Electronic Colloquium on Computational Complexity, Revision 2 of Report No. 82 (2020)

MaxSAT Resolution and Subcube Sums*

YUVAL FILMUS, Technion – Israel Institute of Technology, Israel
MEENA MAHAJAN, The Institute of Mathematical Sciences (CI of Homi Bhabha National Institute), India
GAURAV SOOD, The Institute of Mathematical Sciences (CI of Homi Bhabha National Institute), India
MARC VINYALS, Technion – Israel Institute of Technology, Israel

We study the MaxSAT Resolution (MaxRes) rule in the context of certifying unsatisfiability. We show that it can be exponentially more powerful than tree-like resolution, and when augmented with weakening (the system MaxResW), p-simulates tree-like resolution. In devising a lower bound technique specific to MaxRes (and not merely inheriting lower bounds from Res), we define a new proof system called the SubCubeSums proof system. This system, which p-simulates MaxResW, can be viewed as a special case of the semialgebraic Sherali–Adams proof system. In expressivity, it is the integral restriction of conical juntas studied in the contexts of communication complexity and extension complexity. We show that it is not simulated by Res. Using a proof technique qualitatively different from the lower bounds that MaxResW inherits from Res, we show that Tseitin contradictions on expander graphs are hard to refute in SubCubeSums. We also establish a lower bound technique via lifting: for formulas requiring large degree in SubCubeSums, their XOR-ification requires large size in SubCubeSums.

CCS Concepts: • **Theory of computation → Proof complexity**.

Additional Key Words and Phrases: MaxSAT, resolution, proof complexity, conical juntas, Sherali–Adams

1 INTRODUCTION

The most well-studied propositional proof system is Resolution (Res), [11, 38]. It is a refutational line-based system that operates on clauses, successively inferring newer clauses until the empty clause is derived, indicating that the initial set of clauses is unsatisfiable. It has just one satisfiability-preserving rule: if clauses $A \vee x$ and $B \vee \neg x$ have been inferred, then the clause $A \vee B$ can be inferred. Sometimes it is convenient, though not necessary in terms of efficiency, to also allow a weakening rule: from clause A, a clause $A \vee x$ can be inferred. While there are several lower bounds known for this system, it is still very useful in practice and underlies many current SAT solvers.

While deciding satisfiability of a propositional formula is NP-complete, the MaxSAT question is an optimization question, and deciding whether its value is as given (i.e. deciding, given a formula and a number k, whether k clauses can be simultaneously satisfied but $k+1$ clauses cannot be satisfied) is potentially harder since it is hard for both NP and coNP. A proof system for MaxSAT was proposed in [14, 28]. This system, denoted MaxSAT Resolution or more briefly MaxRes, operates on multi-sets of clauses. At each step, two clauses from the multi-set are resolved and removed. The resolvent, as well as certain "disjoint" weakenings of the two clauses, are added to the multiset. The invariant maintained is that for each assignment ρ, the number of clauses in the multi-set falsified by ρ remains unchanged. The process stops when the multi-set has a satisfiable instance along with k copies of the empty clause; k is exactly the minimum number of clauses of the initial multi-set that must be falsified by every assignment. [14]

*A preliminary version of this article appeared in the proceedings of the 23rd International Conference on Theory and Applications of Satisfiability Testing – SAT 2020 [17]

Authors' addresses: Yuval Filmus, yuvalfi@cs.technion.ac.il, Technion – Israel Institute of Technology, Computer Science Department, Haifa, Israel; Meena Mahajan, meena@imsc.res.in, The Institute of Mathematical Sciences (CI of Homi Bhabha National Institute), IV Cross Road, CIT Campus, Taramani, Chennai, India; Gaurav Sood, gauravs@imsc.res.in, The Institute of Mathematical Sciences (CI of Homi Bhabha National Institute), IV Cross Road, CIT Campus, Taramani, Chennai, India; Marc Vinyals, marcvinyals@gmail.com, Technion – Israel Institute of Technology, Computer Science Department, Haifa, Israel.

ISSN 1433-8092

Since MaxRes maintains multi-sets of clauses and replaces used clauses, this suggests a "read-once"-like constraint [14]. However, this is not the case; read-once resolution is not even complete [26], whereas MaxRes is a complete system for certifying the MaxSAT value (and in particular, for certifying unsatisfiability). One could use the MaxRes system to certify unsatisfiability, by stopping the derivation as soon as one empty clause is produced. Such a proof of unsatisfiability, by the very definition of the system, can be p-simulated by Resolution. (The MaxRes proof is itself a proof with resolution and weakening, and weakening can be eliminated at no cost.) Thus, lower bounds for Resolution automatically apply to MaxRes and to MaxResW (the augmenting of MaxRes with an appropriate weakening rule) as well. However, since MaxRes needs to maintain a stronger invariant than merely satisfiability, it seems reasonable that for certifying unsatisfiability, MaxRes is weaker than Resolution. (This would explain why, in practice, MaxSAT solvers do not seem to use MaxRes – possibly with the exception of [35], but they instead directly call SAT solvers, which use standard resolution.) Proving this would require a lower bound technique specific to MaxRes.

Associating with each clause the subcube of assignments that falsify it, each MaxRes step manipulates and rearranges multi-sets of subcubes. This naturally leads us to the formulation of a static proof system that we call the SubCubeSums proof system. This system, by its very definition, p-simulates MaxResW. Associating with each subcube the minimal conjunction of literals (called terms) that is satisfied by all assignments in the subcube, SubCubeSums can be viewed as a special case of the semi-algebraic Sherali–Adams proof system (see for instance [3, 5, 10, 19]). Given this position in the ecosystem of simple proof systems, understanding its capabilities and limitations seems an interesting question.

Our contributions and techniques

(1) We observe that for certifying unsatisfiability, the proof system MaxResW p-simulates the tree-like fragment of Res, TreeRes (Lemma 3.1). This simulation seems to make essential use of the weakening rule. On the other hand, we show that even MaxRes without weakening is not simulated by TreeRes (Theorem 3.8). We exhibit a formula, which is a variant of the pebbling contradiction [9] on a pyramid graph, with short refutations in MaxRes (Lemma 3.2), and show that it requires exponential size in TreeRes (Lemma 3.7).

(2) We initiate a formal study of the newly-defined proof system SubCubeSums. We discuss how it is a natural degree-preserving restriction of the Sherali–Adams proof system and touch upon subtleties while defining size. We show that the system SubCubeSums is not simulated by Res, by showing that the Subset Cardinality Formulas, known to be hard for Res, have short SubCubeSums refutations (Theorem 4.1). We also give a direct combinatorial proof that the pigeon-hole principle formulas have short SubCubeSums refutations (Theorem 4.5); this fact is implicit in a recent result from [29].

(3) We show that the Tseitin contradiction on an odd-charged expander graph is hard for SubCubeSums (Theorem 4.9) and hence also hard for MaxResW. While this already follows from the fact that these formulas are hard for Sherali–Adams [3], our lower-bound technique is qualitatively different; it crucially uses the fact that a stricter invariant is maintained in MaxResW and SubCubeSums refutations.

(4) Abstracting the ideas from the lower bound for Tseitin contradictions, we devise a lower-bound technique for SubCubeSums based on lifting (Theorem 4.15). Namely, we show that if every SubCubeSums refutation of a formula F must have at least one wide clause, then every SubCubeSums refutation of the formula $F \circ \oplus$ must have many cubes. We illustrate how the Tseitin contradiction lower bound can be recovered in this way.

The relations among these proof systems are summarized in Figure 1, which also includes two proof systems discussed in Related Work.

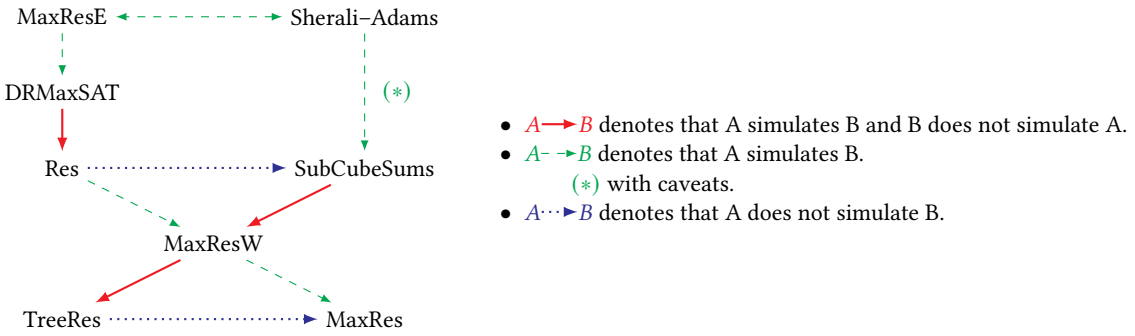

Fig. 1. Relations among various proof systems

Related work

One reason why studying MaxRes is interesting is that it displays unexpected power after some preprocessing. As described in [25] (see also [33]), the PHP formulas that are hard for Resolution can be encoded into MaxHornSAT, and then polynomially many weighted MaxRes steps suffice to expose the contradiction. The underlying proof system, weighted DRMaxSAT, has been studied further in [12], where it is shown to p-simulate general Resolution. While weighted DRMaxSAT gains power from the encoding, the basic steps are MaxRes steps. Thus, to understand how unweighted or weighted DRMaxSAT operates, a better understanding of MaxRes could be quite useful. Since SubCubeSums can easily refute some formulas hard for Resolution, it would be interesting to see how DRMaxSAT relates to SubCubeSums.

Some recent papers [13, 29, 30] study a generalization of the weighted version of MaxRes, under names MaxResE and MaxResSV. This system allows negative weights in the intermediate steps, as long as all the clauses have positive weights at the end. The system is used for certifying the MaxSAT value in [29, 30] and for certifying unsatisfiability in [13]. This difference allows the system to be used in a slightly different way in these papers. Since the satisfiability of a CNF does not change if we assign arbitrary positive weights to the axioms, [13] allows doing this. On the other hand, this is not allowed in [29, 30] because this would make the system unsound for MaxSAT. With this added power the system in [13] is p-equivalent to another recently defined proof system called Circular Resolution [4]; hence by the results in [4], it is also p-equivalent to Sherali–Adams. Though most results in [29] are for general MaxSAT, there is one result for a special case of MaxSAT where all axioms have infinite weight. Because of infinite weights, we get a result similar to that in [13]: the system is p-equivalent to Circular Resolution and Sherali–Adams. As can be seen from [13], the restriction of Circular Resolution where axioms can be used only once is precisely MaxResW; the further restriction of disallowing weakening of axioms is MaxRes.

It is also worth noting that MaxResW appears in [30] as MaxRes with a split rule, or ResS. It is shown in [29, 30] that for certifying the MaxSAT value (that is, the optimization version), weakening provably adds power to MaxRes. However, whether weakening adds power when MaxRes is used only to certify unsatisfiability remains unclear.

In the setting of communication complexity and of extension complexity of polytopes, non-negative rank is an important and useful measure. As discussed in [23], the query-complexity analogue is *conical juntas*; these are non-negative combinations of subcubes. Our SubCubeSums refutations are a restriction of conical juntas to non-negative *integral* combinations. Not surprisingly, our lower bound for Tseitin contradictions is similar to the conical junta degree lower bound established in [22].

Recently, in [18], one of the open problems raised in this paper is resolved; a lower bound for SubCubeSums size is shown for a formula that has short refutations in resolution. Also, in [20], a very close variant of MaxResW called reversible resolution is studied and separated from resolution. This system has the weakening rule and its reverse; that is, resolution is permitted only when the antecedent clauses differ in only one variable, which they have in opposing polarities.

Organisation of the paper

We define the proof systems MaxRes, MaxResW, and SubCubeSums in Section 2. In Section 3 we relate them to TreeRes. In Section 4, we focus on the SubCubeSums proof system, showing the separation from Res (Section 4.1), the lower bound for SubCubeSums (Section 4.2), and the lifting technique (Section 4.3).

2 DEFINING THE PROOF SYSTEMS

A literal is a variable or its negation. A clause is the disjunction of a set of literals (hence, without repetitions). In particular, if A and B are clauses, then $A \vee B$ denotes the clause that is the disjunction of the literals in A and in B without repetitions. A clause is non-tautologous if it has no pair of contradictory literals (x and $\neg x$). We work only with non-tautological clauses throughout.

For set X of variables, let $\langle X \rangle$ denote the set of all total assignments to variables in X. For a (multi-) set F of clauses, $\mathrm{viol}_F \colon \langle X \rangle \to \{0\} \cup \mathbb{N}$ is the function mapping α to the number of clauses in F (counted with multiplicity) falsified by α. A (sub)cube is the set of assignments falsifying a clause, or equivalently, the set of assignments satisfying a conjunction of literals. (We refer to clauses and cubes interchangeably, given the natural bijection between them.) The width of a clause is the number of literals in it, and the width of a (multi-) set F of clauses is the maximum width of the clauses it contains.

The proof system Res has the resolution rule inferring $C \vee D$ from $C \vee x$ and $D \vee \overline{x}$, and optionally the weakening rule inferring $C \vee x$ from C if $\overline{x} \notin C$. A refutation of a CNF formula F is a sequence of clauses $C_1, \ldots, C_t$ where each C_i is either in F or is obtained from some $j, k < i$ using resolution or weakening, and where C_t is the empty clause. The underlying graph of such a refutation has the clauses as nodes, and directed edge from C to D if C is used in the step deriving D. The proof system TreeRes is the fragment of Res where only refutations in which the underlying graph is a tree are permitted. A proof system P simulates (p-simulates) another proof system P' if proofs in P can be transformed into proofs in P' with polynomial blow-up (in time polynomial in the size of the proof). See, for instance, [8], for more details.

2.1 The MaxRes and MaxResW proof systems

The MaxSAT resolution (MaxRes) proof system operates on multi-sets of clauses, and uses the multi-output MaxSAT resolution (MaxRes) rule [14], defined as follows:

$$\begin{array}{ll}
x \vee a_1 \vee \ldots \vee a_s & (x \vee A) \\
\overline{x} \vee b_1 \vee \ldots \vee b_t & (\overline{x} \vee B) \\
\hline
a_1 \vee \ldots \vee a_s \vee b_1 \vee \ldots \vee b_t & \text{(the ``standard resolvent'')} \\
\left.\begin{array}{l}
x \vee A \vee \overline{b}_1 \\
x \vee A \vee b_1 \vee \overline{b}_2 \\
\vdots \\
x \vee A \vee b_1 \vee \ldots \vee b_{t-1} \vee \overline{b}_t
\end{array}\right\} & \text{(weakenings of } x \vee A\text{)} \\
\left.\begin{array}{l}
\overline{x} \vee B \vee \overline{a}_1 \\
\overline{x} \vee B \vee a_1 \vee \overline{a}_2 \\
\vdots \\
\overline{x} \vee B \vee a_1 \vee \ldots \vee a_{s-1} \vee \overline{a}_s
\end{array}\right\} & \text{(weakenings of } \overline{x} \vee B\text{)}
\end{array}$$

The weakening rule for MaxSAT resolution replaces a clause A by the two clauses $A \vee x$ and $A \vee \overline{x}$. While applying either of these rules, the antecedents are removed from the multi-set and the non-tautologous consequents are added. The point of the MaxSAT resolution rule is that if F' is obtained from F by applying these rules, then viol_F and $\mathrm{viol}_{F'}$ are the same function.

In the proof system MaxRes, a refutation of F is a sequence $F = F_0, F_1, \ldots, F_s$ where each F_i is a multi-set of clauses, each F_i is obtained from F_{i-1} by an application of the MaxSAT resolution rule, and F_s contains the empty clause $\square$. In the proof system MaxResW, F_i may also be obtained from F_{i-1} by using the weakening rule. The size of the proof is the number of steps, s. In [14, 28], MaxRes is shown to be complete for MaxSAT; i.e. if any assignment must falsify at least k clauses, then at least k copies of the empty clause can be derived using MaxRes. Hence MaxRes is also complete for unsatisfiability. Since the proof system MaxRes we consider here is a refutation system rather than a system for MaxSAT, we can stop as soon as a single $\square$ is derived.

2.2 The SubCubeSums proof system

The SubCubeSums proof system is a static proof system. For an unsatisfiable CNF formula F (over variable set X), a SubCubeSums proof is a multi-set G of clauses (or subcubes) over X satisfying $\mathrm{viol}_F(\alpha) = 1 + \mathrm{viol}_G(\alpha)$ for all assignments $\alpha \in \langle X \rangle$. The combinatorial size of the proof is the number of clauses in G (counting with multiplicity), and the width of the proof is the width of G.

Stated in this form, SubCubeSums may not be a proof system in the sense of Cook-Reckhow [16], since proofs may not be polynomial-time verifiable. However, proofs in SubCubeSums can be verified in randomized polynomial time. To see this, we consider an arithmetization of SubCubeSums proofs.

Let F be a CNF formula with m clauses in variables $x_1, \ldots, x_n$. Each clause C_i, $i \in [m]$, is translated into a polynomial equation $f_i = 0$. A Boolean assignment either satisfies clause C_i and equation $f_i = 0$, or falsifies clause C_i and satisfies equation $f_i = 1$. (Encoding e: $e(x_j) = (1 - x_j)$; $e(\neg x_j) = x_j$; $e(\bigvee_r \ell_r) = \prod_r e(\ell_r)$. So, e.g., clause $x \vee \neg y \vee z$ translates to the equation $(1 - x)y(1 - z) = 0$. Note that for any non-tautologous clause, each such polynomial f_i is multilinear and has the form $p_{A,B} \triangleq \prod_{i \in A} x_i \prod_{j \in B}(1 - x_j)$ for disjoint $A, B \subseteq [n]$.)

Given an alleged SubCubeSums proof G of an F that we wish to verify, define the polynomial

$$p_0(x) = \sum_{A,B\subseteq[n]:A\cap B\neq\emptyset} \alpha_{A,B} \prod_{i\in A} x_i \prod_{j\in B}(1-x_j)$$

where the coefficient $\alpha_{A,B}$ is the number of copies in G of the clause whose encoding is $p_{A.B}$. Define the polynomial $Q(x) = -\sum_{i\in[m]} f_i(x) + p_0(x) + 1$. That is,

$$Q(x) = -\left(\sum_{i\in[m]} f_i(x)\right) + \left(\sum_{A,B\subseteq[n]:A\cap B\neq\emptyset} \alpha_{A,B} \prod_{i\in A} x_i \prod_{j\in B}(1-x_j)\right) + 1$$

Note that for any Boolean assignment α to the variables, $Q(\alpha) = -\mathrm{viol}_F(\alpha) + \mathrm{viol}_G(\alpha) + 1$. Thus G is a SubCubeSums proof for F if and only if $Q(x)$ vanishes on all Boolean assignments.

Now note that $Q(x)$ has two nice properties with useful consequences for us:

(1) $Q(x)$ is multilinear.
Hence, $Q(x)$ vanishes on all Boolean assignments of and only if $Q(x)$ vanishes everywhere; i.e. $Q(x) = 0$ is a polynomial identity. (See for instance [27, Ex. 2.23 on p. 76])

(2) $Q(x)$ can be computed by an algebraic circuit that has $O(n(|F| + |G|))$ binary operations, and has variables or the constants $-1, +1$ at the leaves. ($O(n)$ operations to encode each copy of each clause, and then $O(|F| + |G|)$ operations to add them all up.)
Hence, whether $Q(x)$ is identically 0 can be tested by a randomized algorithm in time polynomial in $n, |F|, |G|$. (Polynomial identity testing can be done, using randomization, in time polynomial in the size of the circuit representation; see for instance [1].)

2.3 SubCubeSums as a subsystem of the Sherali–Adams proof system

The arithmetization of SubCubeSums proofs discussed above naturally recalls to mind the semi-algebraic Sherali–Adams proof system over the reals, typically with integer coefficients. We recapitulate below the definition of the proof system and observe that SubCubeSums is a subsystem of a specific type.

A Sherali–Adams proof of unsatisfiability of a CNF formula F is a sequence of polynomials g_i, $i \in [m]$; q_j, $j \in [n]$; and a polynomial p_0 of the form

$$p_0 = \sum_{A,B\subseteq[n]:A\cap B=\emptyset} \alpha_{A,B} p_{A,B} = \sum_{A,B\subseteq[n]:A\cap B=\emptyset} \alpha_{A,B} \prod_{j\in A} x_j \prod_{j\in B}(1-x_j)$$

where each $\alpha_{A,B} \geq 0$, such that the following polynomial identity holds:

$$\left(\sum_{i\in[m]} g_i f_i\right) + \left(\sum_{j\in[n]} q_j(x_j^2 - x_j)\right) + p_0 + 1 = 0$$

(As before, the polynomials f_i encode the clauses of F. The axioms $x_j^2 - x_j = 0$ for $j \in [n]$, called the Boolean axioms, are used to restrict the set of assignments to Boolean values.)

Note that each $p_{A,B}$, and hence p_0, is multilinear. The degree or rank of the proof is the maximum degree of any $g_i f_i$, $q_j(x_j^2 - x_j)$, and $p_{A,B}$.

The polynomials f_i corresponding to the clauses of F, as well as the polynomials $p_{A,B}$ in p_0, are conjunctions of literals, thus special kinds of d-juntas (Boolean functions depending on at most d variables). So p_0 is a non-negative linear combination of non-negative juntas, that is, in the nomenclature of [23], a *conical junta*.

Consider the following restriction of Sherali–Adams:

(1) Each $g_i = -1$.
(2) Each $\alpha_{A,B} \in \mathbb{Z}^{\geq 0}$ (non-negative integers).
(3) Each $q_j = 0$.

Hence, for some non-negative integral $\alpha_{A,B}$, a proof as restricted above is the following polynomial identity:

$$-\sum_{i\in[m]} f_i + \left(\sum_{A,B\subseteq[n]:A\cap B=\emptyset} \alpha_{A,B} \prod_{j\in A} x_j \prod_{j\in B}(1-x_j)\right) + 1 = 0$$

This is exactly the form of the arithmetization of SubCubeSums proofs discussed in the previous subsection. That is, any SubCubeSums proof gives rise to such a restricted Sherali–Adams proof. The converse is also true – each such restricted Sherali–Adams proof corresponds in a natural way to a SubCubeSums proof as follows: each $p_{A,B}$ in p_0 encodes a clause (equivalently, the subcube of assignments falsifying the clause). For each disjoint pair $A, B \subseteq [n]$, the SubCubeSums proof has $\alpha_{A,B}$ copies of the corresponding clause/sub-cube.

It is worth noting that in this equivalence, when we translate a SubCubeSums proof G of a formula F into a restricted Sherali–Adams proof, the resulting degree is the maximum of the width of F and the width of G. Conversely, when we translate a restricted Sherali–Adams proof into a SubCubeSums proof, the width of the resulting SubCubeSums proof is no more than the original degree.

SubCubeSums: The algebraic view with twinned variables. A Sherali–Adams system may require large number of monomials for some formulas simply because a clause C with w negated literals gives rise to a polynomial f with 2^w monomials. The standard approach to handle this is to use twinned variables, one variable for each literal (i.e. $\overline{x}$ is a new variable), and include in the set of Boolean axioms the equations $1 - x_i - \overline{x_i} = 0$. This makes no difference to the degree of the proof. (The encoding e is modified to $e(x_j) = \overline{x_j}$; $e(\neg x_j) = x_j$; $e(\bigvee_r \ell_r) = \prod_r e(\ell_r)$. So, e.g., clause $x \vee \neg y \vee z$ translates to the equation $\overline{x}y\overline{z} = 0$.)
Thus a Sherali–Adams proof is now a sequence of polynomials g_i, $i \in [m]$; q_j, r_j, $j \in [n]$; and a polynomial p_0 of the form

$$p_0 = \sum_{A,B\subseteq[n]:A\cap B=\emptyset} \alpha_{A,B} \prod_{j\in A} x_j \prod_{j\in B} \overline{x_j}$$

where each $\alpha_{A,B} \geq 0$, such that

$$\left(\sum_{i\in[m]} g_i f_i\right) + \left(\sum_{j\in[n]} q_j(x_j^2 - x_j)\right) + \left(\sum_{j\in[n]} r_j(1 - x_j - \overline{x_j})\right) + p_0 + 1 = 0$$

We will use this formulation with twinned variables.

The unary size of a Sherali–Adams proof is the sum of (the absolute values of) the coefficients of the polynomials occurring in the proof. We can also define unary reduced size which excludes the Boolean axioms and the polynomials q_j and r_j above. (We can also define binary size, accounting for coefficient bit-sizes when represented in binary, or monomial size, ignoring coefficient sizes altogether and only counting distinct monomials. All these measures have been considered in the literature in different papers and different contexts; see for instance [2, 3, 6, 19, 21, 31]. For the purposes of this paper, unary and unary reduced size are most relevant.) The degree or rank of the proof is the maximum degree of any $g_i f_i$, $q_j(x_j^2 - x_j)$, $r_j x_j$ and $p_{A,B}$.

Now, the restriction where each $g_i = -1$, each $\alpha_{A,B} \in \mathbb{Z}^{\geq 0}$ (non-negative integers), and each $q_j = 0$, gives the SubCubeSums proof system; an algebraic SubCubeSums proof is a polynomial identity of the form

$$-\left(\sum_{i\in[m]} f_i\right) + \left(\sum_{j\in[n]} r_j(1 - x_j - \overline{x_j})\right) + \left(\sum_{A,B\subseteq[n]} \alpha_{A,B} \prod_{j\in A} x_j \prod_{j\in B} \overline{x_j}\right) + 1 = 0.$$

(To be precise, a SubCubeSums proof corresponds to an equivalence class of Sherali–Adams proofs modulo Boolean axioms).

With this algebraic view of SubCubeSums in mind, we can define the *algebraic size* of a SubCubeSums proof to be the unary size of the smallest corresponding Sherali–Adams proof (note that this includes the Boolean axioms and r_j). We can also define the *algebraic reduced size* of a SubCubeSums proof to be unary reduced size of the smallest corresponding Sherali–Adams proof. With these definitions, the following relations are immediate:

For any SubCubeSums proof G of a formula $|F|$,

$$\text{(combinatorial size of } G) + |F| = \text{(algebraic reduced size of } G) \leq \text{(algebraic size of } G).$$

$$\max\{\text{width}(G), \text{width}(F)\} = \text{(algebraic degree of } G).$$

2.4 Relating various measures for SubCubeSums and MaxResW

In the combinatorial view of SubCubeSums, the natural complexity measures are combinatorial size (number of subcubes) and width. In the algebraic view, there are two measures for size depending on whether or not we count the monomials from the Boolean axioms (the contributions from $r_j(1 - x_k - \overline{x_j})$): algebraic size, and algebraic reduced size.

In the algebraic view, there are also two measures for degree: (1) the usual degree of the Sherali-Adams restriction, and (2) the conical junta degree, or the degree of the polynomial p_0 alone. As discussed above, the degree equals the maximum of the initial formula width and the SubCubeSums proof width, while the conical-junta-degree equals the SubCubeSums width.

$$\text{width}(G) = \text{(conical-junta-degree of } G).$$

It is worth noting that the combinatorial measures can be significantly smaller than the algebraic measures. If F is the negation of the complete tautology on n variables, then the SubCubeSums proof is the empty set, of combinatorial size and width 0. However, the algebraic degree is n, and the algebraic size and algebraic reduced size are 2^n, simply because of the contribution from the initial formula.

Strictly speaking we do not know if unary Sherali–Adams (or even Sherali–Adams with size measured as the sum of the binary bit-sizes of all coefficients, that is, the usual Sherali–Adams) simulates SubCubeSums with respect to combinatorial size; hence the caveat in Figure 1. (The simulation holds with respect to algebraic size, as well as with respect to degree.) However, upper bounds on SubCubeSums algebraic size imply upper bounds on Sherali–Adams unary size, while known lower bounds on Sherali–Adams unary reduced size imply lower bounds on SubCubeSums algebraic reduced size. Hence for all practical purposes we can think as if it did.

The following proposition shows why the proposed restriction of Sherali–Adams to SubCubeSums remains complete, and gives combinatorial and algebraic size bounds in terms of MaxResW refutation size.

Proposition 2.1. *SubCubeSums p-simulates MaxResW.*

For any unsatisfiable formula with n variables and m clauses, a MaxResW refutation of size s can be converted (in polynomial time) to a SubCubeSums proof of both combinatorial size and algebraic size $O(m + ns)$.

Proof. If an unsatisfiable CNF formula F with m clauses and $n \geq 3$ variables has a MaxResW refutation with s steps, then this derivation produces $\{\square\} \cup G$ where the number of clauses in G is at most $m + (n-2)s - 1$. (A weakening step increases the number of clauses by 1, without creating an empty clause. A MaxRes step increases it by at most $n-2$, and creates at most one empty clause.) The subcubes falsifying the clauses in G give a SubCubeSums proof.

The simulation still holds if we measure algebraic size. To see that, observe that we can simulate a weakening step by introducing at most 5 new monomials; deriving clauses $A \vee x$ and $A \vee \neg x$ from A corresponds to rewriting the monomial m encoding A as $mx + m\overline{x} + m(1 - x - \overline{x})$. More generally, given a monomial m and a set of literals $A = a_1, \ldots, a_s$, the polynomial

$$\begin{aligned} W(m, A) \stackrel{\text{def}}{=}\ & ma_1 + m(1 - \overline{a_1} - a_1) \\ & + m\overline{a_1}a_2 + m\overline{a_1}(1 - \overline{a_2} - a_2) \\ & + \cdots \\ & + m\overline{a_1} \cdots \overline{a_{s-1}}a_s + m\overline{a_1} \cdots \overline{a_{s-1}}(1 - \overline{a_s} - a_s) \\ & + m\overline{a_1} \cdots \overline{a_s} \end{aligned}$$

is identically equal to m. It describes the weakening of m by the literals of A using the twinning axioms, and has algebraic size $4s + 1 \leq 5s$. Further, given monomials $m_A = \overline{x} \cdot e(A)$ and $m_B = x \cdot e(B)$ encoding clauses $x \vee A$ and $\overline{x} \vee B$, we can simulate the MaxRes resolution rule by writing

$$\begin{aligned} m_A + m_B = & W(m_A, B \setminus A) - m_A \cdot e(B \setminus A) \\ & + W(m_B, A \setminus B) - m_B \cdot e(A \setminus B) \\ & + e(A \cup B) \\ & - e(A \cup B) \cdot (1 - \overline{x} - x). \end{aligned}$$

The algebraic size of this expression is $(4|B \setminus A| + 1) + (4|A \setminus B| + 1) + 6 \leq 8n$.

Hence we can simulate a weakening step with 5 monomials and a resolution step with at most $8n$ monomials. □

In Section 4.1 we establish combinatorial size upper bounds in SubCubeSums for certain formulas. To show that these upper bounds also apply to algebraic size, we observe that the measures are equivalent in proofs of constant positive or negative degree. More formally, defining the positive (negative) degree of a proof as the degree counting only x_i variables (resp. $\overline{x_i}$) in f_i and p_0, the following holds.

Proposition 2.2. *A SubCubeSums proof of combinatorial size s and positive (negative) degree d has algebraic size* $O(2^d(|F| + s))$.

Proof. We use the following claim.

Claim 2.1. *Let p be a polynomial with integer coefficients that*

1. *is multilinear, on $2n$ variables $\{x_i, \overline{x_i} \mid j \in [n]\}$,*
2. *has $\#mon(p) = s$ monomials (with repetition, i.e when written with coefficients ± 1),*
3. *has positive (negative) degree d, and*
4. *vanishes on all Boolean assignments to the variables.*

Then there is a polynomial q of the form $\sum_{j\in[n]} r_j(1-x_j-\overline{x_j})$, with $\sum_{j\in[n]} \#mon(r_j(1-x_j-\overline{x_j})) \leq 3\cdot(2^d-1)\cdot s$, such that $p+q=0$ (here we count the monomials with repetition).

To see why the proposition follows from the claim, consider a SubCubeSums proof of size $s = |p_0|$ and positive (negative) degree d. It has the form $\sum_{i\in[m]} f_i = p_0 + 1$ modulo Boolean (twinning) axioms. Applying the claim to the polynomial $p = -\sum_{i\in[m]} f_i + p_0 + 1$, which has $|F| + |p_0| + 1$ monomials, we obtain a polynomial q such that $-\sum_{i\in[m]} f_i + p_0 + 1 + q$ is a a Sherali–Adams representative of size at most $\left(1 + 3\cdot(2^d-1)\right)\cdot(|F|+|p_0|+1)$. □

Proof. (of Claim) We prove the claim for positive degree; the negative degree argument is identical. We proceed by induction on d.

Base case: $d = 0$. Then p is multilinear on the n variables $\{\overline{x_i} \mid i \in [n]\}$, and vanishes at all 2^n Boolean assignments to its variables. Since the multilinear polynomial interpolating Boolean values on the Boolean hypercube is unique, and since the zero polynomial is such an interpolating polynomial, we already have $p = 0$ and can choose $q = 0$.

Inductive Step: For each monomial in p with positive degree d, pick a positive variable x in the monomial arbitrarily, and rewrite the monomial mx as $m - m\overline{x} - m(1-\overline{x}-x)$. So p is rewritten as $p' + q''$, where q'' collects the parts $m(1-\overline{x}-x)$ introduced above and p' collects the remaining monomials.

Note that the monomials m, $m\overline{x}$ have positive degree $d-1$, so p' is a multilinear polynomial with positive degree at most $d-1$. Also, it has at most $2s$ monomials. Since p and q'' vanish on all Boolean assignments, so does p'. The inductive claim applied to p' yields $q' = \sum_{j\in[n]} r'_j(1-\overline{x_j}-x_j)$ such that $p'+q' = 0$. Hence for $q = q' - q''$, $p + q = 0$. The polynomial q is of the desired form $\sum_{j\in[n]} r_j(1-x_j-\overline{x_j})$. Counting monomials, q'' contributes at most $3s$ monomials by construction, and the number of monomials contributed by q' is bounded by induction, so $\sum_{j\in[n]} \#mon(r_j(1-x_j-\overline{x_j})) \leq 3s + 3\cdot(2^{d-1}-1)\cdot 2s = 3\cdot(2^d-1)\cdot s$.

□

SubCubeSums is also implicationally complete in the following sense. We say that $f \geq g$ if for every truth assignment x, $f(x) \geq g(x)$.

Proposition 2.3. *If f and g are polynomials with $f \geq g$, then there are subcubes h_j and non-negative numbers c_j such that on the Boolean hypercube, $f - g = \sum_j c_j h_j$. Further, if f, g are integral on the Boolean hypercube, so are the c_j.*

Proof. A brute-force way to see this is to consider subcubes of degree n, i.e. a single point (or assignment). For each $\beta \in \{0,1\}^n$, define $c_\beta = (f-g)(\beta) \in \mathbb{R}^{\geq 0}$. □

3 MAXRES, MAXRESW, AND TREERES

Since TreeRes allows reuse only of input clauses, while MaxRes does not allow any reuse of clauses but produces multiple clauses at each step, the relative power of these fragments of Res is intriguing. In this section, we show that MaxRes with the weakening rule, MaxResW, p-simulates TreeRes, is exponentially separated from it, and even MaxRes (without weakening) is not simulated by TreeRes.

Lemma 3.1. *For every unsatisfiable CNF F, $size(F \vdash_{MaxResW} \square) \leq 2size(F \vdash_{TreeRes} \square)$.*

Proof. Let T be a tree-like derivation of $\square$ from F of size s. Without loss of generality, we may assume that T is regular [41]; i.e. no variable is used as pivot twice on the same path.

Since a MaxSAT resolution step always adds the standard resolvent, each step in a tree-like resolution proof can be performed in MaxResW as well, provided the antecedents are available. However, a tree-like proof may use an

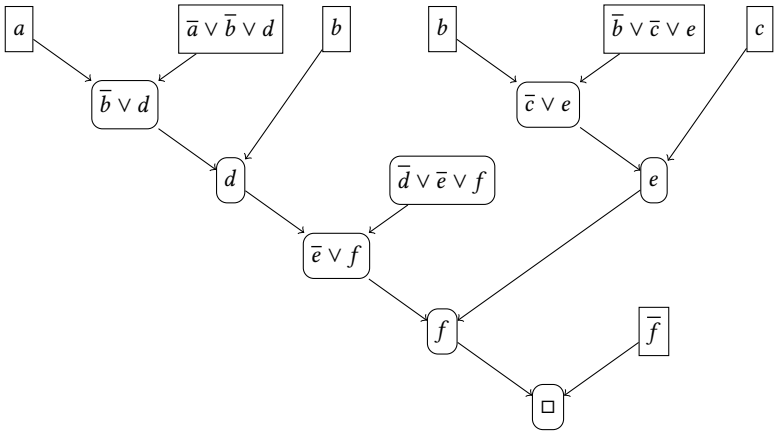

Fig. 2. A tree-like resolution proof

axiom (a clause in F) multiple times, whereas after it is used once in MaxResW it is no longer available, although some weakenings are available. So we need to work with weaker antecedents. We describe below how to obtain sufficient weakenings.

For each axiom $A \in F$, consider the subtree T_A of T defined by retaining only the paths from leaves labeled A to the final empty clause. We will produce multiple disjoint weakenings of A, one for each leaf labelled A. Start with A at the final node (where T_A has the empty clause) and walk up the tree T_A towards the leaves. If we reach a branching node v with clause A', and the pivot at v is x, weaken A' to $A' \vee x$ and $A' \vee \overline{x}$. Proceed along the edge contributing x with $A' \vee x$, and along the other edge with $A' \vee \overline{x}$. Since T is regular, no tautologies are created in this process, which ends with multiple "disjoint" weakenings of A.

After doing this for each axiom, we have as many clauses as leaves in T. Now we simply perform all the steps in T.

Since each weakening step increases the number of clauses by one, and since we finally produce at most s clauses for the leaves, the number of weakening steps required is at most s. □

As an illustration, consider the tree-like resolution proof in Figure 2. Following the procedure in the proof of the Lemma, the axiom b is weakened to $b \vee e$ and $b \vee \neg e$, since e is the pivot variable at the branching point where b is used in both sub-derivations.

We now show that even without weakening, MaxRes has short proofs of formulas exponentially hard for TreeRes. We denote the literals $\overline{x}$ and x by x^0 and x^1 respectively. The formulas that exhibit the separation are *composed* formulas of the form $F \circ g$, where F is a CNF formula, $g\colon \{0,1\}^\ell \to \{0,1\}$ is a Boolean function, there are ℓ new variables $x_1, \ldots, x_\ell$ for each original variable x of F, and there is a block of clauses $C \circ g$, a CNF expansion of the expression $\bigvee_{x^b \in C}(g(x_1, \ldots x_\ell) = b)$, for each original clause $C \in F$. We use the pebbling formulas on single-sink directed acyclic graphs: there is a variable for each node, variables at sources must be true, the variable at the sink must be false, and at each node v, if variables at origins of incoming edges are true, then the variable at v must also be true.

We denote by $\mathrm{PebHint}(G)$ the standard pebbling formula with additional hints $u \vee v$ for each pair of siblings (u, v)—that is, two incomparable vertices with a common predecessor—, and we prove the separation for $\mathrm{PebHint}(G)$ composed with the OR function. More formally, if G is a DAG with a single sink z, we define $\mathrm{PebHint}(G) \circ \mathrm{OR}$ as follows. For each vertex $v \in G$ there are variables v_1 and v_2. The clauses are

- For each source v, the clause $v_1 \vee v_2$.
- For each internal vertex w with predecessors u, v, the expression $((u_1 \vee u_2) \wedge (v_1 \vee v_2)) \to (w_1 \vee w_2)$, expanded into 4 clauses.
- The clauses $\overline{z_1}$ and $\overline{z_2}$ for the sink z.
- For each pair of siblings (u, v), the clause $u_1 \vee u_2 \vee v_1 \vee v_2$.

Note that the first three types of clauses are also present in standard composed pebbling formulas, while the last type are the hints.

We prove a MaxRes upper bound for the particular case of pyramid graphs. Let P_h be a pyramid graph of height h and $n = \Theta(h^2)$ vertices.

LEMMA 3.2. *The* $\mathrm{PebHint}(P_h) \circ \mathrm{OR}$ *formulas have* $\Theta(n)$ *size MaxRes refutations.*

PROOF. We derive the clause $s_1 \vee s_2$ for each vertex $s \in P_n$ in layered order, and left-to-right within one layer. If s is a source, then $s_1 \vee s_2$ is readily available as an axiom. Otherwise assume that for a vertex s with predecessors u and v and siblings r and t – in this order – we have clauses $u_1 \vee u_2 \vee s_1 \vee s_2$ and $v_1 \vee v_2$, and let us see how to derive $s_1 \vee s_2$. (Except at the boundary, we don't have the clause $u_1 \vee u_2$ itself, since it has been used to obtain the sibling r and doesn't exist anymore.) We also make sure that the clause $v_1 \vee v_2 \vee t_1 \vee t_2$ becomes available to be used in the next step.

In the following derivation we skip $\vee$ symbols, and we colour-code clauses so that green clauses are available by induction, axioms are blue, and red clauses, on the right side in steps with multiple consequents, are additional clauses that are obtained by the MaxRes rule but not with the usual resolution rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\overline{u_1}\overline{v_1}s_1s_2 \quad u_1u_2s_1s_2}{u_2\overline{v_1}s_1s_2} \quad \dfrac{u_1u_2v_1s_1s_2 \quad \overline{u_1}\overline{v_2}s_1s_2}{u_2v_1\overline{v_2}s_1s_2} \quad \overline{u_2}\overline{v_2}s_1s_2}{\overline{u_2}\overline{v_1}s_1s_2} \quad \dfrac{v_1\overline{v_2}s_1s_2 \quad v_1v_2}{v_1s_1s_2} \quad v_1v_2\overline{s_1} \quad v_1v_2s_1\overline{s_2} \quad s_1s_2t_1t_2}{\overline{v_1}s_1s_2} \quad \dfrac{v_1v_2s_1t_1t_2}{s_1s_2}}{v_1v_2t_1t_2}$$

The case where some of the siblings are missing is similar: if r is missing then we use the axiom $u_1 \vee u_2$ instead of the clause $u_1 \vee u_2 \vee s_1 \vee s_2$ that would be available by induction, and if t is missing then we skip the steps that use $s_1 \vee s_2 \vee t_1 \vee t_2$ and lead to deriving $v_1 \vee v_2 \vee t_1 \vee t_2$.

Finally, once we derive the clause $z_1 \vee z_2$ for the sink, we resolve it with axiom clauses $\overline{z_1}$ and $\overline{z_2}$ to obtain a contradiction.

A constant number of steps suffice for each vertex, for a total of $\Theta(n)$. □

We can prove a tree-like lower bound along the lines of [8], but with some extra care to respect the hints. As in [8] we derive the hardness of the formula from the *pebble game*, a game where the single player starts with a DAG and a set of pebbles, the allowed moves are to place a pebble on a vertex if all its predecessors have pebbles or to remove a pebble at any time, and the goal is to place a pebble on the sink using the minimum number of pebbles. Denote by $\mathrm{bpeb}(P \to w)$ the cost of placing a pebble on a vertex w assuming there are free pebbles on a set of vertices $P \subseteq V$ – in other words, the number of pebbles used outside of P when the starting position has pebbles in P. For a DAG G with a single sink z, $\mathrm{bpeb}(G)$ denotes $\mathrm{bpeb}(\emptyset \to z)$. For $U \subseteq V$ and $v \in V$, the subgraph of v modulo U is the set of vertices u such that there exists a path from u to v avoiding U.

Lemma 3.3 ([15]). $\mathrm{bpeb}(P_h) = h + 1$.

Lemma 3.4 ([8]). *For all P, v, w, we have* $\mathrm{bpeb}(P \to v) \leq \max(\mathrm{bpeb}(P \to w), \mathrm{bpeb}(P \cup \{w\} \to v) + 1)$.

We deviate slightly from [8] and, instead of directly translating a proof to a pebbling strategy, we go through query complexity as an intermediate step. The canonical search problem of a formula F is the relation $\mathrm{Search}(F)$ where inputs are variable assignments $\alpha \in \{0,1\}^n$ and the valid outputs for α are the clauses $C \in F$ that α falsifies. Given a relation f, we denote by $\mathrm{DT}_1(f)$ the 1-query complexity of f [32], that is the minimum over all decision trees computing f of the maximum of 1-answers that the decision tree receives.[1]

Lemma 3.5. *For all G we have* $\mathrm{DT}_1(\mathrm{Search}(\mathrm{PebHint}(G))) \geq \mathrm{bpeb}(G) - 1$.

Proof. We give an adversarial strategy. Let R_i be the set of variables that are assigned to 1 at round i. We initially set $w_0 = z$, and maintain the invariant that

(1) there is a distinguished variable w_i and a path π_i from w_i to the sink z such that a queried variable v is 0 iff $v \in \pi_i$; and
(2) after each query the number of 1 answers so far is at least $\mathrm{bpeb}(G) - \mathrm{bpeb}(R_i \to w_i)$.

Assume that a variable v is queried. If v is not in the subgraph of w_i modulo R_i then we answer 0 if $v \in \pi_i$ and 1 otherwise. Otherwise we consider $p_0 = \mathrm{bpeb}(R_i \to v)$ and $p_1 = \mathrm{bpeb}(R_i \cup \{v\} \to w_i)$. By Lemma 3.4, $\mathrm{bpeb}(R_i \to w_i) \leq \max(p_0, p_1 + 1)$. If $p_0 \geq p_1$ then we answer 0, set $w_{i+1} = v$, and extend π_i with a path from w_{i+1} to w_i that does not contain any 1 variables (which exists by definition of subgraph modulo R_i). This preserves item 1 of the invariant, and since $p_0 \geq \mathrm{bpeb}(R_i \to w_i)$, item 2 is also preserved. Otherwise we answer 1 and since $p_1 \geq \mathrm{bpeb}(R_i \to w_i) - 1$ the invariant is also preserved.

This strategy does not falsify any hint clause, because all 0 variables lie on a path, or the sink axiom, because the sink is assigned 0 if at all. Therefore the decision tree ends at a vertex w_t that is set to 0 and all its predecessors are set to 1, hence $\mathrm{bpeb}(R_t \to w_t) = 1$. By item 2 of the invariant the number of 1 answers is at least $\mathrm{bpeb}(G) - 1$. □

To complete the lower bound we use the Pudlák–Impagliazzo Prover–Delayer game [37] where Prover points to a variable, Delayer may answer 0, 1, or $*$, in which case Delayer obtains a point in exchange for letting Prover choose the answer, and the game ends when a clause is falsified.

Lemma 3.6 ([37]). *If Delayer can win p points, then all TreeRes proofs require size at least 2^p.*

Lemma 3.7. *$F \circ \mathrm{OR}$ requires size $\exp(\Omega(\mathrm{DT}_1(\mathrm{Search}(F))))$ in tree-like resolution.*

Proof. We use a strategy for the 1-query game of $\mathrm{Search}(F)$ to ensure that Delayer gets $\mathrm{DT}_1(F)$ points in the Prover–Delayer game. If Prover queries a variable x_i then

- If x is already queried we answer accordingly.
- Otherwise we query x. If the answer is 0 we answer 0, otherwise we answer $*$.

Our strategy ensures that if both x_1 and x_2 are assigned then $x_1 \vee x_2 = x$. Therefore the game only finishes at a leaf of the decision tree, at which point Delayer earns as many points as 1s are present in the path leading to the leaf. The lemma follows by Lemma 3.6. □

[1] Essentially the same notion of one-sided query complexity is used in [36] under the name *positive depth*.

The formulas $\mathrm{PebHint}(P_n) \circ \mathrm{OR}$ are easy to refute in MaxRes (Lemma 3.2), but from Lemmas 3.3,3.5, and 3.7, they are exponentially hard for TreeRes. Hence,

Theorem 3.8. *TreeRes does not simulate MaxResW and MaxRes.*

Note that $\mathrm{DT}_1(f) \leq \mathrm{DT}(f)$ for any relation f, therefore Lemma 3.5 also holds for the standard measure of query complexity. The reason behind using one-sided query complexity is Lemma 3.7, which is false if we replace DT_1 by DT. A counterexample is the standard pebbling formula where the signs of all literals have been flipped, which we denote by $\mathrm{Peb}'(G)$: on the one hand we have that $\mathrm{DT}(\mathrm{Search}(\mathrm{Peb}'(G))) = \Omega(n/\log n)$, and on the other hand there is a tree-like proof of $\mathrm{Peb}'(G) \circ \mathrm{OR}$ of length $\mathrm{O}(n)$.

Alternatively we could use standard query complexity in Lemma 3.7 if we composed our formula with $\oplus$ instead of OR, but that would make the upper bound in Lemma 3.2 more intricate.

4 THE SUBCUBESUMS PROOF SYSTEM

In this section, we explore the power and limitations of the SubCubeSums proof system. On the one hand we show (Theorem 4.1) that it has short proofs of the subset cardinality formulas, known to be hard for resolution but easy for Sherali–Adams. We also give a direct combinatorial argument to show that the pigeonhole principle formulas, known to be hard for resolution but easy in MaxRes with extension, are easy for SubCubeSums. On the other hand we show a lower bound for SubCubeSums for the Tseitin formulas on odd-charged expander graphs (Theorem 4.9). Finally, we establish a technique for obtaining lower bounds on SubCubeSums size: a degree lower bound in SubCubeSums for F translates to a size lower bound in SubCubeSums for $F \circ \oplus$ (Theorem 4.15).

4.1 Res does not simulate SubCubeSums

We now show that Res does not simulate SubCubeSums. We will give two independent proofs using two different formulas: Subset cardinality formulas and the PHP formulas. The result for PHP formulas is implicit in [29], but we provide a new combinatorial proof.

4.1.1 *The Subset Cardinality formulas.*

The first separation is achieved using subset cardinality formulas [34, 39, 42]. These are defined as follows: we have a bipartite graph $G(U \cup V, E)$, with $|U| = |V| = n$. The degree of G is 4, except for two vertices that have degree 5. There is one variable for each edge. For each left vertex $u \in U$ we have a constraint $\sum_{e \ni u} x_e \geq \lceil d(u)/2 \rceil$, while for each right vertex $v \in V$ we have a constraint $\sum_{e \ni v} x_e \leq \lfloor d(v)/2 \rfloor$, both expressed as a CNF. In other words, for each vertex $u \in U$ we have the clauses $\bigvee_{i \in I} x_i$ for $I \in \binom{E(u)}{\lfloor d(u)/2 \rfloor + 1}$, while for each vertex $v \in V$ we have the clauses $\bigvee_{i \in I} \overline{x_i}$ for $I \in \binom{E(v)}{\lfloor d(v)/2 \rfloor + 1}$.

Theorem 4.1. *Subset cardinality formulas have SubCubeSums proofs of combinatorial and algebraic size* $\mathrm{O}(n)$ *but require resolution length* $\exp(\Omega(n))$.

The lower bound requires G to be an expander, and is proven in [34, Theorem 6]. The upper bound is the following lemma.

Lemma 4.2. *Subset cardinality formulas have SubCubeSums proofs of combinatorial and algebraic size* $\mathrm{O}(n)$.

To obtain the size upper bound, it is convenient to use the algebraic formulation of SubCubeSums. Our proof below is presented in this framework. For completeness, we also describe, after this proof, the direct presentation of the subcubes

and a combinatorial argument of correctness. The combinatorial proof is simply an unravelling of the algebraic proof, but can be read independently.

Proof. Our plan is to reconstruct each constraint independently, so that for each vertex we obtain the original constraints $\sum_{e \ni u} x_e \geq \lceil d(u)/2 \rceil$ and $\sum_{e \ni v} \overline{x_e} \geq \lceil d(v)/2 \rceil$, and then add all of these constraints together.

Formally, if F_u is the set of polynomials that encode the constraint corresponding to vertex u, we want to find suitable subcubes h_j and write

$$\sum_{f \in F_u} f - \left(\lceil d(u)/2 \rceil - \sum_{e \ni u} x_e \right) = \sum_j c_{u,j} h_j \tag{1}$$

and

$$\sum_{f \in F_v} f - \left(\lceil d(v)/2 \rceil - \sum_{e \ni v} \overline{x_e} \right) = \sum_j c_{v,j} h_j \tag{2}$$

with $c_{u,j}, c_{v,j} \geq 0$ and $\sum_j c_{u,j} = \mathrm{O}(1)$, so that

$$\begin{aligned}
\sum_{f \in F} f &= \sum_{u \in U} \sum_{f \in F_u} f + \sum_{v \in V} \sum_{f \in F_v} f \\
&= \sum_{u \in U} \left(\lceil d(u)/2 \rceil - \sum_{e \ni u} x_e + \sum_j c_{u,j} h_j \right) + \sum_{v \in V} \left(\lceil d(v)/2 \rceil - \sum_{e \ni v} \overline{x_e} + \sum_j c_{v,j} h_j \right) \\
&= \sum_{u \in U} \lceil d(u)/2 \rceil + \sum_{v \in V} \lceil d(v)/2 \rceil - \sum_{e \in E} (x_e + \overline{x_e}) + \sum_j c_j h_j \\
&= \left(1 + \sum_{u \in U} 2 \right) + \left(1 + \sum_{v \in V} 2 \right) - \sum_{e \in E} 1 + \sum_j c_j h_j \\
&= (2n+1) + (2n+1) - (4n+1) + \sum_j c_j h_j = 1 + \sum_j c_j h_j
\end{aligned}$$

where $c_j = \sum_{v \in U \cup V} c_{v,j} \geq 0$. Hence we can write $\sum_{f \in F} f - 1 = \sum_j c_j h_j$ with $\sum_j c_j = \mathrm{O}(n)$.

It remains to show how to derive equations (1) and (2). The easiest way is to appeal to the implicational completeness of SubCubeSums, Proposition 2.3. We continue deriving equation (1), assuming for simplicity a vertex of degree d and incident edges $[d]$. Let $\overline{x_I} = \prod_{i \in I} \overline{x_i}$, and let $\left\{ \overline{x_I} : I \in \binom{[d]}{d-k+1} \right\}$ represent a constraint $\sum_{i \in [d]} x_i \geq k$. Let $f = \sum_{I \in \binom{[d]}{d-k+1}} \overline{x_I}$ and $g = k - \sum_{i \in [d]} x_i$. For each point $x \in \{0,1\}^d$ we have that either x satisfies the constraint, in which case $f(x) \geq 0 \geq g(x)$, or it falsifies it, in which case we have on the one hand $g(x) = s > 0$, and on the other hand $f(x) = \binom{d-k+s}{d-k+1} = \frac{(d-k+s)\cdots s}{(d-k+1)\cdots 1} \geq s$.

We proved that $f \geq g$, therefore by Proposition 2.3 we can write $f - g$ as a sum of subcubes of size at most $2^d = \mathrm{O}(1)$.

Equation (2) can be derived analogously, completing the proof for SubCubeSums algebraic reduced size, which is the same as combinatorial size.

Since the proof has constant degree, Proposition 2.2 implies that combinatorial and algebraic size are at most a constant factor apart, hence the proof also has algebraic size $\mathrm{O}(n)$. □

In proving the upper bound in Lemma 4.2, we invoked implicational completeness from Proposition 2.3. However, in our case the numbers are small enough that we can show how to derive equation (1) explicitly, by solving the appropriate LP, and without relying on Proposition 2.3. As a curiosity, and in preparation for the combinatorial proof,

Clause \ Vertex Type	$w \in U$ and $\deg(w) = 4$	$w \in U$ and $\deg(w) = 5$	$w \in V$ and $\deg(w) = 4$	$w \in V$ and $\deg(w) = 5$
For $A \in \binom{E_w}{3}$: $\bigvee_{e \in A} x_e$	1 in f_w	1 in f_w		
For $A \in \binom{E_w}{3}$: $\bigvee_{e \in A} \overline{x_e}$			1 in f_w	1 in f_w
$\bigvee_{e \in E_w} x_e$	2 in h_w	7 in h_w	2 in h_w	2 in h_w
$\bigvee_{e \in E_w} \overline{x_e}$	2 in h_w	2 in h_w	2 in h_w	7 in h_w
For $e \in E_w$: $x_e \vee \bigvee_{f \in E_w \setminus \{e\}} \overline{x_f}$	1 in h_w	1 in h_w		2 in h_w
For $e \in E_w$: $\overline{x_e} \vee \bigvee_{f \in E_w \setminus \{e\}} x_f$		2 in h_w	1 in h_w	1 in h_w

Table 1. The sets f_w and h_w: The entries give the multiplicity of the clause in the clause sets depending on the type of vertex w.

we display them next. We have

$$\overline{x_{1,2,3}} + \overline{x_{1,2,4}} + \overline{x_{1,3,4}} + \overline{x_{2,3,4}} - (2 - x_1 - x_2 - x_3 - x_4) = \tag{3}$$
$$2x_1x_2x_3x_4 + x_1x_2x_3\overline{x_4} + x_1x_2\overline{x_3}x_4 + x_1\overline{x_2}x_3x_4 + \overline{x_1}x_2x_3x_4 + 2\overline{x_1x_2x_3x_4}$$

and

$$\begin{aligned}&\overline{x_{1,2,3}} + \overline{x_{1,2,4}} + \overline{x_{1,2,5}} + \overline{x_{1,3,4}} + \overline{x_{1,3,5}} + \overline{x_{1,4,5}} + \overline{x_{2,3,4}} + \overline{x_{2,3,5}} + \overline{x_{2,4,5}} \\ &+ \overline{x_{3,4,5}} - (3 - x_1 - x_2 - x_3 - x_4 - x_5) = \\ &2x_1x_2x_3x_4x_5 + x_1x_2x_3x_4\overline{x_5} + x_1x_2x_3\overline{x_4}x_5 + x_1x_2\overline{x_3}x_4x_5 + x_1\overline{x_2}x_3x_4x_5 \\ &+ \overline{x_1}x_2x_3x_4x_5 + 2\overline{x_1x_2x_3x_4}x_5 + 2\overline{x_1x_2x_3}x_4\overline{x_5} \\ &+ 2\overline{x_1x_2}x_3\overline{x_4x_5} + 2\overline{x_1}x_2\overline{x_3x_4x_5} + 2x_1\overline{x_2x_3x_4x_5} + 7\overline{x_1x_2x_3x_4x_5}\end{aligned} \tag{4}$$

We now give the direct combinatorial proof for the Subset Cardinality Formulas. The Subset Cardinality Formula SCF says that G has a spanning subgraph where each $u \in U$ has degree at least 2, the degree-5 vertex in U has degree at least 3, but each $v \in V$ has degree at most 2.

For $w \in W = U \cup V$, $E_w \subseteq E(G)$ denotes the set of edges incident on w.

For a vertex w, f_w is the set of clauses enforcing the condition at vertex w, and F is the union of these sets. A SubCubeSums proof should give a clause multiset H such that

$$\forall \alpha \in \{0,1\}^{|E(G)|} : \mathrm{viol}_F(\alpha) = 1 + \mathrm{viol}_H(\alpha). \tag{5}$$

In short, $\mathrm{viol}_F = 1 + \mathrm{viol}_H$.

We describe such an H whose clauses are also naturally associated with vertices, so H is the union of clause multisets h_w for each $w \in W$. The clause sets f_w and h_w are described in Table 1.

Towards proving Equation 5, we introduce clause multisets f'_w and h'_w, described in Table 2. (They are not part of the SubCubeSums proof.) Note that h'_w has only empty clauses, so every assignment falsifies all clauses in all the h'_w put together, totalling $4n + 2$. The f'_w clauses together have two clauses per edge $e = (u, v)$: the unit clause x_e in f'_u and the unit clause $\overline{x_e}$ in f'_v. Thus every assignment falsifies exactly $|E| = 4n + 1$ of the clauses in all the f'_w sets put together.

Clause \ Vertex Type	$w \in U$ and $\deg(w) = 4$	$w \in U$ and $\deg(w) = 5$	$w \in V$ and $\deg(w) = 4$	$w \in V$ and $\deg(w) = 5$
For $e \ni w : \overline{x_e}$	1 in f'_w	1 in f'_w		
For $e \ni w : x_e$			1 in f'_w	1 in f'_w
$\square$	2 in h'_w	3 in h'_w	2 in h'_w	3 in h'_w

Table 2. The sets f'_w and h'_w: The entries give the multiplicity of the clause in the clause sets depending on the type of vertex w.

The multisets f'_w and h'_w are related to the multisets f_w and h_w by Equation 6 below, which can be verified by inspection (see Equations 3 and 4 for an example).

$$\forall \alpha \in \{0,1\}^{E(G)}; \forall w \in W : \mathrm{viol}_{f_w}(\alpha) + \mathrm{viol}_{f'_w}(\alpha) = \mathrm{viol}_{h_w}(\alpha) + \mathrm{viol}_{h'_w}(\alpha). \tag{6}$$

Hence

$$\begin{aligned}
\mathrm{viol}_F = \sum_{w \in W} \mathrm{viol}_{f_w} &= \sum_{w \in W} \left(\mathrm{viol}_{h_w} + \mathrm{viol}_{h'_w} - \mathrm{viol}_{f'_w} \right) \\
&= \left(\sum_{w \in W} \mathrm{viol}_{h_w} \right) + \left(\sum_{w \in W} \mathrm{viol}_{h'_w} \right) - \left(\sum_{w \in W} \mathrm{viol}_{f'_w} \right) \\
&= \mathrm{viol}_H + (2|U| + 1) + (2|V| + 1) - \sum_{e \in E(G)} \left(\mathrm{viol}_{x_e} + \mathrm{viol}_{\overline{x_e}} \right) \\
&= \mathrm{viol}_H + (4n + 2) - (4n + 1) = \mathrm{viol}_H + 1
\end{aligned}$$

4.1.2 The Pigeonhole Principle formulas. Recall the definition of the Pigeonhole Principle (PHP) formulas:

Definition 4.3 (PHP$_m$). The clauses of PHP$_m$ are defined as follows:

- Pigeon axioms – For each $i \in [m+1]$, P_i is the clause $\bigvee_{j=1}^{m} x_{i,j}$
- Hole axioms – For each $j \in [m]$, H_j is the collection of clauses $H_{i,i',j} : \neg x_{i,j} \vee \neg x_{i',j}$ for $1 \leq i < i' \leq m+1$.

These formulas are known to be hard for Resolution ([24]).

In [29] the authors show that these formulas are easy to refute in MaxResE, an extended version of MaxRes. This extended version allows intermediate clauses with negative weights, and, interpreting viol as the sum of the weights of the falsified clauses, rather than merely the number of falsified clauses, all rules preserve viol. The system allows introducing certain clauses "out of nowhere" preserving this invariant; in particular, it allows the introduction of triples of weighted clauses of the form $(\square, -1), (x, 1), (\neg x, 1)$. Consider the following set of clauses, called the "residual" of PHP and denoted PHP$^\delta$:

Definition 4.4 (PHP$^\delta$ from Theorem 5 of [29]). The clause set PHP$^\delta$ is the set

$$\bigcup_{i \in [m+1]} P_i^\delta \cup \bigcup_{j \in [m]} H_j^\delta$$

where P_i^δ and H_j^δ are defined as follows:

- The clause set P_i^δ encodes that pigeon i goes into at most one hole. It is the set

$$P_i^\delta = \left\{ \neg x_{i,j} \vee \left(\bigvee_{j<\ell<k} x_{i,\ell} \right) \vee \neg x_{i,k} \;\middle|\; 1 \le j < k \le m \right\}.$$

- The clause set H_j^δ says that hole j has at least one and at most two pigeons. It is defined as $H1_j^\delta \cup H2_j^\delta$, where
  - $H1_j^\delta$ has a single clause encoding that hole j is not empty.

$$H1_j^\delta = \left\{ \bigvee_{i=1}^{m+1} x_{i,j} \right\}.$$

  - $H2_j^\delta$ is a set of clauses encoding that no hole has more than two pigeons. It is the set

$$H2_j^\delta = \left\{ \neg x_{i,j} \vee \left(\bigvee_{i<\ell<k} x_{\ell,j} \right) \vee \neg x_{k,j} \vee \neg x_{i',j} \;\middle|\; 1 \le i < k < i' \le m+1 \right\}.$$

Theorem 4.5 (implicit in [29] Theorem 5). $\mathrm{viol}_{PHP^\delta} = \mathrm{viol}_{PHP} - 1$.

In the proof of Theorem 5 in [29], a MaxResE derivation transforming PHP to $\mathrm{PHP}^\delta \cup \{\square\}$ is described. Each step in the derivation preserves the weighted sum of violations. (At intermediate stages, some clauses have negative weight, hence weighted sum.)

More precisely, the three weighted clauses $(\square, -1), (x, 1), (\neg x, 1)$ have weighted viol = 0: Every assignment falsifies one of the unit clauses with weight +1 and falsifies the empty clause with weight −1, so the total weight of falsified clauses is 0. The derivation in [29] adds m such triples. It uses the weighted-viol-preserving rules of MaxResE to transform $\mathrm{PHP}_m \cup \{(\square, -m)\} \cup \{x_{1,j}, \neg x_{1,j} \mid j \in [m]\}$ to $\mathrm{PHP}^\delta \cup \{\square\}$. Here all clauses of PHP_m initially have weight 1, and all clauses of PHP^δ finally have weight 1. Thus the proof establishes the following statement:

Corollary 4.6. *PHP_m has a SubCubeSums refutation of combinatorial size polynomial in m.*

Proof. The cubes falsifying the $O(m^4)$ clauses of PHP^δ are the SubCubeSums refutation of PHP_m. □

In [29] the authors say (just before Theorem 5 and in the footnote) that it is not obvious that the refutation is complete though we know this because PHP_m is minimally unsat. Actually the fact that PHP^δ is satisfiable is obvious: the assignment that sets $x_{i,i} = 1$ for $i \in [m]$ and all other variables to 0 satisfies PHP^δ. (Any matching of size m satisfies PHP^δ.) Thus, since PHP is minimally unsatisfiable, the MaxSAT value of PHP and $\{\square\} \cup \mathrm{PHP}^\delta$ is the same. However, it is not obvious why $\mathrm{viol}_{\mathrm{PHP}^\delta} = \mathrm{viol}_{\mathrm{PHP}} - 1$. We show how to prove this directly without using the MaxResE derivation route. For every assignment A to the variables of PHP, we show below that $\mathrm{viol}_{\mathrm{PHP}}(A) = \mathrm{viol}_{\mathrm{PHP}^\delta}(A)$.

(1) Let $A \in \{0,1\}^{(m+1)\times m}$ be an assignment to the variables of PHP_m.
(2) Denote the column-sums by $c_j = \sum_{i\in[m+1]} A_{i,j}$ for $j \in [m]$.
(3) Denote the row-sums by $r_i = \sum_{j\in[m]} A_{i,j}$ for $i \in [m+1]$.
(4) Denote the total sum by M; $M = \sum_i r_i = \sum_j c_j$.

It is straightforward to see that

$$\mathrm{viol}_{\mathrm{PHP}}(A) = \#\{i \in [m+1] : r_i = 0\} + \sum_{j\in[m]} \binom{c_j}{2}.$$

To describe $\mathrm{viol}_{\mathrm{PHP}^\delta}(A)$, consider the three sets of clauses separately.

(1) For pigeon i, if $r_i = 0$ or $r_i = 1$, then there are no violations in P_i^δ since each clause has two negated literals. If $r_i \geq 2$, let the positions of the 1s in the ith row be $j_1, j_2, \ldots, j_{r_i}$ in increasing order. Then the only clauses falsified are of the form

$$\neg x_{i,j_p} \vee \left(\bigvee_{\ell=j_p+1}^{j_{p+1}-1} x_{i,\ell} \right) \vee \neg x_{i,j_{p+1}}$$

for $p \in [r_i - 1]$, and all these clauses are falsified. So $\mathrm{viol}_{P_i^\delta}(A) = r_i - 1$.

(2) The clause in $H1_j^\delta$ is falsified iff $c_j = 0$.

(3) For hole j, if $c_j \leq 2$, then there are no violations in $H2_j^\delta$ since each clause has three negated literals. If $c_j \geq 3$, then suppose the 1s are in positions $i_1, i_2, \ldots, i_{c_j}$ in increasing order. Then the clauses violated are exactly those of the form

$$\neg x_{i_q,j} \vee \left(\bigvee_{i=i_q+1}^{i_{q+1}-1} x_{i,j} \right) \vee \neg x_{i_{q+1},j} \vee \neg x_{i_{q+1+k},j}$$

for $q, k \geq 1$ and $q + 1 + k \leq c_j$. So the number of violations is $(c_j - 2) + (c_j - 3) + \ldots + 1 = \binom{c_j - 1}{2}$.

Putting this together, we have

$$\mathrm{viol}_{\mathrm{PHP}^\delta}(A) = \sum_{i \in [m+1] : r_i \geq 2} (r_i - 1) + \#\{j \in [m] : c_j = 0\} + \sum_{j \in [m] : c_j \geq 3} \binom{c_j - 1}{2}.$$

Consider the following manipulations:

$$\begin{aligned}
\sum_{i \in [m+1] : r_i \geq 2} (r_i - 1) &= \sum_{i \in [m+1]} (r_i - 1) - \sum_{i \in [m+1] : r_i = 0} (r_i - 1) \\
&= \left(\sum_{i \in [m+1]} r_i - \sum_{i \in [m+1]} 1 \right) - \Big((-1) \times \text{number of 0-rows} \Big) \\
&= M - (m+1) + \text{number of 0-rows}
\end{aligned}$$

$$\begin{aligned}
\sum_{j \in [m] : c_j \geq 3} \binom{c_j - 1}{2} &= \sum_{j \in [m] : c_j \geq 1} \binom{c_j - 1}{2} = \sum_{j \in [m] : c_j \geq 1} \left[\binom{c_j}{2} - (c_j - 1) \right] \\
&= \sum_{j \in [m] : c_j \geq 1} \binom{c_j}{2} - \sum_{j \in [m] : c_j \geq 1} (c_j - 1) \\
&= \sum_{j \in [m]} \binom{c_j}{2} - \sum_{j \in [m]} c_j + \sum_{j \in [m] : c_j \geq 1} 1 \\
&= \sum_{j \in [m]} \binom{c_j}{2} - M + (m - \text{number of 0-columns})
\end{aligned}$$

Putting this together, we obtain

$$\begin{aligned}
\mathrm{viol}_{\mathrm{PHP}^\delta} &= \sum_{i\in[m+1]:r_i\geq 2}(r_i-1) + \#\{j\in[m] : c_j = 0\} + \sum_{j\in[m]:c_j\geq 3}\binom{c_j-1}{2} \\
&= M-(m+1) + \text{number of 0-rows} \\
&\quad +\text{number of 0-columns} \\
&\quad + \sum_{j\in[m]}\binom{c_j}{2} - M + (m - \text{number of 0-columns}) \\
&= \text{number of 0-rows} + \sum_{j\in[m]}\binom{c_j}{2} - 1 \\
&= \mathrm{viol}_{\mathrm{PHP}} - 1
\end{aligned}$$

as claimed.

In particular, we have the identity:

Proposition 4.7. *For any $A \in \{0,1\}^{(m+1)\times m}$, with row sums $r_i = \sum_j A_{i,j}$ and column sums $c_j = \sum_i A_{i,j}$,*

$$\begin{aligned}
&\#\{i\in[m+1] : r_i = 0\} + \sum_{j\in[m]}\binom{c_j}{2} \\
= \quad & 1 + \#\{j\in[m] : c_j = 0\} + \sum_{i\in[m+1]:r_i\geq 2}(r_i-1) + \sum_{j\in[m]:c_j\geq 3}\binom{c_j-1}{2}
\end{aligned}$$

We can improve Corollary 4.6 to a stronger claim about algebraic size.

Corollary 4.8. *PHP_m has a refutation in SubCubeSums with algebraic size polynomial in m.*

Proof. Viewing the SubCubeSums proof in Corollary 4.6 from the algebraic viewpoint, the degree of the proof is linear. However, the negative degree is 3. So we can still use Proposition 2.2 to conclude that there is a refutation with algebraic size $O(m^4)$. □

4.2 A lower bound for SubCubeSums

Fix any graph G with n nodes and m edges, and let I be the node-edge incidence matrix. Assign a variable x_e for each edge e. Let b be a vector in $\{0,1\}^n$ with $\sum_i b_i \equiv 1 \bmod 2$. The Tseitin contradiction asserts that the system $IX = b$ has a solution over $\mathbb{F}_2$. The CNF formulation has, for each vertex u in G, with degree d_u, a set S_u of 2^{d_u-1} clauses expressing that the parity of the set of variables $\{x_e \mid e \text{ is incident on } u\}$ equals b_u.

For these formulas, Res refutations require exponential size [40], and hence MaxResW refutations also require exponential size. We now show that SubCubeSums refutations also require exponential combinatorial size (and hence also algebraic size). By Theorem 4.1, this lower bound cannot be inferred from hardness for Res.

We will use these standard facts:

Fact 4.1. *For connected graph G, over $\mathbb{F}_2$,*

1. *if $\sum_i b_i \equiv 1 \bmod 2$, then the equations $IX = b$ have no solution.*
2. *If $\sum_i b_i \equiv 0 \bmod 2$, then $IX = b$ has exactly 2^{m-n+1} solutions.*
3. *Furthermore, for any assignment a, and any vertex u, a falsifies at most one clause in S_u.*

A graph is a c-expander if for all $V' \subseteq V$ with $|V'| \leq |V|/2$, $|\delta(V')| \geq c|V'|$, where $\delta(V') = \{(u,v) \in E \mid u \in V', v \in V \setminus V'\}$.

Theorem 4.9. *Let G be a d-regular c-expander on n vertices where n is odd, and c, d be constants with $c > 10$. Let b be the all-1s vector. All SubCubeSums refutations of the Tseitin contradiction corresponding to G, b require combinatorial size exponential in n.*

We prove this using the combinatorial view of SubCubeSums. At a high level, the proof proceeds as follows. The Tseitin contradiction F has $m = dn/2$ variables and $n2^{d-1}$ clauses. The assignments can be partitioned into disjoint sets X_i, where X_i consists of assignments falsifying exactly i clauses of F. By Fact 4.1, X_i is empty for even i. We focus on X_1, X_3, and X_5 for the lower bound.

Let C be a SubCubeSums refutation of F, that is, $\text{viol}_C = \text{viol}_F - 1 = g$. Define a matrix M with rows indexed by assignments to variables and columns indexed by clauses/cubes of C, and entries as follows.

$$M(a, C) = \begin{cases} 1 & \text{if } a \text{ falsifies } C \\ 0 & \text{otherwise} \end{cases}$$

For each $a \in X_i$, row a of M has exactly $(i-1)$ 1s. Thus the submatrix $X_3 \times C$ has $2|X_3|$ 1s, and the submatrix $X_5 \times C$ has $4|X_5|$ 1s. We say that a clause is heavy if it contributes many more 1s in the X_5 rows than in the X_3 rows; otherwise it is light.

The proof idea is to show that a significant fraction of the 1s in $X_3 \times C$ come from light clauses (Lemma 4.10 below), and that a light clause can contribute only an exponentially small fraction of the 1s in $X_3 \times C$ (Lemma 4.11 below). It then follows that C must have exponentially many light clauses.

For a clause $C \in C$, let $N_i(C)$ denote the number of 1s it contributes to M in the rows corresponding to X_i. That is viewing C as the cube of its falsifying assignments, $N_i(C) = |C \cap X_i|$. Define the relative density of a clause C, denoted rel-density(C), to be the ratio $N_5(C)/N_3(C)$. Say that a clause is *light* if rel-density$(C) \leq n^2/9$. That is, for a light C,

$$\text{rel-density}(C) \triangleq \frac{\text{number of 1s in } X_5 \times \{C\}}{\text{number of 1s in } X_3 \times \{C\}} \leq \frac{n^2}{9}.$$

In particular, if C is light, $|C \cap X_3|$ is not zero; hence there is at least one assignment $a \in X_3$ that falsifies C. This fact will be significant.

Lemma 4.10.

$$\frac{\textit{number of 1s in } X_3 \times C \textit{ contributed by light clauses}}{\textit{number of 1s in } X_3 \times C} \geq \frac{1}{10}$$

Lemma 4.11. *For a light clause $C \in C$,*

$$N_3(C) \triangleq |C \cap X_3| \leq \frac{3|X_3|}{2^{n(0.1c-1)}}$$

Before proving these lemmas, we show why they imply the theorem.

Proof. (of Theorem 4.9, assuming Lemmas 4.10,4.11)

$$
\begin{aligned}
2|X_3| &= (\text{number of 1s in } X_3 \times C) \\
&\le 10 \times (\text{number of 1s in } X_3 \times C \text{ contributed by light clauses}) && \text{(by Lemma 4.10)} \\
&\le 10 \times (\text{number of light clauses}) \times (\text{max number of 1s contributed by a light clause}) \\
&\le 10 \times |C| \times \frac{3|X_3|}{2^{n(0.1c-1)}} && \text{(by Lemma 4.11)} \\
\text{Hence } |C| &\ge \frac{2^{n(0.1c-1)}}{15} = 2^{\Omega(n)}.
\end{aligned}
$$

□

Here is a simple proposition that will be used in proving both Lemmas.

Proposition 4.12. *For each odd i, $|X_i| = \binom{n}{i} 2^{m-n+1}$.*

Proof. An assignment in X_i lies in i cubes of f. Each cube corresponds to a distinct vertex because the 2^{d-1} cubes corresponding to any single vertex are disjoint. Once the i vertices are fixed and b flipped in those coordinates to get b', there are 2^{m-n+1} 0-1 solutions to $Ix = b'$ (Fact 4.1(2)). □

Now we prove that many 1s in $X_3 \times C$ are contributed by light clauses.

Proof. (of Lemma 4.10) Consider the following probability distribution μ on C:

$$\mu(C) \triangleq \frac{|C \cap X_3|}{\text{number of 1s in } X_3 \times C} = \frac{|C \cap X_3|}{2|X_3|}.$$

This distribution is useful because it can be used to neatly express the quantity we want to bound from below, as follows:

$$
\begin{aligned}
&\frac{\text{number of 1s in } X_3 \times C \text{ contributed by light clauses}}{\text{number of 1s in } X_3 \times C} \\
&\qquad = \frac{\sum_{C \in \mathcal{C}; C \text{ light}} |C \cap X_3|}{2|X_3|} \\
&\qquad = \sum_{C \in \mathcal{C}; C \text{ light}} \mu(C) \\
&\qquad = \Pr_{C \sim \mu}[C \text{ is light}] \\
&\qquad = 1 - \Pr_{C \sim \mu}\left[\text{rel-density}(C) > \frac{n^2}{9}\right] \\
&\qquad \ge 1 - \frac{\mathbb{E}_{C \sim \mu}[\text{rel-density}(C)]}{n^2/9} \text{ (by Markov's inequality)}
\end{aligned}
$$

So it suffices to show that if a clause C is sampled from $\mathcal{C}$ according to distribution μ, its expected rel-density(C) is small.

Claim 4.2.

$$\mathbb{E}_{C \sim \mu}[\textit{rel-density}(C)] \le \frac{n^2}{10}.$$

Proof. (of claim)

$$\begin{aligned}
\mathbb{E}_{C\sim\mu}\left[\text{rel-density}(C)\right] &= \sum_{C\in\mathcal{C}:\mu(C)\neq 0} \mu(C)\frac{|C\cap X_5|}{|C\cap X_3|} \\
&= \sum_{C\in\mathcal{C}:\mu(C)\neq 0} \frac{|C\cap X_5|}{2|X_3|} && \text{(each row in } X_3\times C \text{ has exactly 2 1s)} \\
&= \frac{1}{2|X_3|}\sum_{C\in\mathcal{C}:\mu(C)\neq 0} |C\cap X_5| \\
&\leq \frac{4|X_5|}{2|X_3|} && \text{(each row in } X_5\times C \text{ has exactly 4 1s)} \\
&= \frac{2\binom{n}{5}}{\binom{n}{3}} && \text{(by proposition 4.12)} \\
&\leq \frac{n^2}{10}.
\end{aligned}$$

□

With this claim established, the proof of the Lemma is complete. □

Now we need to show that light clauses cannot contribute many 1s, Lemma 4.11. We will first obtain, for any $C \in \mathcal{C}$, estimates for $|C \cap X_3|$ and $|C \cap X_5|$ in terms of the width $w(C)$ of C; Lemma 4.13 below. Then we will show that if C is light, then it is wide; Lemma 4.14. Putting these together will prove Lemma 4.11.

To state Lemmas 4.13,4.14 we first need to discuss a suitable subgraph of G. Consider a clause $C \in \mathcal{C}$ with non-empty $C \cap X_3$. Since $\text{viol}_C = \text{viol}_F - 1$, no assignment in X_1 falsifies C. We rewrite the system $IX = b$ as $I'X' + I_C X_C = b$, where X_C are the variables fixed in cube C (to a_C, say). So $I'X' = b + I_C a_C$. An assignment a is in $C \cap X_r$ iff it is of the form $a'a_C$, and a' falsifies exactly r equations in $I'X' = b'$ where $b' = b + I_C a_C$. This is a system for the subgraph G_C where the edges in X_C have been deleted. This subgraph may not be connected, so we cannot use our size expressions from Proposition 4.12 directly. Consider the vertex sets $V_1, V_2, \ldots$ of the components of G_C. The system $I'X' = b'$ can be broken up into independent systems; $I'(i)X'(i) = b'(i)$ for the ith connected component. Say a component is odd-charged if $\sum_{j\in V_i} b'(i)_j \equiv 1 \bmod 2$, even-charged otherwise. Let $|V_i| = n_i$ and $|E_i| = m_i$. Any a' falsifies an odd/even number of equations in an odd-charged/even-charged component.

Pick any $a' \in C \cap X_3$; at least one such assignment exists by assumption. It must falsify three equations overall, so G_C must have either one or three odd-charged components. If it has only one odd-charged component, then there is another assignment in C falsifying just one equation (from this odd-charged component), so $C \cap X_1 \neq \emptyset$, a contradiction. Hence G_C has exactly three odd-charged components, with vertex sets V_1, V_2, V_3 of sizes n_1, n_2, n_3 respectively, and overall $k \geq 3$ components.

We now estimate $|C \cap X_3|$ and $|C \cap X_5|$ in terms of these parameters $n_1, n_2, n_3, k, w(C)$, where $w(C)$ denotes the width of the clause C. Recall that $m = nd/2$ is the number of edges in G and hence the number of variables in F.

Lemma 4.13. *If a clause $C \in \mathcal{C}$ has $|C \cap X_3| \neq 0$, then $|C \cap X_3| = n_1 n_2 n_3 2^{m-w(C)-n+k}$ and*

$$|C \cap X_5| \geq n_1 n_2 n_3 2^{m-w(C)-n+k}\left(\frac{1}{3}\sum_{i=1}^{k}\binom{n_i-1}{2}\right).$$

Proof. An $a \in C \cap X_3$ falsifies exactly one equation in the subsystems $I(1), I(2), I(3)$ corresponding to the odd-charged components of G_C. We thus arrive at the expression

$$|C \cap X_3| = \left(\prod_{i=1}^{3} n_i 2^{m_i - n_i + 1}\right)\left(\prod_{i \geq 4} 2^{m_i - n_i + 1}\right) = n_1 n_2 n_3 2^{m - w(C) - n + k}.$$

Similarly, an $a \in C \cap X_5$ must falsify five equations overall. One each must be from V_1, V_2, V_3. The remaining 2 must be from the same component. Hence

$$\begin{aligned} |C \cap X_5| &= \left(\binom{n_1}{3} n_2 n_3 + n_1 \binom{n_2}{3} n_3 + n_1 n_2 \binom{n_3}{3}\right) 2^{m - w(C) - n + k} \\ &\quad + n_1 n_2 n_3 \sum_{i=4}^{k} \binom{n_i}{2} 2^{m - w(C) - n + k} \\ &\geq n_1 n_2 n_3 2^{m - w(C) - n + k} \left(\frac{1}{3} \sum_{i=1}^{k} \binom{n_i - 1}{2}\right) \end{aligned}$$

□

Now we use the structure and parameters of G_C to show that light clauses must be wide.

Lemma 4.14. *For any clause $C \in \mathcal{C}$, if rel-density$(C) = \frac{|C \cap X_5|}{|C \cap X_3|} \leq \frac{n^2}{9}$, then $w(C) \geq \frac{cn}{10}$.*

Proof. Each literal in C removes one edge from G while constructing G_C. Counting the sizes of the cuts that isolate components of G_C, we count each deleted edge twice. So

$$2w(C) = \sum_{i=1}^{k} |\delta(V_i, V \setminus V_i)| = \sum_{i : n_i \leq n/2} \underbrace{|\delta(V_i, V \setminus V_i)|}_{Q1} + \sum_{i : n_i > n/2} \underbrace{|\delta(V_i, V \setminus V_i)|}_{Q2}$$

By the c-expansion property of G, $Q1 \geq cn_i$.
If $n_i > n/2$, it still cannot be too large because C is light. Recall

$$\frac{n^2}{9} \geq \frac{|C \cap X_5|}{|C \cap X_3|} \geq \frac{1}{3} \sum_{i=1}^{k} \binom{n_i - 1}{2}$$

If any n_i is very large, say larger than $5n/6$, then the contribution from that component alone, $\frac{1}{3}\binom{n_i - 1}{2}$, will exceed $\frac{n^2}{9}$. So each $n_i \leq 5n/6$. Thus even when $n_i > n/2$, we can conclude that $n_i/5 \leq n/6 \leq n - n_i < n/2$. By expansion of $V \setminus V_i$, we have $Q2 \geq c(n - n_i) \geq cn_i/5$.

$$\begin{aligned} 2w(C) &= \sum_{i : n_i \leq n/2} \underbrace{|\delta(V_i, V \setminus V_i)|}_{Q1} + \sum_{i : n_i > n/2} \underbrace{|\delta(V_i, V \setminus V_i)|}_{Q2} \\ &\geq \sum_{i : n_i \leq n/2} cn_i + \sum_{i : n_i > n/2} \frac{cn_i}{5} \geq cn/5 \end{aligned}$$

Hence $w(C) \geq cn/10$ as claimed. □

Now we have all that is needed to prove Lemma 4.11.

Proof. (of Lemma 4.11) Let C be a light clause. As discussed above, let G_C be the subgraph of G where edges whose variables are set by C are deleted, let k be the number of components of G_C, and let n_1, n_2, n_3 be the number of vertices

in the three odd-charged components.

$$\begin{aligned}
|C \cap X_3| &= n_1 n_2 n_3 2^{m-w(C)-n+k} && \text{(by Lemma 4.13)}\\
&= \frac{n_1 n_2 n_3 2^{m-w(C)-n+k}}{\binom{n}{3} 2^{m-n+1}} \times |X_3| && \text{(by Proposition 4.12)}\\
&= \frac{n_1 n_2 n_3}{\binom{n}{3}} 2^{k-w(C)-1} \times |X_3|\\
&\le 6 \times 2^{n-w(C)-1} \times |X_3| = 3 \cdot 2^{n-w(C)} \cdot |X_3|\\
&\le 3 \cdot 2^{n-cn/10} \cdot |X_3| && \text{(by Lemma 4.14)}\\
&= \frac{3|X_3|}{2^{n(0.1c-1)}} && \text{as claimed.}
\end{aligned}$$

This completes the proof of Theorem 4.9. □

Remark. As noted in Section 2, the SubCubeSums proof system can be viewed algebraically as a subsystem of Sherali-Adams, for which this lower bound is already known. However, our proof is specific to the SubCubeSums proof system, where all the multipliers for the axiom polynomials are -1. This is implicit in our proof; we use the equation $\mathrm{viol}_C = \mathrm{viol}_F - 1$, and thus we assume that the axiom polynomials from F are multiplied only by -1.

4.3 Lifting degree lower bounds to size

We describe a general technique to lift lower bounds on width, or conical junta degree, to lower bounds on combinatorial size for SubCubeSums. This is an adaptation of the well-known xorification technique of Alekhnovich and Razborov (see [7]), which also consists of applying a random restriction to a formula composed with parity.

Theorem 4.15. *Let d be the minimum width, or conical junta degree, of a SubCubeSums refutation of an unsatisfiable CNF formula F. Then every SubCubeSums refutation of $F \circ \oplus$ has combinatorial size* $\exp(\Omega(d))$.

Before proving this theorem, we establish two lemmas. For a function $h\colon \{0,1\}^n \to \mathbb{R}$, define the function $h \circ \oplus\colon \{0,1\}^{2n} \to \mathbb{R}$ as $(h \circ \oplus)(\alpha_1, \alpha_2) = h(\alpha_1 \oplus \alpha_2)$, where $\alpha_1, \alpha_2 \in \{0,1\}^n$ and the $\oplus$ in $\alpha_1 \oplus \alpha_2$ is taken bitwise.

Lemma 4.16. $\mathrm{viol}_F(\alpha_1 \oplus \alpha_2) = \mathrm{viol}_{F\circ\oplus}(\alpha_1, \alpha_2)$.

Proof. Fix assignments α_1, α_2 and let $\alpha = \alpha_1 \oplus \alpha_2$. We claim that for each clause $C \in F$ falsified by α there is exactly one clause $D \in F \circ \oplus$ that is falsified by $\alpha_1\alpha_2$. Indeed, by the definition of composed formula the assignment $\alpha_1\alpha_2$ falsifies $C \circ \oplus$, hence the assignment falsifies some clause $D \in C \circ \oplus$. However, the clauses in the CNF expansion of $C \circ \oplus$ have disjoint subcubes, hence $\alpha_1\alpha_2$ falsifies at most one clause from the same block. Observing that if α does not falsify C, then $\alpha_1\alpha_2$ does not falsify any clause in $C \circ \oplus$ completes the proof. □

Note that Lemma 4.16 may not be true for gadgets other than $\oplus$.

Corollary 4.17. $\mathrm{viol}_{F\circ\oplus} - 1 = ((\mathrm{viol}_F) \circ \oplus) - 1 = (\mathrm{viol}_F - 1) \circ \oplus$.

Proof. $((\mathrm{viol}_F - 1) \circ \oplus)(\alpha_1, \alpha_2) = (\mathrm{viol}_F - 1)(\alpha_1 \oplus \alpha_2) = (\mathrm{viol}_F)(\alpha_1 \oplus \alpha_2) - 1 = (\mathrm{viol}_{F\circ\oplus})(\alpha_1, \alpha_2) - 1$. □

Lemma 4.18. *If $f \circ \oplus$ has a (integral) conical junta of size s, then f has a (integral) conical junta of degree $d = \mathrm{O}(\log s)$.*

Proof. Let J be a conical junta of size s that computes $f \circ \oplus$. Let ρ be the following random restriction: for each original variable x of f, pick $i \in \{0,1\}$ and $b \in \{0,1\}$ uniformly and set $x_i = b$. Consider a term C of J of degree at least $d > \log_{4/3} s$. The probability that C is not zeroed out by ρ is at most $(3/4)^d < 1/s$, hence by a union bound the probability that the junta $J\restriction_\rho$ has degree larger than d is at most $s \cdot (3/4)^d < 1$. Hence there is a restriction ρ such that $J\restriction_\rho$ is a junta of degree at most d, although not one that computes f. Since for each original variable x, ρ sets exactly one of the variables x_0, x_1, flipping the appropriate surviving variables—those where x_i is set to 1—gives a junta of degree at most d for f. □

Now we can prove Theorem 4.15.

Proof. We prove the contrapositive: if $F \circ \oplus$ has a SubCubeSums proof of combinatorial size s, then there is an integral conical junta for $g = \mathrm{viol}_F - 1$ of degree $\mathrm{O}(\log s)$.

Let H be the collection of cubes in the SubCubeSums proof for $F \circ \oplus$. So $\mathrm{viol}_{F\circ\oplus} - 1 = \mathrm{viol}_H$. By Corollary 4.17, there is an integral conical junta for $(\mathrm{viol}_F - 1) \circ \oplus$ of size s. By Lemma 4.18 there is an integral conical junta for $\mathrm{viol}_F - 1$ of degree $\mathrm{O}(\log s)$. □

Recovering the Tseitin lower bound: This theorem, along with the $\Omega(n)$ conical junta degree lower bound of [22], yields an exponential lower bound for the SubCubeSums and MaxResW refutation size for Tseitin contradictions. However, this construction duplicates every edge of the original graph and therefore does not give a lower bound for all expanders.

A candidate for separating Res from SubCubeSums: We conjecture that the SubCubeSums degree of the pebbling contradiction on the pyramid graph, or on a minor modification of it (a stack of butterfly networks, say, at the base of a pyramid), is $n^{\Omega(1)}$. This, along with Theorem 4.15 would imply that $F \circ \oplus$ is hard for SubCubeSums, thereby separating it from Res. However we have not yet been able to prove the desired degree lower bound. We do know that SubCubeSums degree is not exactly the same as Res width – for small examples, a brute-force computation has shown SubCubeSums degree to be strictly larger than Res width.

5 DISCUSSION

We placed MaxRes and MaxResW in a propositional proof complexity frame and compared it to more standard proof systems, showing that MaxResW is between tree-like resolution (strictly) and resolution. With the goal of also separating MaxRes and resolution we devised a new lower bound technique, captured by SubCubeSums, and proved lower bounds for MaxRes without relying on Res lower bounds.

Perhaps the most conspicuous problem left open in this paper is whether our conjecture that pebbling contradictions composed with XOR separate Res and SubCubeSums holds. (Very recently, in [18], this has been resolved by showing precisely such a separation.) It remains open to show that MaxRes simulates TreeRes – or even MaxResW – or that they are incomparable instead.

Acknowledgments

Part of this work was done when the last author was at TIFR, Mumbai, India. Some of this work was done in the conducive academic environs of the Chennai Mathematical Institute (during the CAALM workshop of CNRS UMI ReLaX, 2019), Banff International Research Station BIRS (seminar 20w5144) and Schloss Dagstuhl Leibniz Centre for

Informatics (seminar 20061). The authors thank Susanna de Rezende, Tuomas Hakoniemi, and Aaron Potechin for useful discussions.

REFERENCES

[1] Sanjeev Arora and Boaz Barak. 2009. *Computational Complexity – A Modern Approach.* Cambridge University Press. I–XXIV, 1–579 pages.

[2] Albert Atserias and Tuomas Hakoniemi. 2018. *Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs.* Technical Report 1811.01351. arXiv.org.

[3] Albert Atserias and Tuomas Hakoniemi. 2019. Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs. In *Proceedings of the 34th Computational Complexity Conference (CCC '19)*. 24:1–24:20.

[4] Albert Atserias and Massimo Lauria. 2019. Circular (Yet Sound) Proofs. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT '19)*. 1–18.

[5] Albert Atserias, Massimo Lauria, and Jakob Nordström. 2014. Narrow Proofs May Be Maximally Long. In *Proceedings of the 29th Annual IEEE Conference on Computational Complexity (CCC '14)*. 286–297.

[6] Albert Atserias, Massimo Lauria, and Jakob Nordström. 2016. Narrow Proofs May Be Maximally Long. *ACM Transactions on Computational Logic* 17, 3, Article 19 (May 2016), 19:1–19:30 pages. Preliminary version in *CCC '14*.

[7] Eli Ben-Sasson. 2009. Size-Space Tradeoffs for Resolution. *SIAM J. Comput.* 38, 6 (May 2009), 2511–2525. Preliminary version in *STOC '02*.

[8] Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. 2004. Near Optimal Separation of Tree-Like and General Resolution. *Combinatorica* 24, 4 (Sept. 2004), 585–603.

[9] Eli Ben-Sasson and Avi Wigderson. 2001. Short Proofs are Narrow—Resolution Made Simple. *J. ACM* 48, 2 (March 2001), 149–169. Preliminary version in *STOC '99*.

[10] Christoph Berkholz. 2018. The Relation between Polynomial Calculus, Sherali-Adams, and Sum-of-Squares Proofs. In *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS '18) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 96)*. 11:1–11:14.

[11] Archie Blake. 1937. *Canonical expressions in Boolean algebra.* Ph.D. Dissertation. University of Chicago.

[12] Maria Luisa Bonet, Sam Buss, Alexey Ignatiev, João Marques-Silva, and António Morgado. 2018. MaxSAT Resolution With the Dual Rail Encoding. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence, (AAAI '18)*. 6565–6572.

[13] Maria Luisa Bonet and Jordi Levy. 2020. Equivalence Between Systems Stronger Than Resolution. In *Theory and Applications of Satisfiability Testing – SAT 2020*, Luca Pulina and Martina Seidl (Eds.). Springer International Publishing, 166–181.

[14] María Luisa Bonet, Jordi Levy, and Felip Manyà. 2007. Resolution for Max-SAT. *Artificial Intelligence* 171, 8 (2007), 606 – 618.

[15] Stephen A. Cook. 1974. An Observation on Time-Storage Trade Off. *J. Comput. System Sci.* 9, 3 (1974), 308–316. Preliminary version in *STOC '73*.

[16] Stephen A. Cook and Robert A. Reckhow. 1979. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic* 44, 1 (1979), 36–50.

[17] Yuval Filmus, Meena Mahajan, Gaurav Sood, and Marc Vinyals. 2020. MaxSAT Resolution and Subcube Sums. In *Theory and Applications of Satisfiability Testing – SAT 2020*, Luca Pulina and Martina Seidl (Eds.). Springer International Publishing, 295–311.

[18] Noah Fleming, Mika Göös, Stefan Grosser, and Robert Robere. 2022. On Semi-Algebraic Proofs and Algorithms. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA (LIPIcs, Vol. 215)*, Mark Braverman (Ed.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 69:1–69:25. https://doi.org/10.4230/LIPIcs.ITCS.2022.69

[19] Noah Fleming, Pravesh Kothari, and Toniann Pitassi. 2019. Semialgebraic Proofs and Efficient Algorithm Design. *Foundations and Trends in Theoretical Computer Science* 14, 1-2 (2019), 1–221.

[20] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. 2022. Separations in Proof Complexity and TFNP. To appear in the proceedings of the 63rd IEEE Symposium on Foundations of Computer Science (FOCS 2022). Preprint available at https://doi.org/10.48550/arXiv.2205.02168.

[21] Dima Grigoriev, Edward A Hirsch, and Dmitrii V Pasechnik. 2002. Complexity of Semi-algebraic Proofs. In *Proceedings of the 19th International Symposium on Theoretical Aspects of Computer Science (STACS '02) (Lecture Notes in Computer Science, Vol. 2285)*. Springer, 419–430.

[22] Mika Göös, Rahul Jain, and Thomas Watson. 2018. Extension Complexity of Independent Set Polytopes. *SIAM J. Comput.* 47, 1 (Feb. 2018), 241–269.

[23] Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. 2016. Rectangles Are Nonnegative Juntas. *SIAM J. Comput.* 45, 5 (Oct. 2016), 1835–1869. Preliminary version in *STOC '15*.

[24] Amin Haken. 1985. The intractability of Resolution. *Theoretical Computer Science* 39 (1985), 297–308.

[25] Alexey Ignatiev, António Morgado, and Joao Marques-Silva. 2017. On Tackling the Limits of Resolution in SAT Solving. In *Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing (SAT '17)*. 164–183.

[26] Kazuo Iwama and Eiji Miyano. 1995. Intractability of Read-Once Resolution. In *Structure in Complexity Theory Conference*. IEEE Computer Society, 29–36.

[27] Stasys Jukna. 2012. *Boolean Function Complexity - Advances and Frontiers.* Algorithms and combinatorics, Vol. 27. Springer. https://doi.org/10.1007/978-3-642-24508-4

[28] Javier Larrosa, Federico Heras, and Simon de Givry. 2008. A logical approach to efficient Max-SAT solving. *Artifical Intelligence* 172, 2-3 (2008), 204–233.

[29] Javier Larrosa and Emma Rollon. 2020. Augmenting the Power of (Partial) MaxSAT Resolution with Extension. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*.

[30] Javier Larrosa and Emma Rollon. 2020. Towards a Better Understanding of (Partial Weighted) MaxSAT Proof Systems. In *Theory and Applications of Satisfiability Testing – SAT 2020*, Luca Pulina and Martina Seidl (Eds.). Springer International Publishing, 218–232.

[31] Massimo Lauria and Jakob Nordström. 2017. Tight Size-Degree Bounds for Sums-of-Squares Proofs. *Computational Complexity* 26, 3 (Dec. 2017), 911–948. Preliminary version in *CCC '15*.

[32] Bruno Loff and Sagnik Mukhopadhyay. 2019. Lifting Theorems for Equality. In *Proceedings of the 36th Symposium on Theoretical Aspects of Computer Science (STACS '19)*. 50:1–50:19.

[33] Joao Marques-Silva, Alexey Ignatiev, and António Morgado. 2017. Horn Maximum Satisfiability: Reductions, Algorithms and Applications. In *18th EPIA Conference on Artificial Intelligence*. 681–694.

[34] Mladen Mikša and Jakob Nordström. 2014. Long Proofs of (Seemingly) Simple Formulas. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*. 121–137.

[35] Nina Narodytska and Fahiem Bacchus. 2014. Maximum Satisfiability Using Core-Guided MaxSAT Resolution. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*. 2717–2723.

[36] Theodoros Papamakarios and Alexander Razborov. 2022. Space Characterizations of Complexity Measures and Size-Space Trade-Offs in Propositional Proof Systems. In *49th International Colloquium on Automata, Languages, and Programming (ICALP 2022) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 229)*, Mikołaj Bojańczyk, Emanuela Merelli, and David P. Woodruff (Eds.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 100:1–100:20.

[37] Pavel Pudlák and Russell Impagliazzo. 2000. A Lower Bound for DLL Algorithms for k-SAT (Preliminary Version). In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00)*. 128–136.

[38] John Alan Robinson. 1965. A machine-oriented logic based on the resolution principle. *J. ACM* 12 (1965), 23–41.

[39] Ivor Spence. 2010. sgen1: A Generator of Small but Difficult Satisfiability Benchmarks. *Journal of Experimental Algorithmics* 15, Article 1.2 (March 2010), 1.2:1–1.2:15 pages.

[40] Alasdair Urquhart. 1987. Hard Examples for Resolution. *J. ACM* 34, 1 (Jan. 1987), 209–219.

[41] Alasdair Urquhart. 1995. The Complexity of Propositional Proofs. *Bulletin of Symbolic Logic* 1, 4 (1995), 425–467.

[42] Allen Van Gelder and Ivor Spence. 2010. Zero-One Designs Produce Small Hard SAT Instances. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*. 388–397.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il