Satisfiability and Derandomization for Small Polynomial Threshold Circuits

Valentine Kabanets
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
kabanets@sfu.ca

Zhenjian Lu
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
zla54@sfu.ca

Abstract

A polynomial threshold function (PTF) is defined as the sign of a polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. A PTF circuit is a Boolean circuit whose gates are PTFs. We study the problems of exact and (promise) approximate counting for PTF circuits of constant depth.

- **Satisfiability (#SAT).** We give the first zero-error randomized algorithm faster than exhaustive search that counts the number of satisfying assignments of a given constant-depth circuit with a super-linear number of wires whose gates are s-sparse PTFs, for s almost quadratic in the input size of the circuit; here a PTF is called s-sparse if its underlying polynomial has at most s monomials. More specifically, we show that, for any large enough constant c, given a depth-d circuit with $(n^{2-1/c})$-sparse PTF gates that has at most $n^{1+\varepsilon_d}$ wires, where ε_d depends only on c and d, the number of satisfying assignments of the circuit can be computed in randomized time $2^{n-n^{\varepsilon_d}}$ with zero error. This generalizes the result by Chen, Santhanam and Srinivasan (CCC, 2016) who gave a SAT algorithm for constant-depth circuits of super-linear wire complexity with linear threshold function (LTF) gates only.
- **Quantified derandomization.** The quantified derandomization problem, introduced by Goldreich and Wigderson (STOC, 2014), asks to compute the majority value of a given Boolean circuit, under the promise that the minority-value inputs to the circuit are very few. We give a quantified derandomization algorithm for constant-depth PTF circuits with a super-linear number of wires that runs in quasi-polynomial time. More specifically, we show that for any sufficiently large constant c, there is an algorithm that, given a degree-Δ PTF circuit C of depth d with n^{1+1/c^d} wires such that C has at most $2^{n^{1-1/c}}$ minority-value inputs, runs in quasi-polynomial time $\exp\left((\log n)^{O(\Delta^2)}\right)$ and determines the majority value of C. (We obtain a similar quantified derandomization result for PTF circuits with n^{Δ}-sparse PTF gates.) This extends the recent result of Tell (STOC, 2018) for constant-depth LTF circuits of super-linear wire complexity.
- **Pseudorandom generators.** We show how the classical Nisan-Wigderson (NW) generator (JCSS, 1994) yields a nontrivial pseudorandom generator for PTF circuits (of unrestricted depth) with sub-linearly many gates. As a corollary, we get a PRG for degree-Δ PTFs with the seed length $\exp\left(\sqrt{\Delta \cdot \log n}\right) \cdot \log^2(1/\epsilon)$.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases constant-depth circuits, polynomial threshold functions, circuit analysis algorithms, SAT, derandomization, quantified derandomization, pseudorandom generators.

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2018.46

Related Version A full version of this paper is available at `https://eccc.weizmann.ac.il/report/2018/115` as ECCC TR18-115.

Acknowledgements We thank Suguru Tamaki for suggesting to us to consider sparse PTFs in the case of satisfiability, and for clarifying the MAX-k-SAT algorithm in [20] for us.

1 Introduction

Satisfiability and derandomization are famous examples of "circuit analysis" problems that, apart from being important algorithmic problems in their own right, are also intimately related to the notoriously difficult problem of proving circuit lower bounds. In this paper, we give several algorithmic results for these problems for the class of Boolean circuits with polynomial-threshold functions (PTFs) as gates.

Circuit-SAT. Circuit-SAT asks to determine whether a given Boolean circuit has a satisfying assignment. As a canonical NP-complete problem, it is not believed to have a polynomial-time (or subexponential-time) algorithm. However, it is still very interesting to look for nontrivial algorithms for Circuit-SAT running faster than naive exhaustive search. More specifically, given a circuit of polynomial size on n variables, is there a satisfiability algorithm that runs in time at most $2^n/n^{\omega(1)}$?

It turns out that this task is challenging even for very restricted classes of circuits. The difficulty of obtaining such a SAT algorithm can be partially explained by the work of Williams [26, 27] showing that a Circuit-SAT algorithm faster than exhaustive search for a given class of circuits can often be used to prove nontrivial circuit lower bounds against that same class of circuits (given that the class of circuits satisfies some mild conditions). In fact, Williams designed such a Circuit-SAT algorithm for ACC circuits (constant-depth circuits with AND, OR, NOT, and modular counting gates) that runs in time $2^{n-n^{1/\exp(d)}}$ (recently improved to $2^{n-n^{1/\mathrm{poly}(d)}}$ by [6]), where d is the depth of the circuit, and then used this algorithm to show that NEXP contains a language that is not computable by any family of polynomial-size constant-depth ACC circuits, a breakthrough result in circuit complexity.

Given the connections between nontrivial Circuit-SAT algorithms and circuit lower bounds, one of the next big goals in circuit complexity is to design such an algorithm for the class of TC^0 circuits, constant-depth circuits with majority gates. Lower bounds against the class of polynomial-size TC^0 circuits is currently one of the most important open problems in complexity.

Derandomization. A central problem in derandomization is to give an efficient deterministic algorithm for computing the majority value of a given Boolean circuit, under the promise that the fraction of minority-value inputs to the circuit is at most $1/3$. That is, given a circuit that outputs some unknown value $b \in \{0,1\}$ on all but at most $1/3$ fraction of inputs, we need to determine this majority value b, efficiently deterministically.

As for Circuit-SAT, it is also known that a "faster-than-brute-force" algorithm solving the aforementioned derandomization problem for a circuit class $\mathcal{C}$ (satisfying some mild conditions) implies lower bounds against that class $\mathcal{C}$ [26].

Black-Box Derandomization: Pseudorandom generators. One way to solve the derandomization problem for a class $\mathcal{C}$ of circuits is to construct a pseudorandom generator (PRG) for $\mathcal{C}$. A PRG for a class $\mathcal{C}$ of n-input Boolean circuits is an efficiently deterministically computable function G mapping short binary strings (seeds) to longer binary strings so that every $C \in \mathcal{C}$ accepts G's output on a uniformly random seed with about the same probability as that for an actual uniformly random string. More precisely, we say that a generator $G\colon \{0,1\}^r \to \{0,1\}^n$ is *ϵ-fooling* for a class $\mathcal{C}$ of Boolean circuits if for every $C\colon \{0,1\}^n \to \{0,1\}$ from $\mathcal{C}$, $|\mathbf{Pr}[C(G(x)) = 1] - \mathbf{Pr}[C(y) = 1]| \leq \epsilon$, for uniformly random $x \in \{0,1\}^r$ and $y \in \{0,1\}^n$. The parameter r is called the seed length of the PRG. Then

given a PRG that fools $\mathcal{C}$, for every $C \in \mathcal{C}$, we can estimate the fraction of accepted inputs to within an additive error ϵ, by trying all possibles seeds. This gives a deterministic algorithm solving the derandomization problem in time approximately 2^r.

Note that a PRG yields *black-box* derandomization in the sense that we do not need to be given as input a circuit $C \in \mathcal{C}$ in order to decide the set of 2^r query points for C; the set of 2^r query points is the same for all circuits in class $\mathcal{C}$.

Quantified derandomization. As standard derandomization appears difficult even for weak circuit classes, one considers relaxations. One relaxation is to assume that a given n-input circuit C outputs an unknown value $b \in \{0,1\}$ on all but "very few" inputs, e.g., $2^n/n^{\omega(1)}$ inputs rather than $2^n/3$ in the case of standard derandomization. Goldreich and Wigderson [8] named this a *quantified derandomization problem*. More formally, for a class $\mathcal{C}$ of circuits, and a function $B\colon \mathbb{N} \to \mathbb{N}$, the $(\mathcal{C}, B)$-quantified derandomization problem is the following: given a circuit $C \in \mathcal{C}$ such that C has at most $B(n)$ minority-value inputs in $\{0,1\}^n$, determine the majority value $b \in \{0,1\}$ for C.

It was immediately observed by [8] that for "sufficiently powerful" circuit classes (e.g., $\mathsf{AC}^0[\oplus]$, polynomial-size constant-depth circuits with unbounded fan-in AND, OR, parity gates, and negation gates), quantified derandomization is *equivalent* to standard derandomization, as one can perform efficient pseudo-random sampling (via randomness extractors) within the same circuit class. Thus, quantified derandomization may be possible to achieve (given our current knowledge) only for "very weak" circuit classes. [8] gave quantified derandomization algorithms for AC^0 (later strengthened by [22]) and some other classes. Recently, Tell [24] showed that quantified derandomization is also possible for constant-depth LTF circuits of small super-linear wire complexity (and that improving this to slightly higher super-linear wire complexity is as hard as getting nontrivial standard derandomization for the circuit class TC^0, which in turn would imply TC^0 circuit lower bounds).

PTF circuits. The focus of the present paper is on circuits whose gates are polynomial threshold functions. An n-variate polynomial threshold function (PTF) is defined as the sign $\mathrm{sgn}(p)$ of a multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. Here, for $v \in \mathbb{R}$, we define the sign function $\mathrm{sgn}(v)$ to be 1 on $v > 0$, and 0 on $v < 0$. There are two common complexity measures for PTFs: *degree*, which is the degree of p, and *sparsity*, which is the number of monomials in p, where a monomial is of the form $\prod_{i \in S}(x_i \oplus b_i)$ where $S \subseteq [n]$ and $b_i \in \{0,1\}$ for each $i \in S$. We call the PTF s-sparse if $p(x_1, \ldots, x_n)$ is the sum of at most s monomials. PTFs of degree 1 are called linear threshold functions (LTFs). Thus an s-sparse PTF can be equivalently defined as an LTF of at most s terms, where each term is an AND of literals (variables and their negations).

Polynomial threshold circuits are circuits whose gates are PTFs. We will study both circuits with low-degree PTF gates and circuits with sparse PTF gates. We call a circuit degree-Δ PTF circuit if its gates are degree-Δ PTFs. Similarly, a circuit is called s-sparse PTF circuit if its gates are s-sparse PTFs. We note that when discussing circuits, the word "sparse" is often used to describe circuits with a small number of wires (recall that the number of wires is the sum of fan-ins over all gates of the circuit). To avoid ambiguity, we clarify that in this paper the word "sparse" always refers to PTFs. For example, a sub-quadratically sparse PTF circuit means a circuit with gates that are sub-quadratically sparse PTFs (i.e., PTFs that have a sub-quadratic number of monomials).

1.1 Our results

Circuit-SAT for sub-quadratically sparse PTF circuits with $n^{1+\varepsilon}$ wires. PTFs are very powerful even for small sparsity. For example, s-sparse PTFs can encode MAX-SAT with s clauses and exponential weights, a problem known how to solve nontrivially only for a sub-quadratic number of clauses. Therefore, a nontrivial SAT algorithm for PTFs of quadratic sparsity would break the current barrier of solving MAX-SAT with exponential weights. In fact, since a polynomial of degree-2 has at most a quadratic number of monomials, such an algorithm would also give a nontrivial SAT algorithm for degree-2 PTFs, which is currently unknown[1].

We give the first nontrivial #SAT algorithms (counting the number of satisfying assignments of a given circuit) for the class of constant-depth circuits with PTF gates, where the PTF circuit has small super-linear wire complexity (defined as the sum of fan-ins over all gates of the circuit) and each PTF gate has sub-quadratic sparsity. Our main result is the following.

▶ **Theorem 1** (#SAT algorithm for sub-quadratically sparse PTF circuits)**.** *There is a constant $b_1 > 1$ such that, for every $c \geq b_1$ and $d > 0$, there is a zero-error randomized algorithm that counts the number of satisfying assignments of any given depth-d, n-variate circuit with*

- *$(n^{2-1/c})$-sparse PTF gates, and*
- *at most $n^{1+\varepsilon_d}$ wires.*

The running time of this #SAT algorithm is at most $2^{n-n^{\varepsilon_d}}$, where $\varepsilon_d = c^{-3^d}$.

We also get an algorithm with better parameters if we further assume that the sparse PTF gates in the circuit have low degree. Let $\mathcal{G}_{\Delta,c}$ denote the class of Boolean functions where each function can be computed as an LTF of at most $n^{2-1/(c\cdot\Delta^2)}$ arbitrary Δ-variate Boolean functions.

▶ **Theorem 2** (#SAT algorithm for sub-quadratically sparse PTF circuits with low-degree)**.** *There exists a constant $b_2 > 1$ such that, for every $d, \Delta > 0$ and $c \geq b_2$, there is a zero-error randomized algorithm that counts the number of satisfying assignments of any given depth-d, n-variate circuit with*

- *gates from $\mathcal{G}_{\Delta,c}$, and*
- *at most $n^{1+\varepsilon_{d,\Delta}}$ wires.*

The running time of this #SAT algorithm is at most $2^{n-n^{\varepsilon_{d,\Delta}}}$, where $\varepsilon_{d,\Delta} = \left(c\cdot\Delta^2\right)^{-d}$.

Quantified derandomization for PTF circuits with $n^{1+\varepsilon}$ wires in quasi-polynomial time.

▶ **Theorem 3** (Quantified derandomization for low-degree (or sparse) PTF circuits)**.** *For any constant $c \geq 122$ and any $\Delta, d > 0$ such that $\Delta \ll \sqrt{\log n/(c^d \cdot \log\log n)}$, let $\mathcal{C} = \mathcal{C}(n, d, \Delta, c)$ be the class of n-variate, depth d PTF circuits with*

- *degree-Δ PTF gates (or n^{Δ/c^d}-sparse PTF gates), and*
- *at most n^{1+1/c^d} wires.*

The $\left(\mathcal{C}, 2^{n^{1-7/\sqrt{c}}}\right)$-quantified derandomization problem is solvable in time $2^{(\log n)^{O\left(\Delta^2\right)}}$.

---

[1] Sakai, Seto, Tamaki and Teruyama [20] recently reported a faster-than-brute-force algorithm for MAX-k-SAT for any constant k with arbitrary weights (which implies a satisfiability algorithm for degree-k PTFs). However, their algorithm is conditional in that it relies on an assumption that one can *efficiently* reduce the weights of a given n-variate LTF to integral weights of magnitude at most $2^{O(n\log n)}$. While it is known that such small weights exist for every LTF [17], it is currently not known how to find them efficiently.

PRG for PTF circuits with few gates.

▶ **Theorem 4** (PRG for PTF Circuits). *There exists a constant $E > 0$ such that the following holds. For any positive integers α and Δ, let $\mathcal{C} = \mathcal{C}(n, \alpha, \Delta)$ be the class of degree-Δ PTF circuits on n inputs with at most $s = n^{\frac{1}{\alpha+1}} / \left(E \cdot 5^{\alpha \cdot \Delta} \cdot \log^2(n) \cdot \log(n/\epsilon)\right)$ gates. There exists a $\mathsf{poly}(n)$-time computable PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ ϵ-fooling $\mathcal{C}$, where the seed length is $r = n^{2/(\alpha+1)}$.*

We get the following PRG for a single PTF (by setting α appropriately).

▶ **Corollary 5** (PRG for PTFs). *There exists a PRG $G\colon \{0,1\}^r \to \{0,1\}^n$, computable in deterministic time $\mathsf{poly}(n)$, that ϵ-fools degree-Δ PTFs on n variables with the seed length $r = \exp\left(O\left(\sqrt{\Delta \cdot \log n}\right)\right) \cdot \log^2(1/\epsilon)$.*

1.2 Our techniques

A common way to analyze constant-depth circuits is to apply (random) restrictions, getting some depth reduction, and iterate, until the resulting circuit becomes very simple. Our #SAT algorithms and quantified derandomization algorithms for constant-depth PTF circuits also follow this approach, mainly relying on the ideas of [5] for depth reduction, and [11] for (pseudo-) random restrictions for PTFs. Our PRG is based on the celebrated Nisan-Wigderson "hardness-based" generator (NW PRG) [19]. We give more details in the following sections as we discuss each of the results.

1.3 Related work and comparison

Circuit Satisfiability. Impagliazzo, Paturi and Schneider [10] gave a Circuit-SAT algorithm for depth-2 LTF circuits with few wires; this result was improved by Chen and Santhanam [4]. Recently, Alman, Chan and Williams [1] and Tamaki [21] both gave Circuit-SAT algorithms for depth-2 LTF circuits with an almost quadratic number of gates.

The most closely related previous work is by Chen, Santhanam and Srinivasan [5] who gave a Circuit-SAT algorithm for circuits with a super-linear number of wires whose gates are LTFs. In particular, they show that the satisfiability of a depth-d, n-variate circuit with LTF gates and at most $n^{1+\varepsilon_d}$ wires can be solved by a zero-error randomized algorithm in time $2^{n-n^{\varepsilon_d}}$, where $\varepsilon_d = c^{-d}$ for some constant c. Our results extend their algorithm to the more general case of circuits with *sparse PTF* gates. In particular, our algorithm in Theorem 2 for $\Delta = 1$ subsumes the Circuit-SAT algorithm for LTFs in [5]. Also note that the sparsity of the PTF gates in our model is almost quadratic in n, which is the input size of the circuit. "Opening up" the PTF gates in the circuit and expressing them as LTFs of terms will result in a (constant-depth) LTF circuit that can have an almost quadratic number of wires, and such a circuit cannot be analyzed by the result in [5].

Quantified derandomization. The quantified derandomization problem was first introduced by Goldreich and Wigderson in [8], where they obtained a polynomial time algorithm that finds the majority output of a given AC^0 circuit that has at most $2^{n^{0.999}}$ minority-value inputs. The key tool in their algorithm is a derandomized version of Håstad's switching lemma [9] with logarithmic seed length. In addition, they obtain quantified derandomization results for log-space algorithms and arithmetic circuits. The quantified derandomization algorithm for AC^0 was generalized by Tell [22] to handle AC^0 circuits with at most $2^{\Omega(n/\log^{d-2} n)}$ minority-value inputs, where d is the depth, with an increase of the running time to $2^{\tilde{O}(\log^3 n)}$. As

mentioned above, Tell [24] has recently obtained a quantified derandomization algorithm for depth-d LTF circuits with $n^{1+1/\exp(d)}$ wires with at most $2^{n^{1-1/5d}}$ minority-value inputs, running in time $n^{(\log\log n)^2}$. Our result extends this to low-degree PTF circuits and sparse PTF circuits, at the expense of increasing the running time to quasi-polynomial (for constant degree and polynomial sparsity). For the results on reducing standard derandomization to quantified derandomization, see [8, 22, 23, 24].

PRGs. There has been a long sequence of works on constructing PRGs (of varying strength) for various sub-classes of $\mathsf{P/poly}$. Among these known PRG constructions, some are NW-style "hardness-based" generators, while others are *ad hoc* constructions (often using such standard pseudorandomness tools as hashing, limited-wise independence, expander graphs, etc.) The previous PRGs for PTFs due to [16, 13] are of the latter kind. The construction uses hashing and limited-wise independence. The analysis is quite involved, and depends on a number of analytic tools for polynomials (concentration and anti-concentration results, the invariance principle, hypercontractivity, regularization, etc.). In contrast, our PRG for PTFs (of Corollary 5) is the NW-style construction, whose analysis is simple, assuming an average-case lower bound for an appropriate class of functions.

For constant degree PTFs and constant error ϵ, the PRG of [16, 13] has exponential stretch (mapping a seed of length $O(\log n)$ to an n-bit string fooling n-input PTFs). However, these PRGs has polynomial dependence in the error $1/\epsilon$ and cannot handle small error. Our PRG cannot achieve such exponentially long stretch for constant error, but it can achieve even exponentially small error ϵ with a nontrivial (sub-linear) seed size, which is impossible for the PRGs of [16, 13].

In their work studying correlation bounds for AC^0 circuits with few symmetric gates [14], Lovett and Srinivasan obtained an average-case hard function for constant depth poly-size AC^0 circuits with few LTF gates and used it to construct a PRG fooling such circuits with polynomial stretch and exponentially small error, also based on the generic construction of Nisan and Wigderson. Since a PTF can be viewed as a depth-2 circuit computing an LTF of ANDs, such a PRG also fools small PTF circuits. While the PRG in [14] can fool a more general model, which is constant-depth AC^0 circuits augmented with LTF gates, it can have only polynomial seed stretch and the circuit can have only constant depth. Our work here focuses on circuits with only PTF gates. Our PRG can have sub-polynomial seed length and it can fool PTF circuits regardless of the depth as long as the number of gates is small. In particular, our PRG for a single PTF with sub-polynomial seed length (Corollary 5) can be used to construct a PRG for degree-2 PTFs with a seed length that is logarithmic in the input size and *sub-polynomial* in the error (see [12]).

Threshold circuits. It is well known that the class of constant-depth polynomial-size TC^0 circuits is equivalent to the class of constant-depth polynomial-size circuits with LTF gates [7]. LTF circuits have been intensively studied in complexity theory. PTF circuits have been previously studied for lower bounds [18, 11]. Threshold circuits are also studied as a model of artificial neural networks [15] (see also [2]), where a threshold gate is also called a neuron.

Remainder of the paper. We give the necessary background in Section 2. We prove our satisfiability algorithms (Theorem 1). We describe our quantified derandomization result for low-degree PTF Circuits in Section 4, and our PRG result in Section 5. We conclude with some open problems in Section 6. For lack of space, we omit many proofs in this extended abstract and refer the reader to the full version for those proofs.

2 Preliminaries

2.1 Notation

For a positive integer n, let $[n]$ denote the set $\{1, 2, \ldots, n\}$. For a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, we define the *majority value* of f to be the bit value $b \in \{0,1\}$ that maximizes the quantity $\mathbf{Pr}_{x\sim\{0,1\}^n}[f(x) = b]$, and we call $1-b$ the *minority value*. We say that two Boolean functions f and g are δ-*close* if $\mathbf{Pr}_x[f(x) \neq g(x)] \leq \delta$. We say that a function f is δ-close to an *explicit* constant if f is δ-close to some constant function and such a constant function can be efficiently determined from f.

We will often view an s-sparse PTF as an LTF of at most s AND gates. It is well known that every LTF on m variables has a canonical representation, where the coefficients are integers of magnitude at most $2^{O(m \log m)}$ [17]. Therefore, every s-sparse PTF is equivalent to some s-sparse PTF whose coefficients are integers of magnitude at most $2^{O(s \log s)}$. Without loss of generality, for a circuit with s-sparse PTF gates, we assume the coefficients of all gates have bit complexity $\mathsf{poly}(s)$.

2.2 Random restrictions

A random restriction is a process that randomly fixes the values of a subset of variables. We will often view a random restriction as a two-step process: the first step is selecting (in some random manner) a subset of unrestricted variables and the second step is fixing (in some random manner) the values of all the other variables. Depending on different contexts, we will consider different types of random restrictions based on how the unrestricted variables are picked and how the restricted variables are fixed.

Truly random restriction. The first type is the *(truly) r-random restriction*, for a parameter $0 < r < 1$. It is the process that leaves each variable, independently, free with probability r, and otherwise assigns it 0 or 1 uniformly at random.

Pseudorandom restriction using limited-wise independence. We can also pick and fix variables in a pseudorandom manner. One way to do this is to use a limited-wise independent distribution. For integers $n, m > 0$, a distribution X on $[m]^n$ is called k-wise independent if any k coordinates of X are uniformly distributed. That is, for any $1 \leq i_1, \ldots, i_k \leq n$ and every $b_1, \ldots, b_k \in [m]$, we have $\mathbf{Pr}[X_{i_1} = b_1, \ldots, X_{i_k} = b_k] = m^{-k}$. A k-wise independent distribution over $[m]^n$ can be constructed using $k \cdot \log n$ random bits for $m \leq n$ (see, e.g., [25]).

For a random restriction ρ, we say that ρ picks the unrestricted variables k-wise independently, each with probability r, if each variable is set to be unrestricted by ρ with probability r, and any k of the variables are independent. Note that this process can be done using a k-wise independent distribution over $[1/r]^n$, where n is the number of variables. Also, we say that ρ fixes the variables k-wise independently if each variable is assigned 0 or 1 uniformly at random by ρ and any k of the variables are independent.

Random block restriction. A random block restriction picks the set of unrestricted variables by picking a block from some arbitrary predetermined partition of variables. More formally, an *m-block random restriction* for a function is the following process: given an arbitrary partitioning of input variables into m disjoint blocks, a random m-block restriction picks a uniformly random block $\ell \in [m]$ and fixes all variable outside the chosen block ℓ to 0 or 1 according to some distribution. Note that we can use a random block restriction to simulate

the first two types of (pseudo-)random restrictions above. For example, to simulate a truly r-random block restriction, we first randomly partition the variables into $m = 1/r$ disjoint blocks, where each variable is assigned to block $i \in [m]$, independently, with probability $1/m$. Then we apply an m-random block restriction based on the partition in the previous step, by fixing the variables outside the selected block uniformly at random. Similarly, to simulate a pseudorandom restriction using limited-wise independence, we can partition (hash) the variables into disjoint blocks limited-wise independently, and apply a random block restriction where we fix the variables also using a limited-wise independent distribution.

2.3 Useful tools for analyzing PTFs

▶ **Definition 6** (δ-concentrated PTFs)**.** Let $p\colon \{0,1\}^n \to \mathbb{R}$ be a degree-Δ multi-linear polynomial and $f = \operatorname{sgn}(p)$. For parameters $0 < \delta \leq 1/2$ and $\lambda \geq 1$, we call p (and f) (δ, λ)-*concentrated* if $\mathbf{Var}[p] \leq (162 \cdot \log(1/\delta))^{-\lambda \cdot \Delta} \cdot \mathbf{Exp}\left[p^2\right]$, where $\mathbf{Exp}$ and $\mathbf{Var}$ denote the *expectation* and the *variance*, respectively, under the uniform distribution over $\{0,1\}^n$. We refer to $(\delta, 1)$-concentrated polynomials as δ-concentrated.

A useful property of concentrated PTFs is that they are close to an explicit constant.

▶ **Lemma 7** (Concentrated implies close to constant)**.** *For any $0 < \delta \leq 1/2$, if a PTF $f = \operatorname{sgn}(p)$ is δ-concentrated, then f is δ-close to the constant function* $\operatorname{sgn}(\mathbf{Exp}[p])$.

For a multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$, it easy to see that the constant function $\operatorname{sgn}(\mathbf{Exp}[p])$ from Lemma 7 is efficiently computable for a given polynomial p.

The following is a random restriction lemma for PTFs which says that a low-degree PTF is likely to become concentrated under a (truly) random block restriction.

▶ **Lemma 8** (Random block restriction lemma [11])**.** *For any $0 < \delta < 1$ and any positive integers m, λ, let $\mathcal{B}_m$ be a m-block random restriction that fixes variables uniformly at random. Then for degree-Δ PTF f whose variables are partitioned into m blocks, we have*

$$\mathbf{Pr}_{\rho \sim \mathcal{B}_m}\left[f_\rho \text{ is not } (\delta, \lambda)\text{-concentrated}\right] \leq m^{-1/2} \cdot (\log m \cdot \log(1/\delta))^{O\left(\lambda \cdot \Delta^2\right)}.$$

There is also a derandomized version of the above random block restriction lemma.

▶ **Lemma 9** (Pseudorandom block restriction lemma [11])**.** *For any $0 < \delta, \gamma < 1$ and any positive integers m, λ, there is a polynomial-time algorithm for sampling a m-block random restriction $\mathcal{B}'_m$, that uses at most $m^\gamma \cdot \log n$ random bits, so that the following holds. For any n-variate degree-Δ PTF f whose variables are partitioned into m blocks, we have*

$$\mathbf{Pr}_{\rho \sim \mathcal{B}'_m}\left[f_\rho \text{ is not } (\delta, \lambda)\text{-concentrated}\right] \leq m^{-1/2} \cdot (\log m \cdot \log(1/\delta))^{O\left(\lambda \cdot \Delta^2/\gamma\right)}.$$

Moreover, $\mathcal{B}'_m$ fixes the variables $(192 \cdot \Delta \cdot \log(1/\delta))$-wise independently.

3 #SAT algorithm for PTF circuits

To get our Circuit-SAT algorithm for circuits with sparse PTF gates, we generalize the analysis of the Circuit-SAT algorithm for small LTF circuits in [5]. An oversimplified description is as follows. We show that for a depth-d circuit with a slightly super-linear number of wires, whose gates are sparse PTFs, there exists a shallow decision tree such that, for most of the leaves, the circuit restricted to that leaf can be "approximated" by some depth-$(d-1)$ circuit. Then we recursively apply a Circuit-SAT algorithm to depth-$(d-1)$ circuits. However, to actually implement this idea, we need three ingredients, which we describe in detail below.

Satisfiability for conjunctions of sparse PTFs. First, we need a base-case algorithm. In the case of LTF circuits in [5], the base case is a conjunction of LTFs, and there is a known algorithm by Williams [28] for such circuits. In contrast, in our case, the base case is a conjunction of sparse PTFs. Using the polynomial method in circuit complexity, we are able to design a Circuit-SAT algorithm for such circuits. More specifically, the algorithm is based on the framework for designing satisfiability algorithms developed by Williams [27, 28]. The idea is to transform a given constant-depth circuit into a low-degree probabilistic polynomial and solve satisfiability by evaluating the polynomial on all points in a faster-than-brute-force manner. Applying this idea naively, we get a randomized SAT algorithm that makes error. Such a base-case algorithm would result in the final SAT algorithm for PTF circuits that also makes error. However, using some derandomization ideas similar to those in [3, 21], we are able to obtain a *deterministic* base-case algorithm that can count the number of satisfying assignments. This allows us to make our final SAT algorithm for PTF circuits to be *zero-error* randomized algorithm that *counts* the number of satisfying assignments.

▶ **Lemma 10.** *There exists a deterministic algorithm that counts the number of satisfying assignments of every n-variate circuit C that is a conjunction of k s-sparse PTF gates, where $s \geq n$, such that the algorithm runs in time at most $2^{n-\left(\frac{n}{\sqrt{s}\cdot(\log s)^{O(1)}\cdot\log^2 k}\right)^{1/2}}$.*

Depth reduction for sparse PTF circuits with few wires. Secondly, to construct the aforementioned decision tree, we need a random restriction lemma showing that, under a (truly) random restriction, a gate in the circuit is likely to be close to constant. More specifically, Chen et al. [5] showed that using such a random restriction lemma, one can get a shallow decision tree such that, restricted to most of the leaves, a circuit with few wires will have many of its bottom-layer gates becoming close to constant, so that we can replace them with actual constants, and the depth decreases by one if we futher remove the rest of the few bottom-layer gates that are not close to constant. In our case, we need a similar restriction lemma for sparse PTFs. It is easy to see that a sparse PTF is likely to become a low-degree PTF under a mild random restriction. Combining this observation with the block restriction lemma for low-degree PTFs (Lemma 8), it is not difficult to show that for any $\left(s \leq n^{O(1)}\right)$-sparse PTF f, $\mathbf{Pr}_{\rho\sim\mathcal{R}_r}[f_\rho \text{ is not } \delta\text{-close to an explicit constant}] \leq r^{\Omega(1)}$, for some very small δ, where $\mathcal{R}_r$ denotes the truly r-random restriction. Using this structual result for sparse PTFs and the idea in [5] (see Section 4.1.1 of [5]), we get the following.

▶ **Lemma 11.** *For any integer $d \geq 2$ and any $(\log n)^{-1} \ll \varepsilon < 1$, let*

- *$\beta = E \cdot \varepsilon$ where E is some constant, $s \leq n^{O(1)}$, $\delta = \exp\left(-n^{\Omega(\beta^3)}\right)$, and*
- *C be any depth-d, n-variate, s-sparse PTF circuit with at most $w = n^{1+\varepsilon}$ wires.*

Then there exists a decision tree T of depth $n - n^{1-\beta}$ such that, for a random leaf σ of T, with probability at least $1 - \exp(-n^{\varepsilon})$, we have the following: C_σ is a depth-d circuit of wire complexity at most w such that its bottom layer has at most n gates that are δ-close to an explicit constant and at most n^β gates that are not δ-close to an explicit constant. Moreover, such a tree can be constructed in zero-error randomized time $\tilde{O}\left(2^{n-n^{1-\beta}}\right)$.

Enumerating minority outputs of sparse PTFs. Finally, in our main algorithm, we will need to apply the depth reduction lemma (Lemma 11) to the circuit to conclude that many of the gates at the bottom layer will become close to constant so that we can replace them with actual constants. This changes the function of the circuit and we need to deal with the inputs where these gates do not evaluate to their majority values. This issue can be handled

if given a sparse PTF we can find the set of all inputs where it evaluates to its minority value, in a relatively efficient way. As shown in [5], there is an efficient way to do this for functions whose satisfiability can be decided in polynomial time, such as LTFs. However, we cannot apply this for sparse PTFs since there is no known polynomial-time SAT algorithm for sparse PTFs. We overcome this issue for sparse PTFs by reducing to the case of LTFs, using the following observation from Chen and Santhanam [4], which says that for a collection of sub-quadratic many monomials, there exists a decision tree of not-too-many leaves such that, under each leaf, each of the monomials becomes a single literal. As a result, we get the following way to enumerate the set of minority-value inputs for a sub-quadratically sparse PTFs in non-trivial time. (See Section A for the proof).

▶ **Lemma 12.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be a s-sparse PTF with coefficients of bit complexity $\mathsf{poly}(n)$, where $s \geq n$ and let S be the set of inputs on which f evaluates to 0 (or 1). Then S can be enumerated in time $\left(2^{n-\Omega(n^2/s)} + |S|\right) \cdot \mathsf{poly}(n)$.*

Putting it all together. Let $\varepsilon_d = 1/\left(E^{3^{d-1}}\right)$ and $\beta_d = E \cdot \varepsilon_d$, where E is a sufficiently large constant. We can show the following.

▶ **Theorem 13.** *For any integer $d \geq 1$, the number of satisfying assignments of a depth-d, n-variate circuit with $(n^{2-10\beta_2})$-sparse PTF gates and at most $n^{(1+\varepsilon_d)}$ wires can be computed by a zero-error randomized algorithm in time $\mathsf{poly}(n) \cdot 2^{n-n^{\Omega\left(\beta_d^3\right)}}$.*

▶ **Definition 14** (Skew Circuits)**.** We say that a circuit C is (d,n,t,s)-skew if it is a n-variate circuit that can be expressed as a conjunction of some circuit C' and at most t s-sparse PTFs, where C' is a depth-d circuit with s-sparse PTF gates and has at most $w = n^{1+\varepsilon_d}$ wires. We call C' the skew subscircuit of C.

Let $\mathcal{T}(d,n,t,s)$ denote the supremum, over all (d,n,t,s)-skew circuits C, of the randomized running time of counting the number of satisfying assignments of C.

Using the depth reduction lemma (Lemma 11) and the enumeration lemma (Lemma 12), we can obtain the following lemma which says that we can reduce the task of counting satisfying assignments of depth-d circuits to that of depth-$(d-1)$ circuits.

▶ **Lemma 15.** *If $s \leq n^{2-5\beta_d}$, then*

$$\mathcal{T}(d,n,t,s) \leq 2^{n-n^{1-2\beta_d}} \cdot 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right) + \mathsf{poly}(n) \cdot 2^{n-n^{\Omega\left(\beta_d^3\right)}}.$$

The proof of the above lemma is similar to that in [5]. We give a detailed proof in Sectoin B.

Given the recursion in Lemma 15, it is not difficult to prove Theorem 13 by solving the recursion and using the SAT algorithm for conjunctions of sparse PTFs (Lemma 10) as the base case. We refer the reader to the full version for the proof.

4 Quantified derandomization for PTF circuits

At a high level, our quantified derandomization algorithm follows the approach of [8]. Given a circuit C with at most B minority-value inputs, we find a restriction ρ such that ρ leaves a large number, say n', of variables unrestricted, and that C_ρ is very close to some simple function $\tilde{C}$ (say they agree on all but at most $1/6$ fraction of inputs). Then the number of minority-value inputs for $\tilde{C}$ is at most $B + 2^{n'}/6$. If B is also at most $2^{n'}/6$, then we can determine the required majority value for C by finding the majority value $\tilde{C}$, which is a

simple function. This approach is also used by Tell [24] to get a quantified derandomization algorithm for LTF circuits with a slightly super-linear number of wires.

Let's first consider a depth-2 LTF circuit with few wires. In [5], Chen, Santhanam and Srinivasan proved a random restriction lemma for LTFs, which says that under a random restriction, an LTF is likely to become very close to an explicit constant. Using this result, one gets that under such a random restriction, many of the gates in the bottom layer of the circuit are expected to become close to constants. Since the circuit has only a few wires, one can further fix a small number of variables so that only those gates that are close to constants are left. Finally, by replacing these gate with their majority values, we obtain a single LTF that is close to the original depth-2 circuit.

Such a random restriction lemma was extended to low-degree PTFs in [11], so we can conclude the same for low-degree PTF circuits. One important issue, though, is that the above "depth reduction" argument only holds for *random* restrictions (but with high probability). So to get quantified derandomization, one will need to consider all possible restrictions. To handle this issue, Tell [24] derandomized the random restriction lemma for LTFs mentioned above so that such a restriction can be sampled using few random bits. As a result, one only needs to consider a much smaller sample space of restrictions.

Pseudorandom restrictions for PTFs. The pseudorandom restriction lemma for LTFs in [24] is obtained using a PRG for LTFs. One way to extend it to PTFs is to use a PRG for PTFs. However, unlike LTFs, for which a PRG with a very short seed is known, all known PRGs for PTFs have a large seed length (for small error, which is needed for the argument). In fact, the only PRG that we can use in this case is the one in Corollary 5, and it would give a quantified derandomization algorithm running in time $\exp\left(\exp\left(\sqrt{\Delta \cdot \log n}\right)\right)$. To get quasi-polynomial running time, we use a powerful pseudorandom block restriction lemma for PTFs (Lemma 9), which needs only a poly-logarithmic number of random bits, and convert it into the following pseudorandom restriction lemma. (The proof is in Section C.)

▶ **Lemma 16** (Pseudorandom restriction lemma for low-degree PTFs). *For any constant $c > 0$, any $\alpha < 1$ and any positive integer Δ such that $\Delta \ll \sqrt{\alpha \cdot \log n / \log\log n}$, there is a random restriction $\mathcal{R}$ such that the following holds:*

- *$\mathcal{R}$ picks the unrestricted variables $(\log n)$-wise independently, each with probability $n^{-\alpha}$.*
- *$\mathcal{R}$ fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently.*
- *$\mathcal{R}$ can be sampled in polynomial time using $(\log n)^{O(\Delta^2)}$ random bits.*
- *For any degree-Δ PTF f on n variables, $\mathbf{Pr}_{\rho\sim\mathcal{R}}[f_\rho$ is not $(n^{-c}, 3)$-concentrated$] \leq n^{-\alpha/3}$.*

Bias preservation for PTFs. Now we need to apply the above idea to a depth-d circuit C. It seems that all we need to do is applying the pseudorandom restriction $d-1$ times. While this is true, the analysis is much more subtle. For example, after applying the first pseudorandom restriction ρ_1, we get a new circuit $\tilde{C}$ of depth $(d-1)$ on some n' variables so that it agrees with C_{ρ_1} on all but at most say $2^{n'}/6$ inputs. Now consider a subsequent restrictions ρ'. Note that the final number of unrestricted variable n'' after ρ' is much smaller than n'. Therefore, $(C_{\rho_1})_{\rho'}$ and $\tilde{C}_{\rho'}$ can disagree on all the inputs (since $2^{n'}/6 \gg 2^{n''}$) so $\tilde{C}_{\rho'}$ cannot be used to determine the correct output of $(C_{\rho_1})_{\rho'}$, which is also the correct output of C. This issue can be handled if those bottom layer gates that become close to constant after applying one step of pseudorandom restriction will remain close to the *same* constant for subsequent pseudorandom restrictions. Such a "bias preservation lemma" for LTFs is also proved in [24], again using a PRG for LTFs. For PTFs, we use an observation in [11], which

says that a concentrated PTF is likely to remain concentrated under any random restriction that fixes variables limited-wise independently.

▶ **Lemma 17** (see Lemma 4.2 and Claim 7.7 of [11]). *Let $f = \mathrm{sgn}(p)$ be any degree-Δ PTF that is $(\delta, \lambda+1)$-concentrated. Let ρ be a random restriction that fixes any subset of variables according to some $(192 \cdot \Delta \cdot \log(1/\delta))$-wise independent distribution. Then with probability at least $1-\delta$ we have: (1) f_ρ is (δ, λ)-concentrated, and (2) $\mathrm{sgn}(\mathbf{Exp}(p_\rho)) = \mathrm{sgn}(\mathbf{Exp}(p))$.*

The above lemma means that if a PTF is $(\delta, 2)$-concentrated and hence close to some constant. Then the restricted PTF is likely to remain close to the same constant.

Quantified derandomization for low-degree PTF circuits. Let $\mathcal{G}$ be a class of Boolean functions, we say that a circuit C is a $(n, d, w, \Delta, \mathcal{G})$-low-degree PTF circuits if: (1) C is an n-variate circuit of depth-d with at most w wires, and (2) C has degree-Δ PTFs as its gates except for the top gate, which is a function from $\mathcal{G}$.

For a class of Boolean functions $\mathcal{G}$, we denote by $\mathrm{Appr}_{n,\epsilon}(\mathcal{G})$ the running time, given an n-variate function g from $\mathcal{G}$, of approximating the acceptance probability of g to within an additive error ϵ. We can show the following.

▶ **Theorem 18.** *For any constant $E \geq 11$ and any positive integers Δ and d such that $\Delta \ll \sqrt{\varepsilon_d \cdot \log n / \log\log n}$, where $\varepsilon_d = E^{-2(d-1)}$, let $\mathcal{C}$ be the class of $\left(n, d, n^{1+\varepsilon_d}, \Delta, \mathcal{G}\right)$-low-degree PTF circuits. Then the $\left(\mathcal{C}, 2^{n^{1-7/E}}\right)$-quantified derandomization problem can be solved in time $2^{(\log n)^{O\left(\Delta^2\right)}} \cdot \mathrm{Appr}_{n,1/6}(\mathcal{G})$.*

Theorem 18 implies Theorem 3 for low-degree PTF circuits since we can always add a dummy gate (e.g., AND) to the top of a PTF circuits; this only increase the depth by 1.

Theorem 18 is obtained by iteratively applying the pseudorandom restriction lemma (Lemma 16) to reduce the depth of the circuit until the circuit has depth 1. The following lemma shows how to do this in one step.

▶ **Lemma 19.** *For any constants $E \geq 11$, $c > 0$, any $\varepsilon \leq 1/(7E)$, and any positive integer Δ such that $\Delta \ll \sqrt{E \cdot \varepsilon \cdot \log n / \log\log n}$, there is a polynomial time algorithm that, given a $\left(n, d, n^{1+\varepsilon}, \Delta, \mathcal{G}\right)$-low-degree PTF circuit C and a random seed of length $(\log n)^{O\left(\Delta^2\right)}$, outputs the following with probability at least $1 - n^{\varepsilon}$:*

- *A restriction $\rho \in \{0, 1, *\}^n$ that leaves $n' = n^{1-3E\cdot\varepsilon}$ variables unrestricted and that the restricted variables are fixed $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently.*
- *A $\left(n', d-1, (n')^{1+7E\cdot\varepsilon}, \Delta, \mathcal{G}\right)$-sparse PTF circuit $\tilde{C}$ such that for all subsequent random restriction ρ' that fixes the variables in a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent manner, with probability $1 - n^{-c}$ over ρ', it holds that $\tilde{C}_{\rho'}$ is n^{-c}-close to $(C_\rho)_{\rho'}$.*

The proof of Lemma 19 is similar to that in [24], which is based on the argument in [5], but requires some critical modifications. A sketched proof is given in Section D. For the proof of Theorem 18, we refer the reader to the full version.

5 PRG for PTF circuits

In this section, we give a high-level description of our PRG for small PTF circuits. The detailed proof is presented in Section E.

Our PRG is based on the Nisan-Wigderson generator (NW PRG) [19]. To fool a class $\mathcal{C}$ of Boolean functions f, the NW PRG construction requires a "hard function" h that cannot be computed correctly on significantly more than a half of all possible inputs by any Boolean

function g in a related class $\tilde{\mathcal{C}}$ of "slightly more powerful" functions than those from $\mathcal{C}$. Thus, sufficiently strong average-case lower bounds against the class $\tilde{\mathcal{C}}$ can be used to build a PRG fooling the class $\mathcal{C}$. In our case, the class $\mathcal{C}$ contains all those n-variate Boolean functions that are computable by constant depth-d circuits with at most $s \ll n$ PTF gates of degree-Δ. Our main observation is that the corresponding class $\tilde{\mathcal{C}}$ (for which we require average-case lower bounds) is the class of Boolean functions computable by constant depth-d circuits with at most s PTF gates of degree $\Delta' = \alpha \cdot \Delta$, for some parameter $\alpha \geq 1$ that we can control (and which will determine the seed size of our PRG). That is, the class $\tilde{\mathcal{C}}$ is the same as $\mathcal{C}$, except for a somewhat higher degree Δ' of the allowed PTF gates.

To illustrate the idea of our analysis of the NW PRG for PTF circuits, we consider the special case of a single n-variate PTF f of degree Δ. That is, $f = \mathrm{sgn}(p(x_1, \ldots, x_n))$ for some degree-Δ multi-linear polynomial $p\colon \{0,1\}^n \to \mathbb{R}$. Suppose that the NW generator based on some "hard" Boolean function h failed to ϵ-fool this PTF f. First, the standard NW analysis shows that the function $h(z)$ can be computed, with probability at least $1/2 + \epsilon/n$, by (possibly the negation of) the function

$$g(z) = f(h_1(z), h_2(z), \ldots, h_i(z), b_{i+1}, \ldots, b_n), \tag{1}$$

for some $1 \leq i \leq n$, fixed bits $b_{i+1}, \ldots, b_n$, and Boolean functions $h_1, \ldots, h_i$, where each $h_j(z)$ depends on at most some α bits in z, for a parameter $a \geq 1$ coming from the NW construction (the maximum overlap between pairs of sets in the NW design; see Section E for details). It is well known that every Boolean function on α inputs can be written as a multi-linear polynomial of degree α over the reals. Plugging in these polynomials for the function h_j's in Equation (1), we get that $g(z)$ is a PTF of degree at most $\Delta' = \alpha \cdot \Delta$. Hence, to ensure that this NW generator based on h is indeed ϵ-fooling for degree-Δ PTFs, we just need h to be such that no PTF of degree-$(\alpha \cdot \Delta)$ can compute $h(z)$ on more than $1/2 + \epsilon/n$ of inputs z. Such hard functions h turn out to be easy to construct. For example, we use the average-case hard function for low-degree PTF circuits due to Nisan [18].

The parameters of our PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ (its error ϵ and seed length r) depend on the strength of the average-case lower bound for the hard function h. To get a short seed r, one needs to maximize the aforementioned parameter α, ideally setting $\alpha = \log n$ (as is the case for a standard application of the NW construction). However, we also need to prove (average-case) lower bounds against PTFs of degree $\alpha \cdot \Delta$, where virtually nothing is known for the degree $\log n$. Thus we are forced to set $\alpha \ll \log n$, which limits the stretch of our PRG to be at most only super-polynomial. On the other hand, for such a small α, our hard function h has exponentially small correlation with degree-$(\alpha \cdot \Delta)$ PTFs, thereby allowing our PRG to have an exponentially small error ϵ.

6 Open problems

An important open problem is to get a nontrivial Circuit-SAT algorithm for circuits with degree-2 PTF gates. Our algorithm only works for the case where the PTF gates have a sub-quadratic number of monomials, so it does not work for the degree-2 case in general. Such an algorithm is not known even for a single degree-2 PTF. Another interesting open problem is to derandomize our zero-error randomized algorithms to get deterministic #Circuit-SAT algorithms of similar time complexity. Can we get any nontrivial standard derandomization for constant-depth PTF (LTF) circuits of small wire complexity? For PRGs, can we get a nontrivial PRG for depth-2 LTF circuits with a super-linear number of gates?

References

1 Josh Alman, Timothy M. Chan, and R. Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *FOCS*, pages 467–476, 2016.

2 Martin Anthony. *Discrete Mathematics of Neural Networks: Selected Topics*. SIAM monographs on discrete mathematics and applications. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2001. doi:10.1137/1.9780898718539.

3 Timothy M. Chan and Ryan Williams. Deterministic APSP, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. In *SODA*, pages 1246–1255, 2016.

4 Ruiwen Chen and Rahul Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *SAT*, pages 33–45, 2015.

5 Ruiwen Chen, Rahul Santhanam, and Srikanth Srinivasan. Average-case lower bounds and satisfiability algorithms for small threshold circuits. In *CCC*, pages 1:1–1:35, 2016.

6 Shiteng Chen and Periklis A. Papakonstantinou. Depth-reduction for composites. In *FOCS*, pages 99–108, 2016.

7 Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority gates vs. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992.

8 Oded Goldreich and Avi Wigderson. On derandomizing algorithms that err extremely rarely. In *STOC*, pages 109–118, 2014.

9 Johan Håstad. Almost optimal lower bounds for small depth circuits. In S. Micali, editor, *Randomness and Computation*, pages 143–170, Greenwich, Connecticut, 1989. Advances in Computing Research, vol. 5, JAI Press.

10 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *FOCS*, pages 479–488, 2013.

11 Valentine Kabanets, Daniel M. Kane, and Zhenjian Lu. A polynomial restriction lemma with applications. In *STOC*, pages 615–628, 2017.

12 Daniel Kane and Sankeerth Rao. A PRG for Boolean PTF of degree 2 with seed length subpolynomial in ϵ and logarithmic in n. In *CCC*, 2018.

13 Daniel M. Kane. A structure theorem for poorly anticoncentrated gaussian chaoses and applications to the study of polynomial threshold functions. In *FOCS*, pages 91–100, 2012.

14 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *APPROX/RANDOM*, pages 640–651, 2011.

15 Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

16 Raghu Meka and David Zuckerman. Pseudorandom generators for polynomial threshold functions. *SIAM J. Comput.*, 42(3):1275–1301, 2013.

17 Saburo Muroga, Iwao Toda, and Satoru Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271:376–418, 1961.

18 Noam Nisan. The communication complexity of threshold gates. In *Proceedings of Combinatorics, Paul Erdős is Eighty*, pages 301–315, 1994.

19 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

20 Takayuki Sakai, Kazuhisa Seto, Suguru Tamaki, and Junichi Teruyama. Bounded depth circuits with weighted symmetric gates: Satisfiability, lower bounds and compression. In *MFCS*, pages 82:1–82:16, 2016.

21 Suguru Tamaki. A satisfiability algorithm for depth two circuits with a sub-quadratic number of symmetric and threshold gates. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:100, 2016.

22 Roei Tell. Improved bounds for quantified derandomization of constant-depth circuits and polynomials. In *CCC*, pages 13:1–13:48, 2017.

23 Roei Tell. A note on the limitations of two black-box techniques in quantified derandomization. *Electronic Colloquium on Computational Complexity (ECCC)*, 24:187, 2017.

24 Roei Tell. Quantified derandomization of linear threshold circuits. In *STOC*, 2018.

25 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

26 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *STOC*, pages 231–240, 2010.

27 Ryan Williams. Non-uniform ACC circuit lower bounds. In *CCC*, pages 115–125, 2011.

28 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *STOC*, pages 194–202, 2014.

A Enumerating minority-value inputs: proof of Lemma 12

We first need the folowing.

▶ **Proposition 20** (see Section 4.1 of [4]). *Let $\phi_1, \ldots, \phi_s$ be a sequence of terms whose literals are from a set of n variables, where $s \geq n$. There exists a decision tree with at most $2^{n-\Omega(n^2/s)}$ leaves such that restricted to each leaf of the tree, ϕ_i contains at most 1 literal, for all $i \in [s]$.*

Proof of Lemma 12. We view f as an LTF of s AND gates. By Proposition 20, there exists a decision tree for f with at most $2^{n-\Omega(n^2/s)}$ leaves such that Φ restricted to each leaf is an LTF. We then go through each leaf σ and enumerate the set of inputs on which f_σ evaluates to 0. Let S_σ be the size of such set. For an LTF, this enumeration takes time $S_\sigma \cdot \mathsf{poly}(n)$ (see, e.g., Proposition 5.2 of [5]). The total running time is the time for going through the leaves of the decision tree, which is at most $2^{n-\Omega(n^2/s)}$, and the time to enumerate the set of inputs evaluating to 0, which is at most $\sum_\sigma S_\sigma \cdot \mathsf{poly}(n) \leq |S| \cdot \mathsf{poly}(n)$. ◀

B Recursion for depth-d circuits: proof of Lemma 15

Let C be any (d, n, t, s)-skew circuit, where its skew subcircuit is C'. To count the number of satisfying assignments of C. We first apply Lemma 11 to C' to get a decision tree with the claimed property. We then count the number of satisfying assignments at each leaves. For those "bad" leaves for which the conditions in Lemma 11 are not satisfied, we will simply do brute force on all $n^{1-2\beta_d}$ variables. The time to perform this is

$$2^{n-n^{1-2\beta_d}} \cdot \exp(-n^{\varepsilon_d}) \cdot 2^{n^{1-2\beta_d}} \leq 2^{n-n^{\varepsilon_d}}. \tag{2}$$

Next, consider a "good" leaf σ that satisfies the conditions in Lemma 11. We now describe how to count the number of satisfying assignments of C_σ. We call a gate *imbalanced* if it is δ-close to an explicit constant and *balanced* otherwise. Let $(g_1, \ldots, g_{\ell \leq n})$ be the set of imbalanced gates and $(a_1, \ldots, a_\ell)$ be their majority values. Let $(h_1, \ldots, h_{t \leq n^{\beta_d}})$ be the set of balanced gates.

We first count the number of satisfying assignments of C_σ in the following subset of inputs $S = \{x : \exists\, i \in [\ell] \text{ for which } g_i(x) \neq a_i\}$. To do so, for each of the imbalanced gates, we enumerate the set of inputs on which it evaluates to its minority value, and keep those that satisfy the circuit C_σ. By Lemma 12, the running time for this is

$$\mathsf{poly}(n) \cdot \left(2^{n^{1-2\beta_d} - \Omega\left(\frac{n^{2\cdot(1-2\beta_d)}}{s}\right)} + 2^{n^{1-2\beta_d}} \cdot \delta \right), \tag{3}$$

where $\delta = \exp\left(-n^{\Omega\left(\beta_d^3\right)}\right)$. Note that for $s \leq n^{2-5\beta_d}$, we have

$$2^{n^{1-2\beta_d}-\Omega\left(\frac{n^{2\cdot(1-2\beta_d)}}{s}\right)} \leq 2^{n^{1-2\beta_d}-\Omega\left(n^{\beta_d}\right)} \leq 2^{n-n^{\Omega\left(\beta_d^3\right)}},$$

so Equation (3) is at most

$$\mathsf{poly}(n) \cdot 2^{n^{1-2\beta_d}-n^{\Omega\left(\beta_d^3\right)}}. \tag{4}$$

We do this for every imbalanced gate and obtain a set of satisfying inputs. In the end we simply take the union of these sets to get the satisfying assignments in S.

Next, we counts the number of satisfying assignments in $T = \{0,1\}^n - S$. Let $C'_{\sigma,a}$ be the circuit with those imbalanced gates in C'_σ replaced with their majority values (i.e., the values given by $(a_1, \ldots, a_\ell)$). Instead of counting the number of satisfying assignments for the original circuit C_σ, we consider the following circuit:

$$D = C'_{\sigma,a} \wedge \bigwedge_{i\colon a_i=-1} g_i \wedge \bigwedge_{i\colon a_i=1} \neg g_i.$$

It is easy to see that $D(x) = 0$ for every $x \in S$ and $D(x) = C_\sigma(x)$ for every $x \in T$. We now need to count the number of satisfying assignments of D. We first partition T into 2^t subsets, each of which is indexed by some $b = (b_1, \ldots, b_t) \in \{0,1\}^t$, where the subset T_b given by the index b is

$$T_b = \{x : x \in T,\, h_1(x) = b_1, \ldots,\, h_t(x) = b_t\}.$$

To count the number of satisfying assignments of D in T_b. We consider the following circuit:

$$E_b = D_b \wedge \bigwedge_{i\colon b_i=-1} h_i \wedge \bigwedge_{i\colon b_i=1} \neg h_i,$$

where D_b is the circuit D with the balanced gates replaced by the values $b_1, \ldots, b_t \in \{0,1\}$. Again, we have $E_b(x) = 0$ for every $x \in [n] - T_b$ and $E_b(x) = D(x)$ for every $x \in T_b$. Now our task is reduced to counting the number of satisfying assignments of E_b for each $b \in \{0,1\}^t$. But note that each E_b is a conjunction of some depth-$(d-1)$ circuit (i.e., the skew subcircuit of E_b) and k s-sparse PTFs, where $k = t + n + n^\beta \leq t + 2n$. Also, the skew subcircuit has at most $n^{1+\varepsilon_d}$ wires, and we have

$$n^{1+\varepsilon_d} \leq \left(n^{1-2\beta_d}\right)^{1+\varepsilon_{d-1}}.$$

Therefore, each E_b is a $\left(d-1, n^{1-2\beta_d}, t+2n, s\right)$-skew circuits, and its number of satisfying assignments can be computed in time $\mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right)$. Then the total time for counting the number of satisfying assignments of the original circuit C_σ in the subset T is

$$2^t \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right) \leq 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right). \tag{5}$$

Therefore, by Equation (4) and Equation (5), counting the number of satisfying assignments of C_σ can be done in time

$$\mathsf{poly}(n) \cdot 2^{n^{1-2\beta_d}-n^{\Omega\left(\beta_d^3\right)}} + 2^{n^{\beta_d}} \cdot \mathcal{T}\left(d-1, n^{1-2\beta_d}, t+2n, s\right). \tag{6}$$

There are at most $L = 2^{n-n^{1-2\beta_d}}$ such leaves. Multiplying L by the running time in Equation (6) and combining Equation (2) yields the desired running time.

C Pseudorandom restriction lemma for PTFs: proof of Lemma 16

We define $\mathcal{R}$ by describing the following process of sampling a random restriction from $\mathcal{R}$:

1. Hash the variables into n^α blocks $(\log n)$-wise independently.
2. Apply a $\left(n^{\alpha/2}\right)$-block pseudorandom restriction from Lemma 9 for degree-Δ PTFs, with parameters
   - $\delta = n^{-c}$ and $\lambda = 3$.
   - $\gamma = (c_1 \cdot \Delta^2 \cdot \log\log n)/(\alpha \cdot \log n)$, where c_1 is a sufficiently large constant (note that $\gamma < 1$ for $\Delta \ll \sqrt{\alpha \cdot \log n / \log\log n}$).

We now argue that the random restriction $\mathcal{R}$ has the desired properties. For the first item, it is easy to see from the above that $\mathcal{R}$ picks the set of unrestricted variables $(\log n)$-wise independently, each with probability $1/n^{\alpha/2}$. The second item follows from Lemma 9 that the pseudorandom block restriction fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. For the third item, note that to sample from $\mathcal{R}$, we need $\mathsf{polylog}(n)$ random bits for its first step, and the number of random bits for its second step is

$$n^{\alpha\cdot\gamma} \cdot \log n \leq (\log n)^{O(\Delta^2)}.$$

Finally, for the last item, note that in the above process of sampling $\mathcal{R}$, for any partition into n^α blocks generated in the first step, by Lemma 9, the probability over the restrictions in the second step that the restricted PTF is not $(n^{-c}, 3)$-concentrated is at most $n^{-\alpha/2} \cdot (\log n)^{O(\Delta^2/\gamma)}$. Thus,

$$\mathbf{Pr}_\rho[f_\rho \text{ is not } (n^{-c}, 3)\text{-concentrated}] \leq n^{-\alpha/2} \cdot (\log n)^{O(\Delta^2/\gamma)} \leq n^{-\alpha/2} \cdot n^{\alpha/6} \leq n^{-\alpha/3}.$$

D Depth reduction via pseudorandom restrictions: proof of Lemma 19

Let $\beta = E \cdot \varepsilon$ and $p = n^{-\beta}$. The restriction ρ consists of three sub-restrictions.

ρ_1: Preprocessing. Fix each of the variables with fan-out greater than $2n^\varepsilon$ using a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent distribution. Since the number of wires is at most $n^{1+\varepsilon}$, it can be easily seen that the number of variables needed to be fixed is at most $n^{1+\varepsilon}/(2n^\varepsilon) = n/2$.

ρ_2: Pseudorandom restriction to simplify PTFs. Let ρ_2 be a random restriction from Lemma 16 with parameters $\alpha_2 = \beta$ and $c_2 = 2c$. Note that ρ_2 fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. Now by Lemma 16, after ρ_2, we expect all but at most a fraction of $n^{-\beta/5}$ of the gates in the bottom layer to become $(n^{-2c}, 3)$-concentrated. Moreover, since the number of unrestricted variables is picked in a $(\log n)$-wise independent manner, by a Chernoff-type concentration bound (for k-wise independence), the fan-in of each of the non-concentrated gates (there are only about a fraction of $n^{-\beta/5}$ of such gates) will shrink by a factor of p with high probability, assuming they have large fan-ins. Then we can expect to eliminate all those non-concentrated gates by fixing a small number of variables. As for the gates with small fan-ins, using a simple graph theoretic argument along with the condition given by the preprocessing step, we can also eliminate those gate by fixing a few variables. More precisely, as in [5, 24], it can be shown that with probability except $O\left(n^{-\beta/10}\right)$, over the random restriction ρ_2, the following holds: there is a set T of variables such that all the bottom layer gates that are not $(n^{-2c}, 3)$-concentrated can be replaced by constants after fixing the variables in T. The number of unrestricted variables after applying ρ_2 and fixing T is at least $n^{1-3E\cdot\varepsilon}$.

ρ_3: Eliminate non-concentrated gates. We will use a $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independent distribution to fix the variable in the set T described above. Note that the number of unrestricted variables is at least $n' = n^{1-3E\cdot\varepsilon}$. We may further fix additional variables so that the number of unrestricted variables is exactly n'. Although this restriction eliminates all non-concentrated gates in the bottom layer, it may also cause some concentrated gates to become non-concentrated. However, by Lemma 17, the probability that each of these gate is not $(n^{-2c}, 2)$-concentrated is at most n^{-2c}. By the union bound, we get with probability all but $n^{-2c} \cdot n^{1+\varepsilon} \leq n^{-c}$, all these gates remain $(n^{-2c}, 2)$-concentrated.

Obtaining $\tilde{C}$. By above, with probability at least $1 - O\left(n^{-\beta/10}\right)$, we have a restriction ρ such that all the bottom layer gates of C_ρ are $(n^{-2c}, 2)$-concentrated and hence close to some associated constants. Let's call these constants V. $\tilde{C}$ is the circuit obtained from C_ρ by replacing those concentrated gates in the bottom with the constants V. Let's argue that $\tilde{C}_{\rho'}$ and $(C_\rho)_{\rho'}$ are n^{-c}-close to each other for any subsequent random restriction ρ' that fixes the variables $(600 \cdot c \cdot \Delta \cdot \log n)$-wise independently. Consider such a subsequent random restriction ρ' and the restricted circuit $(C_\rho)_{\rho'}$. By Lemma 17, with probability except n^{-2c}, the bottom layer gates of $(C_\rho)_{\rho'}$, which are just the bottom layer gates of C_ρ, are still (n^{-2c})-concentrated. Moreover, they are close to the same constants V. Now by replacing these gates in $(C_\rho)_{\rho'}$ with the constants V, we obtain a circuit C'. By a union bound, C' and $(C_\rho)_{\rho'}$ are (n^{-c})-close to each other. On the other hand, consider the circuit $\tilde{C}$, which is obtained by replacing the concentrated gates in the bottom C_ρ with the constant V. Note that $\tilde{C}_{\rho'} = C'$. Thus, $\tilde{C}_{\rho'}$ and $(C_\rho)_{\rho'}$ are n^{-c}-close to each other. Finally, we need to show that $\tilde{C}$ is a $\left(n', d-1, (n')^{1+4E\cdot\varepsilon}, (n')^{\Delta\cdot E\cdot\varepsilon}, \mathcal{G}\right)$-sparse PTF circuit. As for the number of wires in $\tilde{C}$, note that

$$(n')^{1+7E\cdot\varepsilon} = n^{(1-3E\cdot\varepsilon)\cdot(1+7E\cdot\varepsilon)} \geq n^{1+\varepsilon}.$$

Also, we have $(n')^{2\Delta\cdot\varepsilon} = n^{(1-3E\cdot\varepsilon)\cdot 2\Delta\cdot\varepsilon} \geq n^{\Delta\cdot\varepsilon}$.

E PRG for PTF circuits: proof of Theorem 4

In this section, we present our NW-style PRG for low-degree PTF circuits with few gates.

▶ **Theorem 21.** *There exists a constant $E > 0$ such that for any positive integers α, Δ and any degree-Δ PTF circuit C on n variables with at most $s = n^{\frac{1}{\alpha+1}} \cdot \left(E \cdot 5^{\alpha\cdot\Delta} \cdot \log^2(n) \cdot \log(n/\epsilon)\right)^{-1}$ gates, there exists a* $\mathsf{poly}(n)$*-time computable PRG $G\colon \{0,1\}^r \to \{0,1\}^n$ ϵ-fooling C, with the seed length $r = n^{2/(\alpha+1)}$.*

We first need a (average-case) hard function for such circuits.

▶ **Theorem 22** ([18])**.** *There exists a constant $E > 0$ such that for any degree $\Delta \geq 1$, there exists a polynomial-time computable function $f\colon \{0,1\}^n \to \{0,1\}$ such that for any error parameter ϵ and any n-variate degree-Δ PTF circuit C with at most $n \cdot \left(E \cdot 5^{\Delta} \cdot \log^2(n) \cdot \log(1/\epsilon)\right)^{-1}$ gates, we have*

$$\mathbf{Pr}_{x\sim\{0,1\}^n}\left[C(x) = f(x)\right] \leq \frac{1}{2} + \epsilon.$$

Next we apply the Nisan-Wigderson construction to the hard function of Theorem 22. We will use the following (standard) combinatorial designs.

▶ **Claim 23** (NW Designs [19])**.** *For any positive integers n, α, there exists an efficiently computable family of sets $S_1, \ldots, S_n$ such that*

- $S_i \subset [r]$, $\forall i \in [n]$, *where* $r = n^{2/(\alpha+1)}$,
- $|S_i| = \ell = n^{1/(\alpha+1)}$, $\forall i \in [n]$, *and*
- $|S_i \cap S_j| \leq \alpha$, $\forall i, j \in [n]$ *such that* $i \neq j$.

Proof Theorem 21. For $\ell = n^{1/(\alpha+1)}$, let $f\colon \{0,1\}^\ell \to \{0,1\}$ be the hard function for degree-$(\alpha \cdot \Delta)$ PTF circuits from Theorem 22. By Theorem 22 and assuming E is a sufficiently large constant, we have that for any degree-$(\alpha \cdot \Delta)$ PTF circuit D on ℓ variables of size at most s,

$$\mathbf{Pr}_{z \sim \{0,1\}^\ell}[D(z) = f(z)] \leq \frac{1}{2} + \epsilon/n. \tag{7}$$

Let $S_1, \ldots, S_n$ be the sets from Claim 23. Define the generator $G_{\alpha,\Delta}\colon \{0,1\}^r \to \{0,1\}^n$ as follows:

$$G_{\alpha,\Delta}(y) = f(y|_{S_1}), \ldots, f(y|_{S_n}),$$

where, for $i \in [n]$, $y|_{S_i}$ denotes the substring of y indexed by the set S_i.

Toward a contradiction, suppose

$$|\mathbf{Pr}_{x \sim \{0,1\}^n}[C(x) = 1] - \mathbf{Pr}_{y \sim \{0,1\}^r}[C(G_{\alpha,\Delta}(y)) = 1]| > \epsilon. \tag{8}$$

By a standard argument via "reduction from distinguishing to predicting" as in [19], Equation (8) implies that there exist an $i \in [n]$, and bits $b_{i+1}, \ldots, b_n \in \{0,1\}$, such that

$$\mathbf{Pr}_{z \sim \{0,1\}^\ell}[C'(h_1(z), \ldots, h_i(z), b_{i+1}, \ldots, b_n) = f(z)] > 1/2 + \epsilon/n, \tag{9}$$

where

- $C' = C$ or $C' = \neg C$, and
- $h_1, \ldots, h_i$ are Boolean functions such that each depends on at most α bits of its input z.

First, note that each gate in C' is always a PTF of degree at most Δ. Next, observe that every Boolean function that depends on at most α variables can be computed by a multi-linear polynomial of degree at most α over the reals. Replacing our functions $h_1, \ldots, h_i$ with such degree α polynomials $p_1, \ldots, p_i$ inside C', we get

$$C'(p_1(z), \ldots, p_i(z), b_{i+1}, \ldots, b_n).$$

Now we can we merge the polynomials p_i's into every PTF gate in the circuit that reads from them. This yields a new circuit with *exactly the same* number of gates, and of degree at most $\alpha \cdot \Delta$. Denote this new circuit by C''. Note that C'' is a degree-$(\alpha \cdot \Delta)$ PTF circuit on ℓ variables of size at most s. By Equation (9), this PTF circuit C'' computes the function f with probability greater than $1/2 + \epsilon/n$, contradicting Equation (7). ◀

Then the PRG in Corollary 5 for PTFs can be obtained from the result in Theorem 21 by picking

$$\alpha = \frac{2 \log n}{L \cdot (\sqrt{\Delta \cdot \log n + 2 \log \log(1/\epsilon)})} - 1 \leq \frac{2}{L} \cdot \sqrt{\log n / \Delta},$$

where L is a sufficiently large constant. For this value of α, we get that the PRG in Theorem 21 fools any degree-Δ PTF circuit of size at least 1, and has seed length at most $\exp\left(O\left(\sqrt{\Delta \cdot \log n}\right)\right) \cdot \log^2(1/\epsilon)$.