $\epsilon \ll s^{-O(1)}$, but shrinks any formula to a decision tree of depth $s^{o(1)} \cdot \mathrm{polylog}(1/\epsilon)$ with formulas of size $p^{2-o(1)} \cdot s$ at its leaves; the seed length for this procedure is $s^{o(1)} \cdot \mathrm{polylog}(n/\epsilon)$.

For quantified derandomization we do not need the strong concentration bounds above on the shrinkage of the formula, and shrinkage in expectation suffices. For this application, Chen, Jin, and Williams [19]

showed a procedure that uses seed length only $O(\log(n))$ and indeed obtains shrinkage in expectation:

Proposition B.4 (pseudorandom restrictions for formulas; see [19])**.** Let $p\colon \mathbb{N} \to \mathbb{N}$. Then, there exists an algorithm that gets as input a random seed of length $O(\log(n))$, runs in time $\mathrm{poly}(n)$, and outputs a restriction $\rho \in \{0,1,\star\}^n$ such that:

1. With probability at least $2/3$ it holds that ρ keeps at least $pn/2$ variables alive.
2. For every n-variable formula it holds that

$$\mathbb{E}[L(F\restriction_\rho)] \leq \left(p^2 \cdot L(F) + p \cdot \sqrt{L(F)}\right) \cdot n^{c/\mathrm{loglog}(n)} \,,$$

   where $c > 1$ is a universal constant.

Proposition B.4 is the main technical result underlying the algorithm for QD_B of formulas in Theorem 3.8. In addition, the pseudorandom restriction in [19] is even stronger, since it guarantees the existence of a circuit C of size $\mathrm{polylog}(n)$ that gets as input the random seed (of length $O(\log(n))$) and an index $i \in [n]$ of an output, an prints the i^{th} coordinate of the restriction ρ.

C

Extractors Computable by Low-depth Circuits and Formulas

In this section I describe the technical results underlying the reductions of CAPP to QD_B that were presented in Section 7. These technical results are constructions of extractors that are computable in *weak circuit classes*. The precise notion of being computable in a weak circuit class will differ across the constructions presented below, but in general it will be at least as strict as the one in Definition 8.2 (and hence the limitation in Theorem 8.1 applies to the results that use these constructions).

In general, there are very efficient constructions of extractors with good parameters: For example, each output bit of Trevisan's [105] extractor (and of its improvement in [84]) is just a parity of the input. However, in the following results we will be interested in computing extractors by circuits or formulas *that are too weak to even compute the parity* of their input.

C.1 Extractors computable by $\mathcal{AC}^0$ circuits

Goldreich and Wigderson [44, Theorem 3.4 in the full version] constructed an $\mathcal{AC}^0$ circuit computing a function that can be thought of as a middle-point between a standard extractor (which outputs a

distribution close to uniform) and a non-black-box extractor as referred to in Section 8.3 (which outputs a distribution that only looks uniform to a circuit whose description is given to the non-black-box extractor). Specifically, the output distribution of their function looks uniform to any $\mathcal{AC}^0$ observer; this is equivalent to a sampler that only samples correctly subsets that are decidable by $\mathcal{AC}^0$ circuits. Their function was computable by $\mathcal{P}$-uniform $\mathcal{AC}^0$ circuits, had $n_0 = n^{\Omega(1)}$ output bits, and supported min-entropy $k = 2^{n/\mathrm{polylog}(n)}$.

Their construction was later superseded by a construction of standard extractors that are computable by $\mathcal{P}$-uniform $\mathcal{AC}^0$ circuits, which was shown by Cheng and Li [27]. (That is, the construction of [27] is of a standard extractor rather than of a non-black-box one, and also has better parameters than the one in [44].) In fact, there are various different such constructions in [27], supporting different trade-offs between the parameters; let me mention one such construction of theirs:

Proposition C.1 (extractors in uniform $\mathcal{AC}^0$; see [27, Theorem 4.11])**.** For any $d \geq 7$ there exists an extractor family $\{\mathsf{Ext}_n \colon \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^{n_0}\}_{n\in\mathbb{N}}$ with seed length $\ell = O(\log(n))$, output length $n_0 = \left\lfloor n^{1/3600} \right\rfloor$, min-entropy $k = \Theta(n/\log^{d-7}(n))$, and error $n^{-1/600}$, such that the function mapping $(z,s) \in \{0,1\}^n \times \{0,1\}^\ell$ to $\mathsf{Ext}_n(z,s)$ is computable by $\mathcal{P}$-uniform $\mathcal{AC}^0$ circuits of depth d and size $\mathrm{poly}(n)$.

The parameters of Proposition C.1 are close to the best possible (and various optimizations and tradeoffs appear in [27]). This follows from a lower bound of Viola [109] (see also [43]), which asserts that $\mathcal{AC}^0$ circuits of size $\mathrm{poly}(n)$ and depth d can compute extractors for min-entropy at most $k = n/\log^{d-1}(n)$, even if the seed is very long compared to the output length (i.e., even if the seed is of length n_0^{999}). A similar lower bound follows by combining Theorem 8.1 with Håstad's switching lemma [47]. (In fact, Theorem 8.1 yields a more general approach for showing such lower bounds, since the simplifier set need not be a subcube and may even partially depend on the circuit that it simplifies (as explained in Section 8).)

C.2 Extractors computable by extremely sparse threshold circuits

Recall that the parity function can be computed by LTF circuits of depth d and size $n^{1+c_\oplus^{-d}}$, for some constant $c_\oplus \geq \frac{1+\sqrt{5}}{2}$ (see [9], [82]). Thus, if we instantiate Trevisan's [105] extractor Ext with seed length close to $\log(n)$ and output length n^ϵ for a small constant $\epsilon > 0$, we can compute the mapping $z \mapsto \{\mathsf{Ext}(z,s)\}_s$ by a uniform $\mathcal{TC}^0$ circuit of super-quadratic size. (This is since this extractor only computes parities of the input, and since for these parameters the circuit that prints the outputs of the extractor on all seeds has $n^{1+O(\epsilon)}$ output bits.)

As far as I know, the first extractor that is computable by uniform $\mathcal{TC}^0$ circuits of *super-linear* size was constructed in [100]; each output bit of this extractor is still a parity of the input, but these parities are computed "in a batch" rather than paying $n^{1+c_\oplus^{-d}}$ per each output bit. This construction was later improved by Chen and the current author [22], who showed a construction with seed length and output length as above that uses only $n^{1+c^{-d}}$ wires, for any $c < c_\oplus$; that is:

Proposition C.2 (extractors in uniform $\mathcal{TC}^0$ of super-linear size)**.** For any $d \geq 7$ and $c < c_\oplus$ there exists an extractor family $\{\mathsf{Ext}_n \colon \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^{n_0}\}_{n\in\mathbb{N}}$ with seed length $\ell = (1+\exp(-d)) \cdot \log(n)$, output length $n_0 = n^{\exp(-d)}$, min-entropy $k = n^{1-\exp(-d)}$, and error $\epsilon > 0$, such that the following holds: The function mapping $z \in \{0,1\}^n$ to the output-set $(\mathsf{Ext}_n(z,s))_{s\in\{0,1\}^\ell}$ is computable by $\mathcal{P}$-uniform $\mathcal{TC}^0$ circuits of depth d and size $n^{1+c^{-d}}$.

Note that the circuits in Proposition C.2 are $\mathcal{TC}^0$ circuits rather than LTF circuits; that is, to compute the extractor we only use unweighted majority gates rather than (the stronger) linear threshold functions.

C.3 Dispersers computable by formulas of subquadratic size

Recall that the parity function can be computed by formulas of size $O(n^2)$. Thus, a naive implmentation of Trevisan's extractor with seed length close to $\log(n)$ and output length n^ϵ for a small constant $\epsilon > 0$ yields formulas of size $O(n^{3+O(\epsilon)})$.

The reduction of CAPP to QD_B by Chen, Jin, and Williams [19] yields formulas of *sub-quadratic* size, using two ideas. The first idea is to combine a standard linear extractor with naive error reduction; the addition of naive error reduction yields slightly poorer extraction properties, but also reduces the computational complexity (intuiviely, since naive error reduction has very low complexity but poor extractor properties). In particular, the combination yields the following construction:

Proposition C.3 (dispersers computable by uniform sub-quadratic formulas)**.** For any $\epsilon \in (0,1)$ and $\delta > 0$ there exists a family of functions $\widehat{\mathsf{Disp}}_n \colon \{0,1\}^n \times \{0,1\}^{O(\log(n))} \to \{0,1\}^{n_0}$, where $n_0 = n^{\Omega_{\delta,\epsilon}(1)}$, that satisfies the following:

1. **Seeds are pairs.** The seed of $\widehat{\mathsf{Disp}}$ is a pair $(s,i) \in \{0,1\}^{O(\log(n))} \times \{0,1\}^{\epsilon \cdot \log(n)}$.
2. **Computable by formulas of sub-quadratic size:** For each fixed $s \in \{0,1\}^{O(\log(n))}$, the mapping of $x \in \{0,1\}^n$ to the tuple $(\widehat{\mathsf{Disp}}_n(x,(s,i)))_{i \in \{0,1\}^{\epsilon \cdot \log(n)}}$ is computable by $\mathcal{P}$-uniform formulas of size $n^{2-\epsilon+\delta}$.
3. **Disperser with density $\Omega(n^{-\epsilon})$:** For every $T \subseteq \{0,1\}^{n_0}$ such that $|T|/2^{n_0} \geq 9/10$, for all but at most 2^{n^ϵ} inputs $x \in \{0,1\}^n$ there exists $i \in \{0,1\}^{\epsilon \cdot \log(n)}$ such that $\Pr_s \left[\widehat{\mathsf{Disp}}(x,(s,i)) \in T\right] \geq 2/3$.

Proof. For two constants $\alpha > 0$ and $\beta < 1$ that will be defined below, and for $n_1 = n^\beta$, let $\mathsf{Ext} \colon \{0,1\}^{n_1} \times \{0,1\}^{O(\log(n_1))} \to \{0,1\}^{n_0}$ be the extractor that is implicit in the work of Li [69, Theorem 3.14] and was explicitly stated in [19, Theorem 4.1], where $n_0 = n_1^{\alpha/2}$; the min-entropy of Ext is n_1^α, its error is $n_1^{-\alpha}$, and it can be computed by $\mathcal{P}$-uniform formulas of size $n_1^{2+\alpha}$. We think of any n-bit string x as a sequence of $r = n/n_1$ disjoint substrings $x_1, ..., x_r$ of length n_1, and define $\widehat{\mathsf{Disp}}(x,(s,i)) = \mathsf{Ext}(x_i, s)$; that is, the random seed of $\widehat{\mathsf{Disp}}$ consists of an index $i \in [r]$ and of a seed s for Ext, and $\widehat{\mathsf{Disp}}$ applies Ext with seed s to the i^{th} substring of n_1 bits in its input x.

The seed length of $\widehat{\mathsf{Disp}}$ is $(1-\beta) \cdot \log(n) + O(\log(n))$, and its output length is $n_0 = n^{\beta \cdot \alpha/2}$. Also, for each fixed s, the mapping $x \mapsto$

$(\widehat{\mathsf{Disp}}_n(x,(s,i)))_{i\in[r]}$ is computable by $\mathcal{P}$-uniform formulas of size $r\cdot n_1^{2+\alpha}$. Now, let $T\subseteq\{0,1\}^{n_0}$ be of density at least $9/10$. For every fixed $i\in[r]$ there exist at most $2^{n_1^\alpha}$ strings $x_i\in\{0,1\}^{n_1}$ such that $\Pr[\mathsf{Ext}(x_i,\mathbf{s})\in T]<9/10-n^{-\alpha}$. Thus, the number of strings $x=(x_1,...,x_r)$ such that for all $i\in[r]$ it holds that $\Pr[\mathsf{Ext}(x_i,\mathbf{s})\in T]<9/10-n^{-\alpha}$ is at most $2^{n_1^\alpha\cdot r}$. Hence, for all but at most $2^{n_1^\alpha\cdot r}$ of the strings $x\in\{0,1\}^n$ there exists $i\in[r]$ such that $\Pr[\widehat{\mathsf{Disp}}(x,(\mathbf{s},i))\in T]=\Pr[\mathsf{Ext}(x_i,\mathbf{s})\in T]\geq 9/10-o(1)>2/3$.

To conclude we need to choose $\alpha>0$ and $\beta<1$ such that $n_1^\alpha\cdot r\leq n^\epsilon$ (for the number of exceptional inputs) and $r\cdot n_1^{2+\alpha}\leq n^{2-\epsilon+\delta}$ (for the size bound) and $(1-\beta)\cdot\log(n)<\epsilon\cdot\log(n)$ (for the seed length). Choosing $\beta=\frac{1-\epsilon}{1-\alpha}$ and a sufficiently small $\alpha=\alpha_{\epsilon,\delta}>0$ suffices. ∎

The second idea of [19] is that in their reduction, instead of the standard approach of reducing CAPP of a formula F to QD_B for $F'(x)=\bigvee_{s,i}F(\widehat{\mathsf{Disp}}(x,(s,i)))$, they reduce CAPP of F to QD_B for a *probabilistic formula*, defined as follows:

$$\mathbf{F}(x)=\bigvee_{i\in[r]}F(\widehat{\mathsf{Disp}}(x,(\mathbf{s},i)))\ ,$$

where $\mathbf{s}$ (i.e., the first part of the seed) is the only random choice made by the probabilistic formula $\mathbf{F}$. By Proposition C.3, each formula in the support of $\mathbf{F}$ is of size $n^{2-\epsilon+\delta}$, and if F accepts at least $9/10$ of its inputs, then for all but 2^{n^ϵ} of the inputs x for $\mathbf{F}$ it holds that $\Pr[\mathbf{F}(x)=1]\geq 2/3$.

The limitation in Theorem 8.1 still applies to this construction. The limitation in Theorem 8.1 is proved under the hypothesis that the distribution of simplifier sets simplifies every circuit in the class (in the current setting this will refer to every formula of bounded size) with probability at least $1/2$. This hypothesis suffices to deduce a limitation on extractor-based construction. In the setting of formulas the known distribution of simplifier sets has a considerably higher success probability (i.e., $1-n^{-O(1)}$ instead of $1/2$), and thus its existence suffices to deduce a limitation also on disperser-based constructions as in Proposition C.3.

In particular, the following claim asserts that a disperser construction as in Proposition C.3 cannot be computed by formulas of size $n^{2-2\epsilon+o(1)}$ (as in Corollary 7.16) instead of $n^{2-\epsilon+\delta}$. The claim even rules out a weaker disperser construction, in which we do not have a density guarantee (as in Item (3)) and in which only require the disperser to be computable by formulas of the corresponding size on each fixed seed (rather than requiring a batch-computation property as in Item (2)).

Claim C.1. For any $\epsilon > 0$, there does not exist an $(n^{\epsilon}, .01)$-disperser $\mathsf{Disp}\colon \{0,1\}^n \times \{0,1\}^{O(\log(n))} \to \{0,1\}^{n_0}$, where $n_0 = n^{\Omega(1)}$, such that for every fixed $s \in \{0,1\}^{O(\log(n))}$ it holds that $\mathsf{Disp}^{(s)}(x) = \mathsf{Disp}(x,s)$ is computable by a formula of size $n^{2-2\epsilon+o(1)}$.

Proof. Assume towards a contradiction that such construction exists, and let $\varphi = \varphi(\epsilon) > 0$ be a sufficiently small constant. For $p = n^{-1+\epsilon+\varphi}$, let $\mathbf{X}$ be a distribution over subcubes $X \subset \{0,1\}^n$ of size at least $2^{p \cdot n/2} = 2^{n^{\epsilon+\varphi}/2}$ that shrinks every formula of size S to be of size $p^2 \cdot S^{1+o(1)}$, with probability at least $1 - S^{-c}$ for an arbitrarily large constant $c > 1$ (see [55, Lemma 4.8]).[1]

Let $\mathcal{F} = \left\{\mathsf{Disp}^{(s)}\right\}_{s \in \{0,1\}^{O(\log(n))}}$. Note that there are $\mathrm{poly}(n)$ functions in $\mathcal{F}$, and each function has $n_0 = n^{\Omega(1)}$ output bits. Taking the constant $c > 1$ in the error bound above to be sufficiently large, there exists $X \sim \mathbf{X}$ such that the formula size of every function $\mathsf{Disp}^{(s)} \in \mathcal{F}$ decreases by a factor of $p^2 \cdot n^{o(1)}$; in particular, each $\mathsf{Disp}^{(s)}$ is computable by a formula of size $p^2 \cdot n^{2-2\epsilon+o(1)} = n^{2\varphi+o(1)}$.[2]

[1] The subsets in the support of the distribution from [55] are of size $p \cdot n/2$ only with very high probability (rather than always). I ignore this issue for simplicity, as we can always modify the distribution such that it is supported only on subsets of sufficiently large size $p \cdot n/2$, while preserving the property that each size-S formula is simplified with probability at least $1 - S^{-c}$.

[2] To elaborate, each $\mathsf{Disp}^{(s)}$ is a multi-output function computable by a collection of n_0 formulas. Let $\mathcal{S}$ be the sub-collection of formulas of size less than n^{φ}/n_0, and let $\mathcal{L}$ be the sub-collection of formulas of size at least n^{φ}/n_0. For each $F \in \mathcal{L}$, with probability $1 - 1/\mathrm{poly}(n)$ its size decreased by a multplicative factor of $p^2 \cdot n^{o(1)}$; and the total contribution to size of the formulas in $\mathcal{S}$ is at most n^{φ}. Thus, with probability $1 - 1/\mathrm{poly}(n)$ the size of $\mathsf{Disp}^{(s)}$ after the restriction is at most $p^2 \cdot S^{1+o(1)} + n^{\varphi} \leq n^{2\varphi+o(1)}$.

It follows that on the subset X, each function $\mathsf{Disp}^{(s)} \in \mathcal{F}$ is sensitive to less than $n^{2\varphi+o(1)}$ input bits. Hence, the support size of Disp when given inputs from X satisfies

$$\left| \bigcup_{x\in X, s\in\{0,1\}^{O(\log(n))}} \mathsf{Disp}(x,s) \right| \leq \mathrm{poly}(n) \cdot 2^{n^{2\varphi+o(1)}} \leq 2^{n^{2\varphi+o(1)}} .$$

Taking φ to be sufficiently small such that $n^{2\varphi} < \sqrt{n_0}$, there exists a set $T \subseteq \{0,1\}^{n_0}$ of size more than $2^{n_0} - 2^{\sqrt{n_0}} = (1-o(1)) \cdot 2^{n_0}$ that avoids Disp on a set $X \subseteq \{0,1\}^n$ of size $2^{n^{\epsilon+\Omega(1)}}$, a contradiction to the hypothesized properties of Disp. ■

D

Quantified Derandomization of Logspace and of Proof Systems

In this appendix I mention two interesting directions that were raised in the original work of Goldreich and Wigderson [44] but have not been explored further so far.

D.1 Quantified derandomization of logspace

Can we simulate probabilisitic logspace machine in deterministic logspace if the number of exceptional random strings is extremely small? As reported in [44], Mike Saks showed in the 1990s that this is indeed possible, even when the number of exceptional random strings is relatively not that small:

Theorem D.1 (quantified derandomization of logspace; attributed to Saks [44, Appendix A of the Full Version])**.** Let $L \subseteq \{0,1\}^*$ be decidable by a probabilistic logspace machine M such that for some constant $\epsilon > 0$, on n-bit inputs M uses $T = T(n)$ bits of randomness and errs on at most $B(T) = 2^{(1-\epsilon)\cdot T}$ random choices. Then, $L \in \mathcal{L}$.

The number $B(T) = 2^{(1-\Omega(1))\cdot T}$ of exceptional random strings in Theorem D.1 matches the non-uniform derandomization in Theorem 3.1, and is indeed significantly larger than in all other settings in this survey

(i.e., in all other settings the number of exceptional random strings was $B(T) = 2^{o(T)}$).

Saks' original quantified derandomization algorithm was non-black-box: Given as input a description of a polynomial-sized read-once branching program (ROBP), the algorithm relies on the description to find its most likely output. (Recall that the ROBP represents the computation of a probabilistic logspace machine on a fixed input as a function of the random coins.) William Hoza [52] strengthened this result by constructing a *black-box* algorithm (i.e., a PRG for biased ROBPs) that yields the same parameters; the proof below presents Hoza's construction.

Proof of Theorem D.1 by William Hoza. For any $\epsilon = \epsilon(n) > 0$ and any $B(n) \leq \epsilon \cdot 2^n$, we construct an ϵ-PRG for of B-biased ROBPs over n input bits of w, whose seed length is $\ell = \ell(n) = \frac{n}{n - \log(B)} \cdot \log(2nw/\epsilon)$. Given seed $s \in \{0,1\}^\ell$, the PRG simply outputs the n-bit string $(s, s, s, ..., s) \in (\{0,1\}^\ell)^{n/\ell}$ (for simplicity we assume that n/ℓ is an integer). Note that this PRG is indeed computable in logspace, and that for $B(n) = 2^{(1-\Omega(1))\cdot n}$ its seed length satisfies $\ell(n) = O(\log(nw/\epsilon))$.

To see that this construction works, fix an ROBP as above, and let $\sigma \in \{0,1\}$ be its less likely output. Index the layers of the ROBP by $0, ..., n$ where 0 is the layer of the starting vertex and n is the last layer, and consider the vertices at layers indexed $0, \ell, ..., i \cdot \ell, ..., n$. For each such vertex v, denote by p_v the probability that a random walk starting from v reaches a vertex in the last layer labeled with σ, and for $s \in \{0,1\}^\ell$ denote by $v(s)$ the vertex reached when starting from v and walking according to s. (For vertices v in the last layer we will only care about p_v, which is either 0 or 1.)

Note that $p_v = \mathbb{E}_{s \in \{0,1\}^\ell}\left[p_{v(s)}\right]$, and hence (by Markov's inequality)

$$\Pr_s\left[p_{v(s)} \geq p_v \cdot (2nw/\epsilon)\right] \leq \epsilon/(2nw) \ .$$

By a union-bound over the $(n+1) \cdot w/\ell < 2nw$ vertices in the relevant layers, with probability more than $1 - \epsilon$ over choice of $s \in \{0,1\}^\ell$, for every vertex in these layers we have that $p_{v(s)} < p_v \cdot (2nw/\epsilon)$. In this case, when starting from the initial vertex v_0 in the ROBP and walking

according to the n-bit string $(s, s, s, ..., s)$ we pass through vertices $v_1, v_\ell, ...$ and reach a vertex v_n, and by induction for each $i \in [n/\ell]$ we have

$$p_{v_i} = p_{v_{i-1}(s)} < p_{v_{i-1}} \cdot (2nw/\epsilon) < ... < p_{v_1} \cdot (2nw/\epsilon)^i \ .$$

In particular, applying the above for $i = n$ and recalling that $p_{v_1} \leq B/2^n$, we have that $p_{v_n} < p_{v_1} \cdot (2nw/\epsilon)^{n/\ell} \leq B/2^n \cdot (2nw/\epsilon)^{n/\ell} < 1$, where the last inequality relied on our choice of ℓ. Hence, v_n is labeled with the more likely output $\neg\sigma$ of the ROBP.

The above proves that with probability at least $1 - \epsilon$ over choice of seed s for the PRG, the ROBP evaluates to its more likely output. The pseudorandomness of this PRG follows because the probability over a uniform input that the ROBP evaluates to its more likely output is at least $1 - B/2^n \geq 1 - \epsilon$. ■

The proof above (as well as Saks' original proof) is elementary, and does not rely on the vast literature concerning derandomization of logspace (or of ROBP). Nevertheless, improving on the result that it yields is still an open problem:

Open Problem 9: Quantified derandomization of logspace with $B(T) = 2^{(1-o(1))\cdot T}$. *Strengthen Theorem D.1 to work with $B(T) = 2^{T-s(T)}$ for some sub-linear function s, or show that such an improvement implies that $\mathcal{BPL} = \mathcal{L}$.*

D.2 Quantified derandomization of Merlin-Arthur protocols

We are interested in derandomizing Merlin-Arthur protocols, and particularly in derandomizing $\mathcal{MA}$ and $\mathcal{AM}$ (see, e.g., [7, Section 8.2] for the standard definitions of these classes). Recall that assuming sufficiently strong lower bounds, both of these classes can be derandomized and equal $\mathcal{NP}$ (see [64]).

Goldreich and Wigderson asked if derandomizing $\mathcal{MA}$ or $\mathcal{AM}$ becomes easier when the verifier is extremely unlikely to err (i.e., to accept an incorrect proof or to reject a correct proof). They showed two complementary results, the first of which is the following quantified derandomization algorithm for a subclass of $\mathcal{MA}$.

Definition D.1 ($\mathcal{MA}$ with restricted verifiers)**.** For a circuit class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$, we say that $L \subseteq \{0,1\}^*$ can be decided by an $\mathcal{MA}$ protocol with $\mathcal{C}$-verifiers if there exists an $\mathcal{MA}$ verifier V that decides L such that the following holds: For every input $x \in \{0,1\}^*$ and proof $w \in \{0,1\}^{\mathrm{poly}(|x|)}$, the decision of V at x with proof w as a function of the $m = \mathrm{poly}(n)$ random coins can be computed by a circuit in $\mathcal{C}_m$.

Theorem D.2 (quantified derandomization of $\mathcal{MA}$ with $\mathcal{AC}^0$ verifiers; see [44, Theorem 7.3 in the Full Version])**.** Assume that $L \subseteq \{0,1\}^*$ can be decided by an $\mathcal{MA}$ protocol with $\mathcal{AC}^0$ verifiers such that the verifier always errs on at most $B(T) = 2^{T^{1-\epsilon}}$ random choices. Then $L \in \mathcal{NP}$.

Theorem D.2 may appear weak, because it only refers to $\mathcal{MA}$ verifiers whose decision as a function of the random coins is an $\mathcal{AC}^0$ circuit. However, if an analogous result holds for $\mathcal{AM}$ verifiers, then $\mathcal{AM} = \mathcal{NP}$! In fact, this conclusion holds even if the verifier's decision is only a CNF, and even for smaller values of $B(T) = 2^{T^{\epsilon}}$.

Definition D.2 ($\mathcal{AM}$ with restricted verifiers)**.** For a circuit class $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$, we say that $L \subseteq \{0,1\}^*$ can be decided by an $\mathcal{AM}$ protocol with $\mathcal{C}$-verifiers if there exists a deterministic procedure V and a polynomial $p\colon \mathbb{N} \to \mathbb{N}$ such that the following holds:

- For every $x \in L$ it holds that $\Pr_{r\in\{0,1\}^{p(n)}}[\exists w \in \{0,1\}^{p(n)}, V(x,w,r) = 1] \geq 2/3$.
- For every $x \notin L$ it holds that $\Pr_{r\in\{0,1\}^{p(n)}}[\forall w \in \{0,1\}^{p(n)}, V(x,w,r) = 0] \geq 2/3$.
- On n-bit inputs V can be computed by a circuit from $\mathcal{C}_{n+2p(n)}$.

Theorem D.3 (threshold values for quantified derandomization of $\mathcal{AM}$ with CNF verifiers; see [44, Theorem 7.4 in the Full Version])**.** Assume that for some $\epsilon \in (0,1)$ the following holds: For any $L \subseteq \{0,1\}^*$ that can be decided by an $\mathcal{AM}$ protocol with CNF verifiers such that the verifier always errs on at most $B(T) = 2^{T^{\epsilon}}$ random choices, we have that $L \in \mathcal{NP}$. Then $\mathcal{AM} = \mathcal{NP}$.

To make sense of Theorems D.2 and D.3, recall that derandomization of $\mathcal{AM}$ is in general a harder problem than derandomization of $\mathcal{MA}$

(since $\mathcal{AM} \supseteq \mathcal{MA}$). Nevertheless, the contrast between the two results is still striking.

The proof of Theorem D.2 amounts to applying the quantified derandomization algorithm of Theorem 7.4 to the verifier's residual decision as a function of the random coins, when the input and the proof are fixed. Similar results can be obtained for analogous classes of $\mathcal{MA}$ with restricted verifiers (such as verifiers computable by formulas) using Theorems 7.5, 7.10, 7.12, and 7.15. However, this approach does not use the power of interaction for the quantified derandomization algorithm, but rather only applies a known quantified derandomization algorithm to the verifier's decision.

Open Problem 10: Quantified derandomization of $\mathcal{MA}$ using the power of interaction. *For any class $\mathcal{C}$, let $\mathcal{MA}^{\mathcal{C}}$ be the set of problems solvable by $\mathcal{MA}$ protocols in which the verifier's decision as a function of the random coins is computable in $\mathcal{C}$. Can we construct a quantified derandomization algorithm for $\mathcal{MA}^{\mathcal{C}}$ with better parameters than the known quantified derandomization algorithm for $\mathcal{C}$, using the interaction with the prover?*

References

[1] S. Aaronson, "$P \stackrel{?}{=} NP$," in *Open Problems in Mathematics*, J. F. Nash Jr. and M. T. Rassias, Eds., Springer International Publishing, 2016, pp. 1–122.

[2] E. Abbe, A. Shpilka, and A. Wigderson, "Reed-Muller codes for random erasures and errors," *IEEE Transactions on Information Theory*, vol. 61, no. 10, 2015, pp. 5229–5252.

[3] L. Adleman, "Two theorems on random polynomial time," in *Proc. 19th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1978, pp. 75–83.

[4] M. Ajtai, "Σ_1^1-formulae on finite structures," *Annals of Pure and Applied Logic*, vol. 24, no. 1, 1983, pp. 1–48.

[5] M. Ajtai and A. Wigderson, "Deterministic simulation of probabilistic constant depth circuits," in *Proc. 26th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 1985.

[6] A. E. Andreev, "On a method for obtaining more than quadratic effective lower bounds for the complexity of π-schemes," *Vestnik Moskovskogo Universiteta. Seriya I. Matematika, Mekhanika*, no. 1, 1987, pp. 70–73, 103.

[7] S. Arora and B. Barak, *Computational complexity: A modern approach*. Cambridge University Press, Cambridge, 2009.

[8] B. Barak, R. Shaltiel, and A. Wigderson, "Computational analogues of entropy," in *Proc. 7th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, 2003, pp. 200–215.

[9] P. Beame, E. Brisson, and R. Ladner, "The complexity of computing symmetric functions using threshold circuits," *Theoretical Computer Science*, vol. 100, no. 1, 1992, pp. 253–265.

[10] E. Ben-Sasson and E. Viola, "Short PCPs with projection queries," in *Proc. 41st International Colloquium on Automata, Languages and Programming (ICALP)*, 2014, pp. 163–173.

[11] M. Bläser, M. Hardt, and D. Steurer, "Asymptotically optimal hitting sets against polynomials," in *Proceedings of the 35th International Colloquium on Automata, Languages and Programming, Part I*, ser. Proc. 35th International Colloquium on Automata, Languages and Programming (ICALP), pp. 345–356, 2008.

[12] M. Bläser and A. Pandey, "Polynomial identity testing for low degree polynomials with optimal randomness," in *Proc. 24th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, Art. No. 8, 13, 2020.

[13] M. Blum and S. Micali, "How to generate cryptographically strong sequences of pseudo-random bits," *SIAM Journal of Computing*, vol. 13, no. 4, 1984, pp. 850–864.

[14] A. Bogdanov, "Pseudorandom generators for low degree polynomials," in *Proc. 37th Annual ACM Symposium on Theory of Computing (STOC)*, 2005, pp. 21–30.

[15] A. Bogdanov, "Small bias requires large formulas," in *Proc. 45th International Colloquium on Automata, Languages and Programming (ICALP)*, 22:1–22:12, 2018.

[16] A. Bogdanov and E. Viola, "Pseudorandom bits for polynomials," *SIAM Journal of Computing*, vol. 39, no. 6, 2010, pp. 2464–2486.

[17] N. H. Bshouty, "Testers and their applications [extended abstract]," in *Proc. 5th Conference on Innovations in Theoretical Computer Science (ITCS)*, pp. 327–351, 2014.

[18] L. Chen, "Non-deterministic quasi-polynomial time is average-case hard for ACC circuits," in *Proc. 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2019.

[19] L. Chen, C. Jin, and R. R. Williams, "Sharp threshold results for computational complexity," in *Proc. 52nd Annual ACM Symposium on Theory of Computing (STOC)*, pp. 1335–1348, 2020.

[20] L. Chen, X. Lyu, and R. R. Williams, "Almost-everywhere circuit lower bounds from non-trivial derandomization," in *Proc. 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020.

[21] L. Chen and H. Ren, "Strong average-case lower bounds from non-trivial derandomization," in *Proc. 52th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 1327–1334, 2020.

[22] L. Chen and R. Tell, "Bootstrapping results for threshold circuits "just beyond" known lower bounds," in *Proc. 51st Annual ACM Symposium on Theory of Computing (STOC)*, pp. 34–41, 2019.

[23] L. Chen and R. Tell, "Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise," *Electronic Colloquium on Computational Complexity: ECCC*, vol. 28, 2021, p. 080.

[24] L. Chen and R. Tell, "Simple and fast derandomization from very hard functions: Eliminating randomness at almost no cost," in *Proc. 53st Annual ACM Symposium on Theory of Computing (STOC)*, 2021.

[25] L. Chen and R. R. Williams, "Stronger Connections Between Circuit Analysis and Circuit Lower Bounds, via PCPs of Proximity," in *Proc. 34th Annual IEEE Conference on Computational Complexity (CCC)*, 19:1–19:43, 2019.

[26] R. Chen, R. Santhanam, and S. Srinivasan, "Average-case lower bounds and satisfiability algorithms for small threshold circuits," in *Proc. 31st Annual IEEE Conference on Computational Complexity (CCC)*, 1:1–1:35, 2016.

[27] K. Cheng and X. Li, "Randomness extraction in AC0 and with small locality," *Electronic Colloquium on Computational Complexity: ECCC*, vol. 23, 2016, p. 18.

[28] A. Cohen and A. Wigderson, "Dispersers, deterministic amplification, and weak random sources," in *Proc. 30th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 14–19, 1989.

[29] G. Cohen and A. Ta-Shma, "Pseudorandom generators for low degree polynomials from algebraic geometry codes," *Electronic Colloquium on Computational Complexity: ECCC*, vol. 20, 2013, p. 155.

[30] I. Diakonikolas, P. Gopalan, R. Jaiswal, R. A. Servedio, and E. Viola, "Bounded independence fools halfspaces," *SIAM Journal of Computing*, vol. 39, no. 8, 2010, pp. 3441–3462.

[31] I. Dinur and O. Meir, "Toward the KRW composition conjecture: Cubic formula lower bounds via communication complexity," *Computational Complexity*, vol. 27, no. 3, 2018, pp. 375–462.

[32] D. Doron, D. Moshkovitz, J. Oh, and D. Zuckerman, "Nearly optimal pseudorandomness from hardness," in *Proc. 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020.

[33] D. Doron, A. Ta-Shma, and R. Tell, "On hitting-set generators for polynomials that vanish rarely," in *Proc. 24th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, Art. 7–22, 2020.

[34] Z. Dvir, S. Kopparty, S. Saraf, and M. Sudan, "Extensions to the method of multiplicities, with applications to Kakeya sets and mergers," *SIAM Journal of Computing*, vol. 42, no. 6, 2013, pp. 2305–2328.

[35] B. Fefferman, R. Shaltiel, C. Umans, and E. Viola, "On beating the hybrid argument," *Theory of Computing*, vol. 9, 2013, pp. 809–843.

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

[37] A. Gál, A. Tal, and A. T. Nuñez, "Cubic formula size lower bounds based on compositions with majority," in *Proc. 10th Conference on Innovations in Theoretical Computer Science (ITCS)*, Art. No. 35, 13, 2019.

[38] J. T. Gill III, "Computational complexity of probabilistic Turing machines," in *Proc. 6th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 91–95, 1974.

[39] M. Goldmann, J. Håstad, and A. Razborov, "Majority gates vs. general weighted threshold gates," in *Proc. 7th Annual Structure in Complexity Theory Conference*, pp. 2–13, 1992.

[40] M. Goldmann and M. Karpinski, "Simulating threshold circuits by majority circuits," *SIAM Journal of Computing*, vol. 27, no. 1, 1998, pp. 230–246.

[41] O. Goldreich, *Computational Complexity: A Conceptual Perspective*. New York, NY, USA: Cambridge University Press, 2008.

[42] O. Goldreich, "In a world of P=BPP," in *Studies in Complexity and Cryptography. Miscellanea on the Interplay Randomness and Computation*, 2011, pp. 191–232.

[43] O. Goldreich, E. Viola, and A. Wigderson, "On randomness extraction in AC0," in *Proc. 30th Annual IEEE Conference on Computational Complexity (CCC)*, pp. 601–668, 2015.

[44] O. Goldreich and A. Widgerson, "On derandomizing algorithms that err extremely rarely," in *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, 2014, pp. 109–118.

[45] P. Gopalan, R. Meka, and O. Reingold, "Dnf sparsification and a faster deterministic counting algorithm," *Computational Complexity*, vol. 22, no. 2, 2013, pp. 275–310.

[46] V. Guruswami, C. Umans, and S. Vadhan, "Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes," *Journal of the ACM*, vol. 56, no. 4, 2009, Art. 20, 34.

[47] J. Håstad, *Computational Limitations of Small-depth Circuits*. MIT Press, 1987.

[48] J. Håstad, "The shrinkage exponent of De Morgan formulas is 2," *SIAM Journal of Computing*, vol. 27, no. 1, 1998, pp. 48–64.

[49] J. Håstad, "On the correlation of parity and small-depth circuits," *SIAM Journal of Computing*, vol. 43, no. 5, 2014, pp. 1699–1708.

[50] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby, "A pseudorandom generator from any one-way function," *SIAM Journal of Computing*, vol. 28, no. 4, 1999, pp. 1364–1396.

[51] P. Hatami, W. M. Hoza, A. Tal, and R. Tell, "Fooling constant-depth threshold circuits," *Electronic Colloquium on Computational Complexity: ECCC*, vol. 28, 2021, p. 002.

[52] W. M. Hoza, Private Communication, 2021.

[53] R. Impagliazzo and V. Kabanets, "Fourier concentration from shrinkage," *Computational Complexity*, vol. 26, no. 1, 2017, pp. 275–321.

[54] R. Impagliazzo, V. Kabanets, and A. Wigderson, "In search of an easy witness: Exponential time vs. probabilistic polynomial time," *Journal of Computer and System Sciences*, vol. 65, no. 4, 2002, pp. 672–694.

[55] R. Impagliazzo, R. Meka, and D. Zuckerman, "Pseudorandomness from shrinkage," in *Proc. 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2012, pp. 111–119.

[56] R. Impagliazzo and N. Nisan, "The effect of random restrictions on formula size," *Random Structures & Algorithms*, vol. 4, no. 2, 1993, pp. 121–133.

[57] R. Impagliazzo, R. Paturi, and M. E. Saks, "Size-depth tradeoffs for threshold circuits," *SIAM Journal of Computing*, vol. 26, no. 3, 1997, pp. 693–707.

[58] R. Impagliazzo and A. Wigderson, "P = BPP if E requires exponential circuits: Derandomizing the XOR lemma," in *Proc. 29th Annual ACM Symposium on Theory of Computing (STOC)*, 1999, pp. 220–229.

[59] V. Kabanets and R. Impagliazzo, "Derandomizing polynomial identity tests means proving circuit lower bounds," *Computational Complexity*, vol. 13, no. 1-2, 2004, pp. 1–46.

[60] V. Kabanets and Z. Lu, "Satisfiability and derandomization for small polynomial threshold circuits," in *Proc. 22nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, Art. No. 46, 19, 2018.

[61] T. Kaufman, S. Lovett, and E. Porat, "Weight distribution and list-decoding size of Reed-Muller codes," *IEEE Transactions on Information Theory*, vol. 58, no. 5, 2012, pp. 2689–2696.

[62] Z. Kelley, "An improved derandomization of the switching lemma," in *Proc. 53rd Annual ACM Symposium on Theory of Computing (STOC)*, 2021.

[63] V. M. Khrapčenko, "A certain method of obtaining estimates from below of the complexity of π-schemes," *Matematicheskie Zametki*, vol. 10, 1971, pp. 83–92.

[64] A. R. Klivans and D. van Melkebeek, "Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses," *SIAM Journal of Computing*, vol. 31, no. 5, 2002, pp. 1501–1526.

[65] A. R. Klivans and D. Spielman, "Randomness efficient identity testing of multivariate polynomials," in *Proc. 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pp. 216–223, 2001.

[66] I. Komargodski and R. Raz, "Average-case lower bounds for formula size," in *Proc. 45th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 171–180, 2013.

[67] I. Komargodski, R. Raz, and A. Tal, "Improved average-case lower bounds for De Morgan formula size: Matching worst-case lower bound," *SIAM Journal of Computing*, vol. 46, no. 1, 2017, pp. 37–57.

[68] D. Lewin and S. Vadhan, "Checking polynomial identities over any field: Towards a derandomization?" In *Proc. 30th Annual ACM Symposium on Theory of Computing (STOC)*, 1998, pp. 438–447.

[69] X. Li, "Improved two-source extractors, and affine extractors for polylogarithmic entropy," in *Proc. 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 168–177, 2016.

[70] Y. Liu and R. Pass, "Characterizing Derandomization Through Hardness of Levin-Kolmogorov Complexity," in *Proc. 37th Annual IEEE Conference on Computational Complexity (CCC)*, vol. 234, 35:1–35:17, 2022.

[71] S. Lovett, "Unconditional pseudorandom generators for low-degree polynomials," *Theory of Computing*, vol. 5, 2009, pp. 69–82.

[72] C.-J. Lu, "Hitting set generators for sparse polynomials over any finite fields," in *Proc. 27th Annual IEEE Conference on Computational Complexity (CCC)*, 2012, pp. 280–286.

[73] C.-J. Lu, O. Reingold, S. Vadhan, and A. Wigderson, "Extractors: Optimal up to constant factors," in *Proc. 35th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 602–611, 2003.

[74] M. Luby, B. Velickovic, and A. Wigderson, "Deterministic approximate counting of depth-2 circuits," in *Proc. 2nd Israel Symposium on Theory and Computing Systems*, pp. 18–24, 1993.

[75] P. B. Miltersen and N. V. Vinodchandran, "Derandomizing Arthur-Merlin games using hitting sets," *Computational Complexity*, vol. 14, no. 3, 2005, pp. 256–279.

[76] C. Murray and R. Williams, "Circuit lower bounds for nondeterministic quasi-polytime: An easy witness lemma for np and nqp," in *Proc. 50th Annual ACM Symposium on Theory of Computing (STOC)*, 2018.

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

[78] Z. Nie and A. Y. Wang, "Hilbert functions and the finite degree Zariski closure in finite field combinatorial geometry," *Journal of Combinatorial Theory. Series A*, vol. 134, 2015, pp. 196–220.

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

[80] N. Nisan and A. Wigderson, "Hardness vs. randomness," *Journal of Computer and System Sciences*, vol. 49, no. 2, 1994, pp. 149–167.

[81] M. S. Paterson and U. Zwick, "Shrinkage of De Morgan formulae under restriction," *Random Structures & Algorithms*, vol. 4, no. 2, 1993, pp. 135–150.

[82] R. Paturi and M. E. Saks, "Approximating threshold circuits by rational functions," *Information and Computation*, vol. 112, no. 2, 1994, pp. 257–272.

[83] J. Radhakrishnan and A. Ta-Shma, "Bounds for dispersers, extractors, and depth-two superconcentrators," *SIAM Journal of Computing*, vol. 13, no. 1, 2000, pp. 2–24.

[84] R. Raz, O. Reingold, and S. Vadhan, "Extracting all the randomness and reducing the error in Trevisan's extractors," *Journal of Computer and System Sciences*, vol. 65, no. 1, 2002, pp. 97–128.

[85] R. Santhanam, "Fighting perebor: New and improved algorithms for formula and QBF satisfiability," in *Proc. 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2010, pp. 183–192.

[86] R. Santhanam and R. Williams, "On medium-uniformity and circuit lower bounds," in *Proc. 28th Annual IEEE Conference on Computational Complexity (CCC)*, 2013, pp. 15–23.

[87] R. Servedio and L.-Y. Tan, "Deterministic search for CNF satisfying assignments in almost polynomial time," in *Proc. 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2017.

[88] R. A. Servedio, "Every linear threshold function has a low-weight approximator," *Computational Complexity*, vol. 16, no. 2, 2007, pp. 180–209.

[89] R. A. Servedio and L.-Y. Tan, "Luby-Veličković-Wigderson revisited: Improved correlation bounds and pseudorandom generators for depth-two circuits," in *Proc. 22nd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, vol. 116, 2018, Art. No. 56, 20.

[90] R. A. Servedio and L.-Y. Tan, "Improved pseudorandom generators from pseudorandom multi-switching lemmas," in *Proc. 23rd International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, Art. No. 45, 23, 2019.

[91] R. Shaltiel and C. Umans, "Simple extractors for all min-entropies and a new pseudorandom generator," *Journal of the ACM*, vol. 52, no. 2, 2005, pp. 172–216.

[92] A. Ta-Shma, C. Umans, and D. Zuckerman, "Lossless condensers, unbalanced expanders, and extractors," *Combinatorica*, vol. 27, no. 2, 2007, pp. 213–240.

[93] M. Sipser, "Expanders, randomness, or time versus space," in *Proc. Conference on Structure in Complexity Theory*, pp. 325–329, 1986.

[94] B. A. Subbotovskaja, "Realization of linear functions by formulas using $\vee$, &, $-$," *Soviet Mathematics. Doklady*, vol. 2, 1961, pp. 110–112.

[95] M. Sudan, L. Trevisan, and S. Vadhan, "Pseudorandom generators without the XOR lemma," *Journal of Computer and System Sciences*, vol. 62, no. 2, 2001, pp. 236–266.

[96] A. Tal, "Shrinkage of De Morgan formulae by spectral techniques," in *Proc. 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 551–560, 2014.

[97] A. Tal, "Formula lower bounds via the quantum method," in *Proc. 49th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 1256–1268, 2017.

[98] A. Tal, "Tight bounds on the fourier spectrum of AC0," in *Proc. 32nd Annual IEEE Conference on Computational Complexity (CCC)*, 15:1–15:31, 2017.

[99] R. Tell, "A note on the limitations of two black-box techniques in quantified derandomization," *Electronic Colloquium on Computational Complexity: ECCC*, vol. 24, 2017, p. 187.

[100] R. Tell, "Quantified derandomization of linear threshold circuits," in *Proc. 50th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 855–865, 2018.

[101] R. Tell, "Improved bounds for quantified derandomization of constant-depth circuits and polynomials," *Computational Complexity*, vol. 28, no. 2, 2019, pp. 259–343.

[102] R. Tell, "Proving that $pr\mathcal{BPP} = pr\mathcal{P}$ is as hard as proving that "almost $\mathcal{NP}$" is not contained in $\mathcal{P}$/poly," *Information Processing Letters*, vol. 152, 2019, p. 105 841.

[103] R. Tell, *On implications of better sub-exponential lower bounds for AC0*, 2020.

[104] S. Toda, "PP is as hard as the polynomial-time hierarchy," *SIAM Journal of Computing*, vol. 20, no. 5, 1991, pp. 865–877.

[105] L. Trevisan, "Extractors and pseudorandom generators," *Journal of the ACM*, vol. 48, no. 4, 2001, pp. 860–879.

[106] L. Trevisan and T. Xue, "A derandomized switching lemma and an improved derandomization of AC0," in *Proc. 28th Annual IEEE Conference on Computational Complexity (CCC)*, 2013, pp. 242–247.

[107] C. Umans, "Pseudo-random generators for all hardnesses," *Journal of Computer and System Sciences*, vol. 67, no. 2, 2003, pp. 419–440.

[108] S. P. Vadhan, "Pseudorandomness," Foundations and Trends® in Theoretical Computer Science, vol. 7, no. 1-3, 2012, pp. 1–336.

[109] E. Viola, "The complexity of constructing pseudorandom generators from hard functions," *Computational Complexity*, vol. 13, no. 3-4, 2005, pp. 147–188.

[110] E. Viola, "On approximate majority and probabilistic time," *Computational Complexity*, vol. 18, no. 3, 2009, pp. 337–375.

[111] E. Viola, "The sum of d small-bias generators fools polynomials of degree d," *Computational Complexity*, vol. 18, no. 2, 2009, pp. 209–217.

[112] E. Warning, "Bemerkung zur vorstehenden arbeit von herrn chevalley," *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, no. 11, 1935, pp. 76–83.

[113] R. Williams, "Non-uniform ACC circuit lower bounds," in *Proc. 26th Annual IEEE Conference on Computational Complexity (CCC)*, 2011, pp. 115–125.

[114] R. Williams, "Improving exhaustive search implies superpolynomial lower bounds," *SIAM Journal of Computing*, vol. 42, no. 3, 2013, pp. 1218–1244.

[115] R. Williams, "Algorithms for circuits and circuits for algorithms: Connecting the tractable and intractable," in *Proc. International Congress of Mathematicians (ICM)*, pp. 659–682, 2014.

[116] A. C.-C. Yao, "Separating the polynomial-time hierarchy by oracles," in *Proc. 26th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 1–10, 1985.

[117] A. C. Yao, "Theory and application of trapdoor functions," in *Proc. 23rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 80–91, 1982.