Electronic Colloquium on Computational Complexity, Report No. 188 (2016)

Strongly Exponential Lower Bounds for Monotone Computation

Toniann Pitassi[*]
University of Toronto
toni@cs.toronto.edu

Robert Robere[*]
University of Toronto
robere@cs.toronto.edu

November 21, 2016

Abstract

For a universal constant $\alpha > 0$, we prove size lower bounds of $2^{\alpha N}$ for computing an explicit monotone function in NP in the following models of computation: monotone formulas, monotone switching networks, monotone span programs, and monotone comparator circuits, where N is the number of variables of the underlying function. Our lower bounds improve on the best previous bounds in each of these models, and are the best possible for any function up to the constant factor α. Moreover, we give one unified proof that is short and fairly elementary.

1 Introduction

Circuit complexity, one of the central areas of study in modern complexity theory, seeks to prove unconditional bounds on the complexity of boolean functions in various models of computational circuits. Despite decades of exciting progress, most of the major problems in circuit complexity remain wide open: classic counting arguments by Shannon [31] show that all but a negligible fraction of boolean functions on n variables require circuits of size $2^n/n$ in all reasonable models, and yet we are not able to exhibit any explicit function that requires boolean circuits of size $\omega(n)$. The situation is only slightly better for restricted models — for instance, we are unable to prove $\omega(n^3)$ lower bounds on the size of boolean formulas, and similarly cannot prove $\omega(n^2)$ lower bounds on the size of branching programs.

Meanwhile, theorists have had *outstanding* success in *monotone* circuit complexity. (Recall that a circuit model is *monotone* if it is not allowed to use negations.) Classic results in this area include Razborov's $2^{\Omega(n^\varepsilon)}$ lower bound for monotone circuits computing the clique function [28], Karchmer and Wigderson's $n^{\Omega(\log n)}$ lower bounds for monotone formulas computing s-t connectivity [18], and Raz and McKenzie's separation of depth $O(\log^i n)$ monotone circuits from depth $O(\log^{i+1} n)$ monotone circuits for each i [26]. More recently, some exciting progress was made by Rossman [30] — he gives the first superpolynomial lower bounds on the monotone formula size for well-approximating a certain monotone boolean function on the *uniform distribution*. This can be viewed as a "distributional" version of *slice function* lower bounds, and it has long been known that strong enough lower bounds on the monotone circuit complexity of the slice functions actually yield lower bounds for non-monotone circuits.

While primarily considered a test-bed for circuit lower bound techniques, monotone circuit complexity also has connections with other areas of theoretical computer science. For example, in proof

[*]Research supported by NSERC.

ISSN 1433-8092

complexity, monotone circuit and monotone formula lower bounds are used to obtain lower bounds on the size of refutations in the *cutting planes* proof system [3, 9, 25]. And in cryptography, monotone span program lower bounds are known to yield lower bounds on the size of *linear secret sharing schemes* [19].

Given the clear applicability of these results, it is natural to wonder what are the *strongest* lower bounds that we can prove for a monotone circuit model computing an explicit monotone boolean function. A simple modification of Shannon's counting argument shows that almost all monotone boolean functions require monotone boolean circuits of size at least $\tilde{\Omega}(2^n/n)$, but the best known lower bound on an *explicit* function — due to Harnik and Raz [15] — is $2^{\Omega(n^{1/3})}$ for a function computable in NP. Even if we restrict ourselves to monotone formulas, the best known lower bounds are of the form $2^{\Omega(n/\log n)}$ for a function computable in NP [13], which is still not asymptotically optimal. Thus, even though all but a negligible fraction of monotone boolean functions have monotone circuit complexity $2^{\Omega(n)}$, we are still unable to find a single explicit example.

The main result in this work resolves this problem for a large number of monotone circuit models.

Theorem 1.1. *There is a universal constant $\alpha > 0$ and an explicit monotone function computable in NP on N variables that requires size $2^{\alpha N}/11$ in the following models of computation: monotone formulas, monotone switching networks, real monotone span programs, and monotone comparator circuits.*

This is the first example of a *strongly exponential* lower bound for an explicit monotone function in any monotone circuit model. Furthermore, since any monotone boolean function on N inputs can be computed by a monotone DNF of size $N2^N$, it follows that our results are asymptotically the best possible for *any* monotone function up to constants in the exponent.

Lower Bound Techniques for Monotone Circuit Models. Historically, the main techniques used for proving lower bounds against the models in Theorem 1.1 have used ideas from *communication complexity*. We begin by discussing monotone formulas. The first superpolynomial lower bounds for monotone formulas are found in the celebrated work of Karchmer and Wigderson [18] showing that the circuit depth (and thus formula size) required for computing any boolean function is exactly captured by the communication complexity of a certain search problem (now called the *Karchmer-Wigderson game*) related to the function. Using this connection, Karchmer and Wigderson proved $n^{\Omega(\log n)}$ lower bounds on the monotone formula size (equivalently, $\Omega(\log^2 n)$ lower bounds on the monotone circuit depth) of the s-t connectivity function. Later, Raz and Wigderson [27] used the same technique to give a $2^{\Omega(\sqrt{n})}$ lower bound on the monotone formula size of the *matching* problem.

Generalizing this approach, Raz and Mckenzie [26] proved the first *lifting theorem* for communication complexity. In modern terms, the main idea of their proof is to show that for certain "structured" boolean functions, the communication complexity of the Karchmer-Wigderson game is tightly related to the *decision tree complexity* of a certain search problem related to the Karchmer-Wigderson game. Since decision tree complexity is typically much easier to lower bound than communication complexity, they were able to employ this lifting theorem to separate the *monotone* NC *hierarchy*, showing that $O(\log^i n)$-depth monotone circuits are strictly weaker than $O(\log^{i+1} n)$-depth monotone circuits. This work has been hugely influential in communication complexity: lifting theorems have now been proved which connect a wide variety of communication models to simpler query models, and have led to strong lower bounds in classical, quantum, and number-on-forehead communication complexity models [6, 12, 21, 33]. In fact, a lifting theorem recently proved by Göös and Pitassi [13] was used to give $2^{\Omega(n/\log n)}$ lower bounds on the size of monotone formulas computing a monotone boolean function in NP, which was the strongest lower bound known until our result.

For monotone switching networks, the strongest lower bounds follow from the monotone formula lower bounds. In particular, it is known that if a monotone boolean function f has a monotone switching network of size s, then it has a monotone formula of size $2^{O(\log^2 s)}$ [4]. Applying the $2^{\Omega(n/\log n)}$ lower bounds on formula size proven by Göös and Pitassi [13] yields monotone switching network lower bounds of $2^{\Omega(\sqrt{n/\log n})}$, which were the strongest previously known. In a parallel, but impressive, pair of works, Potechin [24] and Chan-Potechin [5] gave an ingenious Fourier-analytic technique proving $n^{\Omega(n^{1/10})}$ lower bounds on monotone switching networks computing a certain function in mP.

Monotone span programs have a long history of lower bounds [1, 2, 7, 8, 10, 11, 19]. We give an abridged version. The first lower bounds were on the order of $\Omega(n \log n)$ for monotone span programs computing threshold functions [19]. The first superpolynomial lower bounds of $n^{\Omega(\log n/\log\log n)}$ were proven by Babai et al [1], which was strengthened to $n^{\Omega(\log n)}$ by Gál [10], who also connected span program size to the *rank measure* [29]. Finally, Robere et al [7] used this connection to prove the first exponential lower bounds on the order of $2^{\Omega(n^{\varepsilon})}$ against monotone span programs.

Let us discuss the rank measure further. Soon after the original paper of Karchmer and Wigderson [18], Razborov [29] introduced a simple matrix-theoretic complexity measure called the *rank measure*, and showed that lower bounds on the rank measure imply formula size lower bounds. The rank measure is both elegant and powerful: it immediately gives lower bounds for monotone formulas, monotone switching networks, monotone span programs and even monotone comparator circuits [7, 10]. Despite its strength, the strongest known lower bounds on the rank measure of an explicit boolean function was on the order of $n^{\Omega(\log n)}$ [10] until Robere et al [7] proved the first exponential lower bounds on the order of $2^{\Omega(n^{\varepsilon})}$ on the size of the rank measure for a function in mP. This implied the first exponential lower bounds on both monotone span programs, and the first superpolynomial lower bounds on monotone comparator circuits. The proof from [7] proceeds by proving a lifting theorem for the rank measure — connecting it to an algebraic complexity measure called the *algebraic gap complexity*.

Our Technique. In this work we use a refined version of the lifting theorem of Robere et al to obtain a $2^{\Omega(n)}$ lower bound on the rank measure for a function in NP [7]. Now, lifting theorems for monotone formula size were obtained by both Raz-Mckenzie [26] and Göös-Pitassi [13], which suggests a natural question: what prevented any of these other lifting theorems from obtaining the asymptotically optimal lower bounds we obtain in Theorem 1.1?

To answer this question, it will help to dive a little deeper into how the lifting theorems work. The central idea is this. We start with an unsatisfiable CNF $\mathcal{C}$ on input variables $z_1, z_2, \ldots, z_m$, and consider the following *search problem* $\mathsf{Search}(\mathcal{C})$: given an assignment z to the variables of $\mathcal{C}$, output a clause $C \in \mathcal{C}$ that is falsified by z. We then transform the search problem into a two-party communication problem as follows:

Gadget. Choose a *two-player gadget* $g : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$.

Compose. Replace each variable z_i in the CNF $\mathcal{C}$ with $g(x_i, y_i)$, where x_i, y_i are new variables.

The new communication problem, denoted $\mathsf{Search}(\mathcal{C}) \circ g^m$, is defined as follows: Alice receives $x \in \mathcal{X}^m$, Bob receives $y \in \mathcal{Y}^m$, and their goal is to evaluate the gadget g on each pair of inputs (x_i, y_i) and solve the search problem on the string

$$z = g(x_1, y_1)g(x_2, y_2)\cdots g(x_m, y_m).$$

Using the Karchmer-Wigderson connection [18], lower bounds on the communication complexity $\mathsf{Search}(\mathcal{C}) \circ g^m$ yield lower bounds on the size of a related monotone boolean function [13, 26]. A *lifting theorem* then shows that the communication complexity of $\mathsf{Search}(\mathcal{C}) \circ g^m$ is related to a simpler complexity measure (such as decision tree complexity) on $\mathsf{Search}(\mathcal{C})$.

Now, in any lifting theorem there is an inherent tradeoff between the size of the gadget g (measured by $|\mathcal{X}|, |\mathcal{Y}|$), the resulting communication lower bound against $\mathsf{Search}(\mathcal{C}) \circ g^m$, and the strength of the complexity measure on $\mathsf{Search}(\mathcal{C})$. For instance, the Raz-Mckenzie lifting theorem transforms *decision tree* lower bounds on $\mathsf{Search}(\mathcal{C})$ into communication lower bounds against $\mathsf{Search}(\mathcal{C}) \circ g^m$ [14,26]. Of course, we have very strong lower bounds for decision trees, but in order to translate lower bounds for such a weak measure into communication lower bounds we (apparently) need a large gadget, causing a significant loss in the final lower bound. On the other hand, the Göös-Pitassi lifting theorem transforms *critical block sensitivity* lower bounds on $\mathsf{Search}(\mathcal{C})$ into communication lower bounds [13]. In this lifting theorem we can use a constant-size gadget — yielding an optimal translation of lower bounds — but we do not know how to prove strong enough lower bounds on critical block sensitivity!

Our refined lifting theorem using the rank measure avoids both of these problems. The lifting theorem of Robere et al [7] connects lower bounds on the *algebraic gap complexity* of $\mathsf{Search}(\mathcal{C})$ with lower bounds on the rank measure of a related monotone boolean function. Here, we show that if one can obtain *linear* lower bounds on the algebraic gap complexity of $\mathsf{Search}(\mathcal{C})$, then we can use a constant-size gadget in the lifting theorem and so obtain $2^{\Omega(n)}$ lower bounds for the resulting monotone boolean function. In the previous work [7] the best algebraic gaps were sublinear (specifically, on the order of $\Theta(n/\log n)$) which prevented the use of a constant-size gadget in the lifting theorem. We are then able to obtain linear lower bounds on the algebraic gap complexity of $\mathsf{Search}(\mathsf{Tseitin}_G)$, where $\mathsf{Tseitin}_G$ is the CNF encoding the well known *Tseitin contradictions* on a highly expanding graph G.

Proof Outline. We now give a technical outline of our proof, which generally follows the same outline as in [7]. Let $f : \{0,1\}^n \to \{0,1\}$ be a monotone function. Let $\mathcal{U} \subseteq f^{-1}(1)$ be a subset of the 1-inputs of f, and let $\mathcal{V} \subseteq f^{-1}(0)$ be a subset of the 0-inputs of f. Let A be a $|\mathcal{U}| \times |\mathcal{V}|$ matrix over $\mathbb{R}$ with rows labelled by $u \in \mathcal{U}$ and columns labelled by $v \in \mathcal{V}$. For each underlying input variable x_i of f, define the subrectangle R_i to be the set of pairs $(u, v) \in \mathcal{U} \times \mathcal{V}$ such that $u_i = 1$ and $v_i = 0$. Let $\mathcal{R}_f(\mathcal{U}, \mathcal{V})$ denote the collection of all of these rectangles.

The *rank measure* of A is defined to be the ratio of the rank of A to the maximum rank of the submatrix of A indexed by any of these rectangles

$$\mu_A(f) = \frac{\operatorname{rank} A}{\max\limits_{R \in \mathcal{R}_f(\mathcal{U},\mathcal{V})} \operatorname{rank} A{\restriction}_R}.$$

This measure was originally introduced by Razborov [29], and for any A the measure $\mu_A(f)$ is a lower bound on each of the monotone computation models we have discussed above. Thus, our overall goal is to find a family of matrices $\{A_n\}$ for an explicit monotone function f for which the rank measure is $2^{\Omega(n)}$.

At a high level our lower bound argument proceeds in two steps. First, we use the lifting theorem from [7] connecting the rank measure to an algebraic measure on polynomials called the *algebraic gap complexity*. The lifting theorem uses the seminal *Pattern Matrix Method* due to Sherstov [33], which was followed by many other query-to-communication complexity lifts in the literature [6,13,16,20,21, 26,32]. The second and main step is to prove linear lower bounds on the algebraic gap complexity for some explicitly defined search problem.

Let us proceed in more detail. Let $\mathcal{C}$ be an unsatisfiable d-CNF on m variables $z_1, z_2, \ldots, z_m$ and clauses $C_1, C_2, \ldots, C_s$ — for instance, take a Tseitin formula — and let

$$H = (\{z_1, \ldots, z_m\} \cup \{C_1, \ldots, C_s\}, E)$$

be a bipartite graph encoding the topology of $\mathcal{C}$ (so, there are edges connecting each variable z_i with the constraints C_j containing it). Given H and any alphabet Σ, one can consider the following *monotone* variant of the constraint satisfaction problem $SAT_{\Sigma,H}$ which is defined as follows. The input is a binary string x of length $s \cdot |\Sigma|^d$ encoding truth tables for each of the constraints $C_1, C_2, \ldots, C_s$ (thus, we have "forgotten" that these are clauses in $\mathcal{C}$, and now treat them as arbitrary constraints over the alphabet Σ). Given such a string x, $SAT_{\Sigma,H}(x) = 1$ if and only if the CSP encoded by x is satisfiable. Our lifting theorem transforms *algebraic gaps* (an algebraic query complexity measure) on a search problem associated with $\mathcal{C}$ into lower bounds on the *rank measure* of $SAT_{\Sigma,H}$ for some domain Σ.

Step 1: The Pattern Matrix Lift. Sherstov [33] gave a general method to construct a "pattern matrix" A_p from a boolean function $p : \{0,1\}^m \to \mathbb{R}$ such that the analytic properties of A_p are related to the Fourier analytic properties of the function p. The matrix is constructed as follows: the columns of A_p are indexed by strings $y \in \{0,1\}^n$ for some $n > m$, the rows of A_p are indexed by pairs (x, w) where $x \in [n/m]^m$ is a string of "pointers" to indices in y, $w \in \{0,1\}^m$, and then for each pair $((x,w), y)$ the value $A_p[(x,w), y]$ is $p(y{\restriction_x} \oplus w)$.

We begin with the Tseitin formulas $\mathsf{Tseitin}_G$ on a highly-expanding, constant-degree graph G. The main idea is to use a pattern matrix A_p for some intelligently chosen p to certify a lower bound on the rank measure for $SAT_{\Sigma,H}$ where H encodes the topology of the Tseitin formula and $\Sigma = ([n/m] \times \{0,1\})^m$ is the domain of (x, w). To do this, we show that each row (x, w) of the pattern matrix A_p can be interpreted as a satisfiable CSP instance of $SAT_{\Sigma,H}$, while each column y of A_p can be interpreted as an unsatisfiable CSP. Under this interpretation, each rectangle R in $\mathcal{R}_{SAT_{\Sigma,H}}$ corresponds in a very natural way to a clause C of the underlying Tseitin formula $\mathsf{Tseitin}_G$. A theorem of Sherstov [33] allows us to connect the rank of the matrix A to the Fourier spectrum of p, and similarly the rank of each "rectangle submatrix" $A{\restriction_R}$ to the Fourier spectrum of $p{\restriction_C}$, where $p{\restriction_C}$ is a restriction of the function p obtained naturally from the clause C underlying the rectangle R. The end result is that the matrix A_p will certify a large rank measure if the function p exhibits a large *algebraic gap*, in that the Fourier degree of p is large, but the Fourier degree of each of the restrictions $p{\restriction_C}$ is small.

Step 2: Exhibiting Large Algebraic Gaps. The second step of our argument is to actually construct a function p exhibiting linear-size algebraic gaps for $\mathsf{Tseitin}_G$. First, we show that for each positive integer m the problem of constructing a boolean function $p : \{0,1\}^m \to \mathbb{R}$ with large algebraic gap is equivalent to the satisfiability of a system of linear equations. To show this system is satisfiable, we introduce a new proof system that is similar to width-restricted Gaussian refutations. Our main technical argument is a completeness theorem, showing that this system of linear equations is satisfiable if the (width-restricted) proof system cannot refute the unsatisfiable Tseitin formulas. Finally, we employ the known lower bounds on Gaussian width by the expansion of the underlying graph of the Tseitin formula to get linear algebraic gaps.

2 Definitions

A real-valued boolean function is any function $p : \{0,1\}^n \to \mathbb{R}$. If A is any set and $x \in A^n$ we let x_i denote the ith component of x. If $x, y \in \{0,1\}^n$ we let $x \oplus y \in \{0,1\}^n$ denote the string obtained by taking the bitwise XOR of x and y.

For any n, the collection of all n-ary real-valued boolean functions $\{p : \{0,1\}^n \to \mathbb{R}\}$ forms a vector space under pointwise addition and scalar multiplication. For any $C \subseteq [n]$, the *Fourier character* at C is the function $\chi_C : \{0,1\}^n \to \{-1,1\}$ defined by

$$\chi_C(x) = (-1)^{\sum_{i \in C} x_i}.$$

The collection of characters $\{\chi_C\}_{C \subseteq [n]}$ form an orthonormal basis for the vector space of real-valued boolean functions known as the *Fourier basis*, where the vector space is equipped with the inner product

$$\langle p, q \rangle = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} p(x)q(x).$$

Since this basis is orthonormal, given any function $p : \{0,1\}^n \to \mathbb{R}$, we can represent p in the Fourier basis as

$$p(x) = \sum_{C \subseteq [n]} \langle p, \chi_C \rangle \chi_C(x).$$

This representation is called the *Fourier transform* of p.

We let $\hat{p}(C) = \langle p, \chi_C \rangle$ denote the coefficient of χ_C of p in the Fourier basis — this is the *Fourier coefficient* of p at C. The collection of non-zero Fourier coefficients of p is called the *Fourier spectrum* of p. The *Fourier degree* is the size of the largest non-zero Fourier coefficient of p:

$$\deg p = \max_{S \subseteq [m]} \{|S| \mid \hat{p}(S) \neq 0\},$$

which, equivalently, is the degree of the unique representation of p as a multilinear polynomial over the real numbers. See [23] for a comprehensive survey of boolean function analysis.

If $x, y \in \{0,1\}^n$ then we write $x \leq y$ if $x_i \leq y_i$ for all i. A function $f : \{0,1\}^n \to \{0,1\}$ is *monotone* if $f(x) \leq f(y)$ whenever $x \leq y$. If f is monotone then an input $x \in \{0,1\}^n$ is a *maxterm* of f if $f(x) = 0$ but $f(x') = 1$ for any x' obtained from x by flipping a single bit from 0 to 1; dually, x is a *minterm* if $f(x) = 1$ but $f(x') = 0$ for any x' obtained by flipping a single bit of x from 1 to 0. More generally, if $f(x) = 1$ we call x an *accepting instance* or a *yes instance*, while if $f(x) = 0$ then we call x a *rejecting instance* or a *no instance*. If x is any yes instance of f and y is any no instance of f then there exists an index $i \in [n]$ such that $x_i = 1, y_i = 0$, as otherwise we would have $x \leq y$, contradicting the fact that f is monotone.

Suppose that $\mathcal{U}, \mathcal{V} \subseteq \{0,1\}^n$ are any sets satisfying $f(\mathcal{U}) = 1, f(\mathcal{V}) = 0$. A set $R \subseteq \mathcal{U} \times \mathcal{V}$ is called a *rectangle* if there are sets $\mathcal{U}_0 \subseteq \mathcal{U}, \mathcal{V}_0 \subseteq \mathcal{V}$ such that $R = \mathcal{U}_0 \times \mathcal{V}_0$. For each $i \in [n]$ let

$$X_i = \{x \in \{0,1\}^n \mid x_i = 1\} \times \{x \in \{0,1\}^n \mid x_i = 0\},$$

and let $R_i = X_i \cap (\mathcal{U} \times \mathcal{V})$. Denote by $\mathcal{R}_f(\mathcal{U}, \mathcal{V})$ the collection of rectangles

$$\mathcal{R}_f(\mathcal{U}, \mathcal{V}) = \{R_i \mid i = 1, 2, \ldots, n\}.$$

Since f is a monotone function there is an index i such that $u_i = 1, v_i = 0$ for all $u \in \mathcal{U}, v \in \mathcal{V}$, and so every entry of $\mathcal{U} \times \mathcal{V}$ is covered by some rectangle in $R_f(\mathcal{U}, \mathcal{V})$. Let A be any $|\mathcal{U}| \times |\mathcal{V}|$ matrix with rows labelled by entries of $\mathcal{U}$ and columns labelled by entries of $\mathcal{V}$, and if $S \subseteq \mathcal{U} \times \mathcal{V}$ is any subset of $\mathcal{U} \times \mathcal{V}$ let $A\restriction_S$ be the submatrix indexed by S.

Definition 2.1. Let $f : \{0,1\}^n \to \{0,1\}$ and let $\mathcal{U} \subseteq f^{-1}(1), \mathcal{V} \subseteq f^{-1}(0)$. Let A be any $|\mathcal{U}| \times |\mathcal{V}|$ matrix over $\mathbb{R}$[1]. The *rank measure* of f with respect to A is

$$\mu_A(f) := \frac{\operatorname{rank}(A)}{\max\limits_{R \in \mathcal{R}_f(\mathcal{U}, \mathcal{V})} \operatorname{rank}(A\restriction_R)}.$$

[1]This definition makes sense with respect to any field, but we will work exclusively in the reals.

2.1 Unsatisfiable Formulas and Search Problems

Let $\mathcal{C}$ be an unsatisfiable d-CNF formula over variables $z_1, \ldots, z_m$. The search problem associated with $\mathcal{C}$, $\mathsf{Search}(\mathcal{C})$, takes as input an assignment $\alpha \in \{0,1\}^m$ to $z_1, \ldots, z_m$ and the goal is to output a clause C_i such that $C_i(\alpha) = 0$. Each clause C_i has a unique falsifying assignment, and given a boolean function p on the same variables as $\mathcal{C}$, we let $p\restriction_C$ denote the restriction of the function p by this assignment.

The communication version of the search problem $\mathsf{Search}(\mathcal{C})$ is obtained by composing (or *lifting*) $\mathsf{Search}(\mathcal{C})$ with a two-party gadget $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ to obtain a new problem $\mathsf{Search}(\mathcal{C}) \circ g^m$ in the natural way: Alice gets $x \in \mathcal{X}^m$ as input, Bob gets $y \in \mathcal{Y}^m$ as input, and their goal is to find a clause $C_i \in \mathcal{C}$ that is violated for the input

$$z = g^m(x,y) = (g(x_1,y_1), \ldots, g(x_m,y_m)).$$

Tseitin Formulas. We will be interested in the unsatisfiable *Tseitin formulas* and their associated search problems. Let $G = (V,E)$ be a connected d-regular graph with a *node labelling* $\ell : V \to \{0,1\}$ which has *odd weight*, i.e. $\sum_{u \in V} \ell(v) = 1 \pmod 2$. If $z : E \to \{0,1\}$ is an edge-labelling and $v \in V$ is a vertex, we write

$$z(v) := \sum_{e \ni v} z(e) \bmod 2$$

to be the sum of the labels of the edges adjacent to v modulo 2. Typically we choose G to be a highly expanding d-regular graph on an odd number of vertices and $m = d|V|/2$ edges, and each vertex will be labelled $\ell(v) = 1$.

Given such a graph G, the *Tseitin formula* $\mathsf{Tseitin}_G$ is a d-CNF on $m = d|V|/2$ variables with one variable x_e for each edge e in G. For each vertex $v \in V$, $\mathsf{Tseitin}_G$ contains 2^{d-1} clauses encoding the equation $z(v) = \ell(v)$. The formula is clearly unsatisfiable: $\sum_v z(v)$ is even because every edge is counted twice, but on the other hand the sum $\sum_v l(v) = |V|$ is odd by assumption. The search problem $\mathsf{Search}(\mathsf{Tseitin}_G)$ therefore has a natural semantics: given an edge-labelling $z : E \to \{0,1\}$, find a node $v \in V$ which has a *parity violation* $z(v) \neq l(v)$.

The communication version of the Tseitin problem is obtained by composing $\mathsf{Search}(\mathsf{Tseitin}_G)$ with a two-party gadget $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$. We define our gadget g later, but what is important is that it is *constant-size* in the sense that $|\mathcal{X}|$ and $|\mathcal{Y}|$ are fixed constants independent of m. In the lifted communication search problem $\mathsf{Search}(\mathsf{Tseitin}_G) \circ g^m$, Alice gets $x \in \mathcal{X}^m$ as input, Bob gets $y \in \mathcal{Y}^m$ as input, and their goal is to find a node $v \in V$ that with a parity violation under the edge-labelling

$$z = g^m(x,y) = (g(x_1,y_1), \ldots, g(x_m,y_m)).$$

2.2 From Search Problems to Monotone CSPs

We now define the monotone variant of the constraint satisfaction problem (CSP) for which we obtain strongly exponential rank lower bounds. The function is defined relative to some finite alphabet Σ and a bipartite graph $H = (L \cup R, E)$ encoding the topology of a CSP $\mathcal{H}$. In H, the left vertices L correspond to a collection of Σ-valued variables, and the right vertices R correspond to the constraints of $\mathcal{H}$. Given a constraint $C \in R$, let $\mathsf{vars}(C)$ denote the variables involved in C (or, equivalently, the neighborhood of C in the graph H). We assume that each constraint C satisfies $|\mathsf{vars}(C)| = d$.

Definition 2.2. Let $H = (L \cup R, E)$ be a bipartite graph, let Σ be a finite alphabet, and let $N = |R| \cdot |\Sigma|^d$. The monotone function $SAT_{\Sigma,H} : \{0,1\}^N \to \{0,1\}$ is defined as follows. An input

$x \in \{0,1\}^N$ defines a CSP instance $\mathcal{H}(x)$ with topology H by specifying for each constraint C in $\mathcal{H}$ a truth table $\Sigma^{\mathsf{vars}(C)} \to \{0,1\}$ of satisfying assignments to that constraint. Given an input x, $SAT_{\Sigma,H}(x) = 1$ if and only if the CSP $\mathcal{H}(x)$ is satisfiable.

For any Σ, H observe that $SAT_{\Sigma,H}$ is monotone since replacing a 0 with a 1 in the truth table of any constraint preserves the constraint's satisfying assignments. Note that if H represents the topology of a linear size d-CSP then $N = \Theta(m)$ if $d, |\Sigma|$ are constants. In particular, if H encodes the topology of the Tseitin formula on a constant-degree graph with m edges, then $N = \Theta(m)$ if $|\Sigma|$ is constant.

Suppose that $\mathcal{C}$ is an unsatisfiable d-CNF on variables $z_1, z_2, \ldots, z_m$, and let $H = (L \cup R, E)$ be the constraint graph representing the topology of $\mathcal{C}$. If $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ is a two-party gadget, then there is a natural way to convert inputs $x \in \mathcal{X}^m$ and $y \in \mathcal{Y}^m$ in the lifted search problem $\mathsf{Search}(\mathcal{C}) \circ g^m$ into minterms and maxterms of $SAT_{\mathcal{X},H}$:

Accepting Inputs $\mathcal{U}$. Alice maps her $x \in \mathcal{X}^m$ into the accepting input $Y(x)$ of $SAT_{\mathcal{X},H}$ for which the unique satisfying assignment to the CSP encoded by $Y(x)$ is x. Formally, for each constraint $C \in R$, the truth table $\mathcal{X}^{\mathsf{vars}(C)} \to \{0,1\}$ is entirely 0 except for a single 1 in the position $x\restriction_{\mathsf{vars}(C)}$.

Rejecting Inputs $\mathcal{V}$. Bob maps his $y \in \mathcal{Y}^m$ into the rejecting input $N(y)$ of $SAT_{\mathcal{X},H}$ as follows. For each constraint $C \in R$, the truth table $t_C : \mathcal{X}^{\mathsf{vars}(C)} \to \{0,1\}$ has $t_C(\alpha) = 1$ if and only if the boolean string $g^{\mathsf{vars}(C)}(\alpha, y\restriction_{\mathsf{vars}(C)}) \in \{0,1\}^{\mathsf{vars}(C)}$ satisfies the corresponding clause C of the underlying CNF $\mathcal{C}$.

Note the distinction between the two constraint satisfaction problems above. The unsatisfiable CNF $\mathcal{C}$ is fixed beforehand, and we use its constraint graph H as the underlying topology of the possible input CSPs to $SAT_{\mathcal{X},H}$. The domain of the CSPs in the $SAT_{\mathcal{X},H}$ problem is $\mathcal{X}$, which is the domain of Alice's inputs to $\mathsf{Search}(\mathcal{C}) \circ g^m$.

It is clear that the inputs $Y(x)$ are accepted by $SAT_{\mathcal{X},H}$, since the CSP encoded by $Y(x)$ is satisfied by x. To see that the inputs $N(y)$ are rejecting inputs, suppose otherwise and let $x \in \mathcal{X}^m$ be the satisfying assignment of the CSP encoded by $N(y)$. By definition of $N(y)$, it follows that for each constraint $C \in \mathcal{C}$ we have $t_C(x\restriction_{\mathsf{vars}(C)}) = 1$, which only occurs if $g^m(x,y)$ is a satisfying assignment to the CNF formula $\mathcal{C}$. This is a contradiction since $\mathcal{C}$ is unsatisfiable.

3 Rank Measure Lower Bounds

In this section we state our main theorem and record its consequences.

Theorem 3.1. *Let N be a sufficiently large positive integer. There is an explicit monotone function $f : \{0,1\}^N \to \{0,1\}$ computable in NP, sets $\mathcal{U} \subseteq f^{-1}(1)$, $\mathcal{V} \subseteq f^{-1}(0)$, and a real-valued $|\mathcal{U}| \times |\mathcal{V}|$ matrix A such that*

$$\mu_A(f) \geq \frac{2^{\alpha N}}{11}$$

for some universal constant $\alpha > 0$.

There is a long history of results relating the rank measure μ_A to monotone computational models. For the sake of brevity, we record them all in a single lemma below. First we record some notation representing monotone circuit complexity classes. If $f : \{0,1\}^N \to \{0,1\}$ is a monotone function, let

$\mathsf{mNC}^1(f) =$ monotone formula size of f,

$\mathsf{mL}(f) =$ monotone switching network size of f,

$\mathsf{mSP}_{\mathbf{F}}(f) =$ monotone span program size (over the field $\mathbf{F}$) of f,

$\mathsf{mCC}(f) =$ monotone comparator circuit size of f.

Lemma 3.2. *Let f be a monotone boolean function. Let $\mathbf{F}$ be any field, and let A be any matrix over $\mathbf{F}$. Then*

$$\mu_A(f) \leq \mathsf{mSP}_{\mathbf{F}}(f) \leq \mathsf{mL}(f) \leq \mathsf{mNC}^1(f),$$

and

$$\mu_A(f) \leq \mathsf{mCC}(f) \leq \mathsf{mNC}^1(f).$$

Proof Sketch. We record references for each of the results:

1. $\mathsf{mL}(f) \leq \mathsf{mNC}^1(f)$ is folklore, and can be found in Jukna [17],
2. $\mathsf{mSP}_{\mathbf{F}}(f) \leq \mathsf{mL}(f)$ was proved by Karchmer and Wigderson [19],
3. $\mu_A(f) \leq \mathsf{mSP}_{\mathbf{F}}(f)$ was shown by Gál [10],
4. The two results $\mu_A(f) \leq \mathsf{mCC}(f) \leq \mathsf{mNC}^1(f)$ are proved by Robere et al [7]. □

By the previous lemma, Theorem 3.1 implies strongly exponential lower bounds in all of these models.

Theorem 1.1. *There is a universal constant $\alpha > 0$ and an explicit monotone function computable in NP on N variables that requires size $2^{\alpha N}/11$ in the following models of computation: monotone formulas, monotone switching networks, real monotone span programs, and monotone comparator circuits.*

Note that for any monotone function f on N inputs we have $\mathsf{mNC}^1(f) \leq N2^N$ since we can construct a simple monotone DNF accepting each of the minterms of f. It follows that our lower bound is asymptotically the best possible — up to constants in the exponent — in each of these models for *any* monotone function.

The function f for which we obtain the strongly exponential lower bound in Theorem 3.1 is $SAT_{\mathcal{X},H}$ where $\mathcal{X}$ is some large (but constant-size) domain and H is the constraint graph encoding the Tseitin formula on a highly expanding d-regular graph. The proof of the lower bound closely follows the previous lower bounds on the rank measure in [7]: in Section 4, we prove a *lifting theorem* which reduces constructing a matrix A for Theorem 3.1 to obtaining large *algebraic gaps* on the search problem $\mathsf{Search}(\mathsf{Tseitin}_G)$. Then, in Section 5, we show that $\mathsf{Search}(\mathsf{Tseitin}_G)$ has linear-size algebraic gaps.

4 Reducing the Rank Measure to Algebraic Gaps

The rest of the paper is devoted to the proof of Theorem 3.1. We first define an algebraic query complexity measure that was introduced in [7]. Recall that $\deg p$ is the size of the largest non-zero Fourier coefficient of p.

Definition 4.1. Let $\mathcal{C}$ be an unsatisfiable CNF on m variables. The *algebraic gap complexity* of $\mathsf{Search}(\mathcal{C})$ is the largest integer k for which there is a boolean function $p : \{0,1\}^m \to \mathbb{R}$ such that

$$\deg p = m \quad \text{and} \quad \deg p{\restriction_C} \leq m - k$$

for all clauses C in $\mathcal{C}$.

Next we state and prove our lifting theorem. The proof of this theorem follows the strategy from [7], although we pay closer attention to the size of the gadget. A striking consequence is that if we can find an unsatisfiable CNF $\mathcal{C}$ with constant width and $\Omega(m)$-size algebraic gaps then we obtain *strongly* exponential lower bounds on the rank measure for the corresponding SAT problem. To contrast, the strongest algebraic gap bounds proved in [7] are sublinear, which led to *weakly* exponential lower bounds.

Theorem 4.2. *Let m, d be positive integers and let $\mathcal{C}$ be an unsatisfiable d-CNF on m variables and with constraint graph H. Suppose the algebraic gap complexity of* $\mathsf{Search}(\mathcal{C})$ *is εm for some $\varepsilon > 0$. Let $\lambda > 2^{1/\varepsilon}$ be a positive integer and let $\mathcal{X} = [\lambda] \times \{0,1\}$. Let $\delta = \delta(\varepsilon) = \varepsilon \log_2 \lambda - 1$. There is a matrix A such that*

$$\mu_A(SAT_{\mathcal{X},H}) \geq \frac{2^{\delta m}}{d+1}.$$

Proof. Let $p : \{0,1\}^m \to \mathbb{R}$ be the function witnessing the algebraic gap complexity of $\mathsf{Search}(\mathcal{C})$. Let $\mathcal{X} = [\lambda] \times \{0,1\}$ and let $\mathcal{Y} = \{0,1\}^\lambda$. Define the *Sherstov gadget* $g_\lambda : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ to be the function $g((x,w),y) = y_x \oplus w$, and let

$$A = [p(g_\lambda^m(x,y))]_{x \in \mathcal{X}^m, y \in \mathcal{Y}^m}.$$

(The matrix A is typically called a *pattern matrix* in the literature). Let $\mathcal{U}, \mathcal{V}$ be the collections of minterms and maxterms (resp.) of $SAT_{\mathcal{X},H}$ constructed using the communication problem $\mathsf{Search}(\mathcal{C}) \circ g_\lambda^m$. We lower bound

$$\mu_A(SAT_{\mathcal{X},H}) = \frac{\operatorname{rank} A}{\max\limits_{R \in \mathcal{R}_{SAT_{\mathcal{X},H}}(\mathcal{U},\mathcal{V})} \operatorname{rank} A{\restriction_R}}.$$

To bound the rank of the numerator, we use a lemma by Sherstov [33] showing that the rank of A is completely specified by the Fourier spectrum of p.

Lemma 4.3 (Theorem 4.3 in [33])**.** *The rank of $A = [p(g_\lambda^m(x,y))]_{x \in \mathcal{X}^m, y \in \mathcal{Y}^m}$ is*

$$\operatorname{rank} A = \sum_{S : \hat{p}(S) \neq 0} \lambda^{|S|}.$$

The companion to the previous lemma, proved in [7], shows that the rank of each submatrix $A{\restriction_R}$ is specified by the Fourier spectrum of the restricted function $p{\restriction_C}$, where C is a clause of $\mathcal{C}$.

Lemma 4.4 (Lemma 4.6 in [7])**.** *Consider the communication search problem* $\mathsf{Search}(\mathcal{C}) \circ g_\lambda^m$. *Let (C, α) be any input variable of $SAT_{\mathcal{X},H}$, and let R be the rectangle corresponding to (C,α) in $\mathcal{R}_{SAT_{\mathcal{X},H}}(\mathcal{U},\mathcal{V})$. Then*

$$\operatorname{rank}(A{\restriction_R}) = \sum_{S : \widehat{p{\restriction_C}}(S) \neq 0} \lambda^{|S|}.$$

Note that the above sum is taken over all subsets S of the unrestricted variables of $p{\restriction_C}$.

Since $\deg p = m$, Lemma 4.3 implies that

$$\operatorname{rank} A = \sum_{S:\hat{p}(S)\neq 0} \lambda^{|S|} \geq \lambda^m.$$

On the other hand, let $R \in \mathcal{R}_{SAT_{\mathcal{X},H}}(\mathcal{U},\mathcal{V})$ be chosen arbitrarily and let (C,α) be the input variable of $SAT_{\mathcal{X},H}$ corresponding to R. Note that we may assume that $\hat{p}(S) = 0$ for all $S \subseteq [m]$ with $|S| < m - \varepsilon m$ w.l.o.g. since this does not affect the algebraic gap exhibited by p. Since $\deg p\restriction_C \leq m - \varepsilon m$, it follows that all of the non-zero Fourier coefficients $\widehat{p\restriction_C}(S')$ are obtained as a linear combination of non-zero Fourier coefficients of $\hat{p}(S)$ where $|S| \leq |S'| + d$. Applying Lemma 4.4 and using these two facts, we have

$$\operatorname{rank} A\restriction_R = \sum_{S:\widehat{p\restriction_C}(S)\neq 0} \lambda^{|S|} \leq \sum_{i=0}^{d} \binom{m}{m-\varepsilon m - i} \lambda^{(m-\varepsilon m - i)} \leq 2^m \sum_{i=0}^{d} \lambda^{(m-\varepsilon m-i)} \leq 2^m(d+1)\lambda^{(m-\varepsilon m)}.$$

Putting it all together we get

$$\mu_A(f_{\mathcal{C}}) = \frac{\operatorname{rank} A}{\max_{R\in\mathcal{R}_{f_{\mathcal{C}}}(\mathcal{Y},\mathcal{N})} \operatorname{rank} A\restriction_R} \geq \frac{\lambda^m}{(d+1)2^m\lambda^{(m-\varepsilon m)}} \geq \frac{\lambda^{\varepsilon m}}{(d+1)2^m} \geq \frac{2^{\delta m}}{d+1}$$

where $\delta = \varepsilon \log_2 \lambda - 1$. □

5 Algebraic Gaps for the Tseitin Search Problem

In this section, we will show that if G is a highly expanding d-regular graph, then $\mathsf{Search}(\mathsf{Tseitin}_G)$ has linear algebraic gap complexity. Let $G = (V,E)$ be any d-regular graph with an odd number of nodes, and for any subset of vertices $U \subseteq V$ let $\mathrm{Cut}(U)$ denote the collection of edges with exactly one endpoint in U. Recall that the *(edge-)expansion* of G is

$$\varepsilon(G) = \min\left\{ \frac{|\mathrm{Cut}(U)|}{|U|} : U \subseteq V, 0 < |U| \leq \lfloor |V|/2 \rfloor \right\}.$$

The following theorem is the main result of this section.

Theorem 5.1. *Let d be a positive integer, let $G = (V,E)$ be a d-regular graph with an odd number of nodes and $m = d \cdot |V|/2$ edges. Then $\mathsf{Search}(\mathsf{Tseitin}_G)$ has algebraic gap complexity at least $\varepsilon(G)m/3d$.*

Using Theorems 5.1 and 4.2 we can prove our main theorem.

Theorem 3.1. *Let N be a sufficiently large positive integer. There is an explicit monotone function $f : \{0,1\}^N \to \{0,1\}$ computable in NP, sets $\mathcal{U} \subseteq f^{-1}(1)$, $\mathcal{V} \subseteq f^{-1}(0)$, and a real-valued $|\mathcal{U}| \times |\mathcal{V}|$ matrix A such that*

$$\mu_A(f) \geq \frac{2^{\alpha N}}{11}$$

for some universal constant $\alpha > 0$.

Proof. Recall that *Ramanujan graphs* are d-regular graphs $G = (V, E)$ with expansion

$$\varepsilon(G) \geq 1/2 - 1/\sqrt{d-1}.$$

Marcus, Spielman and Srivastava [22] proved that Ramanujan graphs[2] exist for all d and $n = |V|$. Let $d = 10$ and let $G = (V, E)$ be any Ramanujan graph with $|V|$ odd and $m = d|V|/2$ edges. By Theorem 5.1, the search problem $\mathsf{Search}(\mathsf{Tseitin}_G)$ has algebraic gap complexity at least

$$\varepsilon(G)m/3d \geq \left(\frac{1}{6d} - \frac{1}{3d\sqrt{d-1}}\right) m = \frac{m}{180}.$$

Let $\lambda = 2^{181}$; then $\delta = \log_2(\lambda)/180 - 1 = 181/180 - 1 > 0$. Applying Theorem 4.2, there is a matrix A such that

$$\mu_A(SAT_{\mathcal{X},H}) \geq \frac{2^{\delta m}}{11}$$

where H is the constraint graph of $\mathsf{Tseitin}_G$ and $\mathcal{X} = [\lambda] \times \{0,1\}$. By definition, the function $SAT_{\mathcal{X},H}$ has $N = 2^{d-1}m(2\lambda)^d$ input variables, and thus setting $\alpha = 2\delta/(4\lambda)^d$ we have

$$\mu_A(SAT_{\mathcal{X},H}) \geq \frac{2^{\alpha N}}{11}.$$

Moreover, it is clear that $SAT_{\mathcal{X},H}$ can be computed in NP. $\square$

In the rest of the section we focus on proving the algebraic gap complexity lower bounds for $\mathsf{Search}(\mathsf{Tseitin}_G)$. As a first step, we reformulate algebraic gaps for $\mathsf{Search}(\mathsf{Tseitin}_G)$ as a system of linear equations. We abuse notation and write $\mathrm{Cut}(u)$ to mean $\mathrm{Cut}(\{u\})$ whenever u is a vertex; note that this is simply the set of edges incident to u. Also, when writing Fourier coefficients $\hat{p}(S)$, we will write $\hat{p}(S, D)$ to mean $\hat{p}(S \cup D)$ for the sake of readability. For any positive integer k such that $0 \leq k \leq |E|$ define the following system of linear equations on variables $\hat{p}(S)$ for $S \subseteq E$.

$\mathcal{G}(G, k)$.

High Degree. $\hat{p}(E) = 1$

Vertex Constraints. For each subset $S \subseteq E$ with $|S| \geq |E| - k$, each vertex u with $\mathrm{Cut}(u) \cap S = \emptyset$, and each even-sized subset $C \subseteq \mathrm{Cut}(u)$ add the equation

$$0 = \sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|} \hat{p}(S, D).$$

Lemma 5.2. *Let $G = (V, E)$ be a d-regular, odd-sized graph, and let $0 \leq k \leq |E|$. If $\mathcal{G}(G, k)$ is satisfiable then $\mathsf{Search}(\mathsf{Tseitin}_G)$ has algebraic gap complexity at least $k + 1$.*

Proof Sketch. Let $\hat{p}$ be a solution to the system and define $p(x) = \sum_{|S| \geq m-k} \hat{p}(S)\chi_S(x)$. Since $\hat{p}(E) = 1$ we have that $\deg p = |E|$. Now, consider any vertex v in the graph G, and consider any clause C in the 2^{d-1} clauses enforcing the constraint $z(v) = 1$. If we let C^+ represent the collection of positive literals in C, then

$$\hat{p}\restriction_C(S) = \sum_{D \subseteq \mathsf{vars}(C)} (-1)^{|D \cap C^+|} \hat{p}(S, D).$$

[2]In this construction of Ramanujan graphs the resulting graphs are actually multigraphs — they can have multiple edges — but this does not affect our use of them.

Since the Fourier coefficients $\hat{p}(S)$ of p satisfy the vertex constraints for v it follows that $\hat{p}\restriction_C(S) = 0$ whenever $|S| \geq m - k$, and thus $\deg p\restriction_C \leq m - k - 1$. □

The previous lemma shows that Theorem 5.1 will follow if we can construct a solution for $\mathcal{G}(G, k)$ when $k = \varepsilon(G)m - 1$. As stated, it is not clear how to construct solutions to $\mathcal{G}(G, k)$ even for small k. Our approach is to add a new family of equations $\mathcal{E}^*$ to $\mathcal{G}(G, k)$ so that the new system is (essentially) triangular, which immediately gives us satisfiability.

Instead of working with the vertex constraints directly, it will be convenient to work with a simpler set of equations which imply the vertex constraints. For any set of edges $S \subseteq E$ and any set $U \subseteq V$ we let $\mathcal{E}(S, U)$ denote the equation

$$\hat{p}(S) = (-1)^{|U|}\hat{p}(S \triangle \mathrm{Cut}(U)).$$

Observe that $\mathcal{E}(E, V)$ is simply $\hat{p}(E) = (-1)^{|V|}\hat{p}(E)$, which is equivalent to $\hat{p}(E) = 0$ since $|V|$ is odd by assumption. We equip these equations with some natural deduction rules.

Deduction If U_1, U_2 are sets of vertices and S is a set of edges then

$$\mathcal{E}(S, U_1), \mathcal{E}(S, U_2) \vdash \mathcal{E}(S \triangle \mathrm{Cut}(U_1), U_1 \triangle U_2).$$

Symmetry If U is a set of vertices and S is a set of edges then

$$\mathcal{E}(S, U) \vdash \mathcal{E}(S \triangle \mathrm{Cut}(U), U).$$

As we claimed above it suffices to consider $\mathcal{E}$-equations instead of the vertex constraints.

Lemma 5.3. *For each vertex $u \in V$ and any set $S \subseteq E$ with $\mathrm{Cut}(u) \cap E = \emptyset$, the family of vertex constraints for u and the set S is implied by the family of $\mathcal{E}$-equations*

$$\{\mathcal{E}(S \cup D, \{u\}) \mid D \subseteq \mathrm{Cut}(u)\}.$$

Proof. For each $u \in V$ and $S \subseteq E$ with $\mathrm{Cut}(u) \cap S = \emptyset$, the vertex constraints for u are the equations of the form

$$0 = \sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|}\hat{p}(S, D)$$

for each even-sized subset $C \subseteq \mathrm{Cut}(u)$. Suppose that $\hat{p}$ satisfies all equations in the family

$$\{\mathcal{E}(S \cup D, \{u\}) \mid D \subseteq \mathrm{Cut}(u)\},$$

and let C be an arbitrary even-sized subset of $\mathrm{Cut}(u)$. Then

$$\sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|}\hat{p}(S, D) = \frac{1}{2} \sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|}\hat{p}(S, D) + (-1)^{|(\mathrm{Cut}(u) \setminus D) \cap C|}\hat{p}(S, \mathrm{Cut}(u) \setminus D)),$$

but since $D \cap C$ and $(\mathrm{Cut}(u) \setminus D) \cap C$ partition C and $|C|$ is even it follows that

$$(-1)^{|(\mathrm{Cut}(u) \setminus D) \cap C|} = (-1)^{|D \cap C|}.$$

Since $\hat{p}$ satisfies $\mathcal{E}(S \cup D, \{u\})$ it follows that $\hat{p}(S, D) = -\hat{p}(S, \mathrm{Cut}(u) \setminus D)$ for all $D \subseteq \mathrm{Cut}(u)$, thus

$$\begin{aligned} &\frac{1}{2} \sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|}\hat{p}(S, D) + (-1)^{|(\mathrm{Cut}(u) \setminus D) \cap C|}\hat{p}(S, \mathrm{Cut}(u) \setminus D)) \\ &= \frac{1}{2} \sum_{D \subseteq \mathrm{Cut}(u)} (-1)^{|D \cap C|} \left(\hat{p}(S, D) - \hat{p}(S, D)\right) = 0. \end{aligned}$$

□

Given G, k, define an index set

$$\mathcal{I}(G,k) = \left\{(S,u) \subseteq 2^E \times V \mid m-k \leq |S| \leq m - |\mathrm{Cut}(u)|, S \cap \mathrm{Cut}(u) = \emptyset\right\}.$$

Let $\mathcal{E}^*(G,k)$ be the closure of the collection of equations

$$\bigcup_{(S,u)\in\mathcal{I}(G,k)} \{\mathcal{E}(S \cup D, \{u\}) \mid D \subseteq \mathrm{Cut}(u)\}$$

under the above deduction rules. We first claim that for every equation $\mathcal{E}(S,U) \in \mathcal{E}^*(G,k)$ we have $|S|, |S \triangle \mathrm{Cut}(U)| \geq m-k$. This is because the deduction rules do not allow the introduction of new $\hat{p}(S)$ coefficients — they only relate coefficients which already appear in the initial set of equations. Since each coefficient in the initial set has size at least $m-k$ the claim follows.

Now, define $\mathcal{G}^*(G,k)$ to be $\{\hat{p}(E) = 1\} \cup \mathcal{E}^*(G,k)$. By the previous lemma, if $\mathcal{G}^*(G,k)$ is satisfiable then so is $\mathcal{G}(G,k)$. Moreover, the equation $\mathcal{E}(E,V)$ is $\hat{p}(E) = 0$, and thus if $\mathcal{E}^*(G,k)$ contains $\mathcal{E}(E,V)$ this directly contradicts the equation $\hat{p}(E) = 1$ in $\mathcal{G}^*(G,k)$. The next lemma shows that this is the *only* obstruction to the system's satisfiability.

Lemma 5.4. *For any graph $G = (V,E)$ with $|V|$ odd and for any integer $0 \leq k \leq |E|$, the system $\mathcal{G}^*(G,k)$ is satisfiable unless it contains $\mathcal{E}(E,V)$.*

Proof. Assume that $\mathcal{G}^*(G,k)$ does not contain $\mathcal{E}(E,V)$, and the solution $\hat{p}$ is constructed by Algorithm 1. Intuitively the algorithm sets values $\hat{p}(S)$ from the "top-down" using the equations in $\mathcal{G}^*(G,k)$: it first assigns $\hat{p}(E) = 1$, and then for each of the smaller sets S it chooses an equation $\mathcal{E} \in \mathcal{G}^*(G,k)$ containing $\hat{p}(S)$ arbitrarily and updates $\hat{p}(S)$ according to $\mathcal{E}$.

Algorithm 1: Defining $\hat{p}(S)$

```
Set $\hat{p}(E) = 1$;
**foreach** $i = 1, 2, \ldots k$ **do**
    Let $\mathcal{S}$ be the collection of all sets $S \subseteq E$ of size $m - i$;
    **while** $\exists S \in \mathcal{S}$ *such that* $\mathcal{E}(S,U) \in \mathcal{G}^*(G,k)$ *with* $|S \triangle \mathrm{Cut}(U)| > |S|$ **do**
        Choose any such equation $\mathcal{E}(S,U) \in \mathcal{G}^*(G,k)$ arbitrarily;
        Set $\hat{p}(S) = (-1)^{|U|}\hat{p}(S \triangle \mathrm{Cut}(U))$;
        Remove $S$ from $\mathcal{S}$;
    **end**
    Set $\hat{p}(S) = 0$ for all remaining $S \in \mathcal{S}$;
**end**
Set $\hat{p}(S) = 0$ for all remaining sets $S$;
**return** $\hat{p}$
```

The correctness of the algorithm follows from the next claim.

Claim. Let $\hat{p}$ be defined by Algorithm 1, and let $S \subseteq E$ be any set with $|S| \geq m-k$. For any $i = 0, 1, \ldots, k$ the following holds. Let $U \subseteq V$ be any set of vertices for which $\mathcal{E}(S,U) \in \mathcal{G}^*(G,k)$ and $|S \triangle U|, |S| \geq m - i$. Then $\mathcal{E}(S,U)$ is satisfied by $\hat{p}$.

Proof of Claim. We prove the claim by induction on i. As a base case we have $i = 0$, and thus $|S| = |S \triangle \mathrm{Cut}(U)| = m$. This implies that $S = S \triangle \mathrm{Cut}(U) = E$, and thus U is either V or $\emptyset$. If

$U = V$ then $\mathcal{E}(E, V) \in \mathcal{G}^*(G, k)$, which is a contradiction. On the other hand, if $U = \emptyset$, then note that $\mathcal{E}(E, \emptyset)$ is just the equation $\hat{p}(E) = \hat{p}(E)$ which is trivially satisfied.

Assume that the claim holds for all $j < i$, and we prove the claim when $j = i$. Let $S \subseteq E$ and $U \subseteq V$ be any sets such that $\mathcal{E}(S, U) \in \mathcal{G}^*(G, k)$ and $|S|, |S \triangle \mathrm{Cut}(U)| \geq m - i$. We assume U is non-empty in order to avoid trivialities, and we prove

$$\hat{p}(S) = (-1)^{|U|} \hat{p}(S \triangle \mathrm{Cut}(U)).$$

If both sets satisfy $|S|, |S \triangle \mathrm{Cut}(U)| > m - i$, then the claim follows from the inductive hypothesis, so assume that at least one of the sets has size $m - i$.

Clearly if the algorithm sets $\hat{p}(S) = \hat{p}(S \triangle \mathrm{Cut}(U)) = 0$ simultaneously (i.e. at the end of the for loop) then the equation is satisfied. So, assume that one of the values was assigned first, and by symmetry assume that the first value assigned is $\hat{p}(S \triangle \mathrm{Cut}(U))$. We break into two cases:

Case 1. $|S| < |S \triangle \mathrm{Cut}(U)|$**.**

Observe that this implies that $|S| = m - i$. Consider the time at which $\hat{p}(S)$ is assigned. Since $|S \triangle \mathrm{Cut}(U)| > m - i$ it is clear that $\hat{p}(S)$ will be set by the end of the while loop in which S is considered. If the algorithm uses the equation $\mathcal{E}(S, U)$ to set $\hat{p}(S) = (-1)^{|U|} \hat{p}(S \triangle \mathrm{Cut}(U))$ then we are done, so suppose otherwise. It follows that there is a set U' such that $\mathcal{E}(S, U') \in \mathcal{G}^*(G, k)$, $|S \triangle \mathrm{Cut}(U')| > |S|$, and the algorithm assigns

$$\hat{p}(S) = (-1)^{|U'|} \hat{p}(S \triangle \mathrm{Cut}(U')).$$

Applying the deduction rule to $\mathcal{E}(S, U)$ and $\mathcal{E}(S, U')$ we get $\mathcal{E}(S \triangle \mathrm{Cut}(U), U \triangle U')$ which is

$$\hat{p}(S \triangle \mathrm{Cut}(U)) = (-1)^{|U \triangle U'|} \hat{p}(S \triangle \mathrm{Cut}(U')).$$

Now, $|S \triangle \mathrm{Cut}(U)|, |S \triangle \mathrm{Cut}(U')| > m - i$, and so by the inductive hypothesis $\hat{p}$ satisfies $\mathcal{E}(S \triangle \mathrm{Cut}(U), U \triangle U')$. Thus

$$\hat{p}(S) = (-1)^{|U'|} \hat{p}(S \triangle \mathrm{Cut}(U')) = (-1)^{|U'| + |U \triangle U'|} \hat{p}(S \triangle \mathrm{Cut}(U)) = (-1)^{|U|} \hat{p}(S \triangle \mathrm{Cut}(U)).$$

Case 2. $|S| = |S \triangle \mathrm{Cut}(U)|$**.**

In this case $|S| = |S \triangle \mathrm{Cut}(U)| = m - i$. By assumption $\hat{p}(S \triangle \mathrm{Cut}(U))$ is set first, and so there must be a set U'' such that $\mathcal{E}(S \triangle \mathrm{Cut}(U), U'')$ is in $\mathcal{G}^*(G, k)$, $|S \triangle \mathrm{Cut}(U) \triangle \mathrm{Cut}(U'')| > m - i$, and the algorithm sets

$$\hat{p}(S \triangle \mathrm{Cut}(U)) = (-1)^{|U''|} \hat{p}(S \triangle \mathrm{Cut}(U) \triangle \mathrm{Cut}(U'')).$$

First, observe that $\hat{p}(S)$ must be defined in the while loop (that is, it can not be assigned to 0 at the end of the iteration of the for loop). For suppose otherwise. Applying the deduction rule to $\mathcal{E}(S \triangle \mathrm{Cut}(U), U'')$ and $\mathcal{E}(S \triangle \mathrm{Cut}(U), U)$ yields the equation

$$\mathcal{E}(S \triangle \mathrm{Cut}(U) \triangle \mathrm{Cut}(U), U \triangle U'') = \mathcal{E}(S, U \triangle U'')$$

which is $\hat{p}(S) = (-1)^{|U \triangle U''|} \hat{p}(S \triangle \mathrm{Cut}(U) \triangle \mathrm{Cut}(U''))$. But $|S \triangle \mathrm{Cut}(U) \triangle \mathrm{Cut}(U'')| > m - i$, and so the equation $\mathcal{E}(S, U \triangle U'')$ will cause $\hat{p}(S)$ to be assigned in the while loop.

So, let us suppose that both S and $S \triangle \mathrm{Cut}(U)$ were set in the while loop. As in the previous case, there is a set U' such that $\mathcal{E}(S, U') \in \mathcal{G}^*(G, k)$, $|S \triangle \mathrm{Cut}(U')| > |S|$, and the algorithm assigns

$$\hat{p}(S) = (-1)^{|U'|} \hat{p}(S \triangle \mathrm{Cut}(U')).$$

Applying the deduction rule to $\mathcal{E}(S,U)$ and $\mathcal{E}(S,U')$ yields $\mathcal{E}(S\triangle\mathrm{Cut}(U),U\triangle U')$, and then applying the deduction rule again to $\mathcal{E}(S\triangle\mathrm{Cut}(U),U\triangle U')$ and $\mathcal{E}(S\triangle\mathrm{Cut}(U),U'')$ yields

$$\mathcal{E}(S\triangle\mathrm{Cut}(U)\triangle\mathrm{Cut}(U)\triangle\mathrm{Cut}(U'),U\triangle U'\triangle U'')=\mathcal{E}(S\triangle\mathrm{Cut}(U'),U\triangle U'\triangle U'')$$

which is the equation

$$\hat{p}(S\triangle\mathrm{Cut}(U'))=(-1)^{|U\triangle U'\triangle U''|}\hat{p}(S\triangle\mathrm{Cut}(U)\triangle\mathrm{Cut}(U'')).$$

Both $S\triangle\mathrm{Cut}(U')$ and $S\triangle\mathrm{Cut}(U)\triangle\mathrm{Cut}(U'')$ have greater than $m-i$ elements since $\hat{p}(S)$ and $\hat{p}(S\triangle\mathrm{Cut}(U))$ were assigned in the while loop, and so by the inductive hypothesis the above equation holds. Therefore

$$\begin{aligned}\hat{p}(S)&=(-1)^{|U'|}\hat{p}(S\triangle\mathrm{Cut}(U'))\\&=(-1)^{|U'|+|U\triangle U'\triangle U''|}\hat{p}(S\triangle\mathrm{Cut}(U)\triangle\mathrm{Cut}(U''))\\&=(-1)^{|U'|+|U\triangle U'\triangle U''|+|U''|}\hat{p}(S\triangle\mathrm{Cut}(U))\\&=(-1)^{|U|}\hat{p}(S\triangle\mathrm{Cut}(U)).\end{aligned}$$

□

Applying the Claim when $i=k$, we get that the output $\hat{p}$ of the algorithm satisfies every equation $\mathcal{E}(S,U)$ in $\mathcal{G}^*(G,k)$, and it clearly satisfies $\hat{p}(E)=1$. The proof is complete. □

We now show that if the system $\mathcal{E}^*(G,k)$ contains $\mathcal{E}(U,V)$, then the Tseitin formula $\mathsf{Tseitin}_G$ on G has a width-$2k$ Gaussian proof. Let us recall the Gaussian refutation system. Each line of a Gaussian refutation of $\mathsf{Tseitin}_G$ is a linear equation over $\mathbf{F}_2$ of the form

$$L_U\equiv\bigoplus_{e\in\mathrm{Cut}(U)}x_e=|U|\bmod 2.$$

The axioms are of the form $L_{\{u\}}$, where u is any node in V, and the lines are equipped with a single derivation rule of the form

$$L_{U_1},L_{U_2}\vdash L_{U_1\triangle U_2}.$$

A Gaussian refutation of the Tseitin formula on G is a derivation of $0=1$ from axiom lines $\{L_{\{u\}}\mid u\in V\}$. The *width* of a line L_U in a Gaussian refutation is $|\mathrm{Cut}(U)|$, and the width of a Gaussian refutation is the maximum width of all of the lines in the refutation. Once we have this result in hand, we are finished thanks to the following proposition.

Proposition 5.5. *Let $G=(V,E)$ be a d-regular graph on an odd number of vertices and m edges. Any Gaussian refutation of* $\mathsf{Tseitin}_G$ *requires width at least $\varepsilon(G)2m/3d$.*

Proof. Fix the first line L_U in any Gaussian refutation that is derived from between $|V|/3$ and $2|V|/3$ initial lines. Since $|V|=2m/d$ we have $2m/3d\le|U|\le 4m/3d$. Every edge in $\mathrm{Cut}(U)$ occurs in L_U, and thus the width of L_U is $\varepsilon(G)|U|\ge\varepsilon(G)2m/3d$. □

Lemma 5.6. *Let $G=(V,E)$ be a graph with $|V|$ odd, and let $0\le k\le|E|$. If $\mathcal{E}^*(G,k)$ contains $\mathcal{E}(U,V)$ then the Tseitin formula on G has Gaussian width at most $2k$.*

Proof. Let $\mathcal{P}$ be a proof of $\mathcal{E}(U,V)$ in the $\mathcal{E}$-calculus. We turn $\mathcal{P}$ into a Gaussian refutation as follows. The axioms $\mathcal{E}(S,\{u\})$ are turned into axiom lines $L_{\{u\}}$. Inductively, if we apply the deduction rule $\mathcal{E}(S,U_1),\mathcal{E}(S,U_2)\vdash\mathcal{E}(S\triangle U_1,U_1\triangle U_2)$ then we derive the line $L_{U_1\triangle U_2}$ from the lines L_{U_1},L_{U_2}. The symmetry rule is idempotent (it does not affect the Gaussian refutation). It should be clear that the final line in this inductive construction is $L_V \equiv 0 = 1$, since $|V|$ is odd.

Let $\ell = |\mathrm{Cut}(U)|$ be the width of the widest line L_U in the Gaussian refutation produced from $\mathcal{P}$, and consider the corresponding line $\mathcal{E}(S, U)$ which produced the line L_U. This line is equivalent to the equation

$$\hat{p}(S) = (-1)^{|U|}\hat{p}(S \triangle \mathrm{Cut}(U)).$$

We claim that $\min\{|S|, |S \triangle \mathrm{Cut}(U)|\} \leq m - \ell/2$, and thus $k \geq \ell/2$, which proves the theorem. To see this, first suppose that $|S \cap \mathrm{Cut}(U)| \leq \ell/2$. Then

$$|\mathrm{Cut}(U) \setminus S| \geq \ell - |S \cap \mathrm{Cut}(U)| \geq \ell/2,$$

and thus

$$|S| + |\mathrm{Cut}(U) \setminus S| = |S \cup \mathrm{Cut}(U)| \leq m$$

so $|S| \leq m - \ell/2$. On the other hand, suppose $|S \cap \mathrm{Cut}(U)| > \ell/2$. Then

$$|S \triangle \mathrm{Cut}(U)| + |S \cap \mathrm{Cut}(U)| = |S \cup \mathrm{Cut}(U)| \leq m,$$

and thus

$$|S \triangle \mathrm{Cut}(U)| + \ell/2 \leq m,$$

or equivalently $|S \triangle \mathrm{Cut}(U)| \leq m - \ell/2$. □

Proof of Theorem 5.1. By Lemma 5.4 the system $\mathcal{G}^*(G, k)$ is satisfiable unless it contains $\mathcal{E}(E, V)$. By the previous lemma and proposition, if $\mathcal{G}^*(G, k)$ contains $\mathcal{E}(E, V)$ then $k \geq \varepsilon(G)m/3d$; thus, $\mathcal{G}^*(G, \varepsilon(G)m/3d - 1)$ is satisfiable. Lemma 5.2 therefore implies that $\mathsf{Search}(\mathsf{Tseitin}_G)$ has algebraic gap complexity at least $\varepsilon(G)m/3d$. □

References

[1] László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999.

[2] Amos Beimel, Anna Gál, and Mike Paterson. Lower bounds for monotone span programs. *Computational Complexity*, 6(1):29–45, 1997.

[3] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. *J. Symb. Log.*, 62(3):708–728, 1997.

[4] Allan Borodin. On relating time and space to size and depth. *SIAM J. Comput.*, 6(4):733–744, 1977.

[5] Siu Man Chan and Aaron Potechin. Tight bounds for monotone switching networks via fourier analysis. *Theory of Computing*, 10:389–419, 2014.

[6] Arkadev Chattopadhyay and Anil Ada. Multiparty communication complexity of disjointness. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(002), 2008.

[7] Stephen A. Cook, Toniann Pitassi, Robert Robere, and Benjamin Rossman. Exponential lower bounds for monotone span programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:64, 2016.

[8] László Csirmaz. The dealer's random bits in perfect secret sharing schemes. *Studia Sci. Math. Hungary*, 32(3-4):429–437, 1996.

[9] Yuval Filmus, Pavel Hrubes, and Massimo Lauria. Semantic versus syntactic cutting planes. In Nicolas Ollinger and Heribert Vollmer, editors, *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17-20, 2016, Orléans, France*, volume 47 of *LIPIcs*, pages 35:1–35:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[10] Anna Gál. A characterization of span program size and improved lower bounds for monotone span programs. *Computational Complexity*, 10(4):277–296, 2001.

[11] Anna Gál and Pavel Pudlák. A note on monotone complexity and the rank of matrices. *Inf. Process. Lett.*, 87(6):321–326, 2003.

[12] Mika Göös. Lower bounds for clique vs. independent set. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1066–1076, 2015.

[13] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 847–856, 2014.

[14] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1077–1088. IEEE Computer Society, 2015.

[15] Danny Harnik and Ran Raz. Higher lower bounds on monotone size. In F. Frances Yao and Eugene M. Luks, editors, *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 378–387. ACM, 2000.

[16] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 233–248, 2012.

[17] Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

[18] Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discrete Math.*, 3(2):255–265, 1990.

[19] Mauricio Karchmer and Avi Wigderson. On span programs. In *Proceedings of the Eigth Annual Structure in Complexity Theory Conference, San Diego, CA, USA, May 18-21, 1993*, pages 102–111, 1993.

[20] James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 567–576, 2015.

[21] Troy Lee and Adi Shraibman. Disjointness is hard in the multiparty number-on-the-forehead model. *Computational Complexity*, 18(2):309–336, 2009.

[22] Adam W. Marcus, Daniel A. Spielman, and Nikhil Srivastava. Interlacing families IV: bipartite ramanujan graphs of all sizes. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1358–1377. IEEE Computer Society, 2015.

[23] Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

[24] Aaron Potechin. Bounds on monotone switching networks for directed connectivity. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 553–562, 2010.

[25] Pavel Pudlák and Jirí Sgall. Algebraic models of computation and interpolation for algebraic proof systems. In *Proof Complexity and Feasible Arithmetics, Proceedings of a DIMACS Workshop, New Brunswick, New Jersey, USA, April 21-24, 1996*, pages 279–296, 1996.

[26] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999.

[27] Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.

[28] Alexander A. Razborov. Lower bounds for the monotone complexity of some boolean functions. *Soviet Math. Dokl.*, 31:354–357, 1985.

[29] Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, 1990.

[30] Benjamin Rossman. Correlation bounds against monotone nc^1. In David Zuckerman, editor, *30th Conference on Computational Complexity, CCC 2015, June 17-19, 2015, Portland, Oregon, USA*, volume 33 of *LIPIcs*, pages 392–411. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[31] Claude Shannon et al. The synthesis of two-terminal switching circuits. *Bell System Technical Journal*, 28(1):59–98, 1949.

[32] Alexander A. Sherstov. Communication lower bounds using dual polynomials. *Bulletin of the EATCS*, 95:59–93, 2008.

[33] Alexander A. Sherstov. The pattern matrix method. *SIAM J. Comput.*, 40(6):1969–2000, 2011.