Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 186 (2019)

Lifting with Simple Gadgets and Applications to Circuit and Proof Complexity

Susanna F. de Rezende
KTH Royal Institute of Technology

Or Meir
University of Haifa

Jakob Nordström
University of Copenhagen

Toniann Pitassi
University of Toronto
Institute for Advanced Study

Robert Robere
DIMACS
Institute for Advanced Study

Marc Vinyals
Technion

Abstract

We significantly strengthen and generalize the theorem lifting Nullstellensatz degree to monotone span program size by Pitassi and Robere (2018) so that it works for any gadget with high enough rank, in particular, for useful gadgets such as *equality* and *greater-than*. We apply our generalized theorem to solve two open problems:

- We present the first result that demonstrates a separation in proof power for cutting planes with unbounded versus polynomially bounded coefficients. Specifically, we exhibit CNF formulas that can be refuted in quadratic length and constant line space in cutting planes with unbounded coefficients, but for which there are no refutations in subexponential length and subpolynomial line space if coefficients are restricted to be of polynomial magnitude.
- We give the first explicit separation between monotone Boolean formulas and monotone real formulas. Specifically, we give an explicit family of functions that can be computed with monotone real formulas of nearly linear size but require monotone Boolean formulas of exponential size. Previously only a non-explicit separation was known.

An important technical ingredient, which may be of independent interest, is that we show that the Nullstellensatz degree of refuting the pebbling formula over a DAG G over any field coincides exactly with the reversible pebbling price of G. In particular, this implies that the standard decision tree complexity and the parity decision tree complexity of the corresponding falsified clause search problem are equal.

ISSN 1433-8092

1 Introduction

Lifting theorems in complexity theory are a method of transferring lower bounds in a weak computational model into lower bounds for a more powerful computational model, via function composition. There has been an explosion of lifting theorems in the last ten years, essentially reducing communication lower bounds to query complexity lower bounds.

Early papers that establish lifting theorems include Raz and McKenzie's separation of the monotone NC hierarchy [RM99] (by lifting decision tree complexity to deterministic communication complexity), and Sherstov's pattern matrix method [She11] which lifts (approximate) polynomial degree to (approximate) matrix rank. Recent work has established query-to-communication lifting theorems in a variety of models, leading to the resolution of many longstanding open problems in many areas of computer science. Some examples include the resolution of open questions in communication complexity [GPW15, GLM+15, GKPW17, GJPW17, GPW18], monotone complexity [RPRC16, PR17, PR18], proof complexity [HN12, GP18, dRNV16, GGKS18], extension complexity of linear and semidefinite programs [KMR17, GJW18, LRS15], data structures [CKLM18] and finite model theory [BN16].

Lifting theorems have the following form: given functions $f\colon \{0,1\}^n \to \{0,1\}$ (the "outer function") and $g\colon \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ (the "gadget"), a lower bound for f in a weak computational model implies a lower bound on $f \circ g^n$ in a stronger computational model. The most desirable lifting theorems are the most general ones. First, it should hold for *any* outer function, and ideally f should be allowed to be a partial function or a relation (i.e., a search problem). Indeed, nearly all of the applications mentioned above require lifting where the outer function is a relation or a partial function. Secondly, it is often desirable that the gadget is as small as possible. The most general lifting theorems established so far, for example lifting theorems for deterministic and randomized communication complexity, require at least logarithmically-sized gadgets; if these theorems could be improved generically to hold for constant-sized gadgets then many of the current theorems would be vastly improved. Some notable examples where constant-sized gadgets are possible include Sherstov's degree-to-rank lifting [She11], critical block-sensitivity lifting [GP18, HN12], and lifting for monotone span programs [PR17, PR18, Rob18].

1.1 A New Lifting Theorem

In this paper, we generalize a lifting theorem of Pitassi and Robere [PR18] to use any gadget that has nontrivial rank. This theorem takes a search problem associated with an unsatisfiable CNF, and lifts a lower bound on the Nullstellensatz degree of the CNF to a lower bound on a related communication problem.

More specifically, let $\mathcal{C}$ be an unsatisfiable k-CNF formula. The search problem associated with $\mathcal{C}$, $\mathsf{Search}(\mathcal{C})$, takes as input an assignment to the underlying variables, and outputs a clause that is falsified by the assignments. [PR18] prove that for any unsatisfiable $\mathcal{C}$, and for a sufficiently rich gadget g, deterministic communication complexity lower bounds for the composed search problem $\mathsf{Search}(\mathcal{C}) \circ g^n$ follow from Nullstellensatz degree lower bounds for $\mathcal{C}$.[1] We significantly improve this lifting theorem so that it holds for *any* gadget of large enough rank.

Theorem 1.1. *Let $\mathcal{C}$ be a CNF over n variables, let $\mathbb{F}$ be any field, and let g be any gadget of rank at least r. Then the deterministic communication complexity of $\mathsf{Search}(\mathcal{C} \circ g^n)$ is at least $\mathsf{NS}_{\mathbb{F}}(\mathcal{C})$, the Nullstellensatz degree of $\mathcal{C}$, as long as $r \geq cn/\mathsf{NS}_{\mathbb{F}}(\mathcal{C})$ for some large enough constant c.*

An important special case of our generalized theorem is when the gadget g is the equality function. In this work, we apply our theorem to resolve two open problems in proof complexity and circuit complexity. Both solutions depend crucially on the ability to use the equality gadget.

We note that lifting with the equality gadget has recently been the focus of another paper. Loff and Mukhopadhyay [LM19] observed that a lifting theorem for *total functions* with the equality gadget can

[1] In fact the result is quite a bit stronger—it applies to Razborov's rank measure [Raz90], which is a strict strengthening of deterministic communication complexity.

be proven using a rank argument. Surprisingly, they also observed that it is *not* possible to lift query complexity to communication complexity for arbitrary relations! Concretely, [LM19] give an example of a relation with linear query complexity but whose composition with equality has only polylogarithmic communication complexity. Nonetheless, they are able to prove a lifting theorem for general relations using the equality gadget by replacing standard query complexity with a stronger complexity measure (namely, the 0-query complexity of the relation).

Unfortunately, we cannot use either of the lifting theorems of [LM19] for our applications. Specifically, in our applications we lift a search problem (and therefore cannot use their result for total functions), and this search problem has small 0-query complexity (and therefore we cannot use their lifting theorem for general relations). Indeed, this shows that our lifting theorem is incomparable to the results of [LM19], even when specialized to the equality gadget. We note that our theorem, too, bypasses the impossibility result of [LM19] by using a stronger complexity measure, which in our case is the Nullstellensatz degree.

1.2 A Separation in Proof Complexity

The main application of our lifting theorem is the first separation in proof complexity between cutting planes proofs with high-weight versus low-weight coefficients. The cutting planes proof-system is a proof system that can be used to refute an unsatisfiable CNF by translating it into a system of integer inequalities and showing that this system has no integer solution. The latter is achieved by a sequence of steps that derive new integer inequalities from old ones, until we derive the inequality $0 \geq 1$ (which clearly has no solution). The efficiency of such a refutation is measured by its length (i.e., the number of steps) and its space (i.e., the maximal number of inequalities that have to be stored simultaneously during the derivation).

The standard variant of the cutting planes proof system, commonly denoted by CP, allows the inequalities to use coefficients of arbitrary size. However, it is also interesting to consider the variant in which the coefficients are polynomially bounded, which is commonly denoted by CP*. This gives rise to the natural question of the relative power of CP vs. CP*: are they polynomially equivalent or is there a super-polynomial length separation? This question appeared in [BC96] and remains stubbornly open to date. In this work we finally make progress by exhibiting a setting in which unbounded coefficients afford an exponential increase in proof power.

Theorem 1.2. *There is a family of CNF formulas of size N that have cutting planes refutations of length $\tilde{O}(N^2)$ and space $O(1)$, but for which any refutation in length L and space s with polynomially bounded coefficients must satisfy $s \log L = \tilde{\Omega}(N)$.*

Our result is the first result in proof complexity demonstrating any situation where high-weight coefficients are more powerful than low-weight coefficients. In comparison, for computing Boolean functions, the relative power of high-weight and low-weight linear threshold functions has been understood for a long time. The greater-than function can be computed by high-weight threshold functions, but not by low-weight threshold functions, and weights of length polynomial in n suffice [Mur71] for Boolean functions. For higher depth threshold formulas, it is known that depth-d threshold formulas of high-weight can efficiently be computed by depth-$(d+1)$ threshold formulas of low-weight [GHR92].

In contrast to our near-complete knowledge of high versus low weights for functions, almost nothing is known about the relative power of high versus low weights in the context of proof complexity. Buss and Clote [BC96], building on work by Cook, Coullard, and Turán [CCT87], proved an analog of Muroga's result for cutting planes, showing that weights of length polynomial in the length of the proof suffice. Quite remarkably, this result is not known to hold for other linear threshold proof systems: there is *no* nontrivial upper bound on the weights for more general linear threshold propositional proof systems (such as *stabbing planes* [BFI$^+$18], and Krajíček's threshold logic proof system [Kra95] where one can additionally branch on linear threshold formulas). Prior to our result, there was no separation between high and low weights, for any linear threshold proof system.

1.3 A Separation in Circuit Complexity

A second application of our lifting theorem relates to monotone real circuits, which were introduced by Pudlák [Pud97]. A monotone real circuit is a generalization of monotone Boolean circuits where each gate is allowed to compute any non-decreasing real function of its inputs, but the inputs and output of the circuit are Boolean. A formula is a tree-like circuit, that is, every gate has fan-out one. The first (exponential) lower bound for monotone real circuits was proven already in [Pud97] by extending the lower bound for computing the clique-colouring function with monotone Boolean circuits [Raz85, AB87]. This lower bound, together with a generalization of the interpolation technique [Kra97] which applied only to CP*, was used by Pudlák to obtain the first exponential lower bounds for CP.

Shortly after monotone real circuits were introduced, there was an interest in understanding the power of monotone real computation in comparison to monotone Boolean computation. By extending techniques in [RM99], Bonet et al. prove that there are functions with polynomial size monotone Boolean circuits that require monotone real formulas of exponential size [Joh98, BEGJ00]. This illustrates the power of DAG-like computations in comparison to tree-like. In the other direction, we would like to know whether monotone real circuits are exponentially stronger than monotone Boolean circuits. Rosenbloom [Ros97] presented an elegant, simple proof that monotone real formulas are exponentially stronger than (even non-monotone) Boolean circuits, since slice functions can be computed by linear-size monotone real formulas, whereas by a counting argument we know that most slice functions require exponential size Boolean circuits.

The question of finding explicit functions that demonstrate that monotone real circuits are stronger than general Boolean circuits is much more challenging since it involves proving explicit lower bounds for Boolean circuits—a task that seems currently completely out of reach. A more tractable problem is that of finding explicit functions showing that monotone real circuits or formulas are stronger than *monotone* Boolean circuits or formulas, but prior to this work, no such separation was known either. We provide an explicit separation for *monotone formulas*, that is, we provide a family of explicit functions that can be computed with monotone real formulas of near-linear size but require exponential monotone Boolean formulas. This is the first explicit example that illustrates the strength of monotone real computation.

Theorem 1.3. *There is an explicit family of functions f_n over $O(n \operatorname{polylog} n)$ variables that can be computed by monotone real formulas of size $O(n \operatorname{polylog} n)$ but for which every monotone Boolean formula requires size $2^{\Omega(n/\log n)}$.*

Another motivation for studying lifting theorems with simple gadgets, and in particular the equality gadget, are connections with proving *non-monotone* formula size lower bounds. As noted earlier, lifting theorems have been extremely successful in proving monotone circuit lower bounds, and it has also been shown to be useful in some computational settings that are only “partially” monotone; notably monotone span programs [RPRC16, PR17, PR18] and extended formulations [GJW18, KMR17].

This raises the question of to what extent lifting techniques can help prove *non-monotone* lower bounds. The beautiful work by Karchmer, Raz and Wigderson [KRW95] initiated such an approach for separating P from NC^1—this opened up a line of research popularly known as the *KRW conjecture*. Intriguingly, steps towards resolving the KRW conjecture are closely connected to proving lifting theorems for the equality gadget. The first major progress was made in [EIRS01] where lower bounds for the universal relation game are proven, which is an important special case of the KRW conjecture. Their result was recently improved in several papers [GMWW17, HW93, KM18], and Dinur and Meir [DM18] gave a new top-down proof of the state-of-the-art $\Omega(n^3)$ formula-size lower bounds via the KRW approach.

The connection to lifting using the equality gadget is obtained by observing that the KRW conjecture involves communication problems in which Alice and Bob are looking for a bit on which they differ—this is exactly an *equality* problem. Close examination of the results in [EIRS01, HW93] show that they are equivalent to proving lower bounds for the search problem associated with the pebbling formula when lifted with a 1-bit equality gadget on a *particular* graph [Pit16]. Our proof of Theorem 4.1 actually establishes near-optimal lower bounds on the communication complexity of the pebbling formula lifted with equality for *any* graph, but where the size of the equality is not 1. Thus if our main theorem could be improved with one-bit equality gadgets this would imply the results of [EIRS01, HW93] as a direct corollary and with significantly better parameters.

1.4 Overview of Techniques

We conclude this section by giving a brief overview of our techniques, also trying to convey some of the simplicity of the proofs which we believe is an extra virtue of these results.

Lifting theorem In order to prove their lifting theorem, Pitassi and Robere [PR18] defined a notion of a "good" gadget. They then showed that if we compose a polynomial p with a good gadget g, the rank of the resulting matrix $p \circ g^n$ is determined *exactly* by the non-zero coefficients of p and the rank of g. Their lifting theorem follows by using this correspondence to obtain bounds on the ranks of certain matrices, which in turn yield the required communication complexity lower bound.

In this work, we observe that every gadget g can be turned into a good gadget using a simple transformation. This observation allows us to get an approximate bound on the rank of $p \circ g^n$ for any g with nontrivial rank. While the correspondence we get in this way is only an approximation and not an exact correspondence as in [PR18], it turns out that this approximation is sufficient to prove the required lower bounds. We thus get a lifting theorem that works for every gadget g with sufficiently large rank.

Cutting planes separation The crux of our separation between CP and CP* is the following observation: CP can encode a conjunction of linear equalities with a single equality, by using exponentially large coefficients. This allows CP refutations to obtain a significant saving in space when working with *linear equalities*. This saving is not available to CP*, and this difference between the proof systems allows the separation.

In order to exploit this observation, one of our main innovations is to concoct the separating formula. To do this, we must come up with a candidate formula that can only be refuted by reasoning about a large conjunction of linear equalities, to show that cutting planes (CP) can efficiently refute it, and to show that low-weight cutting planes (CP*) cannot.

To find such a candidate formula family we resort to *pebbling formulas* which have played a major role in many proof complexity trade-off results. Interestingly, pebbling formulas have short resolution proofs that reason in terms of large conjunctions of literals. When we lift such formulas with the equality gadget this proof can be simulated in cutting planes by using the large coefficients to encode many equalities with a single equality. This yields cutting planes refutation of any pebbling formula in quadratic length and constant space.

On the other hand we prove our time-space lower bound showing that any CP* refutation requires large length or large space for the same formulas. To prove this lower bound, the first step is to instantiate the connection in [HN12] linking time/space bounds for many proof systems to communication complexity lower bounds for lifted search problems. This connection means that we can obtain the desired CP*-lower bounds for our formulas $\mathrm{Peb}_G \circ \mathrm{EQ}^n$ by proving communication complexity lower bounds for the corresponding lifted search problem $\mathsf{Search}(\mathrm{Peb}_G) \circ \mathrm{EQ}^n$.

In order to prove the latter communication lower bounds, we prove lower bounds on the Nullstellensatz degree of $\mathsf{Search}(\mathrm{Peb}_G)$, and then invoke our new lifting theorem to translate them into communication lower bounds for $\mathsf{Search}(\mathrm{Peb}_G) \circ \mathrm{EQ}^n$. To show the Nullstellensatz lower bounds, we prove the following lemma, which establishes an equivalence between Nullstellsatz degree and the *reversible* pebbling price, and may be interesting in its own right. (We remark that connections between Nullstellensatz degree and pebbling were previously shown in [BCIP02]; however their result was not tight.)

Lemma 1.4. *For any field $\mathbb{F}$ and any directed acyclic graph G the Nullstellensatz degree of Peb_G is equal to the reversible pebbling price of G.*

We remark that due to known lower and upper bounds in query and proof complexity, this lemma immediately implies that Nullstellensatz degree coincides for (deterministic) decision tree and parity decision tree complexity. We record this here as a corollary, as it may be of independent interest, and provide its proof in Appendix C.

Corollary 1.5. *For any field $\mathbb{F}$ and any directed acyclic graph G, the Nullstellensatz degree over $\mathbb{F}$ of Peb_G, the decision tree depth of $\mathsf{Search}(\mathrm{Peb}_G)$, and the parity decision tree depth of $\mathsf{Search}(\mathrm{Peb}_G)$ coincide and are equal to the reversible pebbling price of G.*

Using the above equivalence, we obtain near-linear Nullstellensatz degree refutations for a family of graphs with maximal pebbling price, which completes our time/space lower bound for CP*. However, in order to separate CP and CP* we require a very specific gadget and lifting theorem. Specifically, the gadget should be strong enough, so that lifting holds for deterministic communication complexity (which can efficiently simulate small time/space CP* proofs), but on the other hand also weak enough, so that lifting does not hold for stronger communication models (randomized, real) that can efficiently compute high-weight inequalities. The reason that we are focusing on the equality gadget is that it hits this sweet spot—it requires large deterministic communication complexity, yet has short randomized protocols, and furthermore equalities can be represented with a single pair of inequalities.

Separation for monotone formulas As was the case for the separation between CP and CP*, to obtain a separation between monotone Boolean formulas and monotone real formulas we must find a function that has just the right level of hardness.

To obtain a size lower bound for monotone Boolean formulas we invoke the characterization of formula depth by communication complexity of the Karchmer–Wigderson game [KW90]. By choosing a function that has the same Karchmer–Wigderson game as the search problem of a lifted pebbling formula, we get a depth lower bound for monotone Boolean formulas from the communication lower bound of the search problem. Note that since monotone Boolean formulas can be balanced, a depth lower bound implies a size lower bound.

In the other direction, we would like to show that these functions are easy for real computation. Analogously to the Karchmer–Wigderson relation, it was shown in [HP18] that there is a correspondence between real DAG-like communication protocols (as defined in [Kra98]) and monotone real circuits. Using this relation, a small monotone real *circuit* can be extracted from a short CP proof of the lifted pebbling formula. However, we would like to establish a monotone real *formula* upper bound. One way to achieve this is by finding small tree-like CP refutations of lifted pebbling formulas. The problem is that for many gadgets lifted pebbling formulas require exponentially long tree-like proofs. Nevertheless, for pebbling formulas lifted with the equality gadget we are able to exhibit a short *semantic* tree-like CP refutation, which via real communication yields a small monotone real formula.

1.5 Organization of This Paper

Section 2 contains formal definitions of concepts discussed above and some useful facts. Our main lifting theorem is proven in Section 3. Section 4 is devoted to proving our separation between high-weight and low-weight cutting planes. In Section 5 we prove the separation between monotone real and Boolean formulas. We conclude in Section 6 with some open problems.

2 Preliminaries

In this section we review some background material from communication complexity and proof complexity.

2.1 Communication Complexity and Lifted Search Problems

Given a function $g : \mathcal{X} \times \mathcal{Y} \to \mathcal{I}$, we denote by $g^n : \mathcal{X}^n \times \mathcal{Y}^n \to \mathcal{I}^n$ the function that takes as input n independent instances of g and applies g to each of them separately. A *total search problem* is a relation $\mathcal{S} \subseteq \mathcal{I} \times \mathcal{O}$ such that for all $z \in \mathcal{I}$ there is an $o \in \mathcal{O}$ such that $(z, o) \in \mathcal{S}$. Intuitively, S represents the computational task in which we are given an input $z \in \mathcal{I}$ and would like to find an output $o \in \mathcal{O}$ that satisfies $(z, o) \in \mathcal{S}$.

An important example of a search problem, which has proved to be very useful for proof complexity results, comes from unsatisfiable k-CNF formulas. Given a k-CNF formula $\mathcal{C}$ over variables $z_1, \ldots, z_n$, the search problem $\mathsf{Search}(\mathcal{C}) \subseteq \{0,1\}^n \times \mathcal{C}$ takes as input an assignment $z \in \{0,1\}^n$ and outputs a clause $C \in \mathcal{C}$ that is falsified by z.

Given a search problem $\mathcal{S} \subseteq \mathcal{I}^n \times \mathcal{O}$ with a product input domain and a function $g : \mathcal{X} \times \mathcal{Y} \to \mathcal{I}$, we define the *composition* $\mathcal{S} \circ g^n \subseteq \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{O}$ in the natural way: $(x, y, o) \in \mathcal{S} \circ g$ if and only if $(g^n(x, y), o) \in \mathcal{S}$. We remark that this composition notation extends naturally to functions: for instance, if $f : \mathcal{I}^n \to \mathbb{F}$ is a function taking values in some field $\mathbb{F}$, for example, then the composition $f \circ g^n$ is a $\mathcal{X}^n \times \mathcal{Y}^n$ matrix over $\mathbb{F}$. Second, we remark that we will sometimes write $S \circ g$ instead of $S \circ g^n$ if n is clear from context.

A *communication search problem* is a search problem with a bipartite input domain $\mathcal{I} = \mathcal{A} \times \mathcal{B}$. A communication protocol for a search problem $\mathcal{S} \subseteq \mathcal{A} \times \mathcal{B} \times \mathcal{O}$ is a strategy for a collaborative game where two players Alice and Bob hold $x \in \mathcal{A}, y \in \mathcal{B}$, respectively, and wish to output an $o \in \mathcal{O}$ such that $((x, y), o) \in \mathcal{S}$ while communicating as few bits as possible. Messages are sent sequentially until one player announces the answer and only depend on the input of one player and past messages. The cost of a protocol is the maximum number of bits sent over all inputs, and the communication complexity of a search problem, which we denote by $\mathsf{P}^{\mathsf{cc}}(\mathcal{S})$, is the minimum cost over all protocols that solve $\mathcal{S}$. For more details on communication complexity, see, for instance, [KN97].

Given a CNF formula $\mathcal{C}$ on n variables $z_1, z_2, \ldots, z_n$ and a Boolean function $g\colon \{0,1\}^q \times \{0,1\}^q \to \{0,1\}$, we define a *lifted formula* $\mathcal{C} \circ g^n$ as follows. For each variable z_i of $\mathcal{C}$, we have $2q$ new variables $x_{i,1}, \ldots, x_{i,q}, y_{i,1}, \ldots, y_{i,q}$. For each clause $C \in \mathcal{C}$ we replace each literal z_i or $\neg z_i$ in C by a CNF encoding of either $g(x_{i,1}, \ldots, x_{i,q}, y_{i,1}, \ldots, y_{i,q})$ or $\neg g(x_{i,1}, \ldots, x_{i,q}, y_{i,1}, \ldots, y_{i,q})$ according to the sign of the literal. We then expand the resulting expression into a CNF, which we denote by $C \circ g$, using de Morgan's rules. The substituted formula is $\mathcal{C} \circ g = \bigcup_{C \in \mathcal{C}} C \circ g$.

For the sake of an example, consider the clause $u \vee \overline{v}$, and we will substitute with the equality gadget on two bits. Formally, we replace u with $x_{u,1}x_{u,2} = y_{u,1}y_{u,2}$ and v with $x_{v,1}x_{v,2} = y_{v,1}y_{v,2}$. We can encode a two-bit equality as the CNF formula

$$(x_1x_2 = y_1y_2) \equiv (\overline{x}_1 \vee y_1) \wedge (x_1 \vee \overline{y}_1) \wedge (\overline{x}_2 \vee y_2) \wedge (x_2 \vee \overline{y}_2),$$

and a two-bit disequality as the CNF formula

$$(x_1x_2 \neq y_1y_2) \equiv (\overline{x}_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee \overline{y}_2) \wedge (\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee y_2) \wedge (x_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2) \wedge (x_1 \vee x_2 \vee y_1 \vee y_2).$$

So, in the clause $u \vee \overline{v}$, we would substitute u for the CNF encoding of $x_{u,1}x_{u,2} = y_{u,1}y_{u,2}$ and $\overline{v}$ with the CNF encoding of $x_{v,1}x_{v,2} \neq y_{v,1}y_{v,2}$; finally, we would convert the new formula to a CNF by distributing the top $\vee$ over the $\wedge$s from the new CNF encodings.

While $\mathsf{Search}(\mathcal{C}) \circ g^n$ is not the same problem as $\mathsf{Search}(\mathcal{C} \circ g^n)$, we can reduce the former to the latter. Specifically, suppose we are given a protocol Π for $\mathsf{Search}(\mathcal{C} \circ g^n)$. Consider the following protocol Π' for $\mathsf{Search}(\mathcal{C}) \circ g^n$: Given an input (x, y), the protocol Π' interprets (x, y) as an input to Π. Now, assume that Π' outputs on (x, y) a clause D of $\mathcal{C} \circ g^n$, which was obtained from a clause C of $\mathcal{C}$. Then, the clause C is a valid $\mathsf{Search}(\mathcal{C})$ on (x, y), so Π' outputs it. Let us record this observation.

Observation 2.1. $\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C} \circ g)) \geq \mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C}) \circ g)$ *for any unsatisfiable CNF $\mathcal{C}$ and any Boolean gadget g.*

2.2 Nullstellensatz

As a proof system, Nullstellensatz allows verifying that a set of polynomials does not have a common root, and it can also be used to refute CNF formulas by converting them into polynomials. It plays an important role in our lower bounds.

Let $\mathbb{F}$ be a field, and let $\mathcal{P} = \{p_1 = 0, p_2 = 0, \ldots, p_m = 0\}$ be an unsatisfiable system of polynomial equations in $\mathbb{F}[z_1, z_2, \ldots, z_n]$. A *Nullstellensatz refutation* of $\mathcal{P}$ is a sequence of polynomials $q_1, q_2, \ldots, q_m \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ such that $\sum_{i=1}^m p_i q_i = 1$ where the equality is syntactic. The *degree* of the refutation is $\max_i \deg(p_i q_i)$; the *Nullstellensatz degree* of $\mathcal{P}$, denoted $\mathsf{NS}_{\mathbb{F}}(\mathcal{P})$, is the minimum degree of any Nullstellensatz refutation of $\mathcal{P}$.

Let $\mathcal{C} = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be an unsatisfiable CNF formula over Boolean variables $z_1, z_2, \ldots, z_n$. We introduce a standard encoding of each clause C_i as a polynomial equation. If C is a clause then let C^+ denote the set of variables occurring positively in C and C^- denote the set of variables occurring negatively in C; with this notation we can write $C = \bigvee_{z \in C^+} z \vee \bigvee_{z \in C^-} \overline{z}$. From C define the polynomial

$$\mathcal{E}(C) \equiv \prod_{z \in C^+} (1 - z) \prod_{z \in C^-} z,$$

over formal variables $z_1, z_2, \ldots, z_n$. Observe that $\mathcal{E}(C) = 0$ is satisfied (over $0/1$ assignments to z_i) if and only if the corresponding assignment satisfies C. We abuse notation and let $\mathcal{E}(\mathcal{C}) = \{\mathcal{E}(C) : C \in \mathcal{C}\} \cup \{z_i^2 - z_i\}_{i \in [m]}$, and note that the second set of polynomial equations restricts the z_i inputs to $\{0, 1\}$ values. The $\mathbb{F}$-*Nullstellensatz degree* of $\mathcal{C}$, denoted $\mathsf{NS}_{\mathbb{F}}(\mathcal{C})$, is the Nullstellensatz degree of refuting $\mathcal{E}(\mathcal{C})$.

How do we know that a Nullstellensatz refutation always exists? One can deduce this from Hilbert's Nullstellensatz, but for our purposes it is enough to use a simpler version proved by Buss et al. (Theorem 5.2 in [BIK+97]): if $\mathcal{P}$ is a system of polynomial equations over $\mathbb{F}[z_1, \ldots, z_n]$ with no $\{0, 1\}$ solutions, then there exists a Nullstellensatz refutation of $\mathcal{P} \cup \{z_i^2 - z_i = 0\}_{i \in [n]}$.

2.3 Cutting Planes

The *Cutting planes (CP)* proof system was introduced in [CCT87] as a formalization of the integer linear programming algorithm in [Gom63, Chv73]. Cutting planes proofs give a formal method to deduce new linear inequalities from old that are *sound over integer solutions*—that is, if some integral vector x^* satisfies a set of linear inequalities $\mathcal{I}$, then x^* will also satisfy any inequality $ax \geq b$ deduced from $\mathcal{I}$ by a sequence of cutting planes deductions. The allowed deductions in a cutting planes proof are the following:

$$\text{Linear combination } \frac{\sum_i a_i x_i \geq A \qquad \sum_i b_i x_i \geq B}{\sum_i (c a_i + d b_i) x_i \geq cA + dB} \qquad \text{Division} \frac{\sum_i c a_i x_i \geq A}{\sum_i a_i x_i \geq \lceil A/c \rceil}$$

where a_i, b_i, c, d, A, and B are all integers and $c, d \geq 0$.

In order to use cutting planes to refute unsatisfiable CNF formulas, we need to translate clauses to inequalities. It is easy to see how to do this by example: we translate the clause $x \vee y \vee \neg z$ to the inequality $x + y + (1 - z) \geq 1$, or, equivalently, $x + y - z \geq 0$ if we collect all constant terms on the right-hand side. For refuting CNF formulas we equip cutting planes proofs with the following additional rules ensuring all variables take $\{0, 1\}$ values:

$$\text{Variable axioms } \frac{}{x \geq 0} \qquad \frac{}{-x \geq -1}$$

The goal, then, is to prove unsatisfiability by deriving the inequality $0 \geq 1$. This is possible if and only if there is no $\{0, 1\}$-assignment satisfying all constraints.

As discussed in the introduction, we are interested in several natural parameters of cutting planes proof—length, space, and the sizes of the coefficients. So, we define a cutting planes refutation as a sequence of *configurations* (this is also known as the *blackboard model*). A configuration is a set of linear inequalities with integer coefficients, and a sequence of configurations $\mathbb{C}_0, \ldots, \mathbb{C}_L$ is a cutting planes refutation of a formula $\mathcal{C}$ if $\mathbb{C}_0 = \emptyset$, $\mathbb{C}_L$ contains the contradiction $0 \geq 1$, and each configuration $\mathbb{C}_{t+1}$ follows from $\mathbb{C}_t$ either by adding an inequality in $\mathcal{C}$, by adding the result of one of the above inference rules where all the premises are in $\mathbb{C}_t$, or by removing an inequality present in $\mathbb{C}_t$. The length of a refutation is then defined to be the number of configurations L; the space[2] is $\max_{t \in [L]} |\mathbb{C}_t|$, the maximum number of

[2]Formally, this is known as the *line space*.

inequalities in a configuration; and the coefficient bit size is the maximum size in bits of a coefficient that appears in the refutation.

For any proof system, it is natural to ask what is the minimal amount of space needed to prove tautologies. Indeed, there has been much work in the literature studying this, and for proof systems such as resolution (e.g. [ET01, ABRW02, BG03, BN08]) and polynomial calculus (e.g. [ABRW02, FLN$^+$15, BG15, BBG$^+$17]) it is known that there are unsatisfiable CNF formulas which unconditionally require large space to refute. In contrast (and quite surprisingly!) it was shown in [GPT15] that for cutting planes proofs, constant line space is always enough. The proof presented in [GPT15] does use coefficients of exponential magnitude, but the authors are not able to show that this is necessary—only that coefficients of at most constant magnitude are not sufficient.

Similarly, one can ask whether cutting planes refutations require large coefficients to realize the full power of the proof system. Towards this, define CP* to be cutting planes proofs with *polynomially-bounded coefficients* or, in other words, a cutting planes refutation Π of a formula $\mathcal{C}$ with n variables is a CP* refutation if the largest coefficient in Π has magnitude $\mathsf{poly}(n, L)$.

The question of how CP* relates to unrestricted cutting planes has been raised in several papers, e.g., [BPR97, BEGJ00]. This question was studied already in [BC96], where it was proven that any cutting planes refutation in length L can be transformed into a refutation with $L^{O(1)}$ lines having coefficents of magnitude $\exp(O(L))$ (here the asymptotic notation hides a mild dependence on the size of the coefficients in the input). The authors write, however, that their original goal had been to show that coefficients of only polynomial magnitude would be enough, i.e., that CP* would be as powerful as cutting planes except possibly for a polynomial loss, but that they had to leave this as an open problem. To the best of our knowledge, there has not been a single example of *any unsatisfiable formula* where CP* could potentially perform much worse than general (high-weight) cutting planes.

Finally, as observed in [BPS07, HN12], we can use an efficient cutting planes refutation of a formula $\mathcal{C}$ to solve $\mathsf{Search}(\mathcal{C})$ by an efficient communication protocol. Since the first configuration $\mathbb{C}_0$ is always true and the last configuration $\mathbb{C}_L$ is always false, the players can simulate a binary search by evaluating the truth value of a configuration according to their joint assignment and find a true configuration followed by a false configuration. It is not hard to see that the inequality being added corresponds to a clause in $\mathcal{C}$ and it is a valid answer to $\mathsf{Search}(\mathcal{C})$.

Lemma 2.2 ([HN12]). *If there is a cutting planes refutation of $\mathcal{C}$ in length L, line space s, and coefficient bit size c, then there is a deterministic communication protocol for* $\mathsf{Search}(\mathcal{C})$ *of cost $O(s(c+\log n)\log L)$.*

3 Rank Lifting from Any Gadget

In this section we discuss our new lifting theorem, restated next.[3]

Theorem 3.1. *Let $\mathcal{C}$ be any unsatisfiable k-CNF on n variables and let $\mathbb{F}$ be any field. For any Boolean valued gadget g with $\mathrm{rank}(g) \geq 12enk/\mathsf{NS}_{\mathbb{F}}(\mathcal{C})$ we have*

$$\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C}) \circ g) \geq \mathsf{NS}_{\mathbb{F}}(\mathcal{C}).$$

This generalizes a recent lifting theorem from [PR18], which only allowed certain "good" gadgets. The main technical step of that proof showed that "good" gadgets can be used to lift the degree of multilinear polynomials to the rank of matrices. In this section, we improve this, showing that *any* gadget with non-trivial rank can be used to lift polynomial degree to rank. Given this result, Theorem 3.1 is proved by reproducing the proof of [PR18] with a tighter analysis. With this in mind, in this section we will prove our new lifting argument for degree to rank, and then relegate the rest of the proof of Theorem 3.1 to Appendix A.

[3]In fact, we prove a somewhat more general theorem (see Theorem A.1 in Appendix A for details). We also remark that this theorem in fact holds for a stronger communication measure (Razborov's *rank measure* [Raz90]), and so implies lower bounds for other models—see Appendix A for details.

Let us now make these arguments formal. We start by recalling the definition of a "good" gadget of [PR18].

Definition 3.2 (Definition 3.1 in [PR18]). Let $\mathbb{F}$ be a field. A gadget $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{F}$ is *good* if for any matrices A, B of the same size we have

$$\mathrm{rank}(\mathbb{1}_{\mathcal{X},\mathcal{Y}} \otimes A + g \otimes B) = \mathrm{rank}(A) + \mathrm{rank}(g)\mathrm{rank}(B)$$

where $\mathbb{1}_{\mathcal{X},\mathcal{Y}}$ denotes the $\mathcal{X} \times \mathcal{Y}$ all-1s matrix.

In [PR18] it is shown that good gadgets are useful because they lift *degree* to *rank* when composed with multilinear polynomials.

Theorem 3.3 (Theorem 1.2 in [PR18]). *Let $\mathbb{F}$ be any field, and let $p \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ be a multilinear polynomial over $\mathbb{F}$. For any good gadget $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{F}$ we have*

$$\mathrm{rank}(p \circ g^n) = \sum_{S:\hat{p}(S)\neq 0} \mathrm{rank}(g)^{|S|}.$$

In the present work, we show that a gadget being good is *not* strictly necessary to obtain the above lifting from degree to rank. In fact, composing with *any* gadget lifts degree to rank!

Theorem 3.4. *Let $p \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ be any multilinear polynomial and let $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{F}$ be any non-zero gadget with $\mathrm{rank}(g) \geq 3$. Then*

$$\sum_{S:\hat{p}(S)\neq 0} (\mathrm{rank}(g) - 3)^{|S|} \leq \mathrm{rank}(p \circ g^n) \leq \sum_{S:\hat{p}(S)\neq 0} \mathrm{rank}(g)^{|S|}.$$

We remark that the lower bound in the theorem can be sharpened to $\mathrm{rank}(g) - 2$ if the gadget g is not full rank. While the previous theorem does not require the gadget g to be good, the notion of a good gadget will still play a key role in the proof. The general idea is that every gadget with non-trivial rank can be transformed into a good gadget with a slight modification. With this in mind, en-route to proving Theorem 3.4 we give the following characterization of good gadgets which may be of independent interest.

Lemma 3.5. *A gadget g is good if and only if the all-1s vector is not in the row or column space of g.*

In the remainder of the section we prove Theorem 3.4 and Lemma 3.5.

3.1 Proof of Lemma 3.5

We begin by proving Lemma 3.5, which is by a simple linear-algebraic argument. Given a matrix M over a field, let $row(M)$ denote the row-space of M and let $col(M)$ denote the column-space of M. The following characterization of when rank is additive will be crucial.

Theorem 3.6 ([MS72]). *For any matrices A, B of the same size over any field, $\mathrm{rank}(A+B) = \mathrm{rank}(A) + \mathrm{rank}(B)$ if and only if $row(A) \cap row(B) = col(A) \cap col(B) = \{\mathbf{0}\}$.*

The previous theorem formalizes the intuition that rank should be additive if and only if the corresponding linear operators act on disjoint parts of the vector space. Using the previous theorem we deduce the following general statement, from which Lemma 3.5 immediately follows.

Lemma 3.7. *Let f, g be matrices over any fixed field $\mathbb{F}$ of the same size. The following are equivalent:*

1. *For all matrices A, B of the same size, $\mathrm{rank}(f \otimes A + g \otimes B) = \mathrm{rank}(f)\mathrm{rank}(A) + \mathrm{rank}(g)\mathrm{rank}(B)$.*

2. $\mathrm{rank}(f + g) = \mathrm{rank}(f) + \mathrm{rank}(g)$.

Proof. By choosing $A = B = (1)$ we instantly deduce (2) from (1). To prove the converse, we use Theorem 3.6. Let A, B be matrices such that $\text{rank}(f \otimes A + g \otimes B) \neq \text{rank}(f)\text{rank}(A) + \text{rank}(g)\text{rank}(B)$. Then by Theorem 3.6 it follows that there is a non-zero vector in the intersection of either the row- or column-spaces of $f \otimes A$ and $g \otimes B$. Suppose that there is a non-zero vector $u \in col(f \otimes A) \cap col(g \otimes B)$, and we prove that there is a non-zero vector in $col(f) \cap col(g)$ implying $\text{rank}(f + g) \neq \text{rank}(f) + \text{rank}(g)$. (A symmetric argument will apply to the row spaces.)

Assume that f and g are $a \times b$ dimensional matrices, and that A and B are $m \times n$ dimensional matrices. Let u be the length am non-zero vector in the column spaces of both $f \otimes A$ and $g \otimes B$, and suppose without loss of generality that $u_1 \neq 0$. It follows that there are length bn vectors x, y such that $(f \otimes A)x = u = (g \otimes B)y$. Write

$$x = (x^1, x^2, \ldots, x^b),$$
$$y = (y^1, y^2, \ldots, y^b)$$

where x^i, y^i are vectors of length n for each i.

Let A_1 denote the first row of A and B_1 denote the first row of B; note they are both vectors of length n. Define the length-b vectors

$$x' = (A_1 x^1, A_1 x^2, \ldots, A_1 x^b),$$
$$y' = (B_1 y^1, B_1 y^2, \ldots, B_1 y^b).$$

Then, by definition, for each $i = 1, 2, \ldots, a$ we have $(fx')_i = u_{(i-1)m+1} = (gy')_i$, and the vector is non-zero since $u_1 \neq 0$ by assumption. Thus $fx' = gy'$ and the column spaces of f and g intersect at a non-zero vector. $\square$

From Lemma 3.7 we can deduce Lemma 3.5 immediately.

Proof of Lemma 3.5. By the previous lemma, g is good if and only if $\text{rank}(\mathbb{1} + g) = \text{rank}(\mathbb{1}) + \text{rank}(g)$. By Theorem 3.6 this is true iff the all-1s vector is not in the row- or column-space of g. $\square$

3.2 Proof of Theorem 3.4

In this section we prove Theorem 3.4 using Lemma 3.5. The theorem follows by induction using the following lemma, and the proof mimics the proof from [PR18, Rob18].

Lemma 3.8. *Let $\mathbb{F}$ be any field, and let $g : \mathcal{X} \times \mathcal{Y} \to \mathbb{F}$ be any gadget with $\text{rank}(g) \geq 3$. For any matrices A, B of the same size we have*

$$\text{rank}(\mathbb{1}_{\mathcal{X},\mathcal{Y}} \otimes A + g \otimes B) \geq \text{rank}(A) + (\text{rank}(g) - 3)\text{rank}(B)$$

where $\mathbb{1}_{\mathcal{X},\mathcal{Y}}$ is the $\mathcal{X} \times \mathcal{Y}$ all-1s matrix.

Proof. Assume without loss of generality that $|\mathcal{X}| \geq |\mathcal{Y}|$ and let $\mathbb{1} = \mathbb{1}_{\mathcal{X},\mathcal{Y}}$. Thinking of g as a matrix, let u be any column vector of g. If we zero the entries of u in g, then the remaining matrix cannot have full rank, implying that some row-vector v of the remaining matrix will become linearly dependent. Let g_1 be the $\mathcal{X} \times \mathcal{Y}$ matrix consisting of the u column and v row of g, and let g_2 be the $\mathcal{X} \times \mathcal{Y}$ matrix obtained by zeroing out u and v in g. Observe $g = g_1 + g_2$, and also since g_2 contains an all-0 row and an all-0 column it is good by Lemma 3.5 (as any linear combination of rows/columns of g must contain a zero coordinate).

Now, observe that

$$\begin{aligned}\text{rank}(\mathbb{1} \otimes A + g \otimes B) &= \text{rank}(\mathbb{1} \otimes A + g_1 \otimes B + g_2 \otimes B) \\ &\geq \text{rank}(\mathbb{1} \otimes A + g_2 \otimes B) - \text{rank}(g_1 \otimes B) \\ &= \text{rank}(\mathbb{1} \otimes A + g_2 \otimes B) - \text{rank}(g_1)\text{rank}(B)\end{aligned}$$

where the inequality follows since adding a rank-R matrix can decrease the rank by at most R. Since g_1 consists of a single non-zero row and column we have $\mathrm{rank}(g_1) \leq 2$; by the construction of g_2 we have $\mathrm{rank}(g_2) = \mathrm{rank}(g) - 1$. Using these facts and the fact that g_2 is good, we have

$$\begin{aligned}\mathrm{rank}(\mathbb{1} \otimes A + g_2 \otimes B) - \mathrm{rank}(g_1)\mathrm{rank}(B) &\geq \mathrm{rank}(A) + \mathrm{rank}(g_2)\mathrm{rank}(B) - 2\,\mathrm{rank}(B)\\ &= \mathrm{rank}(A) + (\mathrm{rank}(g) - 3)\mathrm{rank}(B).\end{aligned}$$ □

With the lemma in hand we can prove Theorem 3.4.

Proof of Theorem 3.4. We prove

$$\mathrm{rank}(p \circ g^n) \geq \sum_{S:\hat{p}(S)\neq 0} (\mathrm{rank}(g) - 3)^{|S|}$$

by induction on n, the number of variables.

Observe that the inequality is trivially true if $n = 0$. Assume $n > 0$, and let $\mathbb{1} = \mathbb{1}_{\mathcal{X},\mathcal{Y}}$. Write $p = q + z_1 r$ for multilinear polynomials $q, r \in \mathbb{F}[z_2, z_3, \ldots, z_n]$. Note that it clearly holds that $p \circ g^n = \mathbb{1} \otimes (q \circ g^{n-1}) + g \otimes (r \circ g^{n-1})$. From the claim we have by induction that

$$\begin{aligned}\mathrm{rank}(p \circ g^n) &= \mathrm{rank}(\mathbb{1} \otimes (q \circ g^{n-1}) + g \otimes (r \circ g^{n-1}))\\ &\geq \mathrm{rank}(q \circ g^{n-1}) + (\mathrm{rank}(g) - 3)\mathrm{rank}(r \circ g^{n-1})\\ &= \sum_{S:\hat{q}(S)\neq 0} (\mathrm{rank}(g) - 3)^{|S|} + (\mathrm{rank}(g) - 3) \sum_{T:\hat{r}(T)\neq 0} (\mathrm{rank}(g) - 3)^{|T|}\\ &= \sum_{\substack{S:\hat{p}(S)\neq 0\\ z_1 \notin S}} (\mathrm{rank}(g) - 3)^{|S|} + (\mathrm{rank}(g) - 3) \sum_{\substack{T:\hat{p}(T)\neq 0\\ z_1 \in T}} (\mathrm{rank}(g) - 3)^{|T|-1}\\ &= \sum_{S:\hat{p}(S)\neq 0} (\mathrm{rank}(g) - 3)^{|S|}.\end{aligned}$$

For the upper bound, by subadditivity of rank we have

$$\begin{aligned}\mathrm{rank}(\mathbb{1} \otimes A + g \otimes B) &\leq \mathrm{rank}(\mathbb{1} \otimes A) + \mathrm{rank}(g \otimes B)\\ &= \mathrm{rank}(\mathbb{1})\mathrm{rank}(A) + \mathrm{rank}(g)\mathrm{rank}(B)\\ &= \mathrm{rank}(A) + \mathrm{rank}(g)\mathrm{rank}(B).\end{aligned}$$

Apply the above induction argument using this inequality *mutatis mutandis*. □

4 Application: Separating Cutting Planes Systems

In this section we prove a new separation between high-weight and low-weight cutting planes proofs in the bounded-space regime.

Theorem 4.1. *There is a family of $O(\log\log n)$-CNF formulas over $O(n \log\log n)$ variables and $\tilde{O}(n)$ clauses that have CP refutations in length $\tilde{O}(n^2)$ and line space $O(1)$, but for which any CP* refutation in length L and line space s must satisfy $s \log L = \Omega(n/\log^2 n)$.*

By the results of [GPT15], *any* unsatisfiable CNF formula has a cutting planes refutation in constant line space, albeit with exponential length and exponentially large coefficients. In Theorem 4.1 we show that the length of such a refutation can be reduced to polynomial for certain formulas, described next.

At a high level, we prove Theorem 4.1 using the reversible pebble game. Given any DAG G with a unique sink node t, the *reversible pebble game* [Ben89] is a single-player game that is played with a set of pebbles on G. Initially the graph is empty, and at each step the player can either place or remove a pebble on a vertex whose predecessors already have pebbles (in particular the player can always place or remove a pebble on a source). The goal of the game is to place a pebble on the sink while using as few pebbles as possible. The reversible pebbling price of a graph, denoted $\mathrm{rpeb}(G)$, is the minimum number of pebbles required to place a pebble on the sink.

The family of formulas witnessing Theorem 4.1 are *pebbling formulas* composed with the equality gadget. Intuitively, the pebbling formula [BW01] Peb_G associated with G is a formula that claims that it is impossible to place a pebble on the sink (using any number of pebbles). Since it is always possible to place a pebble by using some amount of pebbles, this formula is clearly a contradiction.

Formally, the pebbling formula Peb_G is the following CNF formula. For each vertex $u \in V$ there is a variable z_u (intuitively, z_u should take the value "true" if and only if it is possible to place a pebble on u using any number of pebbles). The variables are constrained by the following clauses.

- a clause z_s for each source vertex s (i.e., we can always place a pebble on any source),
- a clause $\bigvee_{u \in \mathrm{pred}(v)} \neg z_u \vee z_v$ for each non-source vertex v with predecessors $\mathrm{pred}(v)$ (i.e., if we can place a pebble on the predecessors of v, then we can place a pebble on v), and
- a clause $\neg z_t$ for the sink t (i.e., it is impossible to place a pebble on t).

Proving Theorem 4.1 factors into two tasks: a lower bound and an upper bound. By applying our lifting theorem from the previous section, the lower bound will follow immediately from a good lower bound on the Nullstellensatz degree of pebbling formulas. In order to prove lower bounds on the Nullstellensatz degree, we show in Section 4.1 that over *every* field, the Nullstellensatz degree required to refute Peb_G is *exactly* the reversible pebbling price of G. We then use it together with our lifting theorem to prove the time-space tradeoff for bounded-coefficient cutting planes refutations of $\mathrm{Peb}_G \circ g$ in Section 4.2 for *any* high-rank gadget g. Finally, in Section 4.3 we prove the upper bound by presenting a short and constant-space refutation of $\mathrm{Peb}_G \circ \mathrm{EQ}$ in cutting planes with unbounded coefficients.

4.1 Nullstellensatz Degree of Pebbling Formulas

In this section we prove that the Nullstellensatz degree of the pebbling formula of a graph G equals the reversible pebbling price of G.

Lemma 4.2. *For any field $\mathbb{F}$ and any graph G, $\mathsf{NS}_{\mathbb{F}}(\mathrm{Peb}_G) = \mathrm{rpeb}(G)$.*

We crucially use the following dual characterization of Nullstellensatz degree by designs [Bus98].

Definition 4.3. Let $\mathbb{F}$ be a field, let d be a positive integer, and let $\mathcal{P}$ be an unsatisfiable system of polynomial equations over $\mathbb{F}[z_1, z_2, \ldots, z_n]$. A *$d$-design* for $\mathcal{P}$ is a linear functional D on the space of polynomials satisfying the following axioms:

1. $D(1) = 1$.
2. For all $p \in \mathcal{P}$ and all polynomials q such that $\deg(pq) \leq d$, we have $D(pq) = 0$.

Clearly, if we have a candidate degree-d Nullstellensatz refutation $1 = \sum p_i q_i$, then applying a d-design to both sides of the refutation yields $1 = 0$, a contradiction. Thus, if a d-design exists for a system of polynomials then there cannot be a Nullstellensatz refutation of degree d. Remarkably, a converse holds for systems of polynomials over $\{0, 1\}^n$.

Theorem 4.4 (Theorems 3, 4 in [Bus98]). *Let $\mathbb{F}$ be a field and let $\mathcal{P}$ be a system of polynomial equations over $\mathbb{F}[z_1, z_2, \ldots, z_n]$ containing the Boolean equations $z_i^2 - z_i = 0$ for all $i \in [n]$. Then $\mathcal{P}$ does not have a degree-d Nullstellensatz refutation if and only if it has a d-design.*

With this characterization in hand we prove Lemma 4.2.

Proof of Lemma 4.2. Let G be a DAG, and consider the pebbling formula G. Following the standard translation of CNF formulas into unsatisfiable systems of polynomial equations, we express Peb_G with the following equations:

Source Equations. The equation $(1 - z_s) = 0$ for each source vertex s.

Sink Equations. The equation $z_t = 0$ for the sink vertex t.

Neighbour Equations. The equation $(1 - z_v) \prod_{u \in \mathrm{pred}(v)} z_u = 0$ for each internal vertex v.

Boolean Equations. The equation $z_v^2 - z_v = 0$ for each vertex v.

We prove that a d-design for the above system exists if and only if $d < \mathrm{rpeb}(G)$, and this implies the lemma. Let D be a d-design for the system. First, note that since the Boolean axioms are satisfied and since D is linear, it follows that D is completely specified by its value on multilinear monomials $z_T := \prod_{i \in T} z_i$ (with this notation note that $z_\emptyset := 1$). Moreover, D must satisfy the following properties:

Empty Set Axiom. $D(z_\emptyset) = 1$.

Source Axioms. $D(z_T) = D(z_T z_s)$ for every source s and every $T \subseteq [n]$ with $|T \cup \{s\}| \leq d$.

Neighbour Axioms. $D(z_T z_{\mathrm{pred}(v)}) = D(z_T z_{\mathrm{pred}(v)} z_v)$ for every non-source vertex v and every $T \subseteq [n]$ with $|T \cup \mathrm{pred}(v) \cup \{v\}| \leq d$.

Sink Axiom. $D(z_T z_t) = 0$ for the sink t and every $T \subseteq [n]$ with $|T \cup \{t\}| \leq d$.

We may assume without loss of generality that $D(z_T) = 0$ for any set T with $|T| > d$.

Given a set S of vertices of G, we think of S as the reversible pebbling configuration in which there are pebbles on the vertices in S and there are no pebbles on any other vertex. We say that a configuration T is *reachable* from a configuration S if there is a sequence of legal reversible pebbling moves that changes S to T while using at most d pebbles at any given point.

Now, we claim that the only way to satisfy the first three axioms is to set $D(x_T) = 1$ for every configuration T that is reachable from $\emptyset$. To see why, observe that those axioms are satisfiable if and only if the empty configuration is assigned the value 1, any configuration containing the sink is labelled 0, and $D(z_S) = D(z_T)$ for any two configurations S, T with at most d pebbles that are mutually reachable via a single reversible pebbling move. Hence, this setting of D is the only one we need to consider.

Finally, observe that this specification of a design D satisfies the sink axiom if and only if $d < \mathrm{rpeb}(G)$, since the sink is reachable from $\emptyset$ using $\mathrm{rpeb}(G)$ pebbles but not with less (by the definition of $\mathrm{rpeb}(G)$). Therefore, a d-design for Peb_G exists if and only if $d < \mathrm{rpeb}(G)$, as required. □

4.2 Time-Space Lower Bounds for Low-Weight Refutations

In this section we prove the lower bound part of the time-space trade-off for CP*.

Lemma 4.5. *There is a family of graphs $\{G_n\}$ with n vertices and constant degree, such that every CP* refutation of $\mathrm{Peb}_{G_n} \circ \mathrm{EQ}$ in length L and line space s must have $s \log L = \Omega(n/\log^2 n)$.*

Our plan is to lift a pebbling formula that is hard with respect to Nullstellensatz degree, and as we just proved it is enough to find a family of graphs whose reversible pebbling price is large. Paul et al. [PTC77] provide such a family (and in fact prove their hardness in the stronger standard pebbling model).

Theorem 4.6. *There is a family of graphs $\{G_n\}$ with n vertices, constant degree, and for which $\mathrm{rpeb}(G_n) = \Omega(n/\log n)$.*

We combine these graphs with our lifting theorem as follows.

Lemma 4.7. *There is a family of graphs $\{G_n\}$ with n vertices and constant degree, such that $\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathrm{Peb}_G \circ \mathrm{EQ})) = \Omega(n/\log n)$.*

Proof. Let Peb_G be the pebbling formula of a graph $G = G_n$ from the family given by Theorem 4.6. By Lemma 4.2 the Nullstellensatz degree of the formula is

$$\mathsf{NS}_{\mathbb{F}}(\mathrm{Peb}_G) = \mathrm{rpeb}(G) = \Omega(n/\log n) \ . \tag{4.1}$$

This allows us to use our lifting theorem, Theorem 3.1, with an equality gadget of arity $q = O(\log(n/\mathsf{NS}_{\mathbb{F}}(\mathrm{Peb}_G)) = O(\log\log n)$, and obtain that the lifted search problem $\mathsf{Search}(\mathrm{Peb}_G) \circ \mathrm{EQ}$ requires deterministic communication

$$\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathrm{Peb}_G) \circ \mathrm{EQ}) \geq NS_{\mathbb{F}}(\mathrm{Peb}_G) = \Omega(n/\log n) \ . \tag{4.2}$$

As we noted in Observation 2.1, this implies that the search problem of the lifted formula also requires deterministic communication

$$\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathrm{Peb}_G \circ \mathrm{EQ})) \geq \mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathrm{Peb}_G) \circ \mathrm{EQ}) = \Omega(n/\log n) \ . \tag{4.3}$$

□

Since we collected our last ingredient, let us finish the proof.

Proof of Lemma 4.5. Let $S = \mathsf{Search}(\mathrm{Peb}_G \circ \mathrm{EQ})$ be the search problem given by Lemma 4.7. Using Lemma 2.2 we have that every cutting planes refutation of the lifted formula in length L, line space s, and coefficient length c must satisfy

$$s(c + \log n)\log L = \Omega(\mathsf{P}^{\mathsf{cc}}(S)) = \Omega(n/\log n) \ . \tag{4.4}$$

Since the size of the lifted formula $\mathrm{Peb}_G \circ \mathrm{EQ}$ is $\tilde{O}(n)$, the coefficients of a CP* refutation are bounded by a polynomial of n in magnitude, and hence by $O(\log n)$ in length. Substituting the value of $c = O(\log n)$ in (4.4) we obtain that

$$s \log L = \Omega(n/\log^2 n) \tag{4.5}$$

as we wanted to show. □

4.3 Time-Space Upper Bounds for High Weight Refutations

We now prove Theorem 4.8, showing that cutting planes proofs with large coefficients can efficiently refute pebbling formulas composed with equality gadgets in constant line space. Let EQ_q denote the equality gadget on q bits.

Theorem 4.8. *Let* Peb_G *be any constant-width pebbling formula. There is a cutting planes refutation of* $\mathrm{Peb}_G \circ \mathrm{EQ}_{\log\log n}$ *in length* $\tilde{O}(n^2)$ *and space* $O(1)$.

We also use the following lemma, which is a "derivational" analogue of the recent result of [GPT15] showing that any set of unsatisfiable integer linear inequalities has a cutting planes refutation in constant space. As the techniques are essentially the same we leave the proof to Appendix B.

Lemma 4.9 (Space Lemma). *Let* $\mathcal{C}$ *be a set of integer linear inequalities over* n *variables that implies a clause* C. *Then there is a cutting planes derivation of* C *from* $\mathcal{C}$ *in length* $O(n^2 2^n)$ *and space* $O(1)$.

Let us begin by outlining the high level proof idea. We would like to refute the lifted formula $\mathrm{Peb}_G \circ \mathrm{EQ}_q$ using constant space. Consider first the unlifted formula Peb_G. The natural way to refute it is the following: Let $v_1, \ldots, v_n$ be a topological ordering of the vertices of G. The refutation will go over the vertices in this order, each time deriving the equation that says that the variable z_{v_i} must take the value "true" by using the equations that were derived earlier for the predecessors of v_i. Eventually, the refutation will derive the equation that says that the sink must take the value "true", which contradicts the axiom that says that the sink must be false.

Going back to the lifted formula $\mathrm{Peb}_G \circ \mathrm{EQ}_q$, we construct a refutation using the same strategy, except that now the equation of z_{v_i} is replaced with the equations

$$x_{v_i,1} = y_{v_i,1}, \ldots x_{v_i,q} = y_{v_i,q}.$$

The main obstacle is that if we implement this refutation in the naive way, we will have to store all the equations simultaneously, yielding a refutation of space $O(q \cdot n)$. The key idea of our proof is that CP can encode the conjunction of many equations using a *single* equation. We can therefore use this encoding in our refutation to store at any given point all the equations that were derived so far in a single equation. The implementation yields a refutation of constant space, as required.

To see how we can encode multiple equations using a single equation, consider the following example. Suppose we wish to encode the equations

$$x_1 = y_1, x_2 = y_2, x_3 = y_3,$$

where all the variables take values in $\{0, 1\}$. Then, it is easy to see that those equations are equivalent to the equation

$$4 \cdot x_1 + 2 \cdot x_2 + x_3 = 4 \cdot y_1 + 2 \cdot y_2 + y_3.$$

This idea generalizes in a straightforward way to deal with more equations, as well as with arbitrary linear gadgets, to be discussed below.

The rest of this section is devoted to the proof of Theorem 4.8. The following notion is central to the proof. Say a gadget $g(x, y) : \{0, 1\}^q \times \{0, 1\}^q \to \{0, 1\}$ is *linear* if there exists a linear expression with integer coefficients

$$L(x, y) = c + \sum_{i=1}^{q} a_i x_i + b_i y_i$$

such that $g(x, y) = 1$ if and only if $L(x, y) = 0$. Note that the equality gadget is linear, as it corresponds to the linear expression $\sum_{i=1}^{q} 2^{i-1}(x_i - y_i)$.

Let g be any linear gadget with corresponding linear expression L. Let $K = 1 + \max_{x,y} |L(x, y)|$, and let G be the underlying DAG of the composed pebbling formula $\mathrm{Peb}_G \circ g^n$. Note that for each vertex u of G the composed formula has corresponding variables $x_u, y_u \in \{0, 1\}^q$. Once and for all, fix an ordering of the vertices of G and assume that all subsets are ordered accordingly. For a subset of vertices $U \subseteq V$ define

$$L(U) := \sum_{u_i \in U} K^i L(x_{u_i}, y_{u_i}).$$

The following claim shows that $L(U)$ encodes the truth of the conjunction $\bigwedge_{u_i \in U} g(x_{u_i}, y_{u_i})$.

Claim 4.10. For a set of vertices U and any $x, y \in \{0, 1\}^{qn}$, $L(U) = 0$ if and only if $\bigwedge_{u_i \in U} g(x_{u_i}, y_{u_i}) = 1$.

Proof. Since g is linear, if $g(x_{u_i}, y_{u_i}) = 1$ for all $u_i \in U$ then it follows that $L(x_{u_i}, y_{u_i}) = 0$ for all $u_i \in U$, which in turn implies $L(U) = 0$. Conversely, suppose $g(x_{u_i}, y_{u_i}) = 0$ for some vertex u_i, and let i be the *largest* such index. It follows that $L(u_i) \neq 0$, and clearly

$$\left| \sum_{j<i} K^j L(x_{u_j}, y_{u_j}) \right| \leq \sum_{j<i} K^j |L(x_{u_j}, y_{u_j})| \leq (K - 1) \sum_{j<i} K^j < K^i. \tag{4.6}$$

This implies $L(U) \neq 0$, since

$$\begin{aligned} |L(U)| &= \left| K^i \cdot L(u_i) + \sum_{j<i} K^j L(x_{u_j}, y_{u_j}) \right| \\ &\geq \left| K^i \cdot L(u_i) \right| - \left| \sum_{j<i} K^j L(x_{u_j}, y_{u_j}) \right| \\ &\geq K^i - \left| \sum_{j<i} K^j L(x_{u_j}, y_{u_j}) \right| > 0 \end{aligned} \qquad \square$$

From here on in the proof, we consider $L(U) = 0$, or $L(U)$ for short, as being syntactically represented in cutting planes as the pair of inequalities $L(U) \geq 0$, $-L(U) \geq 0$. The bulk of the proof lies in the following lemma, which shows how to "encode" and "decode" unit literals in the expressions $L(U)$.

Lemma 4.11 (Coding Lemma). *Let U be any set of vertices. Then*

1. *For any $u \in U$ there is a cutting planes derivation of $L(u)$ from $L(U)$ in length $O(q|U|)$ and space $O(1)$.*
2. *Let $C = \neg z_{u_1} \vee \neg z_{u_2} \vee \cdots \vee \neg z_{u_{k-1}} \vee z_{u_k}$ be an axiom of Peb_G with $u_1, u_2, \ldots, u_{k-1} \in U$. Let ℓ_g and s_g be such that there exists a derivation of $L(u)$ from a CNF encoding of $g(u)$ in length ℓ_g and space s_g. From $L(u_1), L(u_2), \ldots, L(u_{k-1})$ and $C \circ g^n$ there is a cutting planes derivation of $L(u_k)$ in length $O(2^{kq}\ell_g)$ and space $O(s_g)$.*
3. *For any $u \notin U$ there is a cutting planes derivation of $L(U \cup \{u\})$ from $L(U)$ and $L(u)$ in length $O(1)$ and space $O(1)$.*

Let us first use the Coding Lemma to complete the proof. We show a more general statement from which Theorem 4.8 follows immediately by setting $k = 3$ and $g = \mathrm{EQ}_q$, with $q = O(\log \log n)$, and bounding $\ell_{\mathrm{EQ}} = O(q)$ and $s_{\mathrm{EQ}} = O(1)$.

Lemma 4.12. *If Peb_G is a width-k pebbling formula on n variables and g is a linear gadget of arity q then there is a cutting planes refutation of $\mathrm{Peb}_G \circ g^n$ in length $O(n(kqn + 2^{kq}\ell_g))$ and space $O(k + s_g)$.*

Proof. We begin with $L(\emptyset)$, which is represented as the pair of inequalities $0 \geq 0, 0 \geq 0$. By combining Parts (2) and (3) of the Coding Lemma we can derive $L(S)$, where S is the set of sources of G. We then follow a unit-propagation proof of Peb_G, deriving $L(u)$ for each vertex of G in topological order. Suppose at some point during the derivation we have derived $L(U)$ for some subset U of vertices. For any axiom C of Peb_G of the form $C = \neg z_{u_1} \vee \neg z_{u_2} \vee \cdots \vee \neg z_{u_{k-1}} \vee z_{u_k}$ with $u_1, u_2, \ldots, u_{k-1} \in U$ we do the following: first apply Part (1) of the Coding Lemma to obtain $L(u_i)$ for each $i \in [k-1]$. Apply Part (2) to derive $L(u_k)$, forget $L(u_i)$ for each $i \in [k-1]$, and then apply Part (3) to $L(U)$ and $L(u_k)$ to derive $L(U \cup \{u_k\})$. Continue in this way until we derive $L(z)$ where z is the sink vertex of G. Since $\{L(z), \neg z \circ g\}$ is an unsatisfiable set of linear inequalities, it follows by the Space Lemma (Lemma 4.9) that we can deduce a contradiction in length $O(q^2 2^q)$ and space $O(1)$.

In the above proof we need to derive $L(u)$ for each of the n vertices of the graph. Deriving $L(u)$ requires at most $O(k)$ applications of Part (1), one application of Part (2), and one application of Part (3). Thus, in total, we require length $O(n(kqn + 2^{kq}\ell_g))$ and space $O(k + s_g)$. $\square$

It remains to prove the Coding Lemma (Lemma 4.11).

Proof of Coding Lemma. Let $U = \{u_1, u_2, \ldots, u_t\}$ be an arbitrary subset of vertices of size t. Recall the definition $L(U) = \sum_{i=1}^{t} K^{i-1} L(u_i)$. For any $u_i \in U$ a *term* of $L(U)$ will be one of the terms $K^{i-1}L(u_i)$, which is a sum of $2q$ variables itself. We begin by defining two auxiliary operations that allow us to trim both the least and the most significant terms from $L(U)$.

To trim the i least significant terms of an inequality we essentially divide by K^i. More formally, for every variable v with a positive coefficient a_j less than K^i we add the inequality $-a_j v \geq -a_j$, and for every variable with a negative coefficient greater than $-K^i$ we add the inequality $a_j v \geq 0$. This takes length $O(qi)$, since each term contributes $2q$ coefficients, and space $O(1)$.

At this point all the remaining coefficients on the LHS are divisible by K^i, so we can apply the division rule. By construction the RHS is greater than $-K^i - \sum_{j \geq i} cK^j$, therefore when we divide by K^i the coefficient on the RHS becomes $-\sum_{j \geq i} cK^{j-i}$.

Finally, to restore the values of the coefficients to the values they had before dividing, we multiply by K^i at the end to restore them.

To trim the $m - i$ most significant terms of an inequality with m terms we need to use the opposite inequality, since the remaining part only has a semantic meaning when the most significant part vanishes. Hence we first trim the i least significant terms of the opposite inequality, keeping exactly the negation of the terms that we want to discard. Then we add both inequalities so that only the i least significant terms remain. This takes length $O(qi)$ and space $O(1)$.

Using the trimming operations we can prove items 1–3 in the lemma.

1. We must show that for any $u \in U$ there is a cutting planes derivation of $L(u)$ from $L(U)$ in length $O(qt)$ and space $O(1)$. This is straightforward: begin by making copies of the pair of inequalities $L(U) \geq 0$ and $-L(U) \geq 0$ encoding $L(U) = 0$. Trim the terms that are strictly more and strictly less significant than $L(u)$ from both of the inequalities, in length $O(qt)$ and space $O(1)$.

2. Recall that we assumed there is a derivation Π of $L(u_k)$ from the CNF formula $z_{u_k} \circ g$ in length ℓ_g and space s_g, so our goal is to produce the set of clauses $z_{u_k} \circ g$. Any such clause D is implied by the set of inequalities $\{L(u_i)\}_{i=1}^{k-1}$ together with the CNF encoding of $C \circ g^n$, therefore it has a derivation Π_D in length $O(2^{kq})$ and space $O(1)$ by the Space Lemma (Lemma 4.9). Replacing each usage of a clause $D \in z_{u_k} \circ g$ in Π as an axiom by the corresponding derivation Π_D we obtain a sound derivation $L(u_{t+1})$ in length $O(2^{tq}\ell_g)$ and space $O(s_g)$.

3. Simply add $K^{t+1}L(u) \geq 0$ to $L(U) \geq 0$ and $-K^{t+1}L(u) \geq 0$ to $-L(U) \geq 0$; this clearly uses bounded length and space. □

This completes the proof of Theorem 4.8.

Note that the largest coefficient used in the refutation is bounded by K^n. Indeed, the argument can be generalized to give a continuous trade-off between the size of the largest coefficient and the number of inequalities, simply by adding a new pair of empty inequalities once the coefficient required to add a vertex to an existing pair would be too large. This means that if we allow up to ξ inequalities then we can use coefficients of size bounded by $K^{O(n/\xi)}$.

5 Application: Separating Monotone Boolean and Real Formulas

In this section we exhibit an explicit function that exponentially separates the size of monotone Boolean formulas and monotone real formulas.

Theorem 5.1. *There is an explicit family of functions f_n over $O(n \operatorname{polylog} n)$ variables that can be computed by monotone real formulas of size $O(n \operatorname{polylog} n)$ but for which every monotone Boolean formula requires size $2^{\Omega(n/\log n)}$.*

To prove the lower bound part of Theorem 5.1 we use the characterization of formula depth by communication complexity [KW90]. Given a monotone Boolean function f, the monotone Karchmer–Wigderson game of f is a search problem $\mathrm{mKW}(f)\colon \{0,1\}^n \times \{0,1\}^n \to [n]$ defined as $((x, y), i) \in \mathrm{mKW}(f)$ if $f(x) = 1$, $f(y) = 0$, $x_i = 1$, and $y_i = 0$. In other words, given a 1-input x and a 0-input y

for f, the task is to find an index $i \in [n]$ such that $x_i = 1$, and $y_i = 0$. Such an index always exists because f is monotone.

If we denote by $\mathrm{mD}(f)$ the minimum depth of a monotone Boolean formula required to compute a Boolean function f, then we can write the characterization as

Lemma 5.2 ([KW90]). *For every function f, it holds that $\mathrm{mD}(f) = \mathsf{P}^{\mathsf{cc}}(\mathrm{mKW}(f))$.*

The analogue of this characterization for real circuits is in terms of DAG-like real protocols [Kra98, Sok17, HP18]. Since we are only interested in formulas rather than circuits we only consider tree-like protocols, which we call *locally real* protocols to distinguish them from the stronger model of real protocols, also known as real games [Kra98].

A locally real communication protocol, then, is a communication protocol where the set of inputs compatible with a node is defined by one half-space, as opposed to a real protocol where the set of compatible inputs is defined by the intersection of all half-spaces in the path leading to that node.

Formally, a locally real protocol for a search problem $\mathsf{Search}\colon \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$, where $\mathcal{X}$ and $\mathcal{Y}$ are Boolean hypercubes, is a tree where every internal node v is labelled with a half-space $H_v = \{(x, y) \in \mathcal{X} \times \mathcal{Y} \mid \langle xy, c_v\rangle \geq d_v\}$, where $c_v \in \mathbb{R}^{\dim \mathcal{X} + \dim \mathcal{Y}}$ and $d_v \in \mathbb{R}$, and every leaf is additionally labelled with an element $z \in \mathcal{Z}$. The root is labelled with the full space $\mathcal{X} \times \mathcal{Y}$, children are consistent in the sense that if a node w has children u and v then $H_w \subseteq H_u \cup H_v$. Given an input (x, y), the protocol produces a nondeterministic output z as follows. We start at the root and at each internal node we nondeterministically move to a child that contains (x, y), which exists by the consistency condition. The output of the protocol is the label of the resulting leaf. A protocol is correct if for any input $(x, y) \in \mathcal{X} \times \mathcal{Y}$ it holds that $z \in \mathcal{Z}$.

It is not hard to turn a real formula into a locally real protocol, and the converse also holds.

Lemma 5.3 ([HP18]). *Given a locally real protocol for the monotone Karchmer–Wigderson game of a partial function f, there exists a monotone real formula with the same underlying graph that computes f.*

In order to obtain a function whose Karchmer–Wigderson game we can analyse we use the fact that every search problem can be interpreted as the Karchmer–Wigderson game of some function. To state the result we need the notion of a nondeterministic communication protocol, which is a collection N of deterministic protocols such that $((x, y), z) \in S$ if and only if there exists some protocol $\pi \in N$ such that $\pi(x, y) = z$. The cost of a nondeterministic protocol is $\log|N| + \max_{\pi\in N} \mathrm{depth}(\pi)$.

Lemma 5.4 ([Raz90, Gál01], see also [Rob18]). *Let S be a two-party total search problem with nondeterministic communication complexity k. There exists a partial monotone Boolean function $f\colon \{0, 1\}^{2^k} \to \{0, 1, *\}$ such that S is exactly the monotone Karchmer–Wigderson game of f.*

We use as a search problem the falsified clause search problem of a hard pebbling formula composed with equality given by Lemma 4.7. To exhibit a real formula for the function it induces, we first build a tree-like cutting planes proof of small size of the composed pebbling formula.

Theorem 5.5. *If $\mathcal{C}$ is the pebbling formula of a graph of indegree 2, then there is a tree-like semantic cutting planes refutation of $\mathcal{C} \circ \mathrm{EQ}_{\log\log n}$ in length $O(n \log n \log\log n)$.*

It is not hard to see that we can extract an efficient locally real protocol from a tree-like cutting planes refutation of small size, but let us record this fact formally.

Lemma 5.6 (Folklore, see [Sok17]). *Given a semantic cutting planes refutation of a formula F, there is a locally real protocol for $\mathsf{Search}(F)$ with the same underlying graph.*

Before we move into the proof of Theorem 5.5, let us complete the proof of Theorem 5.1.

Proof of Theorem 5.1. Let $S = \mathsf{Search}(\mathrm{Peb}_G \circ \mathrm{EQ}_{\log\log n})$ be the search problem given by Lemma 4.7. The nondeterministic communication complexity of f is $\log(|\mathrm{Peb}_G \circ \mathrm{EQ}_{\log\log n}|) + 2$, since given a certificate consisting of a clause falsified by the inputs each party can independently verify that their part is falsified and communicate so to the other party. Therefore by Lemma 5.4 there is a partial monotone function f^* over $O(n \operatorname{polylog} n)$ variables whose monotone Karchmer–Wigderson game is equivalent to S. By Theorem 5.5 there is a semantic cutting planes refutation of the formula $\mathrm{Peb}_G \circ \mathrm{EQ}_{\log\log n}$ of length $O(n \operatorname{polylog} n)$, which we convert into a locally real protocol for S of size $O(n \operatorname{polylog} n)$ using Lemma 5.6, and then into a monotone real formula for f^* of size $O(n \operatorname{polylog} n)$ using Lemma 5.3. Add a threshold gate on top of the formula to ensure that the output is always Boolean and let f be the total function that the formula computes. Since f extends f^*, by Lemma 5.2 and Lemma 4.7 f requires monotone Boolean formulas of depth $\Omega(n/\log n)$, and therefore size $2^{\Omega(n/\log n)}$. $\square$

5.1 A Short Tree-like Refutation

For simplicity in this section we reinterpret the pebbling formula of a graph G of indegree 2 lifted with equality of q bits as the pebbling formula of a graph G' lifted with equality of 1 bit or XNOR, where G' is the graph where we replace every vertex in G by a blob of q vertices and we replace every edge by a bipartite complete graph between blobs, and with the difference that instead of having axioms asserting that all sinks are false, the axioms assert that some sink is false.

Without further ado, let us prove Theorem 5.5, which follows by setting $q = \log\log n$ in the following Lemma.

Lemma 5.7. *If $\mathcal{C}$ is the pebbling formula of a graph of indegree 2, then there is a tree-like semantic cutting planes refutation of $\mathcal{C} \circ \mathrm{EQ}_q$ in length $O(nq2^q)$.*

As in Section 4.3 we fix a topological order of G and we build a refutation by keeping two inequalities $L(W) \geq 0$ and $-L(W) \geq 0$. The main difference is that we cannot use the Coding Lemma to isolate the value of a single vertex, since then we would lose the information on the rest of vertices, therefore we have to simulate the inference steps in place as we describe next.

Let us set up some notation. If W is a set of vertices, let $g(W) = \bigwedge_{v\in W} \mathrm{XNOR}(v)$. We represent $\mathrm{XNOR}(v)$ with $L(v) = 0$, where $L(v) = x_v - y_v$, and $g(W)$ with $L(W) = \sum_{v_j\in W} 2^j L(v_j)$. We begin with $W = \emptyset$ and with the trivial inequalities $0 = L(\emptyset) = 0$. Let us show how to derive each vertex.

Lemma 5.8. *There is a tree-like semantic derivation of $L(W \cup \{w\}) \geq 0$ from $L(W) \geq 0$ and the axioms in 2^q steps.*

Proof. If w is a source, then the inequality $L(w) \geq 0$ is already an axiom, hence it is enough to multiply $L(W) \geq 0$ by 2 and add $L(w)$.

The complex case is when w has predecessors $u_1, \ldots, u_q$. Let $\ell(v, b) = b + (-1)^b v$ be the literal over variable v and polarity $1 - b$. Consider the 2^q axioms $I_b \geq 0$ indexed by $b \in \{0,1\}^q$ and defined as

$$J_b = \sum_{j=1}^{q} \ell(x_{u_j}, b_j) + \ell(y_{u_j}, b_j) \tag{5.1}$$

$$I_b = L(w) + J_b \ . \tag{5.2}$$

We start with an inequality $L(W) \geq 0$. In order to have enough working space for the axioms we multiply the inequality by 2^q, and using weakening axioms we add a slack term defined as

$$S = \sum_{j=1}^{q} 2^{j-1} x_{u_j} \tag{5.3}$$

to obtain $L_0^+ \geq 0$ with

$$L_0^+ = 2^q L(W) + S \ . \tag{5.4}$$

The coefficients for S are chosen so that if we evaluate S on a string $b \in \{0,1\}^q$, the result is b interpreted as a binary number. We use it to keep track of which axioms we have processed so far, in a similar fashion to how the space-efficient refutation of the complete tautology [GPT15] that we reproduce in Appendix B keeps track of processed truth value assignments.

The next step is to add each axiom to L_0^+, but for this to work we need to represent each intermediate step with one inequality as follows.

Claim 5.9. We can represent the Boolean expression

$$g_B^+ = [\![L(W) \geq 0]\!] \wedge \left(g(W) \rightarrow \bigwedge_{b \leq B} I_b \right) \tag{5.5}$$

with the inequality $L_B^+ \geq 0$ defined as

$$L_B^+ = (B+1)L(w) + L_0^+ \ . \tag{5.6}$$

Proof. Let us begin proving the claim by showing that $g_B^+ \Rightarrow L_B^+ \geq 0$. First consider an assignment α that satisfies $L(W) \geq 0$ but not $g(W)$, that is an assignment where $x_{u_j} = 1$ and $y_{u_j} = 0$ for some predecessor u_j of w. Then $L_0^+\restriction_\alpha \geq 2^q$, hence $L_B^+\restriction_\alpha \geq -(B+1) + 2^q \geq 0$.

Now consider an assignment α that satisfies $g(W)$, hence $L(W) = 0$. If $\alpha_{u_1,\ldots,u_q} = b \leq B$ then, since α falsifies $J_b \geq 1$, α must satisfy $L(w) \geq 0$, so both $L_0^+ \geq 0$ and $L(w) \geq 0$. Otherwise if $\alpha_{u_1,\ldots,u_q} = b > B$ then $S\restriction_\alpha = b \geq B+1$, and we have $L_B^+\restriction_\alpha \geq -(B+1) + b \geq 0$.

Let us finish by showing that $g_B^+ \Leftarrow L_B^+ \geq 0$. First consider an assignment α that falsifies $L(W) \geq 0$. Then $L_B^+\restriction_\alpha \leq (B+1) - 2^q < 0$.

Now consider an assignment α that satisfies $g(W)$ but not an axiom I_b with $b \leq B$. Then in particular α falsifies $L(w) \geq 0$, hence $L_B^+\restriction_\alpha = (B+1)L(w) + S\restriction_\alpha = -(B+1) - b < 0$. This concludes the proof of the claim. ☐

Since $L_{B+1}^+ \geq 0$ follows semantically from $L_B^+ \geq 0$ and I_B, we can derive $g_{2^q}^+ \geq 0$ from $L_0^+ \geq 0$ and the set of axioms $I_b \geq 0$ using 2^q semantic inferences of arity 2. Also, $L_{2^q}^+ \geq 0$ is semantically (but not syntactically) equivalent to $L(W \cup \{w\}) \geq 0$, so we can be ready for the next step with a semantic inference of arity 1. ☐

We can derive the upper bound inequality $-L(W \cup \{w\}) \geq 0$ similarly, the main differences being that we start with $-L(W) \geq 0$ and that we use the other half of the axioms, that is $I_b = -L(w) + J_b$.

We handle the sinks in a slightly different way. Instead of using the pebbling axioms directly, we first use the pebbling axioms of all the sinks together with the axioms enforcing that some sink is false in order to derive a set of inequalities similar to pebbling axioms but with -1 in place of $L(w)$. We then use the same derivation as in Lemma 5.8 using these inequalities in place of the axioms and we obtain $L(W) - 1 \geq 0$ and analogously $-L(W) - 1 \geq 0$. Adding both inequalities leads to the contradiction $-2 \geq 0$.

To conclude the proof it is enough to observe that we do $O(2^q)$ inference steps for each vertex in G', which has order nq, hence the total length of the refutation is $O(nq2^q)$.

6 Concluding Remarks

In this paper, we show that the cutting planes proof system (CP) is stronger than its variant with polynomially bounded coefficients (CP*) with respect to simultaneous length and space. This is the first result in proof complexity demonstrating any situation where high-weight coefficients are more powerful than low-weight coefficients. We also prove an explicit separation between monotone Boolean formulas and monotone real formulas. Previously the result was only known to hold non-constructively. To obtain these results we strengthen a lifting theorem of [PR18] to allow the lifting to work with *any* gadget with sufficiently large rank, in particular with the equality gadget—a crucial ingredient for obtaining the separations discussed above.

This work raises a number of questions. Prior to our result, no explicit function was known separating monotone real circuits or formulas from monotone Boolean circuits or formula. Although we prove an explicit formula separation, it remains open to obtain an explicit function that separates monotone real circuits from monotone Boolean circuits.

The most glaring open problem related to our cutting planes contribution is to strengthen our result to a true length separation, without any assumption on the space complexity. It is natural to ask whether techniques inspired by [Sok17, GGKS18] can be of use. Another thing to note about our trade-off result for CP* is that it is not a "true trade-off": we know that length and space cannot be optimised simultaneously, but we do not know if there in fact exist small space refutations. An interesting problem is, therefore, to exhibit formulas that present "true trade-offs" for CP* but are easy with regard to space and length in CP.

It follows from our results that standard decision tree complexity, parity decision tree complexity, and Nullstellensatz degree are equal for the falsified clause search problem of lifted pebbling formulas. In view of this we can ask ourselves what complexity measure we are actually lifting. We know that for general search problem decision tree complexity is not enough for a lifting result. How about parity decision tree complexity? Or can we leverage the fact that we have "well-behaved" rectangle covers and small certificate complexity to lift weaker complexity models? It would be valuable to have a better understanding of the relation between gadgets, outer functions/relations and complexity measures.

Acknowledgements

Different subsets of the authors would like to acknowledge fruitful and enlightening conversations with different subsets of Arkadev Chattopadhyay, Pavel Hrubeš, Christian Ikenmeyer, Bruno Loff, Sagnik Mukhopadhyay, Igor Carboni Oliveira, Pavel Pudlák, and Dmitry Sokolov. We are also grateful for discussions regarding literature references with Albert Atserias, Paul Beame, and Massimo Lauria.

We are grateful to anonymous referees for their comments; in particular, indicating a simplified proof of Lemma 3.5.

Part of this work was carried out while several of the authors were visiting the Simons Institute for the Theory of Computing in association with the DIMACS/Simons Collaboration on Lower Bounds in Computational Complexity, which is conducted with support from the National Science Foundation.

Or Meir was supported by the Israel Science Foundation (grant No. 1445/16). Toniann Pitassi and Robert Robere were supported by NSERC. Susanna F. de Rezende and Jakob Nordström were supported by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611, as well as by the Knut and Alice Wallenberg grant KAW 2016.0066. Jakob Nordström also received funding from the Swedish Research Council grants 621-2012-5645 and 2016-00782. Marc Vinyals was supported by the Prof. R Narasimhan post-doctoral award.

A Lifting Nullstellensatz Degree for All Gadgets

In this section we prove Theorem 3.1. In fact, we prove the following stronger result, which implies Theorem 3.1 as a corollary.

Theorem A.1. *Let $\mathcal{C}$ be an unsatisfiable k-CNF on n variables and let $\mathbb{F}$ be any field. Let g be any Boolean-valued gadget with* $\mathrm{rank}(g) \geq 4$. *Then*

$$\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C}) \circ g^n) \geq \mathsf{NS}_{\mathbb{F}}(\mathcal{C}) \log \left(\frac{\mathsf{NS}_{\mathbb{F}}(\mathcal{C}) \mathrm{rank}(g)}{en} \right) - \frac{6n \log e}{\mathrm{rank}(g)} - \log k.$$

The proof of the theorem follows the proof of a similar lifting theorem from [PR18]. As such, we will need some notation from that paper. Let us begin by introducing a key notion: Razborov's *rank measure*. Given sets $\mathcal{U}, \mathcal{V}$, a *rectangle cover* $\mathcal{R}$ of $\mathcal{U} \times \mathcal{V}$ is a covering of $\mathcal{U} \times \mathcal{V}$ by combinatorial rectangles.

Definition A.2. Let $\mathcal{U}, \mathcal{V}$ be sets and let $\mathcal{R}$ be any rectangle cover of $\mathcal{U} \times \mathcal{V}$. Let A be any $\mathcal{U} \times \mathcal{V}$ matrix over a field $\mathbb{F}$. The *rank measure* of $\mathcal{R}$ at A is the quantity

$$\mu_{\mathbb{F}}(\mathcal{R}, A) = \frac{\operatorname{rank}(A)}{\max\limits_{R \in \mathcal{R}} \operatorname{rank}(A \upharpoonright R)}.$$

Using the rank measure we can lower bound the deterministic communication complexity of composed CNF search problems as follows. The key observation is that any deterministic communication protocol outputs a rectangles that lie in a "structured" rectangle cover in the following sense. We note below that if A is a collection of tuples from some product set $\mathcal{I}^n$ then we write A_i to mean the projection of A to the ith coordinate and A_I for $I \subseteq [n]$ to mean the projection onto the coordinates in I.

Definition A.3. Let $\mathcal{C}$ be an unsatisfiable k-CNF on n variables and let $g : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ be a gadget. For a clause $C \in \mathcal{C}$, a combinatorial rectangle $R \subseteq \mathcal{X}^n \times \mathcal{Y}^n$ is C-*structured* if $g^n(x, y)$ falsifies C for all $(x, y) \in R$ and for all $i \notin \mathit{Vars}(C)$ we have $R_i = \mathcal{X} \times \mathcal{Y}$. A rectangle cover $\mathcal{R}$ of $\mathcal{X}^n \times \mathcal{Y}^n$ is $\mathcal{C}$-*structured* if every $R \in \mathcal{R}$ is C-structured for some $C \in \mathcal{C}$.

Lemma A.4. *Let $\mathcal{C}$ be an unsatisfiable k-CNF on n variables and let $g : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ be a gadget. Let $\mathbb{F}$ be any field and let A be any $\mathcal{X}^n \times \mathcal{Y}^n$ matrix over $\mathbb{F}$. Then*

$$\mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C}) \circ g) \geq \min_{\mathcal{R}} \log \mu_{\mathbb{F}}(\mathcal{R}, A)$$

where the minimum is taken over $\mathcal{C}$-structured rectangle covers of $\mathcal{X}^n \times \mathcal{Y}^n$.

Proof. Let Π be any communication protocol solving $\mathsf{Search}(\mathcal{C}) \circ g$, and let $\mathcal{T}$ be the monochromatic rectangle partition corresponding to Π. Since Π solves the search problem, for every rectangle $R \in \mathcal{T}$ there is a clause C such that for all $(x, y) \in R$, $g^n(x, y)$ falsifies C. We can write $R = A \times B$ for some sets $A \subseteq \mathcal{X}^{\mathit{Vars}(C)} \times \mathcal{X}^{[n] \setminus \mathit{Vars}(C)}$ and $B \subseteq \mathcal{Y}^{\mathit{Vars}(C)} \times \mathcal{Y}^{[n] \setminus \mathit{Vars}(C)}$. Consider $R' = A' \times B'$ where $A' = A_{\mathit{Vars}(C)} \times \mathcal{X}^{[n] \setminus \mathit{Vars}(C)}$ and $B' = B_{\mathit{Vars}(C)} \times \mathcal{X}^{[n] \setminus \mathit{Vars}(C)}$. Since C only depends on indices in $\mathit{Vars}(C)$ we have that $g^n(x, y)$ falsifies C for all $(x, y) \in R'$ and, moreover, $R'_i = \mathcal{X} \times \mathcal{Y}$ for all $i \notin \mathit{Vars}(C)$. It follows that R' is C-structured. Let $\mathcal{R}$ be the $\mathcal{C}$-structured rectangle covering obtained from $\mathcal{T}$ by relaxing all rectangles of $\mathcal{T}$ in this way.

We now have an $\mathcal{C}$-structured rectangle cover $\mathcal{R}$ such that every $T \in \mathcal{T}$ is contained in some rectangle of $\mathcal{R}$. Razborov [Raz90] proved that if $\mathcal{T}$ is a rectangle partition and $\mathcal{R}$ is a rectangle cover such that for each $T \in \mathcal{T}$ there is an $R \in \mathcal{R}$ such that $T \subseteq R$ it holds that

$$|\mathcal{T}| \geq \mu_{\mathbb{F}}(\mathcal{R}, A)$$

for any matrix A. Since $\log |\mathcal{T}| \leq |\Pi| = \mathsf{P}^{\mathsf{cc}}(\mathsf{Search}(\mathcal{C}) \circ g)$ the lemma follows. $\square$

We now introduce the notion of a *certificate* of an unsatisfiable CNF formula.

Definition A.5. Let $\mathcal{C}$ be an unsatisfiable Boolean formula on n variables in conjunctive normal form, and let C be a clause in $\mathcal{C}$. The *certificate* of C, denoted $\operatorname{Cert}(C)$, is the partial assignment $\pi : [n] \to \{0, 1, *\}$ which falsifies C and sets the maximal number of variables to $*$s. Let $\operatorname{Cert}(\mathcal{C})$ denote the set of certificates of clauses of $\mathcal{C}$.

We say that an assignment $z \in \{0, 1\}^n$ *agrees* with a certificate $\pi \in \operatorname{Cert}(\mathcal{C})$ if $\pi(i) = z_i$ for each i assigned to a $\{0, 1\}$ value by π. Since the CNF formula $\mathcal{C}$ is unsatisfiable, it follows that every assignment in $z \in \{0, 1\}^n$ agrees with some $\{0, 1\}$-certificate of $\mathcal{C}$. Next we introduce an alternative definition of Nullstellensatz degree called the *algebraic gap complexity*.

Definition A.6. Let $\mathbb{F}$ be a field. Let $\mathcal{C}$ be an unsatisfiable CNF on n variables. The $\mathbb{F}$-*algebraic gap complexity* of $\mathcal{C}$ is the maximum positive integer $\operatorname{gap}_{\mathbb{F}}(\mathcal{C}) \in \mathbb{N}$ for which there exists a multilinear polynomial $p \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ such that

$$\deg(p) = n \quad \text{and} \quad \forall \pi \in \operatorname{Cert}(\mathcal{C}) : \deg(p \upharpoonright \pi) \leq n - \operatorname{gap}_{\mathbb{F}}(\mathcal{C}) \ .$$

When the field is clear from context we will write $\operatorname{gap}(\mathcal{C})$.

In [PR18, Rob18] it was shown that the algebraic gap complexity is equal to Nullstellensatz degree.

Theorem A.7. *For any unsatisfiable CNF formula $\mathcal{C}$ on n variables and any field $\mathbb{F}$, $\mathrm{gap}_{\mathbb{F}}(\mathcal{C}) = \mathsf{NS}_{\mathbb{F}}(\mathcal{C})$.*

We now prove a lifting theorem from Nullstellensatz degree to the rank measure, from which Theorem A.1 follows by applying Lemma A.4.

Theorem A.8. *Let $\mathcal{C}$ be an unsatisfiable k-CNF on n variables and let $\mathbb{F}$ be any field. Let g be any Boolean-valued gadget with $\mathrm{rank}(g) \geq 4$. There is a matrix A such that for any $\mathcal{C}$-structured rectangle cover $\mathcal{R}$ we have*

$$\mu_{\mathbb{F}}(\mathcal{R}, A) \geq \frac{1}{k}\left(\frac{\mathsf{NS}_{\mathbb{F}}(\mathcal{C})\mathrm{rank}(g)}{en}\right)^{\mathsf{NS}_{\mathbb{F}}(\mathcal{C})} \exp(-6n/\mathrm{rank}(g)) \ .$$

Proof. Let $p \in \mathbb{F}[z_1, z_2, \ldots, z_n]$ be the polynomial witnessing the algebraic gap complexity $\mathrm{gap}(\mathcal{C})$, and let $A = p \circ g^n$ be the pattern matrix obtained by composing p and g. We need to analyze

$$\mu_{\mathbb{F}}(\mathcal{R}, p \circ g^n) = \frac{\mathrm{rank}_{\mathbb{F}}(p \circ g^n)}{\max_{R \in \mathcal{R}} \mathrm{rank}_{\mathbb{F}}(p \circ g^n \restriction R)} \ .$$

Let us first analyze the denominator. Let R be an arbitrary rectangle from the cover $\mathcal{R}$, and suppose that R is C-structured for the clause $C \in \mathcal{C}$. Let $\pi = \mathrm{Cert}(C)$. We want to show that

$$\mathrm{rank}_{\mathbb{F}}(p \circ g^n \restriction R) \leq \sum_{S : \widehat{p \restriction \pi}(S) \neq 0} \mathrm{rank}(g)^{|S|} \ . \tag{A.1}$$

To prove this, we claim that $p \circ g^n \restriction R$ is column-equivalent to the block matrix

$$[(p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}, (p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}, \ldots, (p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}]$$

for some number of copies of the matrix $(p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}$. Indeed Equation A.1 immediately follows from this claim as

$$\mathrm{rank}_{\mathbb{F}}(p \circ g^n \restriction R) = \mathrm{rank}_{\mathbb{F}}((p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}) \leq \sum_{S : \widehat{p \restriction \pi}(S) \neq 0} \mathrm{rank}(g)^{|S|}$$

by Theorem 3.4. So, we now prove the claim.

Write $R = A \times B$. Fix assignments $\alpha \in A_{\mathit{Vars}(C)}$ and $\beta \in B_{\mathit{Vars}(C)}$, and note that since R is C-structured we have that $g^{\mathit{Vars}(C)}(\alpha, \beta) = \pi$ and $(\alpha, x') \in A$ and $(\beta, y') \in B$ for all x', y'. Thus, by ranging $x_{[n] \setminus \mathit{Vars}(C)}, y_{[n] \setminus \mathit{Vars}(C)}$ over all values yields the matrix $(p \restriction \pi) \circ g^{[n] \setminus \mathit{Vars}(C)}$. Then, ranging $x_{\mathit{Vars}(C)}$ and $y_{\mathit{Vars}(C)}$ over all α, β such that $g^{\mathit{Vars}(C)}(\alpha, \beta) = \pi$ yields the claim and Equation A.1.

Now, consider the rank measure $\mu_{\mathbb{F}}(\mathcal{R})$, which by Theorem 3.4 and Equation A.1 satisfies

$$\mu_{\mathbb{F}}(\mathcal{R}) \geq \frac{\mathrm{rank}_{\mathbb{F}}(p \circ g^n)}{\max_{R \in \mathcal{R}} \mathrm{rank}_{\mathbb{F}}(p \circ g^n \restriction R)} = \frac{\sum_{S : \hat{p}(S) \neq 0} (\mathrm{rank}(g) - 3)^{|S|}}{\max_{\pi \in \mathrm{Cert}(\mathcal{C})} \sum_{S : \widehat{p \restriction \pi}(S) \neq 0} \mathrm{rank}(g)^{|S|}}$$

By definition of $\mathrm{gap}(\mathcal{C})$ we have $\deg p = n$ and thus the numerator is at least $(\mathrm{rank}(g) - 3)^n$. For the denominator, since p witnesses the algebraic gap of $\mathcal{C}$, we have that $\deg p \restriction \pi \leq n - \mathrm{gap}(\mathcal{C})$ for all $\pi \in \mathrm{Cert}(\mathcal{C})$. We may assume that $\hat{p}(S) = 0$ when $|S| < n - \mathrm{gap}(\mathcal{C})$ as the definition of algebraic gaps depends only on the coefficients of monomials of p with degree larger than $n - \mathrm{gap}(\mathcal{C})$. So, for any restriction π:

$$\begin{aligned}
\sum_{S : \widehat{p \restriction \pi}(S) \neq 0} \mathrm{rank}(g)^{|S|} &\leq \sum_{i=0}^{k} \binom{n}{\mathrm{gap}(\mathcal{C}) - i} \mathrm{rank}(g)^{n - \mathrm{gap}(\mathcal{C}) - i} \\
&\leq k \left(\frac{en}{\mathrm{gap}(\mathcal{C})}\right)^{\mathrm{gap}(\mathcal{C})} \mathrm{rank}(g)^{n - \mathrm{gap}(\mathcal{C})}
\end{aligned}$$

Putting it all together, and using the fact that $\mathrm{rank}(g) \geq 6en/\mathrm{gap}(\mathcal{C})$, we have

$$\begin{aligned}\mu_{\mathbb{F}}(\mathcal{R}, p \circ g^n) &\geq \frac{(\mathrm{rank}(g)-3)^n}{k(en/\mathrm{gap}(\mathcal{C}))^{\mathrm{gap}(\mathcal{C})}\mathrm{rank}(g)^{n-\mathrm{gap}(\mathcal{C})}} \\ &= \frac{1}{k}\left(\frac{\mathrm{gap}(\mathcal{C})\mathrm{rank}(g)}{en}\right)^{\mathrm{gap}(\mathcal{C})} \cdot \left(1 - \frac{3}{\mathrm{rank}(g)}\right)^n \\ &\geq \frac{1}{k}\left(\frac{\mathrm{gap}(\mathcal{C})\mathrm{rank}(g)}{en}\right)^{\mathrm{gap}(\mathcal{C})} \exp(-6n/\mathrm{rank}(g)) \enspace .\end{aligned}$$

Since $\mathrm{gap}(\mathcal{C}) = \mathsf{NS}(\mathcal{C})$ the theorem is proved. □

Theorem A.1 follows immediately from Theorem A.8 and Lemma A.4.

B Proof of the Space Lemma

In this section we prove the Space Lemma (Lemma 4.9), restated next.

Lemma B.1. *Let $\mathcal{C}$ be a set of inequalities over n variables that implies a clause C. Then there is a cutting planes derivation of C from $\mathcal{C}$ in length $O(n^2 2^n)$ and space $O(1)$.*

We do so by adapting the proof in [GPT15] that any formula has a cutting planes refutation in constant space in order to show that, in fact, we can derive any clause that follows from a set of inequalities in constant space.

At a bird's eye view, the proof in [GPT15] has two steps. The primary step is building a refutation of the complete tautology, the formula that contains all 2^n clauses with n variables each forbiding one of the possible 2^n assignments, in constant space. The authors come up with an order and a way to encode that the first K clauses are all true in small space for an arbitrary K, and the rest of the primary step consists of showing how to operate with this encoding also in small space, starting with no clause being true and adding clauses one by one until a contradiction arises. The secondary step is to transform the original set of linear inequalities into the complete tautology.

If we do not start with an unsatisfiable set of linear inequalities we obviously cannot reach a contradiction, but given a clause C that follows from $\mathcal{C}$ we can still encode that all the clauses that are a superset of C must be true, and this expression is equivalent to C.

Let us set up some notation. We number the variables from 0 to $n-1$. If α is a total assignment, we denote by C_α the clause over n variables that is falsified exactly by α. We overload notation and also denote by C_α the standard translation of the clause C_α into an inequality. We say that an assignment is less than a natural number B and write $\alpha < B$ if α is lexicographically smaller than the binary representation of B, that is if $\sum_{i=0}^{n-1} 2^i \alpha(x_i) < B$. We write T_B to denote the inequality $\sum_{i=0}^{n-1} 2^i x_i \geq B$ that is falsified exactly by the assignments $\{\alpha \in \{0,1\}^n \mid \alpha < B\}$.

We can reuse the following two intermediate lemmas from [GPT15], corresponding to the primary and the secondary steps.

Lemma B.2 ([GPT15]). *There is a cutting planes derivation of T_B from the set of clauses $\{C_\alpha \mid \alpha < B\}$ in length $O(nB)$ and space $O(1)$.*

Lemma B.3 ([GPT15]). *If a total assignment α falsifies a set of inequalities $\mathcal{C}$, then there is a cutting planes derivation of C_α from $\mathcal{C}$ in length $O(n)$ and space $O(1)$.*

Lemma B.2, which contains the core of the argument, follows from the proof of Lemma 3.2 in [GPT15]. We repeat Claim 3 in that proof, which shows how to inductively derive $T_{B'+1}$ from $T_{B'}$ and $C_{B'}$, not 2^n but B times.

In turn Lemma B.3 follows from the proof of Theorem 3.4 in [GPT15]: since α must falsify some inequality I from $\mathcal{C}$, we only need to reproduce the derivation of C_α from I verbatim.

Proof of Lemma 4.9. Assume for now that $C = x_{n-1} \vee \dots \vee x_{n-k}$. Consider the derivation Π of the inequality $T_{2^{n-k}}$ from $\{C_\alpha \mid \alpha < B\}$, which is equivalent to C, given by Lemma B.2. We build a new derivation Π' extending Π as follows.

Every time that we add an axiom C_α to a configuration in Π, we replace that step by the derivation of C_α from $\mathcal{C}$ given by Lemma B.3. Observe that we only add axioms C_α with $\alpha < 2^{n-k}$, and since any such assignment falsifies $\mathcal{C}$ we meet the conditions to apply Lemma B.3.

Finally we obtain C from $T_{2^{n-k}}$ by considering $\{T_{2^{n-k}}\}$ as a set of inequalities over the k variables $x_{n-1} \dots x_{n-k}$ and applying Lemma B.3 with $\alpha = 0^k$ being the only assignment over these variables that falsifies $T_{2^{n-k}}$. The result is $C_0 = C$.

To derive a general clause C that contains k' negative literals, say $C = \neg x_{n-1} \vee \dots \vee \neg x_{n-k'} \vee x_{n-k'-1} \vee \dots \vee x_{n-k}$, we build a derivation with the same structure as Π', except that we replace every occurrence of x_i by $(1 - x_i)$ for $n - k' \leq i < n$. To do so, we replace each derivation of an axiom C_α with a derivation of the axiom $C_{\alpha+(0^{n-k'}1^{k'})}$ and, for $n - k' \leq i < n$, replace each use of $x_i \geq 0$ and $-x_i \geq -1$ by $-x_i \geq -1$ and $x_i \geq 0$, respectively. Linear combination and division steps go through unchanged, we only observe that at a division step the coefficient on the right hand side differs by a multiple of the divisor, so rounding is not affected. ☐

C Proof of Corollary 1.5

In this appendix, we provide the proof of Corollary 1.5, restated next.

Corollary C.1 (1.5, restated). *For any field $\mathbb{F}$ and any directed acyclic graph G, the Nullstellensatz degree over $\mathbb{F}$ of Peb_G, the decision tree depth of $\mathsf{Search}(\mathrm{Peb}_G)$, and the parity decision tree depth of $\mathsf{Search}(\mathrm{Peb}_G)$ coincide and are equal to the reversible pebbling price of G.*

Our proof uses Lemma 4.2, restated next.

Lemma C.2 (4.2, restated). *For any field $\mathbb{F}$ and any graph G, $\mathsf{NS}_{\mathbb{F}}(\mathrm{Peb}_G) = \mathrm{rpeb}(G)$.*

Let $\mathbb{F}_2$ be the finite field of two elements. Given a search problem $\mathcal{S}$, we denote the (deterministic) decision tree depth and parity decision tree depth of $\mathcal{S}$ by $\mathrm{DT}(\mathcal{S})$ and $\mathrm{PDT}(\mathcal{S})$ respectively. We use the following lemma, which will be proved below.

Lemma C.3 (Folklore). *Let $\mathcal{C}$ be an unsatisfiable CNF over n variables. Then, $\mathsf{NS}_{\mathbb{F}_2}(\mathcal{C}) \leq \mathrm{PDT}(\mathsf{Search}(\mathcal{C}))$.*

Proof of Corollary 1.5 from Lemmas 4.2 and C.3. Let G be a directed acyclic graph. Note that it suffices to prove the corollary for $\mathbb{F} = \mathbb{F}_2$, since Lemma 4.2 implies that $\mathsf{NS}_{\mathbb{F}}(\mathrm{Peb}_G)$ is the same for every finite field $\mathbb{F}$. The corollary follows immediately from the following chain of inequalities:

$$\mathrm{PDT}(\mathsf{Search}(\mathrm{Peb}_G)) \leq \mathrm{DT}(\mathsf{Search}(\mathrm{Peb}_G)) = \mathrm{rpeb}(G) = \mathsf{NS}_{\mathbb{F}_2}(\mathrm{Peb}_G) \leq \mathrm{PDT}(\mathsf{Search}(\mathrm{Peb}_G)).$$

The first inequality is obvious, and the first equality was proved in the work of Chan [Cha13], but for completeness we provide a simplified proof in Appendix D. The second equality follows from Lemma 4.2, and the last inequality follows from Lemma C.3. Thus, the corollary is proved. ☐

In the remainder of this appendix, we prove Lemma C.3. Let $\mathcal{C}$ be an unsatisfiable CNF over variables $z_1, \dots, z_n$, and let T be a parity decision tree of depth d that solves $\mathsf{Search}(\mathcal{C})$. We prove that there exists a Nullstellensatz refutation over $\mathbb{F}_2$ for $\mathcal{C}$ of degree at most d, and this will imply the required result.

Recall that the parity decision tree T takes as input an assignment α to $z_1, \dots, z_n$, queries at most d parities of α, and then outputs a clause C of $\mathcal{C}$ that is violated by α. More formally, every internal node v of T is associated with some linear polynomial p_v in $z_1, \dots, z_n$, and each outgoing edge e of v is associated with bit $b_e \in \{0, 1\}$ (so the edge e is taken if $p_v(\alpha) = b_e$). Every leaf ℓ of T is associated with a clause $C_\ell \in \mathcal{C}$, such that every assignment α that leads T to ℓ violates the clasue C_ℓ.

We construct for each leaf ℓ of T a polynomial $r_\ell(z_1, \dots, z_n)$ of degree at most d that output 1 on an assignment α if the tree T reaches the leaf ℓ when invoked on α, and outputs 0 otherwise. We will use those polynomials later to construct the Nullstellensatz refutation of $\mathcal{C}$. First, for every internal vertex v and an outgoing edge e of v, we associate with e the linear polynomial

$$r_e(z_1, \ldots, z_n) = p_v(z_1, \ldots, z_n) + b_e + 1.$$

Intuitively, the polynomial r_e output 1 on an assignment α if the query of v outputs b_e on α, and 0 otherwise. Now, to construct the polynomial r_ℓ of a leaf ℓ, we multiply the polynomials r_e for all the edges e on the path from the root to ℓ. Since there are at most d edges on that path, it follows that r_ℓ is of degree at most d. Moreover, it is not hard to see that $r_\ell(\alpha) = 1$ if α leads the tree T to the leaf ℓ, and $r_\ell(\alpha) = 0$ otherwise.

Let us denote by r'_ℓ the "multilinearized" version of r_ℓ, that is, r'_ℓ is the polynomial obtained from r_ℓ by reducing the degree of every variable to 1 in each of its occurences in r_ℓ. It is not hard to see that r'_ℓ agrees with r_ℓ on every assignment in $\{0,1\}^n$. Our Nullstellensatz refutation of $\mathcal{C}$ is the polynomial

$$r = \sum_{\text{leaf } \ell \text{ of } T} r'_\ell.$$

Clearly, this polynomial is of degree at most d. In order to prove that the polynomial r is a valid Nullstellensatz refutation of $\mathcal{C}$, we need to prove that r equals 1, and that it can be derived from the axioms of $\mathcal{C}$ (in other words, that r belongs to ideal generated by $\mathcal{E}(\mathcal{C}) \cup \left\{z_i^2 - z_i\right\}_{i \in [n]}$). We start by proving that r equals 1.

Claim C.4. $r = 1$.

Proof. We use the well-known fact that a multilinear polynomial is determined by its values on $\{0,1\}^n$: one way to see it is to observe that the multilinear monomials are a basis of the space of functions from $\{0,1\}^n$ to $\{0,1\}$, and therefore every such function has a unique representation as a multilinear polynomial. Since r is multilinear, it therefore suffices to prove that r outputs 1 on every assignment in $\{0,1\}^n$.

Let $\alpha \in \{0,1\}^n$ be an assignment to $z_1, \ldots, z_n$. Let ℓ be the leaf that T reaches when invoked on α. Then, by the construction of r'_ℓ, it holds that $r'_\ell(\alpha) = 1$ and that $r'_{\ell'}(\alpha) = 0$ for every other leaf ℓ' of T. It follows that $r(\alpha) = 1$, as required. □

It remains to show that r can be derived from the axioms of $\mathcal{C}$. To this end, we will show that each of the polynomials r'_ℓ can be derived from those axioms . It is not hard to show that for every leaf ℓ, the polynomial $r_\ell - r'_\ell$ can be derived from the boolean axioms $\left\{z_i^2 - z_i\right\}_{i \in [n]}$ (in fact, this holds for any difference of a polynomial and its multilinearized version). Thus, it suffices to prove that for every leaf ℓ, the polynomial r_ℓ can be derived from the axioms of $\mathcal{C}$. We prove a stronger statement, namely, that for every leaf ℓ, the polynomial r_ℓ is divisible by the polynomial $\mathcal{E}(C_\ell)$ (i.e., the polynomial encoding of the clause C_ℓ). To this end, we prove the following result.

Claim C.5. Let $p(z_1, \ldots, z_n)$ be a multilinear polynomial over $\mathbb{F}_2$, and let $i \in [n]$. If p vanishes whenever $z_i = 0$, then z_i divides p. Moreover, if p vanishes whenever $z_i = 1$, then $(1 - z_i)$ divides p.

Proof. We can write $p = z_i \cdot a + b$, where a and b are polynomials that do not contain z_i. Suppose first that p vanishes whenever $z_i = 0$. We would like to prove that $b = 0$. Assume that this is not the case. Then, there is an assignment α' to the variables $z_1, \ldots, z_n$ except for z_i on which b does not vanish. Now, if we extend α' to an assignment α to $z_1, \ldots, z_n$ by setting $z_i = 0$, it will follow that α is an assignment on which $z_i = 0$ but p does not vanish, which is a contradiction.

Next, suppose that p vanishes whenever $z_i = 1$. Observe that we can write $p = (1 - z_i) \cdot a + (b - a)$ (here we use the fact that we are working over $\mathbb{F}_2$). We would like to prove that $b - a = 0$. Assume that this is not the case. Then, there is an assignment α' to the variables $z_1, \ldots, z_n$ except for z_i on which $b - a$ does not vanish. As before, if we extend α' to an assignment α to $z_1, \ldots, z_n$ by setting $z_i = 1$, it will follow that α is an assignment on which $z_i = 1$ but p does not vanish, which is a contradiction. □

We turn to prove that for every leaf ℓ, the polynomial r_ℓ is divisible by the polynomial $\mathcal{E}(C_\ell)$. Fix a leaf ℓ, and denote $C = C_\ell$. Recall that we denote by C^+ and C^- the sets of variables that occur positively and negatively in C respectively, and that

$$\mathcal{E}(C) \equiv \prod_{z \in C^+} (1 - z) \prod_{z \in C^-} z.$$

Now, observe that for every variable $z_i \in C^+$, it holds that r_ℓ vanishes whenever $z_i = 1$: to see it, observe that when $z_i = 1$, the assignment does not violate the clause C, and therefore the tree T cannot reach ℓ when invoked on that assignment. Thus, $(1 - z_i)$ divides r_ℓ for every $z_i \in C^+$. Similarly, the variable z_i divides r_ℓ for every $z_i \in C^-$. It follows that r_ℓ is divisible by every factor of $\mathcal{E}(C)$, and therefore it is divisible by $\mathcal{E}(C)$ (here we used the fact that each factor occurs in $\mathcal{E}(C)$ at most once). This concludes the proof.

D Reversible Pebbling is Equivalent to Query Complexity

In this appendix we present a direct proof that the reversible pebbling price of a graph equals the query complexity of the search problem of the pebbling formula of that graph, originally proved by Chan [Cha13].

Theorem D.1 ([Cha13]). *For every DAG G with a single sink, it holds that* $\mathrm{DT}(\mathsf{Search}(\mathrm{Peb}_G)) = \mathrm{rpeb}(G)$.

Let us introduce some notation to talk more formally about pebbling: A *pebbling configuration* is a set of vertices P. A *(reversible) pebbling* is a sequence of configurations $\mathcal{P} = P_1, \ldots, P_\ell$ where P_{i+1} follows from P_i by applying the pebbling rules. Its reverse $R(\mathcal{P}) = P_\ell, \ldots, P_1$ is also a valid pebbling. Its *cost* is the maximal size of a configuration P_i. Unless we call a pebbling *partial*, we assume that $P_1 = \emptyset$. A pebbling *visits* x if $x \in P_\ell$, and *surrounds* x if $\mathrm{pred}(x) \subseteq P_\ell$. The *pebbling price* of a graph G, denoted $\mathrm{rpeb}(G)$, is the minimum cost of all pebblings that visit the sink, and its *surrounding price*, denoted $\mathrm{speb}(G)$, is the minimum cost of all pebblings that surround the sink.

Given a decision tree for $\mathsf{Search}(\mathrm{Peb}_G)$, we associate each node with a state formed by a pair (Q, Z) of queried vertices and the vertices $Z \subseteq Q$ whose queries were answered by 0. It is immediate to verify that at a leaf either the sink z belongs to $Q \setminus Z$, or there is a vertex in Z such that all of its predecessors are in $Q \setminus Z$. It is useful to generalize the definition of the search problem $\mathsf{Search}(\mathrm{Peb}_G)$ to start with intermediate states. Specifically, we associate a state (Q, Z) with the search problem in which we are given an assignment to Peb_G that is promised to assign 0 to the vertices in Z and 1 to the vertices in $Q \setminus Z$, and would like to find a clause that is falsified by this assignment. We denote this search problem by $\mathsf{Search}_G(Q, Z)$ and denote its query complexity by $\mathrm{DT}_G(Q, Z)$. We omit G from the latter notation when it is clear from the context. The crux of our proof is the following lemma, which implies Theorem D.1.

Lemma D.2. *For every DAG G with a single sink z, it holds that* $\mathrm{DT}_G(\{z\}, \{z\}) = \mathrm{speb}(G)$.

We claim that Lemma D.2 implies Theorem D.1. To see why, let G be a DAG with a single sink z, and let G' be the DAG obtained from G by adding a new sink z' and an edge from z to z'. Then, it is not hard to see that $\mathsf{Search}'_G(\{z'\}, \{z'\}) = \mathsf{Search}(\mathrm{Peb}_G))$, and that every pebbling that surrounds the sink of G' is pebbling that visits the sink of G, and vice versa. Hence, it holds that

$$\mathrm{DT}(\mathsf{Search}(\mathrm{Peb}_G)) = \mathrm{DT}_{G'}(\{z'\}, \{z'\}) = \mathrm{speb}(G') = \mathrm{rpeb}(G),$$

where the second equality follows from Lemma D.2. In the rest of this appendix, we focus on proving Lemma D.2. To this end, fix a DAG G with a single sink z.

We first show that the states of an optimal decision tree for $\mathsf{Search}(\{z\}, \{z\})$ are of a special form, which we call "path-like". Specifically, we say that a state (Q, Z) is *path-like* if there is a path P ending at the sink such that $P \cap Q = Z$. Observe that in a path-like state there is a unique such path of maximal length that starts in a vertex in Z. We denote this path by P_Z, and denote its first vertex by $\mathrm{head}(Z)$. We say that a vertex v is *relevant* to a path-like state (Q, Z) if there is a path P_v from v to $\mathrm{head}(Z)$ such that $P_v \cap Q = \{\mathrm{head}(Z)\}$. Observe that starting from a path-like state and querying a relevant vertex yields a path-like state where the new path is $P_Z \cup P_v$ if the answer is 0 and P_Z if it is 1.

We now show that if (Q, Z) is a path-like state, then every optimal decision tree for $\mathsf{Search}(Q, Z)$ only queries relevant vertices. Observe that this implies that all the nodes of the tree correspond to path-like states. Moreover, this result holds in particular for $(\{z\}, \{z\})$, since it is a path-like state.

Lemma D.3. *If (Q, Z) is a path-like state then every optimal decision tree for* $\mathsf{Search}(Q, Z)$ *only queries vertices relevant to (Q, Z).*

Proof. The proof is by induction on the query complexity p of $\mathsf{Search}(Q, Z)$. The base case $p = 0$ holds vacuously: the optimal decision tree does not make any queries. Assume that $p > 0$. Fix an optimal decision tree T for (Q, Z), and let v be the query made by the root of T. Observe that the children of the root of T correspond to the states $(Q \cup v, Z \cup v)$ and $(Q \cup v, Z)$. Let T_0, T_1 be the sub-trees rooted at those children respectively, and note that these trees are optimal for the latter states and that their depth is at most $p - 1$.

Suppose first that v is a relevant query to (Q, Z). In this case, the states of the children of the root are path-like. By the induction assumption, the trees T_0, T_1 only make queries that are relevant to the states $(Q \cup v, Z \cup v)$ and $(Q \cup v, Z)$ respectively. Observe that any such query is necessarily relevant to (Q, Z) as well. Therefore, every query of T is relevant to (Q, Z).

Next, suppose that v is irrelevant to (Q, Z). We consider two separate cases: the case where v belongs to the path P_Z, and the case where it does not belong to P_Z.

v belongs to P_Z. Assume that v belongs to the path P_Z. In this case, the state $(Q \cup v, Z \cup v)$ is path-like, and therefore by the induction assumption the tree T_0 only makes queries that are relevant to that state. We claim that T_0 solves $\mathsf{Search}(Q, Z)$, which contradicts the assumption that T is optimal for $\mathsf{Search}(Q, Z)$.

To see why, suppose for the sake of contradiction that T_0 fails to solve $\mathsf{Search}(Q, Z)$ on some assignment α. Observe that the only case where this can happen is when $\alpha(x_v) = 1$ but T_0 outputs that the violated clause is $C_v = x_v \vee \bigvee_{u \in \mathrm{pred}(v)} \neg x_u$. Let u be the predecessor of v that lies on P_Z (such u must exist since v is queried by T and hence cannot be equal to $\mathrm{head}(Z)$). Then, u cannot belong to Z (or otherwise T_0 could not have output C_v) and therefore it cannot belong to Q (since $P_Z \cap Q = Z$). This means that T_0 must have queried u in order to output C_v. However, u is irrelevant to $(Q \cup v, Z \cup v)$, so we reached a contradiction to the assumption that T_0 does not make irrelevant queries. Hence, T_0 solves $\mathsf{Search}(Q, Z)$, as required.

v does not belong to P_Z. Next, assume that v does not belong to the path P_Z. In this case, the state $(Q \cup v, Z)$ is path-like, and therefore by the induction assumption the tree T_1 only makes queries that are relevant to that state. We claim that T_1 solves $\mathsf{Search}(Q, Z)$, which contradicts the assumption that T is optimal for (Q, Z).

To see why, suppose for the sake of contradiction that T_1 fails to solve $\mathsf{Search}(Q, Z)$ on some assignment α. Observe that the only case where this can happen is when $\alpha(x_v) = 0$ but for some successor w of v the tree T_1 outputs that the violated clause is $C_w = x_w \vee \bigvee_{u \in \mathrm{pred}(w)} \neg x_u$. The query w cannot be relevant to (Q, Z), since otherwise v would have been relevant for (Q, Z). Since w is irrelevant to (Q, Z), it cannot be queried by T_1, and therefore it must belong to Z in order for T_1 to output C_w. Moreover, w cannot be equal to $\mathrm{head}(Z)$, since otherwise v would have been relevant for (Q, Z). Thus, w has a predecessor u in P_Z.

The vertex u cannot belong to Z (or otherwise T_1 could not have output C_w) and therefore it cannot belong to Q (since $P_Z \cap Q = Z$). This means that T_1 must have queried u in order to output C_w. However, u is irrelevant to $(Q \cup v, Z)$, so we reached a contradiction to the assumption that T_1 does not make irrelevant queries. Hence, T_1 solves $\mathsf{Search}(Q, Z)$ □

The following two propositions prove the two directions of Lemma D.2. In the proof of the first proposition we use the following notion: a *pebbling assuming free pebbles on a set S* is a partial pebbling $\mathcal{P}$ such that $P_1 \subseteq S$, and its *cost* is the maximum size of $P_i \setminus S$.

Proposition D.4. *If* $\mathrm{DT}(\{z\},\{z\}) \leq p$, *then* $\mathrm{speb}(G) \leq p$.

Proof. We prove the following stronger claim: if for some path-like state (Q, Z) there is an optimal decision tree T of depth p, then there is a pebbling that surrounds $\mathrm{head}(Z)$ of cost p assuming free pebbles on $Q \setminus Z$. To see that this claim implies the proposition observe that $(\{z\},\{z\})$ is path-like. Therefore, if $\mathrm{DT}(\{z\},\{z\}) \leq p$ then the claim implies that there is a pebbling that surrounds z of cost at most p without free pebbles.

We prove the claim by induction on p. The base case is when $p = 0$, so T consists of a single leaf. This means that there must be some vertex in Z that is surrounded by vertices in $Q \setminus Z$. This vertex must be $\mathrm{head}(Z)$, since any other vertex $v \in Z$ has a predecessor in P_Z, and this predecessor cannot belong to $Q \setminus Z$. Hence, $\{Q \setminus Z\}$ is a surrounding pebbling of $\mathrm{head}(Z)$ of cost 0 assuming free pebbles on $Q \setminus Z$, as required.

We proceed to the induction step. Suppose that $p > 0$. Let v be the query made at the root of T. Let T_0, T_1 be the subtrees rooted at the children of v that corresponds to the states $(Q \cup v, Z \cup v)$ and $(Q \cup v, Z)$ respectively. By Lemma *D.3*, the query v is relevant to (Q, Z), and therefore the latter states are path-like.

Observe that $\mathrm{head}(Z \cup v) = v$. Hence, by applying the induction assumption to T_0, we obtain a pebbling $\mathcal{P}_0$ that surrounds v of cost at most $p - 1$ assuming free pebbles on

$$(Q \cup v) \setminus (Z \cup v) = Q \setminus Z.$$

Furthermore, by applying the induction assumption to T_1, we obtain a pebbling $\mathcal{P}_1$ that surrounds $\mathrm{head}(Z)$ of cost at most $p - 1$ assuming free pebbles on $(Q \cup v) \setminus Z$.

We now construct a pebbling that surrounds $\mathrm{head}(Z)$ assuming free pebbles on $Q \setminus Z$ as follows: we first follow $\mathcal{P}_0$, thus reaching a configuration that surrounds v. Then, we place a pebble on v (unless it is already pebbled). Next, we follow $\{P \cup v \mid P \in R(\mathcal{P}_0)\}$ to remove all the pebbles that were placed by $\mathcal{P}_0$ except for $(Q \cup v) \setminus Z$. Finally, we follow $\mathcal{P}_1$ and reach a configuration that surrounds $\mathrm{head}(Z)$. It is not hard to see that this pebbling has cost at most p assuming free pebbles on $Q \setminus Z$, as required. □

In the proof of the second proposition, we use the following notion: Given a pebbling $\mathcal{P}$, we define $\mathrm{static}(\mathcal{P}) = \cap \mathcal{P}$ to be the set of pebbles that are always present in $\mathcal{P}$, and the *non-static cost* of $\mathcal{P}$ be the maximum size of $P \setminus \mathrm{static}(\mathcal{P})$ for $P \in \mathcal{P}$. We also denote the vertices reachable from v, including v itself, by $\mathrm{desc}(v)$.

Proposition D.5. *If* $\mathrm{speb}(G) \leq p$ *then* $\mathrm{DT}(\{z\},\{z\}) \leq p$.

Proof. We prove the following stronger claim: if for some state (Q, Z) there is a partial pebbling $\mathcal{P}$ that surrounds a vertex $w \in Z$ of non-static cost p with $\mathrm{static}(\mathcal{P}) \subseteq Q \setminus Z$, then there is a decision tree of depth p that solves $\mathsf{Search}(Q, Z)$.

To see that this claim implies the proposition, observe that a pebbling $\mathcal{P}$ that surrounds z of cost p starting from $\emptyset$ has $\mathrm{static}(\mathcal{P}) = \emptyset$. Thus, the claim implies that if such a pebbling exists, it holds that $\mathrm{DT}(\{z\},\{z\}) \leq p$.

If $\mathrm{static}(\mathcal{P})$ surrounds w then all the predecessors of w are in $Q \setminus Z$. Therefore, $\mathsf{Search}(Q, Z)$ can be solved without making any queries: the decision tree can immediately output that the clause $C_w = x_w \vee \bigvee_{u \in \mathrm{pred}(w)} \neg x_u$ is violated.

Otherwise we prove the claim by induction on p. The base case $p = 0$ is a particular instance of $\mathrm{static}(\mathcal{P})$ surrounding w, hence we turn to proving the induction step and suppose that $p > 0$. Having ℓ denote the length of $\mathcal{P}$, let v be the earliest (in time) vertex to be placed at some time $m > 1$ and not removed until time ℓ. Note that v exists because w is not surrounded in some configuration of $\mathcal{P}$.

Let $\mathcal{P}' = P_m, \ldots, P_\ell$ be the subpebbling of $\mathcal{P}$ from time m to time ℓ. Observe that $\mathrm{static}(\mathcal{P}') = \mathrm{static}(\mathcal{P}) \cup \{v\}$ by construction, and thus the non-static cost of $\mathcal{P}'$ is at most $p - 1$. By applying the induction assumption to $\mathcal{P}'$, it follows that there exists a decision tree T_1 for $\mathsf{Search}(Q \cup v, Z)$ of depth $p - 1$.

Next, observe that $R(\mathcal{P}')$ is a partial pebbling that *surrounds* v of non-static cost $p-1$ with $\mathrm{static}(R(\mathcal{P}')) = \mathrm{static}(\mathcal{P}) \cup \{v\}$, therefore $\mathcal{P}'' = \{P \setminus \mathrm{desc}(v) \mid P \in R(\mathcal{P}')\}$ is a partial pebbling that surrounds v of non-static cost $p - 1$ with $\mathrm{static}(\mathcal{P}'') \subseteq \mathrm{static}(\mathcal{P})$. Hence, by applying the induction assumption to $\mathcal{P}''$, it follows that there exists a decision tree T_0 for $\mathsf{Search}(Q \cup v, Z \cup v)$ of depth $p - 1$.

We now construct a decision tree T for $\mathsf{Search}(Q, Z)$ as follows: The tree T queries v. If the answer is 0, the tree proceeds by invoking T_0, and otherwise it invokes T_1. It is not hard to see that T has depth at most p and that it solves $\mathsf{Search}(Q, Z)$, as required. □

It is worth mentioning that we can prove Theorem D.1 directly, without going through Lemma D.2. This can be done by splitting the proofs of each of the propositions into two cases: the case where $Z = \emptyset$ and the case where $Z \neq \emptyset$. In the first case, we need to work with a visiting pebbling of the sink rather than a surrounding pebbling of $\mathrm{head}(Z)$. However, this makes the proof more cumbersome.

References

[AB87] Noga Alon and Ravi B. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica*, 7(1):1–22, March 1987.

[ABRW02] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002. Preliminary version in *STOC '00*.

[BBG$^+$17] Patrick Bennett, Ilario Bonacina, Nicola Galesi, Tony Huynh, Mike Molloy, and Paul Wollan. Space proof complexity for random 3-CNFs. *Information and Computation*, 255:165–176, 2017.

[BC96] Samuel R. Buss and Peter Clote. Cutting planes, connectivity and threshold logic. *Archive for Mathematical Logic*, 35:33–63, 1996.

[BCIP02] Joshua Buresh-Oppenheim, Matthew Clegg, Russell Impagliazzo, and Toniann Pitassi. Homogenization and the polynomial calculus. *Computational Complexity*, 11(3-4):91–108, 2002. Preliminary version in *ICALP '00*.

[BEGJ00] María Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM Journal on Computing*, 30(5):1462–1484, 2000. Preliminary version in *FOCS '98*.

[Ben89] Charles H Bennett. Time/space trade-offs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, August 1989.

[BFI$^+$18] Paul Beame, Noah Fleming, Russell Impagliazzo, Antonina Kolokolova, Denis Pankratov, Toniann Pitassi, and Robert Robere. Stabbing planes. In *Proceedings of the 9th Innovations in Theoretical Computer Science Conference (ITCS '18)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:20, January 2018.

[BG03] Eli Ben-Sasson and Nicola Galesi. Space complexity of random formulae in resolution. *Random Structures and Algorithms*, 23(1):92–109, August 2003. Preliminary version in *CCC '01*.

[BG15] Ilario Bonacina and Nicola Galesi. A framework for space complexity in algebraic proof systems. *Journal of the ACM*, 62(3):23:1–23:20, June 2015. Preliminary version in *ITCS '13*.

[BIK$^+$97] Samuel R. Buss, Russell Impagliazzo, Jan Krajíček, Pavel Pudlák, Alexander A. Razborov, and Jiri Sgall. Proof complexity in algebraic systems and bounded depth Frege systems with modular counting. *Computational Complexity*, 6(3):256–298, 1997.

[BN08] Eli Ben-Sasson and Jakob Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pages 709–718, October 2008.

[BN16] Christoph Berkholz and Jakob Nordström. Near-optimal lower bounds on quantifier depth and Weisfeiler-Leman refinement steps. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '16)*, pages 267–276, July 2016.

[BPR97] María Bonet, Toniann Pitassi, and Ran Raz. Lower bounds for cutting planes proofs with small coefficients. *Journal of Symbolic Logic*, 62(3):708–728, September 1997. Preliminary version in *STOC '95*.

[BPS07] Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. *SIAM Journal on Computing*, 37(3):845–869, 2007. Preliminary version in *ICALP '05*.

[Bus98] Samuel R. Buss. Lower bounds on Nullstellensatz proofs via designs. In *Proof Complexity and Feasible Arithmetics*, volume 39 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 59–71. American Mathematical Society, 1998. Available at http://www.math.ucsd.edu/~sbuss/ResearchWeb/designs/.

[BW01] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48(2):149–169, March 2001. Preliminary version in *STOC '99*.

[CCT87] William Cook, Collette Rene Coullard, and György Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, November 1987.

[Cha13] Siu Man Chan. Just a pebble game. In *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13)*, pages 133–143, June 2013.

[Chv73] Vašek Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4(1):305–337, 1973.

[CKLM18] Arkadev Chattopadhyay, Michal Koucky, Bruno Loff, and Sagnik Mukhopadhyay. Simulation beats richness: New data-structure lower bounds. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, pages 1013–1020, June 2018.

[DM18] Irit Dinur and Or Meir. Toward the KRW composition conjecture: Cubic formula lower bounds via communication complexity. *Computational Complexity*, 27(3):375–462, 2018.

[dRNV16] Susanna F. de Rezende, Jakob Nordström, and Marc Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16)*, pages 295–304, October 2016.

[EIRS01] Jeff Edmonds, Russell Impagliazzo, Steven Rudich, and Jirí Sgall. Communication complexity towards lower bounds on circuit depth. *Computational Complexity*, 10(3):210–246, 2001.

[ET01] Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001. Preliminary versions of these results appeared in *STACS '99* and *CSL '99*.

[FLN$^+$15] Yuval Filmus, Massimo Lauria, Jakob Nordström, Noga Ron-Zewi, and Neil Thapen. Space complexity in polynomial calculus. *SIAM Journal on Computing*, 44(4):1119–1153, August 2015. Preliminary version in *CCC '12*.

[Gál01] Anna Gál. A characterization of span program size and improved lower bounds for monotone span programs. *Computational Complexity*, 10(4):277–296, December 2001. Preliminary version in *STOC '98*.

[GGKS18] Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, pages 902–911, June 2018.

[GHR92] Mikael Goldmann, Johan Håstad, and Alexander A. Razborov. Majority gates VS. general weighted threshold gates. *Computational Complexity*, 2:277–300, 1992. Preliminary version in *CCC '92*.

[GJPW17] Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. In *Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '17)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 52:1–52:15, July 2017.

[GJW18] Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM Journal on Computing*, 47(1):241–269, February 2018.

[GKPW17] Mika Göös, Pritish Kamath, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for P^NP. In *Proceedings of the 32nd Annual Computational Complexity Conference (CCC '17)*, volume 79 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:16, July 2017.

[GLM$^+$15] Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, pages 257–266, June 2015.

[GMWW17] Dmitry Gavinsky, Or Meir, Omri Weinstein, and Avi Wigderson. Toward better formula lower bounds: The composition of a function and a universal relation. *SIAM Journal on Computing*, 46(1):114–131, February 2017.

[Gom63] Ralph E. Gomory. An algorithm for integer solutions of linear programs. In R.L. Graves and P. Wolfe, editors, *Recent Advances in Mathematical Programming*, pages 269–302. McGraw-Hill, New York, 1963.

[GP18] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM Journal on Computing*, 47(5):1778–1806, October 2018. Preliminary version in *STOC '14*.

[GPT15] Nicola Galesi, Pavel Pudlák, and Neil Thapen. The space complexity of cutting planes refutations. In *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 433–447, June 2015.

[GPW15] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. In *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, pages 1077–1088, October 2015.

[GPW18] Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. *Computational Complexity*, 27(2):245–304, June 2018. Preliminary version in *ICALP '16*.

[HN12] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity (Extended abstract). In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 233–248, May 2012.

[HP18] Pavel Hrubeš and Pavel Pudlák. A note on monotone real circuits. *Information Processing Letters*, 131:15–19, March 2018.

[HW93] Johan Håstad and Avi Wigderson. Composition of the universal relation. In *Advances In Computational Complexity Theory*, volume 13 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 119–134. American Mathematical Society, 1993.

[Joh98] Jan Johannsen. Lower bounds for monotone real circuit depth and formula size and tree-like cutting planes. *Information Processing Letters*, 67(1):37–41, July 1998.

[KM18] Sajin Koroth and Or Meir. Improved composition theorems for functions and relations. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM '18)*, volume 116 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 48:1–48:18, August 2018.

[KMR17] Pravesh K. Kothari, Raghu Meka, and Prasad Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC '17)*, pages 590–603, June 2017.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[Kra95] Jan Krajíček. On Frege and extended Frege proof systems. In *Feasible Mathematics II*, pages 284–319, 1995.

[Kra97] Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *Journal of Symbolic Logic*, 62(2):457–486, June 1997.

[Kra98] Jan Krajíček. Interpolation by a game. *Mathematical Logic Quarterly*, 44(4):450–458, 1998.

[KRW95] Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3/4):191–204, September 1995. Preliminary version in *CCC '91*.

[KW90] Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM Journal on Discrete Mathematics*, 3(2):255–265, 1990. Preliminary version in *STOC '88*.

[LM19] Bruno Loff and Sagnik Mukhopadhyay. Lifting theorems for equality. In *Proceedings of the 36th Symposium on Theoretical Aspects of Computer Science (STACS '19)*, volume 126 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 50:1–50:19, March 2019.

[LRS15] James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, pages 567–576, June 2015.

[MS72] George Marsaglia and George P. H. Styan. When does $\mathrm{rank}(A+B) = \mathrm{rank}(A)+\mathrm{rank}(B)$? *Canadian Mathematical Bulletin*, 15(3):451–452, March 1972.

[Mur71] Saburo Muroga. *Threshold logic and its applications*. Wiley, 1971.

[Pit16] Toniann Pitassi. Manuscript, 2016.

[PR17] Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC '17)*, pages 1246–1255, June 2017.

[PR18] Toniann Pitassi and Robert Robere. Lifting Nullstellensatz to monotone span programs over any field. In *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, pages 1207–1219, June 2018.

[PTC77] Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. *Mathematical Systems Theory*, 10:239–251, 1977.

[Pud97] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, September 1997.

[Raz85] Alexander A. Razborov. Lower bounds for the monotone complexity of some Boolean functions. *Soviet Mathematics Doklady*, 31(2):354–357, 1985. English translation of a paper in *Doklady Akademii Nauk SSSR*.

[Raz90] Alexander A. Razborov. Applications of matrix methods to the theory of lower bounds in computational complexity. *Combinatorica*, 10(1):81–93, March 1990.

[RM99] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, March 1999. Preliminary version in *FOCS '97*.

[Rob18] Robert Robere. *Unified Lower Bounds for Monotone Computation*. PhD thesis, University of Toronto, 2018.

[Ros97] Arnold Rosenbloom. Monotone real circuits are more powerful than monotone Boolean circuits. *Information Processing Letters*, 61(3):161–164, February 1997.

[RPRC16] Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen A. Cook. Exponential lower bounds for monotone span programs. In *Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16)*, pages 406–415, October 2016.

[She11] Alexander A. Sherstov. The pattern matrix method. *SIAM Journal on Computing*, 40(6):1969–2000, December 2011. Preliminary version in *STOC '08*.

[Sok17] Dmitry Sokolov. Dag-like communication and its applications. In *Proceedings of the 12th International Computer Science Symposium in Russia (CSR '17)*, volume 10304 of *Lecture Notes in Computer Science*, pages 294–307. Springer, June 2017.