SIAM J. COMPUT.
Vol. 38, No. 4, pp. 1347–1363

RESOLUTION IS NOT AUTOMATIZABLE UNLESS W[P] IS TRACTABLE*

MICHAEL ALEKHNOVICH† AND ALEXANDER A. RAZBOROV‡

Abstract. We show that neither resolution nor tree-like resolution is automatizable unless the class **W[P]** from the hierarchy of parameterized problems is fixed-parameter tractable by randomized algorithms with one-sided error.

Key words. proof complexity, resolution, automatizability

AMS subject classifications. 03F20, 03D15

DOI. 10.1137/06066850X

1. Introduction.

Analysis of the usefulness of proof search heuristics and automated theorem proving procedures based on a proof system P amounts (on the theoretical level) to the following two basic questions:

Question 1. Which theorems in principle possess efficient P-proofs?

Question 2. How can one find the optimal (or, at least, a nearly optimal) proof of a given theorem in P?

Traditional proof complexity mostly dealt, and still deals, with the first question. However, there has been a growing interest in the second one, too. An additional motivation for studying the complexity of finding optimal proofs comes from deep connections with efficient interpolation theorems; we refer the reader to the surveys [9, 19, 22] for more details. These surveys also serve as a good starting point for learning more about propositional proof complexity in general.

One convenient framework for the theoretical study of Question 2 was proposed by Bonet, Pitassi, and Raz in [13]. Namely, they called a proof system P *automatizable* if there exists a deterministic algorithm A which, given a tautology τ, returns its P-proof in time polynomial *in the size of the shortest P-proof of τ*. The definition of a quasi-automatizable proof system is given in the same way, but we only require algorithm A to run in time which is quasi-polynomial (in the same parameter).

One advantage of this definition is that it allows us to completely disregard the first basic question on the *existence* of efficient P-proofs and to indeed concentrate on *finding* efficient proofs *provided* they exist. In particular, the notion of automatizability makes perfect sense for those (weak) proof systems for which hard tautologies are already known. Moreover, the weaker our system is, the more likely it seems to be automatizable. One possible explanation of this phenomenon comes from the connection between automatizability and efficient interpolation (every automatizable proof system has efficient interpolation, and the property of having efficient interpolation is indeed antimonotone w.r.t. the strength of the system). Anyway, given

*Received by the editors August 29, 2006; accepted for publication (in revised form) April 14, 2008; published electronically August 6, 2008. A preliminary version of this paper appeared in the *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science, (Las Vegas, NV)*, 2001.

http://www.siam.org/journals/sicomp/38-4/66850.html

†This author is deceased. Former address: Department of Mathematics, University of California at San Diego, La Jolla, CA 92093.

‡Department of Computer Science, University of Chicago, Chicago, IL 60637 (razborov@cs.uchicago.edu). This author's research was supported by the State of New Jersey and NSF grant CCR-9987077.

this connection, the results from [20, 13] imply that extended Frege and TC^0-Frege proof systems, respectively, are not automatizable under some widely believed cryptographic assumptions. Bonet et al. [11] extended the latter result to bounded-depth Frege but under a much stronger assumption.

In this paper we are primarily interested in the automatizability of resolution and tree-like resolution. It is worth noting that both systems possess efficient interpolation, and therefore their nonautomatizability cannot be proved via techniques similar to those in [20, 13, 11]. Nonetheless, [18] proved that it is **NP**-hard to find the shortest resolution refutation. Alekhnovich et al. [3] proved that if $\mathbf{P} \neq \mathbf{NP}$, then the length of the shortest resolution refutation cannot be approximated to within a constant factor (both for general and tree-like resolution). Under the stronger assumption $\mathbf{NP} \not\subseteq \mathbf{QP}$, they were able to improve the ratio from an arbitrary constant to $2^{\log^{1-\epsilon} n}$ (later, Dinur and Safra [16] obtained a better probabilistically checkable proofs (PCP) characterization of **NP** that allows one to prove the same bound for arbitrary $\epsilon \to 0$ modulo $\mathbf{P} \neq \mathbf{NP}$).

In the opposite direction, Beame and Pitassi [8] observed that tree-like resolution is quasi-automatizable. Thus, it is unlikely to show that this system is not automatizable modulo $\mathbf{P} \neq \mathbf{NP}$, because it would imply quasi-polynomial algorithms for **NP** (in case of general resolution this goal seems also tricky at the moment because there is only one[1] known example [12] for which the proof search algorithm of [10] requires more than quasi-polynomial time). Therefore, any result establishing nonautomatizability of tree-like resolution needs to be formulated within a complexity framework in which the asymptotics $n^{O(1)}$ and $n^{\log n}$ are essentially different.

One natural example of such a framework is *parameterized complexity* introduced by Downey and Fellows (see [17]) in which algorithms working in times $f(k)n^{O(1)}$ and n^k are considered different from the point of view of effectiveness (here k is an integer input parameter that should be thought of as an "arbitrarily large" constant). In this paper we prove that neither resolution nor tree-like resolution is automatizable unless the class $\mathbf{W}[\mathbf{P}]$ (lying very high in the hierarchy of parameterized problems) is fixed-parameter tractable by a randomized algorithm with one-sided error (Theorem 2.7). Our proof goes by a reduction from the optimization problem MINIMUM MONOTONE CIRCUIT SATISFYING ASSIGNMENT (MMCSA) whose decision version is complete for the class $\mathbf{W}[\mathbf{P}]$. An alternative hardness assumption is that there is no *deterministic* fixed-parameter algorithm which *approximates* MMCSA within any constant factor (Theorem 2.5). It is worth noting in this connection that we were able to relate to each other the hardness of finding *exact* and *approximate* solutions for MMCSA without using the PCP theorem (see the proof of Theorem 2.7 given in section 4). This result can be interesting on its own.

The paper is organized as follows. Section 2 contains necessary preliminaries and definitions. In section 3 we present our core reduction from MMCSA to automatizability of resolution, and in section 4 we use (sometimes nontrivial) self-improving techniques to prove our main results, Theorems 2.5 and 2.7. The paper is concluded with some open problems in section 5.

1.1. Recent developments. Since the preliminary version of this paper was released, the following related developments have occurred.

Atserias and Bonet [6] studied a slightly different variant of automatizability that they called *weak automatizability*. Using their techniques, they were also able

[1] See, however, section 1.1.

to produce more examples of poly-size tautologies for which the width-based proof search algorithm from [10] requires more than quasi-polynomial time. In the opposite direction, Alekhnovich and Razborov [4] introduced an enhancement of that algorithm which they called BWBATP (branch-width based automated theorem prover). This algorithm performs better than the width-based algorithm for several important classes of tautologies, and for at least one such class it even achieves complete (that is, polynomial) automatization. Finally, quite unexpectedly our techniques turned out to be useful in the totally different area of computational learning, where they inspired a number of strong hardness results for the so-called model of proper learning [2].

2. Preliminaries and main results.

2.1. Resolution and automatizability. Let x be a Boolean variable, i.e., a variable that ranges over the set $\{0,1\}$. A *literal* of x is either x (denoted sometimes as x^1) or $\bar{x}$ (denoted sometimes as x^0). A *clause* is a disjunction of literals. A *CNF* (conjunctive normal form) is a conjunction of pairwise different clauses.

Let $f(x_1,\dots,x_n)$ be an arbitrary function (possibly, partial) from $\{0,1\}^n$ to some finite domain D. An *assignment to* f is a mapping $\alpha : \{x_1,\dots,x_n\} \to \{0,1\}$. A *restriction of* f is a mapping $\rho : \{x_1,\dots,x_n\} \to \{0,1,\star\}$. We denote by $|\rho|$ the number of assigned variables, $|\rho| \stackrel{\text{def}}{=} |\rho^{-1}(\{0,1\})|$. The *restriction of a function* f *or CNF* τ *by* ρ, denoted by $f|_\rho$ [$\tau|_\rho$], is the function [CNF] obtained from f [τ, respectively] by setting the value of each $x \in \rho^{-1}(\{0,1\})$ to $\rho(x)$, and leaving each $x \in \rho^{-1}(\star)$ as a variable.

The general definition of a propositional proof system was given in the seminal paper [15]. But since we are interested only in resolution (which is one of the simplest and most widely studied concrete systems), we prefer to skip this general definition. *Resolution* operates with clauses and has one rule of inference called *resolution rule*:

$$\frac{A \vee x \qquad B \vee \bar{x}}{A \vee B}.$$

A resolution proof is *tree-like* if its underlying graph is a tree. A *resolution refutation* of a CNF τ is a resolution proof of the empty clause from the clauses appearing in τ.

The *size* of a resolution proof is the overall number of clauses in it. For an unsatisfiable CNF τ, $S(\tau)$ [$S_T(\tau)$] is the minimal size of its resolution refutation (tree-like resolution refutation, respectively). Clearly, $S(\tau) \le S_T(\tau)$.

The *width* $w(C)$ *of a clause* C is the number of literals in C. The *width* $w(\tau)$ *of a set of clauses* τ (in particular, the width of a resolution proof) is the maximal width of a clause appearing in this set. For a CNF τ, let $n(\tau)$ be the overall number of distinct variables appearing in it, and let $|\tau|$ be the overall number of occurrences of variables in τ, i.e., $|\tau| \stackrel{\text{def}}{=} \sum_{C\in\tau} w(C)$. For an unsatisfiable CNF τ, $w(\tau \vdash \emptyset)$ will denote the minimal width of its resolution refutation.

For a nonnegative integer n, let $[n] \stackrel{\text{def}}{=} \{1,2,\dots,n\}$, and let $[n]^k \stackrel{\text{def}}{=} \{\, I \subseteq [n] \mid |I| = k \,\}$.

We will recall the general definition of automatizability from [13] for the special cases of resolution and tree-like resolution.

DEFINITION 2.1. *Resolution (tree-like resolution) is* (quasi-)automatizable *if there exists a deterministic algorithm* A *which, given an unsatisfiable CNF* τ*, returns its resolution refutation (tree-like resolution refutation, respectively) in time which is (quasi-)polynomial in* $|\tau| + S(\tau)$ *(*$|\tau| + S_T(\tau)$*, respectively).*

Remark 1. Note that we do not require that all clauses from τ must necessarily appear in the refutation, and therefore, we cannot a priori expect the inequality $n(\tau) \cdot S(\tau) \geq |\tau|$. This is why we must introduce the term $|\tau|$ into the bound on the running time of A when adapting the general definition of automatizability from [13] to the case of resolution.

2.2. Parameterized complexity and the MMCSA problem. We refer the reader to [17] for a good general introduction to the topic of parameterized complexity.

DEFINITION 2.2 (see [17, Definition 2.4]). *The class* **FPT** (fixed-parameter tractable) *of parameterized problems consists of all languages $L \subseteq \Sigma^* \times \mathbb{N}$ for which there exists an algorithm Φ, a constant c, and a recursive function $f : \mathbb{N} \longrightarrow \mathbb{N}$ such that*

1. *the running time of $\Phi(\langle x, k\rangle)$ is at most $f(k) \cdot |x|^c$;*
2. *$\langle x, k\rangle \in L$ iff $\Phi(\langle x, k\rangle) = 1$.*

Thus, an algorithm is considered to be feasible if it works in time polynomial in $n \stackrel{\text{def}}{=} |x|$ and $f(k)$, where k should be thought of as much smaller than n, and f is an arbitrarily large (recursive) function. A similar feasibility requirement arises in the theory of polynomial-time approximation schemes (PTAS) for **NP**-hard problems: assume that we have an algorithm that approximates a given problem within arbitrary error $\epsilon > 0$ working in time $n^{O(1/\epsilon)}$. Is it possible to get rid of $1/\epsilon$ in the exponent and do it in time $f(1/\epsilon)n^{O(1)}$? (The algorithms which obey the latter bound on the running time are called EPTAS, efficient polynomial-time approximation schemes.)

It turns out that this question is tightly related to the fixed-parameter tractability. Namely, the existence of EPTAS for a given problem implies an *exact* algorithm for the corresponding fixed-parameter version (see [7, 14]).

To study the complexity of parameterized problems, the following *parameterized reduction* (that preserve the property of being in **FPT**) is used.

DEFINITION 2.3 (see [17, Definition 9.3]). *A parameterized problem $L \subseteq \Sigma^* \times \mathbb{N}$* reduces *to another parameterized problem $L' \subseteq \Sigma^* \times \mathbb{N}$ if there exist (arbitrary!) functions $f, g : \mathbb{N} \longrightarrow \mathbb{N}$, and a function $h : \Sigma^* \times \mathbb{N} \longrightarrow \Sigma^*$ such that $h(x, k)$ is computable in time $f(k)|x|^{O(1)}$, and $\langle x, k\rangle \in L$ iff $\langle h(x, k), g(k)\rangle \in L'$.*

For any integer t, the parameterized problem WEIGHTED t-NORMALIZED SATISFIABILITY is defined by restricting the ordinary SATISFIABILITY to a certain class of Boolean formulas depending on t (we omit the exact definition since it is a little bit technical and not needed for our results), and the parameter k bounds the Hamming weight of the satisfying assignment we are searching for. The complexity class $\mathbf{W[t]}$ consists of all problems that can be reduced to WEIGHTED t-NORMALIZED SATISFIABILITY via parameterized reduction, and the class $\mathbf{W[P]}$ (where P stands for polynomial) includes all problems reducible to WEIGHTED CIRCUIT SATISFIABILITY described as follows:

WEIGHTED CIRCUIT SATISFIABILITY:
Instance: A circuit C.
Parameter: A positive integer k.
Question: Does C have a satisfying assignment of Hamming weight (defined as the number of ones) k?

These definitions lead to the following *parameterized hierarchy*, in which every inclusion is believed to be strict:

$$\mathbf{FPT} \subseteq \mathbf{W[1]} \subseteq \mathbf{W[2]} \cdots \subseteq \mathbf{W[P]}.$$

In our paper we construct a randomized parameterized reduction from the automatizability of resolution to the following optimization problem (MMCSA in what follows) that was introduced in [3].

Monotone minimum circuit satisfying assignment:
Instance: A monotone circuit C in n variables over the basis $\{\wedge, \vee\}$.
Solution: An assignment $a \in \{0,1\}^n$ such that $C(a) = 1$.
Objective function: $k(a)$, defined as its Hamming weight.

By $k(C)$ we will denote the minimal value $k(a)$ of a solution a for an instance C of MMCSA.

The following easy observation was made in [3] ("self-improvement").

PROPOSITION 2.4. *For every fixed integer $d > 0$ there exists a polynomial-time computable function π which maps monotone circuits into monotone circuits and such that $k(\pi(C)) = k(C)^d$ for all C.*

Our first result can be now formulated as follows.

THEOREM 2.5. *If either resolution or tree-like resolution is automatizable, then for any fixed $\epsilon > 0$ there exists an algorithm Φ receiving as inputs monotone circuits C which runs in time $\exp\left(k(C)^{O(1)}\right) \cdot |C|^{O(1)}$ and approximates the value of $k(C)$ to within a factor $(1+\epsilon)$.*

The decision version of MMCSA was considered in [17] (under the name WEIGHTED MONOTONE CIRCUIT SATISFIABILITY) in the context of parameterized complexity and was shown to be complete in the class **W[P]**.

In order to formulate our second (and main) result, we need to introduce the obvious hybrid of the classes **R** and **FPT**.

DEFINITION 2.6. *The class* **FPR** *(fixed-parameter randomized) of parameterized problems consists of all languages $L \subseteq \Sigma^* \times \mathbb{N}$ for which there exists a probabilistic algorithm Φ, a constant c, and a recursive function $f : \mathbb{N} \longrightarrow \mathbb{N}$ such that*

1. *the running time of $\Phi(\langle x, k\rangle)$ is at most $f(k) \cdot |x|^c$;*
2. *if $\langle x, k\rangle \in L$, then $\mathbf{P}[\Phi(\langle x, k\rangle) = 1] \geq 1/2$;*
3. *if $\langle x, k\rangle \notin L$, then $\mathbf{P}[\Phi(\langle x, k\rangle) = 1] = 0$.*

Then we have the following.

THEOREM 2.7. *If either resolution or tree-like resolution is automatizable, then* $\mathbf{W[P]} \subseteq$ *co-*$\mathbf{FPR}$.

3. Main reduction from MMCSA to automatizability of resolution. This section is entirely devoted to the proof of the following technical lemma.

LEMMA 3.1. *There exists a polynomial-time computable function τ which maps any pair $\langle C, 1^m\rangle$, where C is a monotone circuit and m is an integer, to an unsatisfiable CNF $\tau(C, m)$ such that*

$$S_T(\tau(C,m)) \leq |C| \cdot m^{O(\min\{k(C),\ \log m\})}$$

and

$$S(\tau(C,m)) \geq m^{\Omega(\min\{k(C),\ \log m\})}. \tag{1}$$

We begin the proof of Lemma 3.1 by describing CNFs, which form the main building block $\tau(C, m)$, and establishing their necessary properties. From now on fix a monotone circuit C in n variables $p_1, \ldots, p_n$. Let $\mathcal{A} \subseteq \{0,1\}^m$. We will call vectors from $\mathcal{A}$ (usually represented as columns) *admissible* and call a 0-1 matrix with m rows $\mathcal{A}$-*admissible* if all its columns are so. Consider the following combinatorial principle

$\mathcal{P}_{C,\mathcal{A}}$ (that may be true or false, depending on the choice of C and $\mathcal{A}$):

$\mathcal{P}_{C,\mathcal{A}}$: *every* $(m \times n)$ 0-1 $\mathcal{A}$*-admissible matrix* $A = (a_{ij})$ *contains a row* $i \in [m]$ *such that* $C(a_{i1}, a_{i2}, \ldots, a_{in}) = 1$.

Let us formulate one sufficient condition for $\mathcal{P}_{C,\mathcal{A}}$ to be true (regardless of proof complexity considerations).

DEFINITION 3.2. *$d_1(\mathcal{A})$ is the maximal d such that for every d vectors from $\mathcal{A}$ there exists a position $i \in [m]$ in which all these vectors have* 1.

$d_1(\mathcal{A})$ can also be easily characterized in terms of minimum covers. Namely, if we associate with every

$$\begin{pmatrix} a_1 \\ \vdots \\ a_m \end{pmatrix} \in \mathcal{A}$$

the subset $\{\, i \in [m] \mid a_i = 0 \,\}$ of $[m]$, then $d_1(\mathcal{A}) + 1$ is exactly the minimal number of such sets needed to cover the whole $[m]$.

LEMMA 3.3. *If $k(C) \le d_1(\mathcal{A})$, then $\mathcal{P}_{C,\mathcal{A}}$ is true.*

Proof. Let A be an $(m \times n)$ 0-1 $\mathcal{A}$-admissible matrix. Let $a = (a_1, \ldots, a_n)$ be such that $C(a_1, \ldots, a_n) = 1$ and $k(a) = k(C)$. Let

$$\mathcal{A}_0 \stackrel{\text{def}}{=} \left\{ \left(\begin{array}{c} a_{1j} \\ \vdots \\ a_{mj} \end{array} \right) \middle| \; a_j = 1 \right\}$$

be the set of all columns in A corresponding to those positions j for which $a_j = 1$. Since $|\mathcal{A}_0| \le k(a) = k(C) \le d_1(\mathcal{A})$, there exists $i \in [m]$ such that $a_{ij} = 1$ whenever $a_j = 1$. This means $a_{ij} \ge a_j$ for all $j \in [n]$ and implies $C(a_{i1}, \ldots, a_{in}) = 1$ since C is monotone. ∎

The proof of Lemma 3.3 suggests that if C and $\mathcal{A}$ with the property $k(C) \le d_1(\mathcal{A})$ are "generic enough," then the optimal propositional proof of the principle $\mathcal{P}_{C,\mathcal{A}}$ should exhaustively search through all $|\mathcal{A}|^{k(C)}$ possible placements of admissible vectors to the columns $\{\, j \mid a_j = 1 \,\}$ and thus have size roughly $|\mathcal{A}|^{k(C)}$. Our task is to find an encoding of (the negation of) $\mathcal{P}_{C,\mathcal{A}}$ as a CNF so that we can prove tight upper and lower bounds on $S_T(\tau(C, \mathcal{A}))$ and $S(\tau(C, \mathcal{A}))$ of (roughly) this order. This encoding is somewhat technical and involves several auxiliary functions (see Definition 3.4 below). In order to convey why we need all of these, let us briefly discuss two "naive" attempts at a simpler proof.

Attempt 1 (no encoding at all). Suppose that we simply enumerate elements of $\mathcal{A}$ by binary strings of length $\log |\mathcal{A}|$ and introduce propositional variables expressing their bits. The main problem with this encoding is that it does not behave well with respect to (random) restrictions. The standard width-reducing argument from [8] that we use in part (c) of Lemma 3.8 below assumes a "reasonably uniform" distribution on the set of those restrictions that "reasonably preserve" the complexity of the tautology. But with the straightforward encoding, any restriction of propositional variables used for enumerating the set $\mathcal{A}$ results in shrinking this set and completely destroys its useful properties.

We circumvent this in a standard way by using "excessive encodings" $F_1, \ldots, F_n : \{0,1\}^s \longrightarrow \mathcal{A}$, where $F_i(x_1, \ldots, x_s)$ are surjective and remain so after restricting not too many variables (Definition 3.5). It is worth noting that even if we may have assumed in our definition of $\tau(C, \mathcal{A})$ that $F_1 = F_2 = \cdots = F_n$, this property will *not*

be invariant under restrictions (and this is why it is more convenient not to make this assumption).

Attempt 2 (same encoding for $\mathcal{A}$ and C). The naive encoding of the circuit C (that is, by propositional variables z_{iv} encoding the intermediate result of the computation by the circuit C at the gate v when its input is the ith row of A) suffers from the same drawback as above, which is that we do not want the values of z_{iv} to be exposed by a random restriction. But why do we not apply to the variables z_{iv} just the same excessive encodings we used above for the elements of $\mathcal{A}$?

It turns out that with this "lighter" version our lower bounds already go through, and the upper bound holds for *general* resolution (in particular, the reader interested in only this case can safely assume this simplification). In the tree-like case, however, the upper bound becomes problematic. Namely, when formalizing the proof of Lemma 3.3, we need to prove the fact $C(a_{i_1},\dots,a_{i_n})=1$, and the natural way of doing this in *tree-like* resolution assumes full access to clauses of the form $(z_{i,v_1}\wedge\cdots\wedge z_{i,v_\mu}\supset z_{i,v})$ (cf. the proof of part (a) of Lemma 3.8). This is not a problem if the variables $z_{i,v}$ are not encoded, but if we encode them in a nontrivial way, then we no longer will have a resolution proof.

In order to balance between these two conflicting requirements, we introduce a more sophisticated encoding scheme that intuitively looks as follows. Imagine that we have many independent copies $C_1,\dots,C_r$ of the circuit C; indices $c\in\{1,2,\dots,r\}$ will be called *controls*. The (unencoded!) variables $z^c_{i,v}$ will again express the protocol of computing the value $C_c(a_{i_1},\dots,a_{i_n})$. But for every individual row $i\in[m]$, our axioms will require this protocol to be valid only for *one* of these r circuits (say, C_{c_i}), and the values c_i *are* excessively encoded by surjective mappings $f_1,\dots,f_m:\{0,1\}^s\longrightarrow[r]$ in the same way as we did with the elements of $\mathcal{A}$.

In order to not obstruct the proof with irrelevant details, we will define our CNFs $\tau(C,\mathcal{A},\vec{F},\vec{f})$ and establish their necessary properties in a situation which is more general than what will be actually needed for completing the proof of Lemma 3.1. If the reader prefers, he/she may think during the course of the proof that $\mathcal{A}$ is an arbitrary set of vectors such that $d_1(\mathcal{A})\ge\Omega(\log m)$ and (see Definition 3.6) $d_0(\mathcal{A})\ge\Omega(\log m)$. Furthermore, $r=\log m$, $s=O(\log m)$, and F_j, f_i will be $(\log m)$-surjective in the sense of Definition 3.5.

DEFINITION 3.4. *Let $C(p_1,\dots,p_n)$ be a monotone circuit, $\mathcal{A}\subseteq\{0,1\}^m$ be a set of vectors, and $F_1,\dots,F_n:\{0,1\}^s\longrightarrow\mathcal{A}$, $f_1,\dots,f_m:\{0,1\}^s\longrightarrow[r]$ be surjective functions, where f_is are possibly partial. For every $j\in[n]$ and $\nu\in[s]$ we introduce a propositional variable x^ν_j, for every $i\in[m]$ and $\nu\in[s]$ introduce a variable y^ν_i, and for every $i\in[m]$, every $c\in[r]$ (elements of this set will be sometimes referred to as* controls*), and every vertex v of the circuit C introduce a variable z^c_{iv}.*

For $j\in[n]$ and $\vec{a}\in\mathcal{A}$, let us denote by $[Column_j=\vec{a}]$ the predicate $F_j(x^1_j,\dots,x^s_j)=\vec{a}$. Likewise, for $i\in[m]$ and $c\in[r]$, let $[Control_i=c]$ denote the predicate "$f_i(y^1_i,\dots,y^s_i)$ is defined and $f_i(y^1_i,\dots,y^s_i)=c$."

The CNF $\tau(C,\mathcal{A},\vec{F},\vec{f})$ consists of all clauses that result from the expansion of the following Boolean predicates as CNFs:

$$(y^1_i,\dots,y^s_i)\in dom(f_i)\ \text{for all } i\in[m]; \tag{2}$$

$$\left.\begin{array}{c}([Column_j=\vec{a}]\wedge[Control_i=c])\supset z^c_{i,p_j}\\ \text{for all } \vec{a}\in\mathcal{A},\ i\in[m]\ \text{such that } a_i=1\ \text{and all } j\in[n],\ c\in[r];\end{array}\right\} \tag{3}$$

$$\left.\begin{array}{c} \big([Control_i = c] \wedge (z^c_{i,v'} * z^c_{i,v''})\big) \supset z^c_{iv} \\ \text{for all } i \in [m],\ c \in [r] \text{ and all internal nodes } v \\ \text{corresponding to the instruction } v \leftarrow v' * v'',\ * \in \{\wedge, \vee\}; \end{array}\right\} \tag{4}$$

$$[Control_i = c] \supset \bar{z}_{i,v_{\text{fin}}}, \text{ where } v_{\text{fin}} \text{ is the output node of } C. \tag{5}$$

It is easy to see that $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ is unsatisfiable (for arbitrary surjective $\vec{F}, \vec{f}$) iff $P_{C,\mathcal{A}}$ is true. Also, as we already mentioned, the only thing we need from $\vec{F}, \vec{f}$ is that they remain surjective after restricting a few variables.

DEFINITION 3.5. *We say that an onto (possibly partial) function $g : \{0,1\}^s \longrightarrow D$ is* r-surjective *if for any restriction ρ with $|\rho| \le r$ the function $g|_\rho$ is still onto.*

Finally, for the lower bound we need a notion dual to $d_1(\mathcal{A})$.

DEFINITION 3.6. *$d_0(\mathcal{A})$ is the maximal d such that for every d positions $i_1, \dots, i_d \in [m]$ there exists $\vec{a} \in \mathcal{A}$ such that $a_{i_1} = \dots = a_{i_d} = 0$.*

Now we are ready to formulate our main technical lemma that provides upper and lower bounds on the size of optimal resolution refutations of $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$. Like many similar proofs in the area, the lower bound is naturally split into two fairly independent parts. The first part provides lower bounds on $w(\tau(C, \mathcal{A}, \vec{F}, \vec{f}) \vdash \emptyset)$, but for technical reasons we need a slightly stronger statement based on a modified notion of width.

DEFINITION 3.7. *For a clause D in the variables of the CNF $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$, let $w_x(D), w_y(D)$, and $w_c(D)$ ($c \in [r]$) be the numbers of x-variables, y-variables, and z-variables of the form $z^c_{i,v}$, respectively, appearing in D. We define the* controlled width $\widetilde{w}(D)$ *as*

$$\widetilde{w}(D) \stackrel{\text{def}}{=} w_x(D) + w_y(D) + r \cdot \min_{c \in [r]} w_c(D).$$

The minimal controlled width $\widetilde{w}(\tau(C, \mathcal{A}, \vec{F}, \vec{f}) \vdash \emptyset)$ *is defined similarly to the minimal refutation width.*

Clearly, $\widetilde{w}(D) \le w(D)$ for any clause D, and thus $\widetilde{w}(\tau(C, \mathcal{A}, \vec{F}, \vec{f}) \vdash \emptyset) \le w(\tau(C, \mathcal{A}, \vec{F}, \vec{f}) \vdash \emptyset)$.

LEMMA 3.8. *Let C be a monotone circuit in n variables, let $\mathcal{A} \subseteq \{0,1\}^m$, and let $F_1, \dots, F_n : \{0,1\}^s \longrightarrow \mathcal{A}$, $f_1, \dots, f_m : \{0,1\}^s \longrightarrow [r]$ be r-surjective functions, where the f_i's are possibly partial and m, r, s are arbitrary integer parameters. Then the following bounds hold:*

(a) (*cf. Lemma* 3.3). *If $k(C) \le d_1(\mathcal{A})$, then $S_T(\tau(C, \mathcal{A}, \vec{F}, \vec{f})) \le O(|C| \cdot 2^{s(k(C)+1)})$.*

(b) $\widetilde{w}(\tau(C, \mathcal{A}, \vec{F}, \vec{f}) \vdash \emptyset) \ge \frac{r}{2} \cdot \min\{k(C), d_0(\mathcal{A})\}$.

(c) $S(\tau(C, \mathcal{A}, \vec{F}, \vec{f})) \ge \exp(\Omega(\frac{r^2}{s} \cdot \min\{k(C), d_0(\mathcal{A})\}))$.

Proof of Lemma 3.8.

Part (a). We show this part by formalizing the proof of Lemma 3.3. Let $k \stackrel{\text{def}}{=} k(C)$ and $a_1, \dots, a_n$ be such that $C(a_1, \dots, a_n) = 1$ and $k(a) = k$. Assume for simplicity

that $a_1 = \cdots = a_k = 1,\ a_{k+1} = \cdots = a_n = 0$. Fix arbitrary admissible vectors

$$\vec{a}_1 \stackrel{\text{def}}{=} \begin{pmatrix} a_{i1} \\ \vdots \\ a_{m1} \end{pmatrix}, \ldots, \vec{a}_k \stackrel{\text{def}}{=} \begin{pmatrix} a_{ik} \\ \vdots \\ a_{mk} \end{pmatrix}$$

and, using the inequality $d_1(\mathcal{A}) \geq k$, pick up an arbitrary $i \in [m]$ (depending in general on $\vec{a}_1, \ldots, \vec{a}_k$) such that $a_{i1} = a_{i2} = \cdots = a_{ik} = 1$. We want to infer from $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ all clauses in the CNF expansion of

$$[\text{Column}_1 \neq \vec{a}_1] \vee \cdots \vee [\text{Column}_k \neq \vec{a}_k] \vee [\text{Control}_i \neq c] \tag{6}$$

for all $c \in [r]$. It is fairly obvious how to do this efficiently in general resolution. Namely, let V be the set of all nodes of the circuit C that are evaluated to 1 by the assignment $(1^k, 0^{n-k})$. Then we may proceed by induction on the construction of C and subsequently infer

$$([\text{Column}_1 = \vec{a}_1] \wedge \cdots \wedge [\text{Column}_k = \vec{a}_k] \wedge [\text{Control}_i = c]) \supset z^c_{iv}$$

for all $v \in V$ until we reach v_{fin}.

In order to get a *tree-like* proof, however, we should employ a dual (top-down) strategy. Namely, enumerate the set V in some order which is consistent with the topology of C: $V = \langle v_1 = p_1, v_2 = p_2, \ldots, v_k = p_k, v_{k+1}, v_{k+2}, \ldots, v_t = v_{\text{fin}} \rangle$; all wires between vertices in V go from left to right. Then, by a reverse induction on $\mu = t, t-1, \ldots, k$ we infer (all clauses in the CNF expansion of) $[\text{Control}_i = c] \supset (\bar{z}^c_{i,v_1} \vee \cdots \vee \bar{z}^c_{i,v_\mu})$. For $\mu = t$ this is (a weakening of) (5), and for the inductive step we resolve with the appropriate axiom in (4). When we descend to $[\text{Control}_i = c] \supset \left(\bar{z}^c_{i,p_1} \vee \cdots \vee \bar{z}^c_{i,p_k}\right)$, we consecutively resolve with the corresponding axioms (3) to get rid of $\bar{z}^c_{i,p_j}$ and arrive at (6). Clearly, this resolution inference of every individual clause in (6) is tree-like and has size $O(|C|)$.

Finally, for every $i \in [m]$, every clause in the variables $\{\, y^\nu_i \mid 1 \leq \nu \leq s \,\}$ appears in one of the CNFs resulting from the predicate $\{\, [\text{Control}_i \neq c] \mid c \in [r] \,\}$ or the predicates in (2), and every clause in the variables $\left\{\, x^\nu_j \mid 1 \leq \nu \leq s \,\right\}$ appears in one of $[\text{Column}_j \neq \vec{a}_j]$. This gives us an obvious tree-like refutation of the set of clauses (2), (6) that has size $O(2^{s(k+1)})$. Combining this refutation with previously constructed inferences of (6) from $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$, we get the desired upper bound.

Part (b). We follow the general strategy proposed in [10]. Note that every one of the axioms (2)–(5) "belongs" to a uniquely defined row; let Row_i be the set of axioms in $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ that correspond to the row i. For a clause D, let $\mu(D)$ be the smallest cardinality of $I \subseteq [m]$ such that $\cup \{\, \text{Row}_i \mid i \in I \,\}$ (semantically) implies D. $\mu(D)$ is subadditive, that is, $\mu(D) \leq \mu(D_1) + \mu(D_2)$ whenever D is obtained from D_1, D_2 via a single application of the resolution rule. It is also obvious that $\mu(A) = 1$ for any axiom $A \in \tau(C, \mathcal{A}, \vec{F}, \vec{f})$.

We claim that $\mu(\emptyset) > d_0(\mathcal{A})$. Indeed, fix any $I \subseteq [m]$ with $|I| \leq d_0(\mathcal{A})$. We need to construct an assignment that satisfies all axioms in $\cup \{\, \text{Row}_i \mid i \in I \,\}$. Pick $\vec{a}$ accordingly to Definition 3.6 in such a way that for all $i \in I (a_i = 0)$. Assign every x^ν_j to α^ν_j, where $\alpha^1_j, \ldots, \alpha^s_j$ is an arbitrary vector such that $F_j(\alpha^1_j, \ldots, \alpha^s_j) = \vec{a}$; assign y^ν_i in an arbitrary way with the only requirement that they satisfy (2), and assign all z-variables to 0. This assignment will satisfy all axioms in $\cup \{\, \text{Row}_i \mid i \in I \,\}$, which proves $\mu(\emptyset) > d_0(\mathcal{A})$.

Thus, any resolution refutation of $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ must contain a clause D with $\frac{1}{2}d_0(\mathcal{A}) \le \mu(D) \le d_0(\mathcal{A})$, and we need only show that this implies $\widetilde{w}(D) \ge \frac{r}{2} \cdot \min\{k(C), d_0(\mathcal{A})\}$. Fix $I \subseteq [m]$ such that $\cup\{\mathrm{Row}_i \mid i \in I\}$ semantically implies D and $|I|$ is minimal with this property; $\frac{1}{2}d_0(\mathcal{A}) \le |I| \le d_0(\mathcal{A})$.

If for every $i \in I$ at least one of the following two events is true, then we are done:

1. the clause D contains at least r variables among $\{y_i^\nu \mid \nu \in [s]\}$;
2. for every control $c \in [r]$ the clause D contains at least one variable among $\{z_{iv}^c \mid v \text{ is a node}\}$.

Indeed, if h is the number of indices i for which 1 is true, then $w_y(D) \ge rh$. For those $i \in I$ for which 1 does not hold, we apply 2 to conclude $\min_{c\in[r]} w_c(D) \ge (|I| - h)$, and altogether we have $\widetilde{w}(D) \ge w_y(D) + r \cdot \min_{c\in[r]} w_c(D) \ge r \cdot |I| \ge \frac{r}{2}d_0(\mathcal{A})$.

Thus, suppose that for some $i_0 \in I$ neither of these two is true. In particular, there exists a control $c_0 \in [r]$ such that no variable of the form $z_{i_0,v}^{c_0}$ appears in D. Fix an arbitrary assignment α that satisfies all axioms in $\{\mathrm{Row}_i \mid i \in I \setminus \{i_0\}\}$ and falsifies D (such an assignment exists due to the minimality of $|I|$).

Let J_0 consist of those $j \in [n]$ for which the clause D contains at least r variables from $\{x_j^\nu \mid \nu \in [s]\}$. If $|J_0| \ge k(C)$, we are also done. If this is not the case, we will show how to alter the assignment α so that it will satisfy all axioms in $\cup\{\mathrm{Row}_i \mid i \in I\}$ (including Row_{i_0}) but still will falsify D, and this will give us the contradiction.

According to Definition 3.6, there exists $\vec{a} \in \mathcal{A}$ such that $a_i = 0$ for all $i \in I$. We alter α as follows.

Step 1. Using that F_j is r-surjective, we change for every $j \notin J_0$ the values of the variables $\{x_j^\nu \mid \nu \in [s]\}$ not appearing in D in such a way that $F_j(x_j^1, \dots, x_j^s) = \vec{a}$.

Step 2. Using the fact that f_{i_0} is r-surjective, we change the values of variables $\{y_{i_0}^\nu \mid \nu \in [s]\}$ not appearing in D in such a way that $f_{i_0}(y_{i_0}^1, \dots, y_{i_0}^s) = c_0$. Finally, we reassign every $z_{i_0,v}^{c_0}$ to the value computed by the node v on the characteristic vector of the set J_0. Note that $z_{i_0,p_j}^{c_0}$ is set to 1 for $j \in J_0$, whereas $z_{i_0,v_{\mathrm{fin}}}^{c_0}$ is set to 0 since $|J_0| < k(C)$.

We claim that this altered assignment α' satisfies all axioms in $\cup\{\mathrm{Row}_i \mid i \in I\}$. Indeed, we made sure in our construction that it satisfies all axioms in Row_{i_0} of types (2), (4), (5), and for $i \in I \setminus \{i_0\}$ axioms of these types are satisfied since we have not touched any variable appearing in them. Thus, we have only to check the axiom (3). If $j \in J_0$ and $i \ne i_0$, this axiom has not been touched, and if $j \notin J_0$, it becomes satisfied because of the first step in our construction of α', and due to the condition $a_i = 0$ ($i \in I$). Finally, if $j \in J_0$ and $i = i_0$, the axiom (3) gets satisfied during the second step (in which we set $z_{i_0,p_j}^{c_0}$ to 1).

But α' also falsifies D since we have not touched variables appearing in it. This contradiction with the fact that $\{\mathrm{Row}_i \mid i \in I\}$ implies that D completes the proof of part (b).

Part (c). We apply the standard argument of width-reducing restrictions (cf. [8]). For doing this we observe that the CNFs of the form $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ behave well with respect to certain restrictions. Namely, let $d \le r$ and $R \subseteq [r]$ be an arbitrary set of controls. Denote by $\mathcal{R}_{d,R}$ the set of all restrictions that arbitrarily assign to a Boolean value d variables in every one of the groups $\{x_j^\nu \mid \nu \in [s]\}, \{y_i^\nu \mid \nu \in [s]\}$ with $j \in [n], i \in [m]$ as well as all the variables z_{iv}^c with $c \notin R$. Then it is easy to see that for $\rho \in \mathcal{R}_{d,R}$, every nontrivial clause in $\tau(C, \mathcal{A}, \vec{F}, \vec{f})|_\rho$, after a suitable re-enumeration of variables and controls, contains a subclause from $\tau(C, \mathcal{A}, \vec{F}|_\rho, (\vec{f}|_\rho)|_R)$ (the *partial* function $(f_i|_\rho)|_R$ is obtained from $f_i|_\rho$ by restricting its domain to $\{y_i \mid f_i|_\rho(y_i) \in R\}$ and range to R).

Pick now $\boldsymbol{\rho}$ uniformly at random from $\mathcal{R}_{r/2,\boldsymbol{R}}$, where $\boldsymbol{R}$ is picked at random from $[r]^{r/2}$. Then $F_j|_{\boldsymbol{\rho}}, (f_i|_{\boldsymbol{\rho}})|_{\boldsymbol{R}}$ will be $(r/2)$-surjective. Therefore, by the already proven part (b), for every refutation P of $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$, $\boldsymbol{\rho}(P)$ will contain a clause of controlled width

$$\Omega\left(r \cdot \min\{k(C), d_0(\mathcal{A})\}\right) \tag{7}$$

with probability 1.

It is easy to see, however, that every clause D whose controlled width $\widetilde{w}(D)$ is that large is killed (that is, set to 1 and hence removed from the proof) by $\boldsymbol{\rho}$ with probability

$$1 - \exp\left(-\Omega\left(\frac{r^2}{s} \cdot \min\{k(C), d_0(\mathcal{A})\}\right)\right).$$

Indeed, according to Definition 3.7, either $w_x(D)$ or $w_y(D)$ or $r \cdot \min_{c\in[r]} w_c(D)$ is bounded from below by a quantity of the form (7). Let $w_{x,j}(D)$ be the number of variables in the corresponding group that appear in D so that $w_x(D) = \sum_{j\in[n]} w_{x,j}(D)$. Then the probability that D is not killed by variables in the jth group is $\exp\big(-\Omega\big(\frac{r\cdot w_{x,j}(D)}{s}\big)\big)$, and these events are independent so the probabilities of survival multiply to $\exp\big(-\Omega\big(\frac{r\cdot w_{x,j}(D)}{s}\big)\big)$. The case when $w_y(D)$ is large is treated in exactly the same way, and the case when $r \cdot \min_{c\in[r]} w_c(D)$ is large is even simpler since for *every* choice of $\boldsymbol{R}$, the number of assigned z-variables is at least $\frac{r}{2} \cdot \min_{c\in[r]} w_c(D)$ (and $s \geq r$).

Therefore, the size of P must be at least $\exp\big(\Omega\big(\frac{r^2}{s} \cdot \min\{k(C), d_0(\mathcal{A})\}\big)\big)$ since otherwise a random restriction $\boldsymbol{\rho}$ would have killed all such clauses with nonzero probability, which is impossible.

Lemma 3.8 is completely proved. □

Proof of Lemma 3.1. Our construction of $\tau(C, m)$ proceeds in polynomial time as follows.

1. Let p be the smallest prime greater than or equal to m. Since $m \leq p \leq 2m$, both bounds in Lemma 3.1 remain unchanged if we replace m by p or vice versa. Therefore, w.l.o.g. we may assume from the beginning that m itself is a prime. Let P_m be the $(m \times m)$ 0-1 *Paley matrix* given by $a_{ij} = 1$ iff $j \neq i$ and $(j-i)$ is a quadratic residue mod m. Let $\mathcal{A} \subseteq \{0,1\}^m$ consist of all columns of P_m. Then $|\mathcal{A}| = m$ and $d_0(\mathcal{A}), d_1(\mathcal{A}) \geq \frac{1}{4}\log m$ (see, e.g., [5]).

2. Fix any $\mathbb{F}_2$-linear code $L \subseteq \{0,1\}^{h\lceil\log m\rceil}$ of dimension $\lceil\log m\rceil$ that is computable (as a language) in time $m^{O(1)}$ and has minimal distance $\geq \lceil\log m\rceil$ ($h > 0$ is an absolute constant). Consider the linear mapping $G : \{0,1\}^{h\lceil\log m\rceil} \longrightarrow \{0,1\}^{\lceil\log m\rceil}$ dual to the inclusion $L \to \{0,1\}^{h\lceil\log m\rceil}$ (that is, we fix in L an arbitrary basis $x_1, \ldots, x_{\lceil\log m\rceil}$ and let $G(y) \stackrel{\text{def}}{=} (\langle x_1, y\rangle, \ldots, \langle x_{\lceil\log m\rceil}, y\rangle)$). By linear duality, the fact that L has minimal distance $\geq \lceil\log m\rceil$ is equivalent to $\lceil\log m\rceil$-surjectivity of G. Set $r \stackrel{\text{def}}{=} \lceil\log m\rceil$ and $s \stackrel{\text{def}}{=} h\lceil\log m\rceil$. Consider arbitrary (polynomial-time computable) surjective mappings $\Pi : \{0,1\}^{\lceil\log m\rceil} \longrightarrow \mathcal{A}$, $\pi : \{0,1\}^{\lceil\log m\rceil} \longrightarrow [r]$ and let $F_j \stackrel{\text{def}}{=} \Pi \cdot G, f_i \stackrel{\text{def}}{=} \pi \cdot G$ for all i, j.

3. Construct $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$. Note that the size of this CNF is polynomial in $|C|, m, 2^s$, which is polynomial in $|C|, m$ due to our choice of parameters.

At this point, Lemma 3.8(c) already implies (1) for $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$. The only remaining problem is that a priori we do not have the condition $k(C) \leq d_1(\mathcal{A})$ needed for part (a) of Lemma 3.8. We circumvent this by a trick similar to one used in [3].

Namely, let τ_m be a fixed unsatisfiable (polynomial-time constructible) CNF with $S(\tau_m), S_T(\tau_m) = m^{\theta(\log m)}$ (for example, one can take a Tseitin tautology with $\theta((\log m)^2)$ variables) and such that its set of variables is disjoint from the set of variables of $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$. We finally set $\tau(C, m) \stackrel{\text{def}}{=} \tau(C, \mathcal{A}, \vec{F}, \vec{f}) \wedge \tau_m$.

Since both $\tau(C, \mathcal{A}, \vec{F}, \vec{f})$ and τ_m satisfy the lower bound (1), the weak feasible disjunction property (see, e.g., [21] and the literature cited therein) for resolution implies that $\tau(C, m)$ satisfies this bound, too. If $k(C) \le \frac{1}{4}\log m$ then, since $d_1(\mathcal{A}) \ge \frac{1}{4}\log m$, we can apply Lemma 3.8(a) to get the required upper bound $S_T(\tau(C, m)) \le |C| \cdot m^{O(k(C))}$. If, on the other hand, $k(C) \ge \frac{1}{4}\log m$, the required upper bound $S_T(\tau(C, m)) \le m^{O(\log m)}$ simply follows from the upper bound for τ_m. This completes the proof of Lemma 3.1. ∎

4. Self-improvement. In this section we complete the proof of Theorems 2.5 and 2.7 by combining Lemma 3.1 with a (nontrivial) self-improvement technique. First, we need to get rid of the dummy parameter m in the statement of Lemma 3.1.

LEMMA 4.1. *If either resolution or tree-like resolution is automatizable, then there exists an absolute constant $h > 1$ and an algorithm Φ working on pairs $\langle C, k\rangle$, where C is a monotone circuit and k is an integer such that*

1. *the running time of $\Phi(\langle C, k\rangle)$ is at most $\exp(O(k^2)) \cdot |C|^{O(1)}$;*
2. *if $k(C) \le k$, then $\Phi(\langle C, k\rangle) = 1$;*
3. *if $k(C) \ge hk$, then $\Phi(\langle C, k\rangle) = 0$.*

Proof. Combining the reduction in Lemma 3.1 with an automatizing algorithm for either resolution or tree-like resolution, we get an integer-valued function $S(C, m)$ computable in time $\left(|C| \cdot m^{\min\{k(C),\ \log m\}}\right)^{h_0}$ and such that

$$m^{\epsilon \cdot \min\{k(C),\ \log m\}} \le S(C, m) \le \left(|C| \cdot m^{\min\{k(C),\ \log m\}}\right)^{h_1}$$

for some absolute constants $\epsilon, h_0, h_1 > 0$. Set the constant h in the statement in such a way that

$$h^2 > \frac{h_1}{\epsilon}(h + 1). \tag{8}$$

Our algorithm Φ works as follows. We set

$$m \stackrel{\text{def}}{=} 2^{h \cdot \max\{k,\ \log |C|/k\}}.$$

Φ simulates $\left(|C| \cdot m^k\right)^{h_0}$ steps in the computation of $S(C, m)$, outputs 1 if the computation halts within this time, and its result $S(C, m)$ satisfies the inequality $S(C, m) \le \left(|C| \cdot m^k\right)^{h_1}$ and outputs 0 in all other cases.

Our choice of m ensures that $m^k \le \exp(O(k^2)) \cdot |C|^{O(1)}$, which implies property 1.

Since $\log m \ge hk \ge k$, under the assumption $k(C) \le k$ the limitations we have imposed on the running time and the output value of the algorithm Φ are less stringent than the bounds known of the underlying algorithm computing $S(C, m)$. This observation implies property 2.

Finally, using again the inequality $\log m \ge hk$, $k(C) \ge hk$ implies that $S(C, m) \ge m^{\epsilon hk}$, and elementary calculations show that, along with (8), this gives us $S(C, m) > \left(|C| \cdot m^k\right)^{h_1}$. Thus, if $k(C) \ge hk$, the algorithm Φ outputs the value 0.

Lemma 4.1 is proved. ∎

Proof of Theorem 2.5. First, we extract from Lemma 4.1 an algorithm which meets the bound on the running time and achieves the ratio of approximation h. For that we consecutively run the algorithm Φ from that lemma on the inputs $\langle C, 1\rangle, \ldots, \langle C, k\rangle, \ldots$ and output the first value k for which we get the answer 0.

Combining this algorithm with the self-improving reduction from Proposition 2.4 (for $d = \lceil \frac{1}{\epsilon} \ln h \rceil$), we get an approximating algorithm with the required properties. □

In the established terminology, what we have seen so far under the assumption of automatizability of (tree-like) resolution is a polynomial-time approximation scheme (PTAS) for MMCSA in the context of parameterized complexity (the latter referring to the term $\exp\left(k(C)^{O(1)}\right)$ in the bound on the running time). Unfortunately, our PTAS is not efficient (see the discussion in section 2.2), as the reduction from Proposition 2.4 blows up the size of the circuit. The task of converting an arbitrary PTAS into an EPTAS seems to be hopeless in general even in the context of parameterized complexity (where it appears to be easier). We nonetheless can perform it (in the latter context) for the specific problem MMCSA using a much trickier self-improvement construction. This construction (that completes the proof of our main theorem, Theorem 2.7) might be of independent interest, and its idea is roughly as follows.

We need to improve the approximation ratio of the algorithm Φ in Theorem 2.5 from (say) 2 to (say) $(1 + \frac{1}{\sqrt{k}})$, and the straightforward way of doing this is by iteratively applying Proposition 2.4 (say) $d = \sqrt{k}$ times. The corresponding reduction will map any circuit $C(x_1, \ldots, x_n)$ into an n-ary tree of C-gates, and of depth d, and the resulting increase in size is too costly to us. What we basically show is that we can circumvent this by replacing the tree with a *random* directed acyclic graph (DAG) of the same depth d and of width polynomial in n.

Proof of Theorem 2.7. Let C be a monotone circuit in n variables and k be an integer such that

$$10 \le k \le \epsilon(\log n / \log\log n)^2 \tag{9}$$

for a sufficiently small constant $\epsilon > 0$ (we will remark later how to get rid of this condition). Our goal is to construct in polynomial time a randomized monotone circuit $\boldsymbol{\pi}(C, k)$ and an integer $\alpha(k)$ (deterministically depending only on k) such that α is recursive and the following conditions hold:

$$k(C) \le k \implies \mathbf{P}[k(\boldsymbol{\pi}(C,k)) \le \alpha(k)] = 1; \tag{10}$$

$$k(C) \ge k+1 \implies \mathbf{P}[k(\boldsymbol{\pi}(C,k)) \ge 2\alpha(k)] \ge 1/2. \tag{11}$$

First, we apply to C the reduction from Proposition 2.4 with $d = 2$ that maps the range $[k, k+1]$ to $[k^2, k^2 + 2k + 1]$. Redenoting k^2 back to k, we may assume w.l.o.g. that in (11) we have the stronger premise

$$k(C) \ge k + 2\sqrt{k} \implies \mathbf{P}[k(\boldsymbol{\pi}(C,k)) \ge 2\alpha(k)] \ge 1/2. \tag{12}$$

Now comes our main reduction. Let N, d be two parameters (to be specified later). The randomized circuit $\boldsymbol{\pi}(C, N, d)$ in (nN) variables consists of d layers. Each layer consists of N independent copies of the circuit C (see Figure 1); thus, it has (nN) inputs and N outputs. We connect input nodes at the $(i+1)$st level to output nodes at the ith level at random. Finally, we pick up an arbitrary output node at the last dth level and declare it to be the output of the whole circuit $\boldsymbol{\pi}(C, N, d)$.

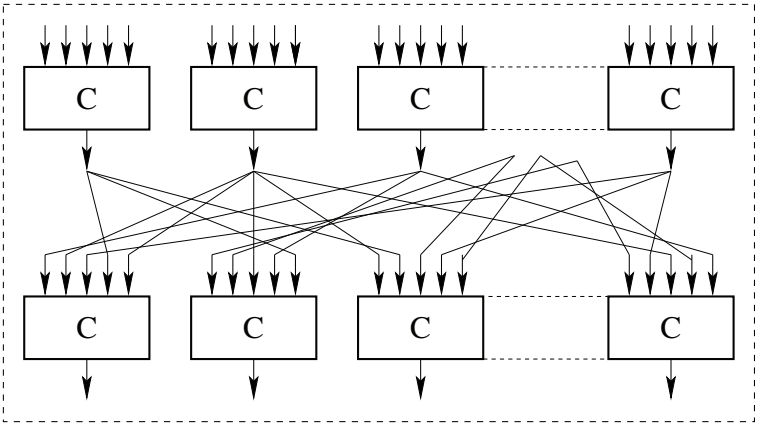

FIG. 1. *One layer of* $\boldsymbol{\pi}(C, N, d)$.

Clearly, this construction is polynomial in $|C|, N, d$. Also, an obvious induction on d shows that

$$k(\boldsymbol{\pi}(C, N, d)) \le k(C)^d \tag{13}$$

with probability 1. In order to get a lower bound on $k(\boldsymbol{\pi}(C, N, d))$, we need the following easy lemma. It is of course yet another version of the well-known fact that a random (bipartite) graph makes an extremely good expander.

LEMMA 4.2. *Let* $\boldsymbol{\chi} : [N] \times [n] \longrightarrow [N]$ *be a randomly chosen function, and let* k, a *be any parameters. Then* $\mathbf{P}\big[\exists V \in [N]^k (|\boldsymbol{\chi}(V \times [n])| \le kn - a)\big] \le N^k \cdot (\frac{4k^2n^2}{N})^a$.

Proof of Lemma 4.2. This event takes place iff there exist $V \in [N]^k$ and disjoint $D_1, \ldots, D_r \subseteq V \times [n]$ such that $|D_1|, \ldots, |D_r| \ge 2$, $\sum_{i=1}^r (|D_i| - 1) = a$, and $\boldsymbol{\chi}|_{D_i} =$ const for all $i \in [r]$. Since the two first properties imply $\sum_{i=1}^r |D_i| \le 2a$, the overall number of all choices of $\langle V, D_1, \ldots, D_r \rangle$ does not exceed $N^k \cdot (2kn)^{2a}$. On the other hand, for every fixed choice, we have

$$\mathbf{P}[\boldsymbol{\chi}|_{D_1} = \text{const}, \ldots, \boldsymbol{\chi}|_{D_r} = \text{const}] = \frac{N^r}{N^{(\sum_{i=1}^r |D_i|)}} = N^{-a}.$$

Lemma 4.2 follows. □

Now we can complete the description of our reduction. Namely, we set $N \stackrel{\text{def}}{=} n^3$, $d \stackrel{\text{def}}{=} \sqrt{k}$ and let $\boldsymbol{\pi}(C, k) = \boldsymbol{\pi}(C, n^3, \sqrt{k})$, $\alpha(k) \stackrel{\text{def}}{=} k^{\sqrt{k}}$.

Equation (10) follows from (13).

In order to check (12), denote by $\boldsymbol{\chi_i} : [N] \times [n] \longrightarrow [N]$ the function used for connecting input nodes at the $(i+1)$st level of $\boldsymbol{\pi}(C, n^3, \sqrt{k})$ to the output nodes at the ith level. Let $k_i \stackrel{\text{def}}{=} (k + \sqrt{k})^{d-i}$. Let us call $\boldsymbol{\pi}(C, N, d)$ *bad* if for at least one of these functions $\boldsymbol{\chi_i}$ there exists a set V of circuits at the $(i+1)$st level such that $|V| = k_{i+1}$ and $|\boldsymbol{\chi_i}(V \times [n])| \le k_{i+1}(n - \sqrt{k})$. Using Lemma 4.2 and (9), we get the

bound

$$\begin{aligned}\mathbf{P}[\boldsymbol{\pi}(C,N,d) \text{ is bad}] &\le \sum_{i=1}^{d-1} N^{k_{i+1}} \cdot \left(\frac{4k_{i+1}^2 n^2}{N}\right)^{\sqrt{k}\cdot k_{i+1}} \\ &= \sum_{i=1}^{d-1} \left(\frac{4k_{i+1}^2}{n^{1-3/\sqrt{k}}}\right)^{\sqrt{k}\cdot k_{i+1}} \\ &\le \sum_{i=1}^{d-1} \left(\frac{1}{3}\right)^{\sqrt{k}\cdot k_{i+1}} \le \frac{1}{2}.\end{aligned}$$

On the other hand, it is easy to see by induction on $i = d, \dots, 1$ that if $k(C) \ge k + 2\sqrt{k}$ and $\boldsymbol{\pi}(C,N,d)$ is good, then every satisfying assignment a should satisfy at least k_i output nodes at the ith level. Indeed, the base $i = d$ is obvious ($k_d = 1$). For the inductive step, assume that a satisfies the output nodes of a set V of circuits at the $(i+1)$st level, $|V| = k_{i+1}$. Then at least $(k + 2\sqrt{k}) \cdot k_{i+1}$ *input* nodes to these circuits should be satisfied. Since $\boldsymbol{\chi_i}$ is good, there are at most $\sqrt{k} \cdot k_{i+1}$ collisions between the $(k + 2\sqrt{k}) \cdot k_{i+1}$ wires leading to these input nodes from the ith level. Therefore, at least $(k + 2\sqrt{k}) \cdot k_{i+1} - \sqrt{k} \cdot k_{i+1} = k_i$ output nodes at the ith level should be satisfied.

In particular, at the first level we will have $\ge (k + \sqrt{k})^{d-1}$ satisfied circuits and $\ge (k + 2\sqrt{k}) \cdot (k + \sqrt{k})^{\sqrt{k}-1} > 2\alpha(k)$ satisfied input nodes. This completes the proof that our probabilistic reduction $\boldsymbol{\pi}(C,k)$ has the properties (10), (12) (and, as we already remarked, improving (12) to (11) takes one more easy step).

Now we finish the proof of Theorem 2.7. Suppose that either resolution or tree-like resolution is automatizable. Since WEIGHTED MONOTONE CIRCUIT SATISFIABILITY is **W[P]**-complete (see [17, Chapter 13]), we have only to show that the language $\{\, \langle C,k\rangle \mid k(C) \le k \,\}$ is in co-**FPR**. Given an input $\langle C,k\rangle$ we check condition (9). If it is violated, we apply a straightforward brute-force algorithm with running time $O(|C|\cdot n^k) \le |C|\cdot f(k)\cdot n^9$ for some recursive f. Otherwise we simply combine our probabilistic reduction $\langle \boldsymbol{\pi}, \alpha\rangle$ with the deterministic algorithm for deciding whether $k(\boldsymbol{\pi}(C,k)) \le \alpha(k)$ or $k(\boldsymbol{\pi}(C,k)) \ge 2\alpha(k)$ provided by Theorem 2.5. Theorem 2.7 is completely proved. □

5. Open problems. The main problem left open by this paper is whether general resolution is quasi-automatizable. Since the width algorithm by Ben-Sasson and Wigderson [10] finds a resolution refutation of any unsatisfiable CNF τ in time $n^{O(w(\tau\vdash\emptyset))}$, a negative solution to this problem must involve a construction of a broad and "tractable" family of CNF τ for which $S(\tau)$ is much smaller than $2^{w(\tau\vdash\emptyset)}$. Such families are not so easy to come by (e.g., our techniques involve showing the *opposite* in the proof of Lemma 3.8(c)), although some progress toward this goal was reported in [6].

As we already mentioned in section 1.1, the same paper [6] also proposed an interesting notion of weak automatizability. Namely, a proof system P is *weakly automatizable* if there exists any automatizable proof system that polynomially simulates P. Is resolution *weakly* automatizable (under any reasonable complexity assumptions in the case of a negative answer)? Paper [6] showed that this is equivalent to another important open question in proof complexity, namely, if the system $Res(2)$ has the feasible interpolation property (for definitions, see, e.g., [22]).

We were not able to derandomize the proof of Lemma 4.2. In the terminology of [1], we need explicit constructions of $(N \times N)$ 0-1 matrices that would be $(k, n, n - O(1))$-expanders for $n \geq N^{\Omega(1)}$ and an arbitrary function $k = k(N)$ tending to infinity. Explicit constructions based on Ramanujan graphs seem to give only $(k, n, n - k^{\epsilon})$-expanders for any *fixed* ϵ which is not sufficient for our purposes. Can we weaken the hardness assumption in Theorem 2.7 to $\mathbf{W[P]} \neq \mathbf{FPT}$ by an explicit construction of better expanders (or by using any other means)?

Acknowledgment. The second author is greatly indebted to all three anonymous referees of the journal version of this paper for their constructive criticism and many useful remarks and suggestions.

REFERENCES

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson, *Pseudorandom generators in propositional proof complexity*, SIAM J. Comput., 34 (2004), pp. 67–88.

[2] M. Alekhnovich, M. Braverman, V. Feldman, A. R. Klivans, and T. Pitassi, *Learnability and automatizability*, in Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (Rome, Italy), 2004, pp. 621–630.

[3] M. Alekhnovich, S. Buss, S. Moran, and T. Pitassi, *Minimum propositional proof length is NP-hard to linearly approximate*, J. Symbolic Logic, 66 (2001), pp. 171–191.

[4] M. Alekhnovich and A. Razborov, *Satisfiability, branch-width and Tseitin tautologies*, in Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science (Vancouver, Canada), 2002, pp. 593–603.

[5] N. Alon, *Tools from higher algebra*, in Handbook of Combinatorics, Vol. II, Elsevier, Amsterdam, 1995, pp. 1749–1783.

[6] A. Atserias and M. Bonet, *On the automatizability of resolution and related propositional proof systems*, Inform. and Comput., 189 (2004), pp. 182–201.

[7] C. Bazgan, *Schémas d'approximation et complexité paramétrée*, Rapport de stage de DEA d'Informatique à Orsay, 1995.

[8] P. Beame and T. Pitassi, *Simplified and improved resolution lower bounds*, in Proceedings of the 37th IEEE Symposium on Foundations of Computer Science (Burlington, VT), 1996, pp. 274–282.

[9] P. Beame and T. Pitassi, *Propositional proof complexity: Past, present and future*, in Current Trends in Theoretical Computer Science: Entering the 21st Century, G. Paun, G. Rozenberg, and A. Salomaa, eds., World Scientific, River Edge, NJ, 2001, pp. 42–70.

[10] E. Ben-Sasson and A. Wigderson, *Short proofs are narrow-resolution made simple*, J. ACM, 48 (2001), pp. 149–169.

[11] M. Bonet, C. Domingo, R. Gavaldá, A. Maciel, and T. Pitassi, *Non-automatizability of bounded-depth Frege proofs*, Comput. Complexity, 13 (2004), pp. 47–68.

[12] M. Bonet and N. Galesi, *A study of proof search algorithms for resolution and polynomial calculus*, in Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (New York, NY), 1999, pp. 422–431.

[13] M. L. Bonet, T. Pitassi, and R. Raz, *On interpolation and automatization for Frege systems*, SIAM J. Comput., 29 (2000), pp. 1939–1967.

[14] M. Cesati and L. Trevisan, *On the efficiency of polynomial time approximation schemes*, Inform. Process. Lett., 64 (1997), pp. 165–171.

[15] S. A. Cook and A. R. Reckhow, *The relative efficiency of propositional proof systems*, J. Symbolic Logic, 44 (1979), pp. 36–50.

[16] I. Dinur and M. Safra, *On the hardness of approximating label-cover*, Inform. Process. Lett., 89 (2004), pp. 247–254.

[17] R. Downey and M. Fellows, *Parameterized Complexity*, Springer-Verlag, New York, 1999.

[18] K. Iwama, *Complexity of finding short resolution proofs*, in Proceedings of the 22nd International Symposium on the Mathematical Foundations of Computer Science (Bratislava, 1997), Lecture Notes in Comput. Sci. 1295, P. Ružička and I. Prívara, eds., Springer-Verlag, New York, 1997, pp. 309–318.

[19] J. Krajíček, *Proof complexity*, in Proceedings of the European Congress of Mathematics (ECM), Stockholm, Sweden, June 27–July 2, 2004, A. Laptev, ed., European Mathematical Society, Zürich, 2005, pp. 221–231.

[20] J. Krajíček and P. Pudlák, *Some consequences of cryptographical conjectures for S_2^1 and EF*, Inform. and Comput., 142 (1998), pp. 82–94.
[21] P. Pudlák, *On reducibility and symmetry of disjoint NP-pairs*, Theoret. Comput. Sci., 295 (2003), pp. 323–339.
[22] N. Segerlind, *The complexity of propositional proofs*, Bull. Symbolic Logic, 13 (2007), pp. 417–481.