Electronic Colloquium on Computational Complexity, Report No. 99 (2016)

Bounded Depth Circuits with Weighted Symmetric Gates: Satisfiability, Lower Bounds and Compression[*]

Takayuki Sakai[†] Kazuhisa Seto[‡] Suguru Tamaki[§] Junichi Teruyama[¶]

Abstract

A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is *weighted symmetric* if there exist a function $g : \mathbb{Z} \to \{0,1\}$ and integers $w_0, w_1, \ldots, w_n$ such that $f(x_1, \ldots, x_n) = g(w_0 + \sum_{i=1}^{n} w_i x_i)$ holds.

In this paper, we present algorithms for the circuit satisfiability problem of bounded depth circuits with AND, OR, NOT gates and a limited number of weighted symmetric gates. Our algorithms run in time super-polynomially faster than 2^n even when the number of gates is super-polynomial and the maximum weight of symmetric gates is nearly exponential. With an additional trick, we give an algorithm for the maximum satisfiability problem that runs in time $\mathrm{poly}(n^t) \cdot 2^{n-n^{1/O(t)}}$ for instances with n variables, $O(n^t)$ clauses and *arbitrary* weights. To the best of our knowledge, this is the first moderately exponential time algorithm even for Max 2SAT instances with arbitrary weights.

Through the analysis of our algorithms, we obtain average-case lower bounds and compression algorithms for such circuits and worst-case lower bounds for majority votes of such circuits, where all the lower bounds are against the generalized Andreev function. Our average-case lower bounds might be of independent interest in the sense that previous ones for similar circuits with arbitrary symmetric gates rely on communication complexity lower bounds while ours are based on the restriction method.

1 Introduction

We are concerned with bounded depth circuits with AND, OR, NOT and (weighted) symmetric gates. Let $\mathbb{Z}$ be the set of integers and $x_1, x_2 \ldots, x_n$ be Boolean variables. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is *weighted symmetric* if there exist a function $g : \mathbb{Z} \to \{0,1\}$ and integers $w_0, w_1, \ldots, w_n$ such that $f(x_1, \ldots, x_n) = g(w_0 + \sum_{i=1}^{n} w_i x_i)$ holds. If $w_1 = w_2 = \cdots = w_n = 1$ holds, then f is *symmetric*.

For example, if we set $g(z) = \mathrm{sgn}(z)$, where $\mathrm{sgn}(z) = 1$ if and only if $z \geq 0$, we obtain *majority* functions as symmetric functions and *linear threshold* functions as weighted symmetric functions. If we define $g(z) = 1$ if and only if $z \equiv 0 \bmod m$ for an integer $m \geq 2$, then we obtain *modulo m* functions as symmetric functions.

A (weighted) symmetric gate is a logic gate that computes a (weighted) symmetric function. We denote by $\mathbf{SYM}_w$ the set of weighted symmetric gates such that $\max_i |w_i| \leq w$ holds. When we consider satisfiability and compression algorithms, we assume that $g(z)$ can be evaluated in time polynomial in $\log_2 |z|$, where $|z|$ denotes the absolute value of z. When we consider circuit lower bounds, we assume that g is computable, i.e., there exists a Turing machine that computes g.

[*] Subsumes the technical report [51].

[†] Oki Electric Industry Co., Ltd.

[‡] Seikei University seto@st.seikei.ac.jp. Supported in part by MEXT KAKENHI 24106003); JSPS KAKENHI (26730007).

[§] Kyoto University tamak@kuis.kyoto-u.ac.jp. Supported in part by MEXT KAKENHI (24106003); JSPS KAKENHI (25240002, 26330011); the John Mung Advanced Program of Kyoto University. Part of the work performed while the author was at Department of Computer Science and Engineering, University of California, San Diego and the Simons Institute for the Theory of Computing, Berkeley.

[¶] National Institute of Informatics, and JST, ERATO, Kawarabayashi Large Graph Project teruyama@nii.ac.jp.

ISSN 1433-8092

1.1 Our contribution

Satisfiability Algorithms: In the *circuit satisfiability problem* (Circuit SAT), our task is, given a Boolean circuit C, to decide whether there exists a 0/1 assignment to the input variables such that C evaluates 1. If input instances are restricted to a class of Boolean circuits $\mathscr{C}$, the problem is called $\mathscr{C}$-SAT. A naïve algorithm can solve Circuit SAT in time $O(\mathrm{poly}(|C|)\cdot 2^n)$, where we denote by $|C|$ the size of C and by n the number of input variables of C respectively. We say an algorithm for $\mathscr{C}$-SAT is *moderately exponential time* if it checks the satisfiability of every $C \in \mathscr{C}$ in time $\mathrm{poly}(|C|)\cdot 2^{n-\omega(\log n)}$, i.e., super-polynomially faster than 2^n. We are interested in for which class $\mathscr{C}$ moderately exponential time satisfiability algorithms exist.

Let $\mathbf{SYM}_w \circ \mathbf{AND}(n,m)$ be the set of n-variate depth 2 circuits with a weighted symmetric gate in $\mathbf{SYM}_w$ at the top and at most m AND gates at the bottom. Let $\mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ be the set of n-variate unbounded fan-in depth $d+1$ layered circuits with AND, OR, NOT gates and a weighted symmetric gate in $\mathbf{SYM}_w$ such that the top gate is the weighted symmetric gate and each layer contains at most m gates. Let $\mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ be the set of n-variate unbounded fan-in depth d layered circuits with AND, OR, NOT gates and at most t weighted symmetric gates in $\mathbf{SYM}_w$ such that each layer contains at most m gates.

In this paper, we show moderately exponential time algorithms for the counting version of $\mathscr{C}$-SAT, where $\mathscr{C} \in \{\mathbf{SYM}_w \circ \mathbf{AND}(n,m), \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m), \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)\}$, as follows.

Theorem 1.1 (depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *We can count the number of satisfying assignments for $C \in \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$ deterministically in time*

$$\mathrm{poly}(n,m,\log w)\cdot 2^{n-\Omega\left((n/\log(mw))^{\log n/4\log(nm)}\right)}$$

and exponential space.

The running time is super-polynomially faster than 2^n when, e.g., $m = n^{o(\log n/\log\log n)}$ and $w = 2^{n^{0.99}}$. Note that $\mathbf{SYM}_{2^n}$ contains all Boolean functions (if we ignore the assumption that $g(z)$ can be evaluated in time polynomial in $\log_2|z|$). The heart of our algorithms is a (seemingly new) *bottom fan-in reduction* technique inspired by recent developments on the analysis of "greedy restriction" by "concentrated shrinkage" [52, 55, 17, 50]. With an additional trick, we give an algorithm for the maximum satisfiability problem that runs in time $\mathrm{poly}(n^t)\cdot 2^{n-n^{1/O(t)}}$ for instances with n variables, $O(n^t)$ clauses and *arbitrary* weights. To the best of our knowledge, this is the first moderately exponential time algorithm even for Max 2SAT instances with arbitrary weights.

We extend the above algorithm with the help of the depth reduction algorithm due to Beame, Impagliazzo and Srinivasan [7].

Theorem 1.2 (depth d, weighted symmetric gate only at the top)**.** *We can count the number of satisfying assignments for $C \in \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ deterministically in time*

$$\mathrm{poly}(n,m,\log w)\cdot 2^{n-\Omega\left((n/2^{2d(\log m)^{4/5}}\log(mw))^{\log n/9\log m}\right)}$$

and exponential space.

The running time is super-polynomially faster than 2^n when, e.g., $m = 2^{(\log n/4d)^{5/4}}$ and $w = 2^{n^{0.49}}$.

We further extend the above algorithm relying on the circuit transformation techniques due to Beigel, Reingold and Spielman [9] and Beigel [8].

Theorem 1.3 (depth d, $t(n)$ weighted symmetric gates)**.** *We can count the number of satisfying assignments for $C \in \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ deterministically in time*

$$\mathrm{poly}(n,m,d,t,\log w)\cdot 2^{n+O(t\log mw)-\Omega\left((n/2^{4d(\log m)^{4/5}}t\log(mw))^{\frac{\log n}{18\log m}}\right)}$$

and exponential space.

The running time is super-polynomially faster than 2^n when, e.g., $m = n^c$, $w = 2^{n^{\frac{1}{40c}}}$ and $t = n^{\frac{1}{40c}}$, where $c \leq \frac{\log^{1/4} n}{2(4d)^{5/4}}$.

Although our algorithms run in time super-polynomially faster than 2^n instead of exponentially faster than 2^n ($2^{(1-\varepsilon)n}$ for a universal constant $\varepsilon > 0$), this seems unavoidable due to the Strong Exponential Time Hypothesis (SETH) [12, 32, 34]: The hypothesis states that for all k, there exists $\varepsilon_k > 0$ such that the satisfiability problem of k-CNF formulas cannot be solved in time $2^{(1-\varepsilon_k)n}$. SETH has been used in proving conditional time lower bounds for several exponential time and polynomial time algorithms, see, e.g., [21, 38, 41].

Circuit Lower Bounds: Through the analysis of our satisfiability algorithms, we obtain the following average-case lower bounds.

Theorem 1.4 (depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *There exists a constant $\alpha > 0$ such that for every m, w and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w}$ such that for every $C \in \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$, it holds that*

$$\Pr_{x \in \{0,1\}^n}[f(x) = C(x)] \leq \frac{1}{2} + 2^{-\Omega\left((n/\log(mw))^{\alpha \log n/\log(nm)}\right)}.$$

We also obtain similar average-case lower bounds for $\mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ and $\mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$, see Theorems 5.2 and 5.3 in Section 5.

Our average-case lower bounds might be interesting in the sense that (1) previous ones for similar circuits with arbitrary symmetric gates rely on communication complexity lower bounds while ours are based on the restriction method and (2) we are not aware of (even worst-case) lower bounds for $\mathbf{SYM}_w \circ \mathbf{AND}$ with $w = n^{\omega(\log n)}$.

Let $\mathscr{C}$ be a set of Boolean circuits and $\mathbf{MAJ} \circ \mathscr{C}$ be the set of Boolean circuits, where $C \in \mathbf{MAJ} \circ \mathscr{C}$ is a majority vote of $\mathscr{C}$ circuits, i.e., $C(x) = \mathrm{sgn}(C_1(x) + \cdots + C_s(x) + w_0)$ holds for some $C_1, \ldots, C_s \in \mathscr{C}$ and an integer w_0.

Combining the above average-case lower bounds and the discriminator lemma due to Hajnal, Maass, Pudlák, Szegedy and Turán [27], we obtain the following worst-case lower bounds.

Theorem 1.5 (majority vote of depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *There exists a constant $\alpha > 0$ such that for every m, w and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w}$ such that any $C \in \mathbf{MAJ} \circ \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$ cannot compute $f_{n,m,w}$ if the majority gate at the top of C has fan-in at most $2^{o\left((n/\log(mw))^{\alpha \log n/\log(nm)}\right)}$.*

We also obtain similar worst-case lower bounds for $\mathbf{MAJ} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$, $\mathbf{MAJ} \circ \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ (and $\mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ with different parameters), see Theorems 6.2, 6.3 and 6.4 in Section 6.

Compression Algorithms: In the *circuit compression problem* (Circuit CMP), our task is, given the truth table of an s-sized Boolean circuit C and an integer $s' \geq s$, to construct a Boolean circuit C' that is at most s'-sized and computes the same function as C. If input instances are restricted to a class of Boolean circuits $\mathscr{C}$, the problem is called $\mathscr{C}$-CMP. In $\mathscr{C}$-CMP, we do not have to construct C' as a circuit in $\mathscr{C}$. Since every n-variate Boolean function can be represented as a $\frac{(1+o(1))2^n}{n}$-sized circuit [40][1], the problem is interesting if $s' \ll 2^n/n$ and in particular we consider the case $s' = 2^{n-\omega(\log n)}$.

A compression algorithm is *efficient* if it runs in time $2^{O(n)}$ given the truth table of an n-variate Boolean function. Note that input length is 2^n and an efficient algorithm runs in polynomial time. The running time analyses of our satisfiability algorithms imply efficient compression algorithms. Let $\mathscr{C} \in \{\mathbf{SYM}_w \circ \mathbf{AND}(n,m), \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m), \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)\}$. We obtain deterministic efficient algorithms for $\mathscr{C}$-CMP if parameters n, m, w, d, t are such that the corresponding algorithms for $\mathscr{C}$-SAT run in time $2^{n-\omega(\log n)}$.

[1] Such a representation can be obtained in time $2^{O(n)}$.

1.2 Background

Bounded Depth Circuits with (Weighted) Symmetric Gates: Let $\mathbf{AC}^0$ be the set of bounded depth circuits with AND, OR and NOT gates, $\mathbf{AC}^0[m]$ be the set of $\mathbf{AC}^0$ circuits with modulo m gates, $\mathbf{AC}^0[\mathbf{MAJ}]$ be the set of $\mathbf{AC}^0$ circuits with majority gates (also known as $\mathbf{TC}^0$), $\mathbf{AC}^0[\mathbf{THR}]$ be the set of $\mathbf{AC}^0$ circuits with linear threshold gates and $\mathbf{AC}^0[\mathbf{SYM}_w]$ be the set of $\mathbf{AC}^0$ circuits with gates in $\mathbf{SYM}_w$. Note that for every linear threshold gate, there exists a polynomial size depth 2 majority circuit that computes it [24].

In their seminal work, Razborov [47] and Smolensky [56] showed exponential lower bounds on the size of $\mathbf{AC}^0[m]$ circuits computing majority or mod q functions when m,q are prime powers and relatively prime. Since then, people have been trying to obtain super-polynomial size lower bounds against stronger circuit classes such as $\mathbf{AC}^0[m]$ with arbitrary m or $\mathbf{AC}^0[\mathbf{MAJ}]$. Despite much effort of researchers, super-polynomial size lower bounds have been only shown for such circuit classes with some restriction, see, e.g., [4, 9, 14, 22, 23, 26, 27, 28] (here we consider circuits computing "explicit" Boolean functions, i.e., functions in NP).

One of the best studied restriction is limiting the number of (weighted) symmetric gates. The following lower bounds are known:

- (Worst-case lower bounds) Exponential lower bounds for $\mathbf{AC}^0[\mathbf{MAJ}]$ circuits with $n^{o(1)}$ majority gates [6, 8] and $\mathbf{AC}^0[\mathbf{THR}]$ circuits with $o(\log n)$ linear threshold gates [45].
- (Average-case lower bounds) super-polynomial lower bounds for $\mathbf{AC}^0[\mathbf{SYM}_1]$ circuits with $o(\log^2 n)$ symmetric gates [59]; arbitrary large polynomial lower bounds for $\mathbf{AC}^0[\mathbf{SYM}_1]$ circuits with $n^{1-o(1)}$ symmetric gates and $\mathbf{AC}^0[\mathbf{THR}]$ circuits with $n^{1/2-o(1)}$ linear threshold gates [39].

The above average-case lower bounds are based on the results of Håstad and Goldmann [29] and Razborov and Wigderson [49] that show average-case lower bounds for $\mathbf{SYM}_1 \circ \mathbf{AND}$ circuits from the communication complexity lower bounds due to Babai, Nisan and Szegedy [5] and also show worst-case lower bounds for $\mathbf{MAJ} \circ \mathbf{SYM}_1 \circ \mathbf{AND}$ circuits using the discriminator lemma.

Circuit Satisfiability: Studying moderately exponential time algorithms for Circuit SAT is motivated by not only the importance in practice, e.g., logic circuit design and constraint satisfaction but also the viewpoint of Boolean circuit complexity. As pointed out by several papers such as [61, 66], there are strong connections between proving circuit lower bounds for $\mathscr{C}$ and designing moderately exponential time algorithms for $\mathscr{C}$-SAT; see also excellent surveys [53, 44, 63]. Typical such connections are:

(1) Some proof techniques such as deterministic/random restriction (shrinkage analysis/switching lemma) simultaneously prove circuit lower bounds for $\mathscr{C}$ and provides $\mathscr{C}$-SAT algorithms [52, 31, 7, 55, 17, 16, 15, 20, 25].

(2) Williams [61, 65] showed that if we obtain a moderately exponential time algorithm for $\mathscr{C}$-SAT and $\mathscr{C}$ satisfies some closure property, then we also have a separation of complexity classes such as $\mathrm{E}^{\mathrm{NP}} \not\subseteq \mathscr{C}$ or $\mathrm{NE} \not\subseteq \mathscr{C}$, where E^{NP} is the set of languages decidable by exponential time Turing machines with NP oracles and NE is the set of languages decidable by non-deterministic exponential time Turing machines; see also [60, 62, 64, 10, 35] for the improvement of such connections. Since then, people have developed moderately exponential time satisfiability algorithms for various circuit classes [33, 18, 30, 1, 3, 2, 43, 19, 58]. In particular, one of the current best lower bounds, $\mathrm{NE} \not\subseteq \mathbf{ACC}^0 \circ \mathbf{THR}$ (also $\mathrm{NE} \not\subseteq \mathbf{ACC}^0 \circ \mathbf{SYM}_1$), was obtained through satisfiability algorithms [64], where $\mathbf{ACC}^0 := \bigcup_m \mathbf{AC}^0[m]$.

Circuit Compression: Circuit CMP is a relaxed version of the circuit minimization problem. Chen, Kabanets, Kolokolova, Shaltiel and Zuckerman [17] established a connection between compression algorithms and circuit lower bounds as follows: If there exists a deterministic efficient algorithm for $\mathscr{C}$-CMP, then $\mathrm{NEXP} \not\subseteq \mathscr{C}$. They also gave efficient compression algorithms for $\mathbf{AC}^0$ circuits, Boolean formulas and branching programs of certain size range. Srinivasan [57] showed an efficient compression algorithm for $\mathbf{AC}^0[m]$ with a prime power m. Carmosino, Impagliazzo, Kabanets and Kolokolova [13] established interesting connections between the tasks of compression/learning and "natural properties" in the sense of Razborov and Rudich [48].

2 Preliminaries

We use random access machines as our computation model. For a set S, we denote by $|S|$ the cardinality of S.

A *literal* is either a Boolean variable or its negation. A *term* is a conjunction of literals. A *Boolean circuit* is a directed acyclic graph whose source nodes are labeled by literals or constants and internal and sink nodes are labeled by logic gates such as AND, OR, NOT, or weighted symmetric gates. A Boolean circuit with a single sink node computes a Boolean function in a natural way. We call source nodes and a sink node *input nodes* and *output node* respectively. The *depth* of a node is defined as the length of the longest path from it to the output node. The *depth* of a Boolean circuit is the maximum value of the depth over all nodes. A Boolean circuit is *layered* if for every edge (u,v), u and v have depth d and $d+1$ for some d.

A Boolean circuit $C : \{0,1\}^n \to \{0,1\}$ is *satisfiable* if there exists a *satisfying assignment* for C, i.e., an assignment $a \in \{0,1\}^n$ such that $C(a) = 1$ holds. For two Boolean functions (or circuits) f,g in the same variables, we write $f \equiv g$ if $f(a) = g(a)$ holds for all $a \in \{0,1\}^n$. A Boolean function $f : \{0,1\}^n \to \{0,1\}$ is *k-junta* if it depends on at most k variables, i.e., there exist $g : \{0,1\}^k \to \{0,1\}$ and $1 \le i_1 < \cdots < i_k \le n$ such that $f(x_1,\dots,x_n) = g(x_{i_1},\dots,x_{i_k})$ holds.

Let $V = \{x_1,\dots,x_n\}$. A *restriction* is a mapping $\rho : V \to \{0,1,*\}$. The meaning of ρ is that if $\rho(x_i) \in \{0,1\}$, then we assign the value $\rho(x_i)$ to x_i, and if $\rho(x_i) = *$, then we leave x_i as it is. Thus, when we *apply* a restriction ρ to a Boolean function f, we obtain the Boolean function $f|_\rho$ defined over the variables $\rho^{-1}(*)$. We also apply a restriction ρ to a Boolean circuit C and obtain a Boolean circuit $C|_\rho$. When we apply a restriction ρ to a Boolean circuit C, we *simplify* a Boolean circuit C using the identities $0 \wedge f \equiv 0$, $1 \wedge f \equiv f$ repeatedly (each appearance of L.H.S. is replaced by R.H.S.).

A *restriction decision tree* T over $x_1,\dots,x_n$ is an ordinary decision tree except that leaves are not necessarily labeled by 0 or 1. The *height* of T is defined as the number of nodes on the longest path from the root to a leaf and the *size* of T is defined as the number of nodes in T. We identify a path from the root to a leaf with a restriction. A *random root-to-leaf path* is sampled by repeatedly selecting a child of the current node uniformly at random from the root. Note that a path of length ℓ is chosen with probability $2^{-\ell}$.

3 A Dynamic Programming Algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k$

We denote by $g \circ \mathbf{AND}_k(n,m,w)$ the set of n-variate Boolean circuits of the form $g(w_0 + \sum_{i=1}^{s} w_i t_i)$, where $g : \mathbb{Z} \to \{0,1\}$, $s \le m$, $w_0, w_1, \dots, w_s \in \mathbb{Z}, \max_{0\le i\le s} |w_i| \le w$, and $t_1,\dots,t_s$ are terms that contain at most k-literals such that $t_i \ne t_j$ holds for $i \ne j$. We define

$$\mathbf{SYM}_w \circ \mathbf{AND}_k(n,m) := \bigcup_{g:\mathbb{Z}\to\{0,1\}} g \circ \mathbf{AND}_k(n,m,w).$$

We specify an element C in $\mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$ as $C = \{g, w_0, (t_1,w_1), \dots, (t_s,w_s)\}$ and call s and $\max_{0\le i\le s} |w_i|$ the *size* and the *maximum weight* of C respectively.

For a restriction ρ, we simplify $C|_\rho = \{g, w_0, (t_1|_\rho, w_1), \dots, (t_s|_\rho, w_s)\}$ repeatedly if there exists a pair (i,j), $1 \le i < j \le s$ such that $t_i|_\rho \equiv t_j|_\rho$ holds. That is, we delete $(t_j|_\rho, w_j)$ and replace $(t_i|_\rho, w_i)$ by $(t_i|_\rho, w_i + w_j)$. If there are multiple such pairs, we may handle them in arbitrary order.

Our first satisfiability algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$ is described in Fig. 1. The algorithm involves two parameters n', m' that are specified in the proof of Theorem 3.1. The basic idea is as follows:
(Step 1) We construct a table T that contains pairs of the form $(C, \#\mathrm{sat}(C))$ for every circuit C in $g \circ \mathbf{AND}_k(n', m', w')$, where $\#\mathrm{sat}(C)$ denotes the number of satisfying assignments for C and n', m', w' are appropriately chosen parameters. Furthermore, pairs are sorted in the lexicographical order with respect to the first coordinate C so that we can use binary search. To do so, we check the number of satisfying assignments for every circuit in $g \circ \mathbf{AND}_k(n', m', w')$ one by one in the lexicographical order using brute force search.
(Step 2) Let C be an input instance in $g \circ \mathbf{AND}_k(n,m,w)$. For each restriction ρ that assigns $*$ to the first n' variables of C, we check the number of satisfying assignments for $C|_\rho$ using binary search in T and output the sum of them.

```
Algorithm1($C = \{g, w_0, (t_1, w_1), \ldots, (t_s, w_s)\}$: instance, $n, m, k, w$: integer)
01: if $C \notin \mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$, return $\perp$.
02: $T \leftarrow \emptyset$. /* table for dynamic programming */
03: for each $C \in g \circ \mathbf{AND}_k(n', m', (s+1)\cdot w)$, /* lexicographical order */
04:     $T \leftarrow T \cup \{(C, \#\mathrm{sat}(C))\}$. /* brute force search */
05: $N \leftarrow 0$.
06: for each $\rho : V \to \{0,1,*\}$ such that $\rho^{-1}(*) = \{x_1, \ldots, x_{n'}\}$,
07:     $N \leftarrow N + \#\mathrm{sat}(C|_\rho)$. /* binary search in $T$ */
08: return $N$.
```

Figure 1: A Dynamic Programming Algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k$

We will show the following theorem.

Theorem 3.1. *We can count the number of satisfying assignments for $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$ deterministically in time*

$$\mathrm{poly}(n, m, \log w) \cdot 2^{n - \Omega((n/\log(mw))^{1/k})}$$

and exponential space.

Proof. We denote by $|g \circ \mathbf{AND}_k(n,m,w)|$ the cardinality of $g \circ \mathbf{AND}_k(n,m,w)$. To evaluate the running time of (Step 1), we upper bound the size of the table T using the following fact.

Fact 3.2. *For all m, we have*

$$|g \circ \mathbf{AND}_k(n,m,w)| \leq (2w+1)^{\sum_{i=0}^{k} 2^i \binom{n}{i}} \leq 2^{(k+1)(2n)^k \log(2w+1)}.$$

Proof. Note that $\sum_{i=0}^{k} 2^i \binom{n}{i}$ is the number of different terms that consist of at most k-literals (including a constant function 1). Each term has a weight in $\{-w, -w+1, \ldots, w-1, w\}$. Thus, we have the first inequality. The second inequality follows from an elementary calculation. □

Thus, we can bound the running time of Lines 03-04 from above by

$$2^{(k+1)(2n')^k \log(2(m+1)w+1)} \times \mathrm{poly}(m', \log(mw)) \cdot 2^{n'},$$

where we set $m' = \sum_{i=0}^{k} 2^i \binom{n'}{i} \leq (k+1)(2n')^k$.

Next we evaluate the running time of (Step 2). Note that the following guarantees that every $C|_\rho$ in Line 06 belongs to $g \circ \mathbf{AND}_k(n', m', (m+1)\cdot w)$.

Fact 3.3. *Let $C = \{g, w_0, (t_1, w_1), \dots, (t_m, w_m)\}$. If $C \in g \circ \mathbf{AND}_k(n, m, w)$ holds, then for all restriction ρ with $|\rho^{-1}(*)| = n'$, we have $C|_\rho \in g \circ \mathbf{AND}_k(n', m', (m+1) \cdot w)$.*

Proof. By the definition of $\mathbf{SYM}_w \circ \mathbf{AND}_k(n, m)$, we have $\sum_{i=0}^{s} |w_i| \leq (m+1)w$. This implies the maximum weight of $C|_\rho$ is at most $(m+1)w$. □

For each $C|_\rho$, binary search in Line 07 takes time at most

$$\log_2 |g \circ \mathbf{AND}_k(n', m', (m+1) \cdot w)| \times \mathrm{poly}(m', \log(mw)) = \mathrm{poly}(m', \log(mw)).$$

Thus, we can bound the running time of Lines 06-07 above by

$$\mathrm{poly}(m, m', \log(mw)) \cdot 2^{n-n'}.$$

If we set $n' = \left(\frac{n}{(k+1)2^{k+1}\log(2(m+1)w+1)}\right)^{1/k} = \Theta((n/\log(mw))^{1/k})$, the total running time of **Algorithm1** is bounded from above by $\mathrm{poly}(n, m, \log w) \cdot 2^{n-\Omega((n/\log(mw))^{1/k})}$. This completes the proof. □

Remark 3.4. *In the case when $g(z) = \mathrm{sgn}(z)$, we can reduce the weight of the top gate of $C|_\rho$ from $(m+1)w$ to $2^{n'^{O(k)}}$ efficiently by Theorem 16 in [42]. With this trick, we can handle Max SAT instances with arbitrary weights.*

4 A Greedy Restriction Algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k$

For a term t, we denote by $|t|$ the width of t, i.e., the number of literals in t and by $\mathrm{var}(t)$ the set of variables that appear in t (possibly negated). Let $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n, m)$ be a circuit $\{g, w_0, (t_1, w_1), \dots, (t_s, w_s)\}$. We define $\mathrm{var}_\ell(C) := \cup_{i:|t_i| \geq \ell} \mathrm{var}(t_i)$, $\mathrm{freq}_\ell(C, x) := |\{t_i \in C \mid x \in \mathrm{var}(t_i), |t_i| \geq \ell\}|$, and $L_\ell(C) := \sum_{i:|t_i| \geq \ell} |t_i|$.

Our second satisfiability algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k(n, m)$ is described in Fig. 2. The basic idea is as follows:
(Step 1) Choose a positive integer ℓ according to the input. We seek for a variable, say x, that occurs most frequently in terms of width at least ℓ. We recursively run the algorithm for $C|_{x=0}$ and $C|_{x=1}$. Here $C|_{x=a}$ denotes the circuit obtained from C by applying a restriction ρ such that $\rho(x) = a \in \{0, 1\}$ and $\rho(x') = *$ for $x' \neq x$.
(Step 2) If there is no term of width at least ℓ, we call **Algorithm1**.

We will show the following theorem which implies Theorem 1.1 by setting $k = n$.

Theorem 4.1. *We can count the number of satisfying assignments for $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n, m)$ deterministically in time*

$$\mathrm{poly}(n, m, \log w) \cdot 2^{n-\Omega\left((n/\log(mw))^{\log n/4\log(km)}\right)}$$

and exponential space.

Proof. Let us define a sequence of random variables $\{C_i\}$ inductively as $C_0 := C$ and $C_{i+1} := C_i|_{x=a}$, where $x = \arg\max_{x \in \mathrm{var}(C_i)} \mathrm{freq}_\ell(C_i, x)$ and a is a uniform random bit.

We can think of the computation of **Algorithm2** as a rooted binary tree. That is, the root node is labeled with C_0, the left and right children of the root are labeled with $C_0|_{x=0}$ and $C_0|_{x=1}$, and so on. Then, if we pick a node of depth $n - n'$ uniformly at random, the distribution of its label is identical to that of the random variable $C_{n-n'}$.

We would like to bound the running time of **Algorithm2**$(C_{n-n'}, n', n', \ell)$. It is obviously bounded from above by $\mathrm{poly}(n, m, \log w) \cdot 2^{n'}$. Furthermore, if $L_\ell(C_{n-n'}) < \frac{n'}{2}$ holds, the running time can be bounded by

```
**Algorithm2(**$C = \{g, w_0, (t_1, w_1), \ldots, (t_s, w_s)\}$**: instance,** $n, n', \ell$**: integer)**
**if** $n > n'$,
    $x = \arg\max_{x \in \mathrm{var}(C)} \mathrm{freq}_\ell(C, x)$.
    $N_0 \leftarrow$ **Algorithm2(**$C|_{x=0}, n-1, n', \ell$).
    $N_1 \leftarrow$ **Algorithm2(**$C|_{x=1}, n-1, n', \ell$).
    **return** $N_0 + N_1$.
**else**
    $N \leftarrow 0$.
    **for each** $\rho : \mathrm{var}(C) \to \{0,1,*\}$ such that $\rho^{-1}(\{0,1\}) = \mathrm{var}_\ell(C)$,
        $w' \leftarrow$ the maximum weight of $C|_\rho$.
        $N \leftarrow N +$ **Algorithm1(**$C|_\rho, n - |\mathrm{var}_\ell(C)|, m', \ell - 1, w'$).
    **return** $N$.
```

Figure 2: A Greedy Restriction Algorithm for $\mathbf{SYM}_w \circ \mathbf{AND}_k$

$2^{n'/2} \times$ (the running time of **Algorithm1**$(C', n'/2, m', \ell - 1, w')$) for $C' \in \mathbf{SYM}_{w'} \circ \mathbf{AND}_{\ell-1}(n'/2, m')$ with $m' = \ell \cdot (n')^{\ell-1}$ and $w' = (m+1)w$. We need the following lemma that is proven in Appendix D.

Lemma 4.2 (Greedy bottom fan-in reduction)**.** *Let $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n, m)$. For all $n' \geq 4$, we have*

$$\Pr\left[L_\ell(C_{n-n'}) \geq 2^\ell \cdot L_\ell(C) \cdot \left(\frac{n'}{n}\right)^{\frac{\ell+2}{2}}\right] < 2^{-n'}.$$

Since $L_\ell(C) \leq km$, if we set $n' = \frac{1}{16}\left(\frac{n}{km}\right)^{2/\ell} \cdot n$ in the above lemma, we have

$$2^\ell \cdot L_\ell(C) \cdot \left(\frac{n'}{n}\right)^{\frac{\ell+2}{2}} \leq \frac{n'}{2},$$

that is, we have $L_\ell(C_{n-n'}) < n'/2$ with probability at least $1 - 2^{-n'}$. If we set $\ell = \frac{4\log(km)}{\log n}$, then the total running time of **Algorithm2** is bounded from above by the sum of

$$\mathrm{poly}(n, m, \log w) \cdot 2^{n-n'} \cdot 2^{-n'} \cdot 2^{n'}$$

and

$$\mathrm{poly}(n, m, \log w) \cdot 2^{n-n'} \cdot (1 - 2^{-n'}) \cdot 2^{n'/2} \cdot 2^{n'/2 - \Omega((n'/(\log(m'w'))^{1/\ell})}$$

according to whether $L_\ell(C_{n-n'}) \geq n'/2$ holds or not. An elementary calculation completes the proof. □

Remark 4.3. *The novelty of our algorithm and its analysis is a new way of reducing the bottom fan-in of circuits in a greedy manner. Intuitively, given a $\mathbf{SYM}_w \circ \mathbf{AND}_k$ circuit with m gates, greedy restriction produces a collection of $\mathbf{SYM}_{w'} \circ \mathbf{AND}_{k'}$ circuits with $k' = O(\log(km)/\log n)$ such that at least one of the circuits in the collection is satisfiable if and only if so is the original circuit. Note that previous techniques such as Schuler's width reduction [54, 11] or the standard random restriction achieve $k' = O(\log(m/n))$ and this bound is not sufficient for our purpose.*

5 Average-Case Circuit Lower Bounds

Through the analysis of our satisfiability algorithms, we obtain the following average-case lower bounds.

Theorem 5.1 (depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *There exists a constant $\alpha > 0$ such that for every m, w and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w}$ such that for every $C \in \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$, it holds that*

$$\Pr_{x \in \{0,1\}^n}[f_{n,m,w}(x) = C(x)] \le \frac{1}{2} + 2^{-\Omega\left((n/\log(mw))^{\alpha \log n/\log(nm)}\right)}.$$

Theorem 5.2 (depth d, weighted symmetric gate only at the top)**.** *There exists a constant $\alpha > 0$ such that for every m, w, d and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w,d}$ such that for every $C \in \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$, it holds that*

$$\Pr_{x \in \{0,1\}^n}[f_{n,m,w,d}(x) = C(x)] \le \frac{1}{2} + 2^{-\Omega\left((n/2^{2d(\log m)^{4/5}} \log(mw))^{\alpha \log n/\log m}\right)}.$$

Theorem 5.3 (depth d, $t(n)$ weighted symmetric gates)**.** *There exists a constant $\alpha > 0$ such that for every m, w and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w,d,t}$ such that for every $C \in \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$, it holds that*

$$\Pr_{x \in \{0,1\}^n}[f_{n,m,w,d,t}(x) = C(x)] \le \frac{1}{2} + 2^{-\Omega\left((n/2^{2d(\log m')^{4/5}} \log(m'w'))^{\alpha \log n/\log m'}\right)},$$

where $m' = m2^{t+1}$ and $w' = (mw)^{2^{t+1}}$.

In the rest of this section, we give a proof of Theorem 5.1. The proof of Theorem 5.2 is similar and we omit it. Theorem 5.3 immediately follows from Theorem 5.2 with Lemma B.3 in Section B.

5.1 Generalized Andreev function

In this section, we review the construction of average-case hard Boolean functions due to [17, 37]. We begin with some definitions.

Definition 5.4 (Statistical distance)**.** *Two distributions X, Y over a set E are ε-*close *if $|\mathbf{Pr}[X \in A] - \mathbf{Pr}[Y \in A]| \le \varepsilon$ holds for every $A \subseteq E$.*

Definition 5.5. *A set $A \subseteq \{0,1\}^n$ is a* subcube of dimension *k if there exist $1 \le i_1 < \cdots < i_k \le n$ and $a_{i_1}, \ldots, a_{i_k} \in \{0,1\}$ such that $A = \{x \in \{0,1\}^n \mid x_{i_1} = a_{i_1}, \ldots, x_{i_k} = a_{i_k}\}$.*

Definition 5.6 (Bit-fixing extractor)**.** *A function $f : \{0,1\}^n \to \{0,1\}^m$ is an (n,k,m,ε)-*bit-fixing extractor *if $f(X)$ and the uniform distribution over $\{0,1\}^m$ are ε-close for every distribution X that is uniform over a subcube of $\{0,1\}^n$ of dimension at least k.*

We need the following explicit construction due to Rao.

Lemma 5.7 (Efficient bit-fixing extractor [46])**.** *There exist constants $\alpha, \beta > 0$ such that for every $k \ge (\log n)^\alpha$, there exists a polynomial time computable* $\mathrm{Ext}_{n,k} : \{0,1\}^n \to \{0,1\}^m$ *that is an (n,k,m,ε)-bit-fixing extractor with $m = 0.9k$ and $\varepsilon \le 2^{-k^\beta}$.*

We also need an efficient and explicit construction of list decodable codes.

Definition 5.8 (List-Decodable Code)**.** *A function $f : \{0,1\}^k \to \{0,1\}^n$ is (p,L)-list-decodable if $|\{y \in \{0,1\}^k \mid \Delta(f(x), f(y)) \le pn\}| \le L$ holds for every $x \in \{0,1\}^k$, where $\Delta(a,b)$ denotes the Hamming distance between a and b.*

Lemma 5.9 (Efficient List-Decodable Code (Folklore), see Theorem 6.4 in [17])**.** *There exists a function* $\mathrm{Enc}_{n,r} : \{0,1\}^{4n} \to \{0,1\}^{2^r}$ *that is* (p,L)*-list-decodable with* $p = 1/2 - O(2^{-r/4})$ *and* $L = O(2^{r/2})$*. Furthermore, there exists an algorithm that, given* $x \in \{0,1\}^{4n}$ *and* $z \in \{0,1\}^{2^r}$*, computes* $(\mathrm{Enc}_{n,r}(x))_z$ *in polynomial time.*

We are ready to define the average-case hard Boolean functions: The generalized Andreev function $A_{n,k} : \{0,1\}^{4n} \times \{0,1\}^n \to \{0,1\}$ is defined as $A_{n,k}(x,y) := (\mathrm{Enc}_{n,0.9k}(x))_{\mathrm{Ext}_{n,k}(y)}$. Let $K(x)$ denote the *Kolmogorov complexity* of a string $x \in \{0,1\}^*$. The following lemma plays an important role in the proofs of our average-case lower bounds.

Lemma 5.10 (Theorem 6.5 in [17])**.** *There exist constants* $\alpha, \gamma > 0$ *such that the following holds. Let* $k \geq (\log n)^\alpha$ *and C be a k-variate circuit whose binary description length is at most n in a some fixed encoding scheme. Let* $\rho : \{x_1, \dots, x_n\} \to \{0,1,*\}$ *be a restriction with* $|\rho^{-1}(*)| = k$*. Fix* $a \in \{0,1\}^{4n}$ *with* $K(a) \geq 3n$ *and define* $f(y) := A_{n,k}(a,y)$*. Then, we have*

$$\Pr_{y' \in \{0,1\}^k}[C(y') = f|_\rho(y')] \leq \frac{1}{2} + \frac{1}{2^{k^\gamma}}.$$

The following fact can be shown by a counting argument.

Fact 5.11. *For every* $0 < p < 1$, $\mathbf{Pr}_{x \in \{0,1\}^n}[K(x) \leq (1-p)n] \leq 2^{-pn+1}$.

5.2 Proof of Theorem 5.1

Fix n, m, w and let $n' = (n/\log(mw))^{\log n/4\log(nm)}$. Select any $a \in \{0,1\}^{4n}$ with $K(a) \geq 3n$ and let $f(y) := A_{n,n'}(a,y)$. We show the following lemma.

Lemma 5.12. *For every* $C \in \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$*, it holds that*

$$\Pr_{y \in \{0,1\}^n}[C(y) = f(y)] \leq \frac{1}{2} + 2^{-\Omega(n'^\gamma)},$$

where $\gamma > 0$ *is a universal constant from Lemma 5.10.*

Assuming this, the proof of Theorem 5.1 is complete since by Fact 5.11, we have

$$\begin{aligned}
\Pr_{x,y}[A_{n,n'}(x,y) = C(x,y)] &\leq \Pr_x[K(x) < 3n] + \Pr_x[K(x) \geq 3n] \Pr_{x,y}[A_{n,n'}(x,y) = C(x,y) \mid K(x) \geq 3n] \\
&\leq 2^{-\Omega(n)} + \max_{x : K(x) \geq 3n} \Pr_y[A_{n,n'}(x,y) = C(x,y)] \\
&\leq 2^{-\Omega(n)} + \frac{1}{2} + 2^{-\Omega(n'^\gamma)}.
\end{aligned}$$

.

Proof of Lemma 5.12. We can see that from the proofs of Theorems 3.1 and 4.1, C can be computed by a restriction decision tree T of height $n - n'$ such that (1) each leaf is labeled by a circuit in $\mathbf{SYM}_{w'} \circ \mathbf{AND}_{k'}(n', m')$ for some m', k', w' and (2) except for a $2^{-n^{\Omega(1)}}$ fraction of leaves, such a circuit can be described by using at most n bits. Let $\sigma(C)$ denote the description length of a circuit C in a fixed encoding scheme. Let ρ be a random restriction sampled by selecting a leaf of T uniformly at random and y_ρ be a uniform random element of $\{0,1\}^{\rho^{-1}(*)}$. Then, we have

$$\begin{aligned}
\Pr_y[C(y) = f(y)] &\leq \Pr_\rho[\sigma(C|_\rho) > n] + \Pr_\rho[\sigma(C|_\rho) \leq n] \Pr_{\rho, y_\rho}[C|_\rho(y_\rho) = f|_\rho(y_\rho) \mid \sigma(C|_\rho) \leq n] \\
&\leq 2^{-n^{\Omega(1)}} + \frac{1}{2} + 2^{-\Omega(n'^\gamma)},
\end{aligned}$$

where the last inequality is by Item (2) above and Lemma 5.10. This completes the proof. □

6 Worst-Case Lower Bounds

From the average-case lower bounds in Section 5, we obtain the following worst-case lower bounds.

Theorem 6.1 (majority vote of depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *There exists a constant $\alpha > 0$ such that for every m, w and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w}$ such that $C \in \mathbf{MAJ} \circ \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$ cannot compute $f_{n,m,w}$ if the majority gate at the top of C has fan-in at most $2^{o\left((n/\log(mw))^{\alpha \log n/\log(nm)}\right)}$.*

Theorem 6.2 (majority vote of depth d, weighted symmetric gate only at the top)**.** *There exists a constant $\alpha > 0$ such that for every m, w, d and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w,d}$ such that any $C \in \mathbf{MAJ} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ cannot compute $f_{n,m,w,d}$ if the majority gate at the top of C has fan-in at most $2^{o\left((n/2^{2d(\log m)^{4/5}} \log(mw))^{\alpha \log n/\log m}\right)}$.*

Theorem 6.3 (majority vote of depth d, $t(n)$ weighted symmetric gates)**.** *There exists a constant $\alpha > 0$ such that for every m, w, d, t and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w,d,t}$ such that any $C \in \mathbf{MAJ} \circ \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ cannot compute $f_{n,m,w,d,t}$ if the majority gate at the top of C has fan-in at most $2^{o\left((n/2^{2d(\log m')^{4/5}} \log(m'w'))^{\alpha \log n/\log m'}\right)}$, where $m' = m2^{t+1}$ and $w' = (mw)^{2^{t+1}}$.*

Theorem 6.4 (depth d, $t(n)$ weighted symmetric gates)**.** *There exists a constant $\alpha > 0$ such that for every m, w, d, t and sufficiently large n, there exists a polynomial time computable function $f_{n,m,w,d,t}$ such that any $C \in \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ cannot compute $f_{n,m,w,d,t}$ if*

$$t = o\left((n/2^{2d(\log m')^{4/5}} \log(m'w'))^{\alpha \log n/\log m'}\right)$$

holds, where $m' = m(t+1)$ and $w' = m^t w^{t+1}$.

We need a corollary of the discriminator lemma that is proven in Section E.

Lemma 6.5 (Discriminator Lemma [27])**.** *If a circuit $C \in \mathbf{MAJ} \circ \mathscr{C}$ is a majority vote of k circuits $C_1, \ldots, C_k \in \mathscr{C}$, then for some $1 \leq i \leq k$, we have*

$$|\Pr_x[C_i(x) = 1 \mid C(x) = 1] - \Pr_x[C_i(x) = 1 \mid C(x) = 0]| \geq \frac{1}{k}.$$

For $f, g : \{0,1\}^n \to \{0,1\}$, let $\mathrm{Corr}(f,g) := |\mathbf{Pr}_x[f(x) = g(x)] - \mathbf{Pr}_x[f(x) \neq g(x)]|$.

Corollary 6.6. *For $\varepsilon \geq 0$, if C in Lemma 6.5 also satisfies that*

$$|\Pr_x[C(x) = 0] - \Pr_x[C(x) = 1]| = 2\varepsilon,$$

then we have $\mathrm{Corr}(f,g) \geq \frac{1}{k} - 2\varepsilon$.

Theorems 6.1, 6.2 and 6.3 immediately follow from Theorems 5.1, 5.2 and 5.3 with Corollary 6.6. Theorem 6.4 can be shown by combining Lemma B.2 in Section B, Theorem 5.2 and Corollary 6.6.

References

[1] A. Abboud, R. Williams, and H. Yu. More applications of the polynomial method to algorithm design. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 218–230, 2015.

[2] K. Amano and A. Saito. A nonuniform circuit class with multilayer of threshold gates having super quasi polynomial size lower bounds against NEXP. In *Proceedings of the 9th International Conference on Language and Automata Theory and Applications (LATA)*, pages 461–472, 2015.

[3] K. Amano and A. Saito. A satisfiability algorithm for some class of dense depth two threshold circuits. *IEICE Transactions*, 98-D(1):108–118, 2015.

[4] J. Aspnes, R. Beigel, M. L. Furst, and S. Rudich. The expressive power of voting polynomials. *Combinatorica*, 14(2):135–148, 1994.

[5] L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.*, 45(2):204–232, 1992.

[6] D. A. M. Barrington and H. Straubing. Complex polynomials and circuit lower bounds for modular counting. *Computational Complexity*, 4:325–338, 1994.

[7] P. Beame, R. Impagliazzo, and S. Srinivasan. Approximating AC^0 by small height decision trees and a deterministic algorithm for $\#\mathrm{AC}^0$ SAT. In *Proceedings of the 27th Conference on Computational Complexity (CCC)*, pages 117–125, 2012.

[8] R. Beigel. When do extra majority gates help? polylog(n) majority gates are equivalent to one. *Computational Complexity*, 4:314–324, 1994.

[9] R. Beigel, N. Reingold, and D. A. Spielman. PP is closed under intersection. *J. Comput. Syst. Sci.*, 50(2):191–202, 1995.

[10] E. Ben-Sasson and E. Viola. Short PCPs with projection queries. In *Proceedings of the 41st International Colloquium on Automata, Languages, and Programming (ICALP), Part I*, pages 163–173, 2014.

[11] C. Calabro, R. Impagliazzo, and R. Paturi. A duality between clause width and clause density for SAT. In *Proceedings of the 21st Annual IEEE Conference on Computational Complexity (CCC)*, pages 252–260, 2006.

[12] C. Calabro, R. Impagliazzo, and R. Paturi. The complexity of satisfiability of small depth circuits. In *Revised Selected Papers from the 4th International Workshop on Parameterized and Exact Computation (IWPEC)*, pages 75–85, 2009.

[13] M. L. Carmosino, R. Impagliazzo, V. Kabanets, and A. Kolokolova. Algorithms from natural lower bounds. In *Proceedings of the 31st Conference on Computational Complexit (CCC)*, pages 10:1–10:24, 2016.

[14] A. Chattopadhyay and K. A. Hansen. Lower bounds for circuits with few modular and symmetric gates. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 994–1005, 2005.

[15] R. Chen. Satisfiability algorithms and lower bounds for Boolean formulas over finite bases. In *Proceedings of the 40th International Symposium on Mathematical Foundations of Computer Science (MFCS), Part II*, pages 223–234, 2015.

[16] R. Chen and V. Kabanets. Correlation bounds and #SAT algorithms for small linear-size circuits. In *Proceedings of the 21st International Conference on Computing and Combinatorics (COCOON)*, pages 211–222, 2015.

[17] R. Chen, V. Kabanets, A. Kolokolova, R. Shaltiel, and D. Zuckerman. Mining circuit lower bound proofs for meta-algorithms. *Computational Complexity*, 24(2):333–392, 2015.

[18] R. Chen, V. Kabanets, and N. Saurabh. An improved deterministic #SAT algorithm for small De Morgan formulas. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS), Part II*, pages 165–176, 2014.

[19] R. Chen and R. Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *Proceedings of the 18th International Conference on Theory and Applications of Satisfiability Testing (SAT)*, pages 33–45, 2015.

[20] R. Chen, R. Santhanam, and S. Srinivasan. Average-case lower bounds and satisfiability algorithms for small threshold circuits. In *Proceedings of the 31st Conference on Computational Complexit (CCC)*, pages 1:1–1:35, 2016.

[21] M. Cygan, H. Dell, D. Lokshtanov, D. Marx, J. Nederlof, Y. Okamoto, R. Paturi, S. Saurabh, and M. Wahlström. On problems as hard as CNF-SAT. In *Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC)*, pages 74–84, 2012.

[22] J. Forster, M. Krause, S. V. Lokam, R. Mubarakzjanov, N. Schmitt, and H. Simon. Relations between communication complexity, linear arrangements, and computational complexity. In *Proceedings of the 21st Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 171–182, 2001.

[23] M. Goldmann. On the power of a threshold gate at the top. *Inf. Process. Lett.*, 63(6):287–293, 1997.

[24] M. Goldmann and M. Karpinski. Simulating threshold circuits by majority circuits. *SIAM J. Comput.*, 27(1):230–246, 1998.

[25] A. Golovnev, A. S. Kulikov, A. Smal, and S. Tamaki. Circuit size lower bounds and #SAT upper bounds through a general framework. In *Proceedings of the 41st International Symposium on Mathematical Foundations of Computer Science (MFCS)*, 2016, to appear.

[26] P. Gopalan and R. A. Servedio. Learning and lower bounds for AC^0 with threshold gates. In *Proceedings of the 13th APPROX and the 14th RANDOM*, pages 588–601, 2010.

[27] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and G. Turán. Threshold circuits of bounded depth. *J. Comput. Syst. Sci.*, 46(2):129–154, 1993.

[28] K. A. Hansen and P. B. Miltersen. Some meet-in-the-middle circuit lower bounds. In *Proceedings of the 29th International Symposium Mathematical Foundations of Computer Science (MFCS)*, pages 334–345, 2004.

[29] J. Håstad and M. Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

[30] R. Impagliazzo, S. Lovett, R. Paturi, and S. Schneider. 0-1 integer linear programming with a linear number of constraints. *Electronic Colloquium on Computational Complexity (ECCC)*, TR14-24, 2014.

[31] R. Impagliazzo, W. Matthews, and R. Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 961–972, 2012.

[32] R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[33] R. Impagliazzo, R. Paturi, and S. Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 479–488, 2013.

[34] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[35] H. Jahanjou, E. Miles, and E. Viola. Local reductions. In *Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP), Part I*, pages 749–760, 2015.

[36] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.*, 9(3):256–278, 1974.

[37] I. Komargodski, R. Raz, and A. Tal. Improved average-case lower bounds for demorgan formula size. In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 588–597, 2013.

[38] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[39] S. Lovett and S. Srinivasan. Correlation bounds for poly-size AC^0 circuits with $n^{1-o(1)}$ symmetric gates. In *Proceedings of the 14th APPROX 2011 and the 15th RANDOM*, pages 640–651, 2011.

[40] O. B. Lupanov. On a method of circuit synthesis (in Russian). *Izvestiâ vysših učebnyh zavedenij, Radiofiz*, 1:120–140, 1958.

[41] D. Marx. Consequences of SETH: Tight bounds for some more problems. In *Fine-Grained Complexity and Algorithm Design Boot Camp*, 2015. “https://simons.berkeley.edu/talks/daniel-marx-2015-09-04” (abstract, slides and archived video).

[42] S. Muroga, I. Toda, and S. Takasu. Theory of majority decision elements. *Journal of the Franklin Institute*, 271(5):376–418, 1961.

[43] A. Nagao, K. Seto, and J. Teruyama. A moderately exponential time algorithm for k-IBDD satisfiability. In *Proceedings of the 14th International Symposium, on Algorithms and Data Structures (WADS)*, pages 554–565, 2015.

[44] I. C. Oliveira. Algorithms versus circuit lower bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, TR13-117, 2013.

[45] V. V. Podolskii. Exponential lower bound for bounded depth circuits with few threshold gates. *Inf. Process. Lett.*, 112(7):267–271, 2012.

[46] A. Rao. Extractors for low-weight affine sources. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity (CCC)*, pages 95–101, 2009.

[47] A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Sci. of the USSR*, 41(4):333–338, 1987.

[48] A. Razborov and S. Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997.

[49] A. Razborov and A. Wigderson. $n^{\Omega(\log n)}$ lower bounds on the size of depth-3 threshold circuits with AND gates at the bottom. *Inf. Process. Lett.*, 45(6):303–307, 1993.

[50] T. Sakai, K. Seto, and S. Tamaki. Solving sparse instances of Max SAT via width reduction and greedy restriction. *Theory Comput. Syst.*, 57(2):426–443, 2015.

[51] T. Sakai, K. Seto, S. Tamaki, and J. Teruyama. A satisfiability algorithm for depth-2 circuits with a symmetric gate at the top and AND gates at the bottom. *Electronic Colloquium on Computational Complexity (ECCC)*, TR15-136, 2015.

[52] R. Santhanam. Fighting perebor: New and improved algorithms for formula and QBF satisfiability. In *Proceedings of the 51th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 183–192, 2010.

[53] R. Santhanam. Ironic complicity: Satisfiability algorithms and circuit lower bounds. *Bulletin of the EATCS*, 106:31–52, 2012.

[54] R. Schuler. An algorithm for the satisfiability problem of formulas in conjunctive normal form. *J. Algorithms*, 54(1):40–44, 2005.

[55] K. Seto and S. Tamaki. A satisfiability algorithm and average-case hardness for formulas over the full binary basis. *Computational Complexity*, 22(2):245–274, 2013.

[56] R. Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC)*, pages 77–82, 1987.

[57] S. Srinivasan. A compression algorithm for $\mathrm{AC}^0[\oplus]$ circuits using certifying polynomials. *Electronic Colloquium on Computational Complexity (ECCC)*, TR15-142, 2015.

[58] A. Tal. #SAT algorithms from shrinkage. *Electronic Colloquium on Computational Complexity (ECCC)*, TR15-114, 2015.

[59] E. Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM J. Comput.*, 36(5):1387–1403, 2007.

[60] F. Wang. NEXP does not have non-uniform quasipolynomial-size ACC circuits of $o(\log\log n)$ depth. In *Proceedings of the 8th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, pages 164–170, 2011.

[61] R. Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013.

[62] R. Williams. Natural proofs versus derandomization. In *Proceedings of the 45th ACM Symposium on Theory of Computing Conference (STOC)*, pages 21–30, 2013.

[63] R. Williams. Algorithms for circuits and circuits for algorithms. In *Proceedings of the 29th Annual IEEE Conference on Computational Complexity (CCC)*, pages 248–261, 2014.

[64] R. Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Proceedings of the 46th Symposium on Theory of Computing (STOC)*, pages 194–202, 2014.

[65] R. Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2, 2014.

[66] F. Zane. *Circuits, CNFs, and satisfiability*. PhD thesis, UC San Diego, 1998.

Appendix

A A Depth Reduction Algorithm for $\mathbf{SYM}_w \circ \mathbf{AC}^0_d$

In this section, we prove the following following theorem:

Theorem A.1. *We can count the number of satisfying assignments for $C \in \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ when $m \leq 2^{(\log n/4d)^{5/4}}$ deterministically in time*

$$\mathrm{poly}(n,m,\log w) \cdot 2^{n-\Omega(n/2^{2d(\log m)^{4/5}}\log(mw))^{\log n/9\log m}}$$

and exponential space.

We represent $C \in \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ as $C = \{g, w_0, (C_1,w_1), \ldots, (C_s,w_s)\}$, where $g : \mathbb{Z} \to \{0,1\}$, $s \leq m$, $w_0, w_1, \ldots, w_s \in \mathbb{Z}$, $\max_{0 \leq i \leq s} |w_i| \leq w$, and $C_1, \ldots, C_s$ are unbounded fan-in circuits of depth at most d with AND, OR and NOT gates.

The following fact is useful for the design and analysis of our algorithm.

Fact A.2. *Let a $\mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ circuit $C = \{g, w_0, (C_1,w_1), \ldots, (C_s,w_s)\}$, where some C_i is a DNF of the form $t_1 \vee t_2 \vee \cdots \vee t_\ell$. If for every $x \in C_i^{-1}(1)$, x satisfies exactly one term of C_i, then it holds that*

$$C \equiv \{g, w_0, (C_1,w_1), \ldots, (C_{i-1},w_{i-1}), (C_{i+1},w_{i+1}), \ldots, (C_s,w_s), (t_1,w_i), \ldots, (t_\ell,w_i)\}.$$

The outline of our algorithm is as follows. First, given a circuit $C = \{g, w_0, (C_1,w_1), \ldots, (C_s,w_s)\}$, we construct a restriction decision tree of depth $n - n/m^{2d/k}$ whose almost all leaves define restrictions ρ such that each $C_i|_\rho$ is a k-junta by using the depth reduction technique due to [7]. Note that a k-junta can be represented as a k-DNF satisfying the condition of Fact A.2. Hence, we can remove the OR gate of each k-DNF and obtain a $\mathbf{SYM}_w \circ \mathbf{AND}_k$ circuit. Using **Algorithm2** in Section 4, we can count the number of satisfying assignments for such a $\mathbf{SYM}_w \circ \mathbf{AND}_k$ circuit. If some $C_i|_\rho$ is not a k-junta, we check all possible 0/1 assignments to the remaining variables. The fraction of such leaves is exponentially small.

The main ingredient of our algorithm is the following depth reduction algorithm due to [7].

Lemma A.3 ([7]). *Let $\mathscr{F}$ be a set of k-DNF formulas over $\{0,1\}^n$ with $|\mathscr{F}| \leq m$ and suppose that $k \leq k' \leq (\log_2 m)^{1/5}$. Then there is a restriction decision tree $T_{\mathscr{F}}$ over $\{0,1\}^n$ of height $n - m^{-2/k'}n$ such that for ρ chosen according to a random root-leaf path in $T_{\mathscr{F}}$, the probability that for some formula $F \in \mathscr{F}$, $F|_\rho$ is not a k'-junta is at most $4k\log_2 m \exp(-2^{-2k-5}k^{-3}n/\log_2 m)$. Moreover, there is an algorithm with the running time $2^{n-m^{-2/k'}n}n^{O(k')}||\mathscr{F}||^{O(1)}$ that constructs $T_{\mathscr{F}}$ given $\mathscr{F}$ as input.*

We say a leaf of a restriction decision tree T_i (or a restriction defined by that leaf) is *good* if all OR gates at level-i reduce to k-juntas after the restriction. Otherwise, it is called *bad*. Now we are ready to prove Theorem A.1.

Proof of Theorem A.1. Without loss of generality, we assume each circuit C_i consists of alternating NOT gates and unbounded fan-in OR gates (and the layer of NOT gates do not contribute to the depth of the circuit).

[depth $d+1$ to d] Clearly, each OR gate at the bottom level is represented as a 1-DNF formula. Setting $k = 1$ and $k' = (\log_2 m)^{1/5}$ in Lemma A.3, there exists a restriction decision tree T_1 of height $n - n/m^{2/k'}$ such that for almost all root-to-leaf paths ρ of T_1, after restriction by ρ, the functions computed by all OR gates at level-1 of the circuit depend only on k variables. Their negations also depend only on k variables and hence, after restriction by ρ, each OR gate at level-2 of C is a k-DNF formula. Thus, we can reduce the depth of C from $d+1$ to d. The probability p_1 that a leaf of T_1 is bad satisfies

$$p_1 \le 4 \log_2 m \exp(-n/128 \log_2 m).$$

[depth ℓ to $\ell - 1$ ($3 \le \ell \le d$)] Let $k = k' = (\log_2 m)^{1/5}$ and $n_1 = n/m^{2d/k}$. Since all good leaves in T_1 creates k-DNF formulas, we can apply Lemma A.3 to the set of OR gates at level-2 and construct a new restriction decision tree T_2 obtained by appending a restriction decision tree to each good leaf. The probability p_2 that a leaf of T_2 is bad satisfies

$$p_2 \le (1-p_1) \cdot 4k \log_2 m \exp(-2^{-2k-5} k^{-3} n_1 / \log_2 m)).$$

We repeat the similar argument for each of the remaining levels up until we have built the tree T_d. Note that at each good leaf of T_d, all OR gates at level-d reduce to k-juntas and these can be represented as k-DNF formulas. Let $n_i = n_{i-1}/m^{2/k}$ for $i \le d$ and p_i be the probability that a leaf of T_i is bad, then we have

$$p_i \le 4k \log_2 m \exp(-2^{-2k-5} k^{-3} n_{i-1} / \log_2 m)) \prod_{j=1}^{i-1} (1 - p_j).$$

Now the circuit $C \in \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ can be reduced to some circuit $C' \in \mathbf{SYM}_w \circ \mathbf{AC}^0_2(n_d, m)$ at good leaves.

[$\mathbf{SYM}_w \circ \mathbf{AC}^0_2$ to $\mathbf{SYM}_w \circ \mathbf{AND}_k$] After the above operations, each good leaf of T_d is corresponded to a collection of k-DNFs. In addition, such k-DNFs satisfy the condition of Fact A.2 since they are constructed from k-juntas. Note that the size may increase by a factor of at most 2^k. Hence, for good leaves, we can eliminate OR gates of all k-DNFs and C' can be reduced to $C'' \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n_d, 2^k m)$.

For all good leaves of T_d, using **Algorithm2**, we can count the number of satisfying assignments for C''. For all bad leaves of $T_1, \ldots, T_d$, we check all 0/1 assignment to the remaining variables. Now we estimate the probability p that some leaves of T_i ($1 \le i \le d$) is bad, where $p = p_1 + p_2 + \cdots + p_d$. Since n_i decreases rapidly, the probabilities that some leaf is bad are bounded by a quickly increasing geometric series whose largest term is associated with the construction T_d.

$$\begin{aligned}
p \quad &\le \quad d \cdot p_d = d \cdot 4k \log m \exp(-2^{2k-5} k^{-3} n_{d-1} / \log m) \prod_{j=1}^{d-1} (1 - p_j) \\
&\le \quad 4dk \log m \exp(-2^{2k-5} k^{-3} n_{d-1} / \log m) = 4dk \log m \exp(-2^{2k-5} k^{-3} n / m^{2(d-1)/k} \log m) \\
&\le \quad \exp(-n/m^{2d/k}) \qquad \text{for sufficiently large } n \text{ and } m.
\end{aligned}$$

It remains to estimate the running time of our algorithm. Let the height of T_i be $n - n_i$. Then $n_0 = n$, $n_i = n_{i-1}/m^{2/k}$ for $i \le d$, and $n_d = n/m^{2d/k}$. The number of leaves of T_d is at most 2^{n-n_d}. We denote the running time of **Algorithm2** for $\mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$ by $T'(n,m,k,w)$ and the running time of our algorithm by $T(n,m,d,w)$, then

$$\begin{aligned}
T(n,m,d,w) \quad &\le \quad (1-p) \cdot 2^{n-n_d} \cdot \left(\prod_{i=1}^{d} n^{O(k)} \right) \cdot T'(n_d, 2^k m, k, w) \cdot m^{O(d)} + p \cdot 2^n \\
&\le \quad (1-p) \cdot 2^{n-n_d} \cdot n^{O(dk)} \cdot T'(n_d, 2^k m, k, w) \cdot m^{O(d)} + p \cdot 2^n \\
&\le \quad n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-n_d} \cdot T'(n_d, 2^k m, k, w) + p \cdot 2^n.
\end{aligned}$$

The second term on the right hand is:

$$p \cdot 2^n \quad = \quad \exp(-n/m^{2d/k}) \cdot 2^n = 2^{n - n \log e / 2^{2d(\log m)^{4/5}}} \le 2^{n - n/2^{2d(\log m)^{4/5}}}.$$

The first term on the right hand is:

$$\begin{aligned} & n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-n_d} \cdot \mathrm{poly}(n_d, 2^k m, \log w) \cdot 2^{n_d - \Omega(n_d/\log(2^k mw))^{\log n_d/4\log(k2^k m)}} \\ = \;& \mathrm{poly}(n_d, 2^k m, \log w) \cdot n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-\Omega(n_d/\log(2^k mw))^{\log n_d/4\log(k2^k m)}}. \end{aligned}$$

For sufficiently large m and small constant $\varepsilon > 0$, $\log(k2^k m) = k + \log k + \log m < (1+\varepsilon)\log m$. Suppose that $m \le 2^{(\log n/4d)^{5/4}}$, then $n_d = n/m^{2d/k} \ge n^{1/2}$. Thus, $\log n_d/4\log(k2^k m) \ge \log n/8(1+\varepsilon)\log m \ge \log n/9\log m$, then

$$\begin{aligned} & \mathrm{poly}(n_d, 2^k m, \log w) \cdot n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-\Omega(n_d/\log(2^k mw))^{\log n_d/4\log(k2^k m)}} \\ \le \;& \mathrm{poly}(n, m, \log w) \cdot n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-\Omega(n_d/\log(m^2 w))^{\log n/9\log m}} \\ \le \;& \mathrm{poly}(n, m, \log w) \cdot n^{O(dk)} \cdot m^{O(d)} \cdot 2^{n-\Omega(n/2m^{2d/k}\log(mw))^{\log n/9\log m}} \\ \le \;& \mathrm{poly}(n, m, \log w) \cdot n^{O(d(\log m)^{1/5})} \cdot m^{O(d)} \cdot 2^{n-\Omega(n/2^{2d(\log m)^{4/5}}\log(mw))^{\log n/9\log m}}. \end{aligned}$$

When $m = 2^{\mathrm{poly}(\log n)}$ and $d = O(\log n)$, the terms of $n^{O(d(\log m)^{1/5})}$ and $m^{O(d)}$ are absorbed in the last term, then we have

$$\mathrm{poly}(n, m, \log w) \cdot 2^{n-\Omega(n/2^{2d(\log m)^{4/5}}\log(mw))^{\log n/9\log m}}.$$

Thus,

$$T(n,m,d,w) = \mathrm{poly}(n, m, \log w) \cdot 2^{n-\Omega(n/2^{2d(\log m)^{4/5}}\log(mw))^{\log n/9\log m}} + 2^{n-n/2^{2d(\log m)^{4/5}}}.$$

The first term dominates the second term, hence the proof is completed. □

Remark A.4. *We use the depth reduction algorithm due to [7] instead of [31] because we have to keep the bottom fan-in of circuits much smaller than* $\log n/\log\log n$*.*

B A Transformation Algorithm for $\mathbf{AC}^0_d[\mathbf{SYM}_w]$

In this section, we give an algorithm that counts the number of satisfying assignments for an $\mathbf{AC}^0_d[\mathbf{SYM}_w]$ circuit.

Theorem B.1. *When* $m(t+1) \le 2^{(\log n/4d)^{5/4}}$*, there is an algorithm that counts the number of satisfying assignments for* $C \in \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ *in time*

$$\mathrm{poly}(n, m, d, t, \log w) \cdot 2^{n+O(t\log mw)-\Omega\left((n/2^{4d(\log m)^{4/5}} t\log(mw))^{\frac{\log n}{18\log m}}\right)}.$$

When a weighted symmetric gate $s \in \mathbf{SYM}_w$ computes a weighted symmetric function $f : \{0,1\}^n \to \{0,1\}$ such that $f(x_1, \ldots, x_n) = g(w_0 + \sum_{i=1}^n w_i x_i)$ holds, we represent the gate s as $s := \langle g, w_0, \ldots, w_n \rangle$. For a function $g : \mathbb{Z} \to \{0,1\}$, we denote by $\overline{g}$ the negation of g, i.e., $\overline{g}(x) := \neg g(x)$ for all $x \in \mathbb{Z}$.

The outline of our algorithm is as follows. The proof of Lemma 29 in [9] gives the procedure of transformation from a circuit in $\mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ to the equivalent circuit in $\mathbf{OR}_{2^t} \circ \mathbf{AND}_{t+1} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$. It is enough to count the number of satisfying assignments for each circuit in $\mathbf{AND}_{t+1} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ due to the property of the transformation. By the idea of the proof of Theorem 5.1 in [8], we transform a circuit in $\mathbf{AND}_{t+1} \circ \mathbf{SYM}_w$ to an equivalent gate in $\mathbf{SYM}_{w'}$ where $w' = w'(m,w,t)$. Now, we obtain a circuit in $\mathbf{SYM}_{w'} \circ \mathbf{AC}^0_d(n, m(t+1))$ and, count the number of satisfying assignments for it by Theorem A.1.

Proof of Theorem B.1. Let $s_1, s_2, \ldots, s_t \in \mathbf{SYM}_w$ be weighted symmetric gates of a given C, such that there is no path from the output of s_i to an input s_j if $i < j$. For each $i \in [t]$, let s_i represent a weighted symmetric function $f_i : \{0,1\}^{m_i} \to \{0,1\}$. Let $s_i := \langle g_i, w_{i,0}, \ldots, w_{i,m_i} \rangle$, where a function $g_i : \mathbb{Z} \to \{0,1\}$ and integers $w_{i,0}, \ldots, w_{i,m_i} \in [w]$ are such that $f_i(x_1, \ldots, x_{n_i}) = g_i(w_0 + \sum_{\ell=1}^{n_i} w_\ell x_\ell)$ holds.

A *subcircuit for s on C* is a circuit with the top gate s such that it consists of all gates on the path from inputs to s on C. Let C_i be a subcircuit for s_i on C. For $i \in [t]$ and $a \in \{0,1\}^t$, let $C_i(a)$ be a circuit by replacing s_j with the constant a_j for all $j < i$ on C_i. The output of $C_i(a)$ is equivalent to the output of s_i on C when the output of s_j is a_j for all $j < i$. Similarly, let $C(a)$ be the $\mathbf{AC}^0_d(n,m)$ circuit obtained by replacing all s_i with a_i on C. Note that all $C_i(a)$ and $C(a)$ are in $\mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ and can be constructed in time $O(mdt)$.

For each $a \in \{0,1\}^t$, we denote the circuit $C'(a) := C(a) \wedge \bigwedge_i C'_i(a)$, where $C'_i(a) = C_i(a)$ if $a_i = 1$ and $C'_i(a) = \neg C_i(a)$ otherwise. Note that $\neg C_i(a)$ is also a $\mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ circuit obtained by replacing the function g_i of the gate s_i with $\overline{g_i}$. Thus, $C'(a) \in \mathbf{AND}_{t+1} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$. Note that the constructing time of $C'(a)$ is at most $O(mdt)$. An assignment satisfies $C'(a)$ for some $a \in \{0,1\}^t$ if and only if it satisfies C. Moreover, a satisfying assignment of C satisfies $C'(a)$ for only one $a \in \{0,1\}^t$. This means that the number of satisfying assignments of C is equal to the sum of ones of $C'(a)$ for all $a \in \{0,1\}^t$. Therefore, it is enough to count satisfiability assignments of $C'(a)$ for all $a \in \{0,1\}^t$.

Now, we construct the circuit $C'' \in \mathbf{SYM}_{w'} \circ \mathbf{AC}^0_d(n, m(t+1))$ such that $C'' \equiv C'(a) \in \mathbf{AND}_{t+1} \circ \mathbf{SYM}_w \circ \mathbf{AC}^0_d(n,m)$ where w' is an integer. For $i \in [t]$, let W_i be the sum of wights of the inputs of s_i, i.e., $W_i := \sum_{j=0}^{m_i} w_{i,j}$. We denote by W_0 the output of $C(a)$. Let $W \max_{i=0,1,\ldots,t} W_i$, then W is at most mw. Using base W, $t+1$ integers $\{W_i\}_{i=0,\ldots,t}$ can be denoted into a single number $N = \sum_{i=0}^t W_i W^i$. Now, we set the symmetric function g' as follows. We set $g'(N) := 1$ when $N = \sum_{i=0}^t W_i W^i$ holds such that $g_i(W_i) = a_i$ for all i and $W_0 = 1$. For other values N, we set $g'(N) := 0$. Note that the time of setting of g' is $O(tW^{t+1}\mathrm{poly}(\log W)) = O(tm^{t+1}w^{t+1}\mathrm{poly}(\log(mw)))$. Using this function g', we replace the top gate and symmetric gates of second layer of $C'(a)$ with a new symmetric gate $s' := \langle g', (w'_{i,\ell})_{i \in \{0,1,\ldots,t\}, \ell \in [m_i]} \rangle$, where $w'_{i,\ell} = w_{i,\ell} \cdot W^i$. We call this circuit C'', then C'' is equivalent to $C'(a)$. By the above argument, the size of each layer in C'' is at most $m(t+1)$ and the maximum weight of s' is at most $wW^t \leq m^t w^{t+1}$. Then, C'' is a $\mathbf{SYM}_{m^t w^{t+1}} \circ \mathbf{AC}^0_d(n, m(t+1))$ circuit. The time of constructing $C'' \in \mathbf{SYM}_{m^t w^{t+1}} \circ \mathbf{AC}^0_d(n, m(t+1))$ is at most $O(mdt) + O(t\mathrm{poly}(\log(mw)) \cdot m^{t+1}w^{t+1}) = O(dt\mathrm{poly}(\log(mw)) \cdot m^{t+1}w^{t+1})$.

When $m(t+1) \leq 2^{(\log n/4d)^{5/4}}$, we can apply Theorem A.1 and count the number of satisfying assignments for C'' i.e., $C'(a)$. We know the number of satisfying assignments for C by summing the numbers of satisfying assignments for $C'(a)$ over all $a \in \{0,1\}^t$. Recall that $T(n,m,d,w)$ is the running time of Theorem A.1. Combining the above argument and Theorem A.1, the running time on $C \in \mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ is at most

$$\begin{aligned}
& 2^t \cdot O(dt\mathrm{poly}(\log(mw)) \cdot m^{t+1}w^{t+1}) \cdot T(n, m(t+1), d, w^{t+1}m^t) \\
= \; & 2^t \cdot O(dt\mathrm{poly}(\log(mw)) \cdot m^{t+1}w^{t+1}) \cdot \mathrm{poly}(n, m(t+1), \log(w^{t+1}m^t)) \\
& \times 2^{n - \Omega\left((n/2^{2d(\log(m(t+1)))^{4/5}} \log(m^{t+1}w^{t+1}t))^{\frac{\log n}{9\log(m(t+1))}} \right)} \\
= \; & \mathrm{poly}(n,m,d,t,\log w) \cdot 2^t m^{t+1} w^{t+1} \cdot 2^{n - \Omega\left((n/2^{4d(\log m)^{4/5}} t \log(mw))^{\frac{\log n}{18\log m}} \right)} \; (\because t < m) \\
= \; & \mathrm{poly}(n,m,d,t,\log w) \cdot 2^{n + O(t\log mw) - \Omega\left((n/2^{4d(\log m)^{4/5}} t \log(mw))^{\frac{\log n}{18\log m}} \right)}.
\end{aligned}$$

□

In the above proof, we have the following lemma.

Lemma B.2. *The circuit class* $\mathbf{AC}^0_d[\mathbf{SYM}_w](n,m,t)$ *is contained in the circuit class* $\mathbf{OR}_{2^t} \circ \mathbf{SYM}_{w'} \circ \mathbf{AC}^0_d(n, m(t+1))$*, where* $w' = m^t w^{t+1}$.

By the idea of the proof of Theorem 5.1 in [8], we also have the following lemma.

Lemma B.3. *The circuit class* $\mathbf{AC}_d^0[\mathbf{SYM}_w](n,m,t)$ *is contained in the circuit class* $\mathbf{SYM}_{w'} \circ \mathbf{AC}_d^0(n, m2^{t+1})$, *where* $w' = (mw)^{2^{t+1}}$.

We use these lemmas in the proofs of our circuit lower bounds.

C Compression Algorithms

In this section, we describe the proof sketch of the following theorems.

Theorem C.1 (depth 2, weighted symmetric gate at the top, AND gates at the bottom)**.** *There exists a deterministic efficient and exponential space compression algorithm for* $C \in \mathbf{SYM}_w \circ \mathbf{AND}(n,m)$ *if*

$$(n/\log(mw))^{\log n/4\log(nm)} = \omega(\log n)$$

holds.

Theorem C.2 (depth d, weighted symmetric gate only at the top)**.** *There exists a deterministic efficient and exponential space compression algorithm for* $C \in \mathbf{SYM}_w \circ \mathbf{AC}_d^0(n,m)$ *if*

$$(n/2^{2d(\log m)^{4/5}} \log(mw))^{\log n/9\log m} = \omega(\log n)$$

holds.

Theorem C.3 (depth d, $t(n)$ weighted symmetric gates)**.** *There exists a deterministic efficient and exponential space compression algorithm for* $C \in \mathbf{AC}_d^0[\mathbf{SYM}_w](n,m,t)$ *if*

$$(n/2^{2d(\log m')^{4/5}} \log(m'w'))^{\log n/9\log m'} = \omega(\log n)$$

holds, where $m' = m2^{t+1}$ *and* $w' = (mw)^{2^{t+1}}$.

We formulate Circuit CMP as the set cover problem (SC) and apply the polynomial time approximation algorithm for SC. First we need some definitions. An (n',m,k,w)-*term* is a conjunction of literals and a circuit in $\mathbf{SYM}_w \circ \mathbf{AND}_k(n'',m)$, where $n'' \leq n'$. An (n',m,k,w)-*DNF* is a disjunction of (n',m,k,w)-terms. Let $\mathscr{S}(n',m,k,w)$ be the set of (n',m,k,w)-terms. Note that

$$|\mathscr{S}(n',m,k,w)| \leq 2^{O(n)} |\mathbf{SYM}_w \circ \mathbf{AND}_k(n',m)|.$$

Given a truth table T of length 2^n, we consider an SC instance $(U,\mathscr{S})$, where the universe $U = \{x \in \{0,1\}^n \mid T(x) = 1\}$ and the family $\mathscr{S} = \{t \in \mathscr{S}(n',m,k,w) \mid t^{-1}(1) \subseteq U\}$. Then $\mathscr{S}' \subseteq \mathscr{S}$ is *set cover* if $U = \cup_{t \in \mathscr{S}'} t^{-1}(1)$ holds.

It is easy to prove the following lemma using the greedy approximation algorithm for SC due to Johnson [36].

Lemma C.4. *Let* $(U,\mathscr{S})$ *be the SC instance defined as above and assume the instance has a set cover of cardinality* s. *If* $|\mathbf{SYM}_w \circ \mathbf{AND}_k(n',m)| = 2^{O(n)}$ *holds, then we can construct a set cover of cardinality at most* $O(ns)$ *deterministically in time* $2^{O(n)}$.

We see the following holds from the proofs of Theorems 1.1, 1.2 and Lemma B.3.

Lemma C.5. *Let* $\mathscr{C} \in \{\mathbf{SYM}_w \circ \mathbf{AND}(n,m), \mathbf{SYM}_w \circ \mathbf{AC}_d^0(n,m), \mathbf{AC}_d^0[\mathbf{SYM}_w](n,m,t)\}$ *such that parameters* n,m,w,d,t *satisfy the corresponding condition of Theorems C.1, C.2 and C.3, respectively. Then* $C \in \mathscr{C}$ *can be represented as a* $2^{n-\omega(\log n)}$*-sized* (n',m',k,w')*-DNF, where* $|\mathbf{SYM}_{w'} \circ \mathbf{AND}_k(n',m')| = 2^{O(n)}$ *holds.*

Combining these lemmas, we complete the proofs of Theorems C.1, C.2 and C.3.

D Proof of Lemma 4.2

The proof given here is essentially due to Chen, Kabanets, Kolokolova, Shaltiel and Zuckerman, see the proof of Lemma 4.3 in [17], except that we introduce $L_\ell(\cdot)$ and modify some parameters to measure the effect of bottom fan-in reduction rather than the shrinkage of De Morgan formulas.

Lemma D.1 (Restatement of Lemma 4.2)**.** *Let $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$. For all $n' \geq 4$, we have*

$$\Pr\left[L_\ell(C_{n-n'}) \geq 2^\ell \cdot L_\ell(C) \cdot \left(\frac{n'}{n}\right)^{\frac{\ell+2}{2}}\right] < 2^{-n'}.$$

We need the notion of super-martingales and a variant of Azuma's inequality for them.

Definition D.2. *A sequence of random variables $X_0, X_1, \ldots, X_n$ is a* super-martingale *with respect to a sequence of random variables $Y_0, Y_1, \ldots, Y_n$ if it satisfies $\mathbf{E}[X_i | Y_0, Y_1, \ldots, Y_{i-1}] \leq X_{i-1}$ for $1 \leq i \leq n$.*

Lemma D.3 (Lemma 4.2 in [17])**.** *Let $\{X_i\}_{i=0}^n$ be a super-martingale with respect to $\{Y_i\}_{i=0}^n$. Define $Z_i := X_i - X_{i-1}$ for $1 \leq i \leq n$. If, for $1 \leq i \leq n$, the random variable Z_i (conditioned on $Y_0, Y_1, \ldots, Y_{i-1}$) takes two values with equal probability, and there exists a constant $c_i \geq 0$ such that $Z_i \leq c_i$ holds, then, for all positive real λ, we have*

$$\Pr[X_n - X_0 \geq \lambda] \leq \exp\left(-\frac{\lambda^2}{2\sum_{i=1}^n c_i^2}\right).$$

We begin with a lemma that estimates the effect of greedy restriction.

Lemma D.4. *Let $C \in \mathbf{SYM}_w \circ \mathbf{AND}_k(n,m)$ and $x = \arg\max_{x \in \mathrm{var}(C)} \mathrm{freq}_\ell(C,x)$. Then, we have*

$$\max\{L_\ell(C|_{x=0}), L_\ell(C|_{x=1})\} \leq L_\ell(C) \cdot \left(1 - \frac{1}{n}\right)$$

and

$$\mathop{\mathbf{E}}_{a \in \{0,1\}}[L_\ell(C|_{x=a})] \leq L_\ell(C) \cdot \left(1 - \frac{1}{n}\right)^{\frac{\ell+2}{2}}.$$

Proof. Pick any t_i such that $|t_i| \geq \ell$ and $x \in \mathrm{var}(t_i)$. If $|t_i| = \ell$, we have $|t_i|_{x=a}| < \ell$ for all $a \in \{0,1\}$. If $|t_i| > \ell$, we have $t_i|_{x=a} \equiv 0$ and $|t_i|_{x=\neg a}| = |t_i| - 1$ for some $a \in \{0,1\}$. Since $\mathrm{freq}_\ell(C,x) \geq \frac{L_\ell(C)}{n}$, we have $\max\{L_\ell(C|_{x=0}), L_\ell(C|_{x=1})\} \leq L_\ell(C) \cdot \left(1 - \frac{1}{n}\right)$ and

$$\begin{aligned}\mathop{\mathbf{E}}_{a \in \{0,1\}}[L_\ell(C|_{x=a})] &\leq L_\ell(C) - \frac{L_\ell(C)}{n} \min\left\{\ell, \left(\frac{1}{2} \cdot (\ell+1) + \frac{1}{2} \cdot 1\right)\right\} \\ &= L_\ell(C)\left(1 - \frac{\ell+2}{2n}\right) \leq L_\ell(C) \cdot \left(1 - \frac{1}{n}\right)^{\frac{\ell+2}{2}}.\end{aligned}$$

□

Recall that we define a sequence of random variables $\{C_i\}$ inductively as $C_0 := C$ and $C_{i+1} := C_i|_{x=a}$, where $x = \arg\max_{x \in \mathrm{var}(C_i)} \mathrm{freq}_\ell(C_i, x)$ and a is a uniform random bit. We denote by Y_i the random bit assigned to the selected variables in step i for $1 \leq i \leq n$ and define $Y_0 := 0$. We define sequences of random variables $\{\mathscr{L}_i\}_{i=0}^n, \{l_i\}_{i=0}^n, \{Z_i\}_{i=1}^n$ as follows: $\mathscr{L}_i := L_\ell(C_i)$, $l_i := \ln \mathscr{L}_i$ and

$$Z_i := l_i - l_{i-1} - \frac{\ell+2}{2} \ln\left(1 - \frac{1}{n-i+1}\right).$$

Note that, given $Y_0, Y_1, \ldots, Y_{i-1}$, the random variable Z_i takes two values with equal probability.

Lemma D.5. *Define* $X_0 := 0$ *and* $X_i := \sum_{j=1}^{i} Z_j$. *Then, the sequence of random variables* $\{X_i\}_{i=0}^{n}$ *is a super-martingale with respect to* $\{Y_i\}_{i=0}^{n}$ *and for each* Z_i*, we have* $Z_i \le c_i := -\frac{\ell}{2}\ln(1-\frac{1}{n-i+1})$.

Proof. By the first inequality of Lemma D.4, we have $l_i \le l_{i-1} + \ln\left(1-\frac{1}{n-i+1}\right)$. This implies $Z_i = l_i - l_{i-1} - \frac{\ell+2}{2}\ln\left(1-\frac{1}{n-i+1}\right) \le -\frac{\ell}{2}\ln\left(1-\frac{1}{n-i+1}\right) = c_i$. By Jensen's inequality, we have $\mathbf{E}[l_i|Y_0,Y_1,\dots,Y_{i-1}] \le \ln\mathbf{E}[\mathscr{L}_i|Y_0,Y_1,\dots,Y_{i-1}]$. By the second inequality of Lemma D.4, the right hand side is at most $\ln\left(\mathscr{L}_{i-1}\cdot\left(1-\frac{1}{n-i+1}\right)^{\frac{\ell+2}{2}}\right) = l_{i-1} + \frac{\ell+2}{2}\ln\left(1-\frac{1}{n-i+1}\right)$. This implies $\mathbf{E}[Z_i|Y_0,Y_1,\dots,Y_{i-1}] \le 0$, that is, $\mathbf{E}[X_i|Y_0,Y_1,\dots,Y_{i-1}] \le \mathbf{E}[X_{i-1}|Y_0,Y_1,\dots,Y_{i-1}] = X_{i-1}$. Thus, $\{X_i\}_{i=1}^{n}$ is a super-martingale. □

Now we are ready to prove Lemma D.1.

Proof of Lemma D.1. Let λ be arbitrary positive real and c_i's be as defined in Lemma D.5. By Lemma D.3 and Lemma D.5, we obtain

$$\Pr\left[\sum_{j=1}^{i} Z_j \ge \lambda\right] \le \exp\left(-\frac{\lambda^2}{2\sum_{j=1}^{i} c_j^2}\right).$$

It is easy to show that $\sum_{j=1}^{i} Z_j = l_i - l_0 - \frac{\ell+2}{2}\ln\frac{n-i}{n}$ by the definition of Z_j. Thus, we have

$$\begin{aligned}\Pr\left[\sum_{j=1}^{i} Z_j \ge \lambda\right] &= \Pr\left[l_i - l_0 - \frac{\ell+2}{2}\ln\left(\frac{n-i}{n}\right) \ge \lambda\right] \\ &= \Pr\left[\mathscr{L}_i \ge e^{\lambda}\mathscr{L}_0\left(\frac{n-i}{n}\right)^{\frac{\ell+2}{2}}\right].\end{aligned}$$

For $1 \le j \le n-n'$, we have $c_j = -\frac{\ell}{2}\ln\left(1-\frac{1}{n-j+1}\right) \le \frac{\ell}{2}\cdot\frac{\sqrt{2\ln 2}}{n-j+1}$, using the inequality $-\ln(1-x) \le \sqrt{2\ln 2}\cdot x$ for $0 < x \le 1/4$. Thus, for $1 \le i \le n-n'$, $\sum_{j=1}^{i} c_j^2$ is at most

$$\begin{aligned}\frac{\ell^2\ln 2}{2}\sum_{j=1}^{i}\left(\frac{1}{n-j+1}\right)^2 &\le \frac{\ell^2\ln 2}{2}\sum_{j=1}^{i}\left(\frac{1}{n-j} - \frac{1}{n-j+1}\right) = \frac{\ell^2\ln 2}{2}\left(\frac{1}{n-i} - \frac{1}{n}\right) \\ &\le \frac{\ell^2\ln 2}{2}\cdot\frac{1}{n-i}.\end{aligned}$$

Setting $i = n-n'$, we obtain

$$\begin{aligned}\Pr\left[\mathscr{L}_{n-n'} \ge e^{\lambda}\mathscr{L}_0\left(\frac{n'}{n}\right)^{\frac{\ell+2}{2}}\right] &\le \exp\left(-\frac{\lambda^2}{2\sum_{j=1}^{n-n'} c_j^2}\right) \\ &\le e^{-\frac{1}{\ell^2\ln 2}\lambda^2 n'}.\end{aligned}$$

Choosing $\lambda = \ell\ln 2$ completes the proof. □

E Proof of Corollary 6.6

Proof. Let us denote $\Pr_x[g(x) = a, f(x) = b]$ by $\Pr[a,b]$ and $\Pr_x[g(x) = a \mid f(x) = b]$ by $\Pr[a|b]$. The values p_0 and p_1 denotes $\Pr[f = 0]$ and $\Pr[f = 1]$, respectively. Without loss of generality, we can suppose that

$p_0 \geq p_1$, i.e., $p_0 = \frac{1}{2} + \varepsilon$ and $p_1 = \frac{1}{2} - \varepsilon$.

$$\begin{aligned}
\mathrm{Corr}(f,g) &= |\,\Pr[0,0] + \Pr[1,1] - \Pr[1,0] - \Pr[0,1]| \\
&= |2(\Pr[0,0] + \Pr[1,1]) - 1| \\
&= |2\,(\Pr[0|0]p_0 + \Pr[1|1]p_1) - 1| \\
&= |\Pr[0|0](1+2\varepsilon) + \Pr[1|1](1-2\varepsilon) - 1| \\
&= |(1-\Pr[1|0])(1+2\varepsilon) + \Pr[1|1](1-2\varepsilon) - 1| \\
&= |2\varepsilon - (1+2\varepsilon)\Pr[1|0] + \Pr[1|1](1-2\varepsilon)| \\
&= |\Pr[1|1] - \Pr[1|0] + 2\varepsilon\{1 - \Pr[1|0] - \Pr[1|1]\}| \\
&\geq \frac{1}{k} - 2\varepsilon\,|\{1 - \Pr[1|0] - \Pr[1|1]\}| \geq \frac{1}{k} - 2\varepsilon
\end{aligned}$$

□