Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 82 (2020)

MaxSAT Resolution and Subcube Sums*

Yuval Filmus[1], Meena Mahajan[2], Gaurav Sood[2], and Marc Vinyals[1]

[1] Technion – Israel Institute of Technology, Haifa, Israel
yuvalfi@cs.technion.ac.il, marcviny@cs.technion.ac.il
[2] The Institute of Mathematical Sciences (HBNI), Chennai, India
meena@imsc.res.in, gauravs@imsc.res.in

Abstract. We study the 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.

1 Introduction

The most well-studied propositional proof system is Resolution (Res), [Bla37,Rob65]. 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 the maximum number of clauses simultaneously satisfiable is exactly k) is potentially harder since it is hard for both NP and coNP. A proof system for MaxSAT was proposed in [BLM07,LHdG08]. This system, denoted MaxSAT

* A preliminary version of this article appeared in the proceedings of the 23rd International Conference on Theory and Applications of Satisfiability Testing – SAT 2020 [FMSV20]

ISSN 1433-8092

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.

Since MaxRes maintains multi-sets of clauses and replaces used clauses, this suggests a "read-once"-like constraint. However, this is not the case; read-once resolution is not even complete [IM95], 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 [NB14], 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 (conjunction of literals) 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, and can be viewed as a special case of the semi-algebraic Sherali–Adams proof system (see for instance [FKP19,ALN14,Ber18,AH19]). 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 4). 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 11). We exhibit a formula, which is a variant of the pebbling contradiction [BW01] on a pyramid graph, with short refutations in MaxRes (Lemma 5), and show that it requires exponential size in TreeRes (Lemma 10).

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 considering monomial 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 12). We also give a direct combinatorial proof that the pigeon-hole principle formulas have short SubCubeSums refutations (Theorem 16); this fact is implicit in a recent result from [LR20a].

3. We show that the Tseitin contradiction on an odd-charged expander graph is hard for SubCubeSums (Theorem 20) and hence also hard for MaxResW. While this already follows from the fact that these formulas are hard for Sherali–Adams [AH19], 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 21). 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.

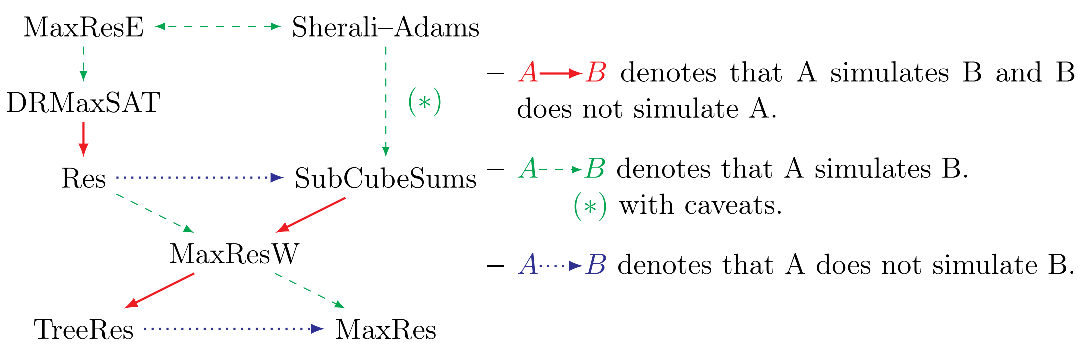


Fig. 1. Relations among various proof systems

Related work

One reason why studying MaxRes is interesting is that it displays unexpected power after some preprocessing. As described in [IMM17] (see also [MIM17]), the PHP formulas that are hard for Resolution can be encoded into MaxHornSAT, and then polynomially many MaxRes steps suffice to expose the contradiction. The underlying proof system, DRMaxSAT, has been studied further in [BBI+18], where it is shown to p-simulate general Resolution. While DRMaxSAT gains power from the encoding, the basic steps are MaxRes steps. Thus, to understand how 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 [LR20a,LR20b,BL20] 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 [LR20a,LR20b] and for certifying unsatisfiability in [BL20]. This difference allows the system to be used in slightly different way in these two papers. Since the satisfiability of a CNF does not change if we assign arbitrary positive weights to the axioms, [BL20] allows doing this. On the other hand, this is not allowed in [LR20a,LR20b] because this would make the system unsound for MaxSAT. With this added power the system in [BL20] is p-equivalent to another recently defined proof system Circular Resolution [AL19], and hence also to Sherali–Adams. Though most results in [LR20a] 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 [BL20]: the system is p-equivalent to Circular Resolution and Sherali–Adams.

It is also worth noting that MaxResW appears in [LR20b] as MaxRes with a split rule, or ResS. It is shown in [LR20a,LR20b] 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 [GLM$^+$16], 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 [GJW18].

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

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. 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 $C_1,\dots,C_t$ 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, [BIW04], for more details.

The MaxRes and MaxResW proof systems

The MaxRes proof system operates on sets of clauses, and uses the MaxSAT resolution rule [BLM07], defined as follows:

$$
\begin{array}{ll}
x \vee a_1 \vee \ldots \vee a_s & (x \vee A) \\
\overline{x} \vee b_1 \vee \ldots \vee b_t & (\overline{x} \vee B) \\
\hline
a_1 \vee \ldots \vee a_s \vee b_1 \vee \ldots \vee b_t & \text{(the "standard resolvent")} \\
\text{(weakenings of } x \vee A) & \text{(weakenings of } \overline{x} \vee B) \\
x \vee A \vee \overline{b}_1 & \overline{x} \vee B \vee \overline{a}_1 \\
x \vee A \vee b_1 \vee \overline{b}_2 & \overline{x} \vee B \vee a_1 \vee \overline{a}_2 \\
\vdots & \vdots \\
x \vee A \vee b_1 \vee \ldots \vee b_{t-1} \vee \overline{b}_t & \overline{x} \vee B \vee a_1 \vee \ldots \vee a_{s-1} \vee \overline{a}_s
\end{array}
$$

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. 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 [BLM07,LHdG08], MaxRes is shown to be complete for MaxSAT, hence also 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.

The SubCubeSums proof system

The SubCubeSums proof system is a static proof system. For an unsatisfiable CNF formula F, a SubCubeSums proof is a multi-set G of sub-cubes (or terms, or conjunctions of literals) satisfying $\mathrm{viol}_F \equiv 1 + \mathrm{viol}_G$. The size of the proof is the number of sub-cubes in G, and the width of the proof is the maximum number of literals in a single term in G.

Stated in this form, SubCubeSums may not be a proof system in the sense of Cook-Reckhow [CR79], since proofs may not be polynomial-time verifiable. However, as we discuss below, proofs are verifiable in randomized polynomial-time, and for a related size measure, they are also polynomial-time-verifiable deterministically.

We can view SubCubeSums as a subsystem of the semialgebraic Sherali–Adams proof system as follows. 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 satisfies clause C_i iff it satisfies equation $f_i = 0$.
(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 $(1-x)y(1-z) = 0$.)
Boolean values are forced through the axioms $x_j^2 - x_j = 0$ for $j \in [n]$. A Sherali–Adams proof is a sequence of polynomials g_i, $i \in [m]$; q_j, $j \in [n]$; and a polynomial p_0 of the form

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

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

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

The degree or rank of the proof is the maximum degree of $g_i f_i$, $q_j(x_j^2 - x_j)$, and p_0. The size of the proof is the total number of monomials in the polynomials $g_i f_i$, $q_j(x_j^2 - x_j)$, and p_0.

The polynomials f_i corresponding to the clauses, as well as the terms 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 [GLM$^+$16], a *conical junta*.

A proof in Sherali–Adams is verified by checking that the claimed polynomial identity is valid. This can be checked deterministically in time polynomial in the monomial size. If the polynomials are not given explicitly as sums of monomials but are represented in some other form, say as circuits, then the identity can be checked by a randomized algorithm in time polynomial in the total representation size.

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), and $\alpha_{A,B} > 0 \implies A \cap B = \emptyset$.

This implies each q_j must be 0, since the rest of the expression is multilinear. Hence, for some non-negative integral $\alpha_{A,B}$,

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

This is exactly the SubCubeSums proof system: the terms in p_0 are subcubes, and the right-hand-side is viol_F. For each disjoint pair A, B, the SubCubeSums proof has $\alpha_{A,B}$ copies of the corresponding sub-cube. The degree of this Sherali–Adams proof is the maximum number of literals appearing in a conjunction; that is, the width of the SubCubeSums proof.

To be pedantically correct, the degree is the maximum of the width of the SubCubeSums proof and the width of the initial formula F. The constraint $g_i = -1$ means that for bounded CNF formulas, the degree of a SubCubeSums proof is essentially the degree of p_0, i.e. the degree of the juntas.

The size of the SubCubeSums proof is the number of subcubes, that is, $\sum_{A,B} \alpha_{A,B}$. This could be much smaller than the monomial size when viewed as a Sherali–Adams proof. However, the Sherali–Adams system may also require large size for some formulas even 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, and include in the set of Boolean axioms the equations $1 - x_i - \overline{x_i} = 0$. (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

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

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 implicitly use this formulation. The restriction where each $g_i = -1$, each $\alpha_{A,B} \in \mathbb{Z}^{\geq 0}$, (non-negative integers), and $\alpha_{A,B} > 0 \implies A \cap B = \emptyset$, gives the SubCubeSums proof system; a proof is an identity

$$-\left(\sum_{i \in [m]} f_i \right) + \left(\sum_{j \in [n]} q_j (x_j^2 - x_j) \right) + \left(\sum_{j \in [n]} r_j (1 - x_j - \overline{x_j}) \right) + \sum_{A,B \subseteq [n]} \alpha_{A,B} \prod_{j \in A} x_j \prod_{j \in B} \overline{x_j} + 1 = 0.$$

(To be precise, a SubCubeSums proof corresponds to an equivalence class of Sherali–Adams proofs modulo Boolean axioms). Now degree is precisely the maximum of the SubCubeSums width and the intitial formula width, and monomial size modulo the Boolean axioms is SubCubeSums size plus the number of initial clauses.

With this algebraic view of SubCubeSums in mind, we can define the **extended size** of a SubCubeSums proof to be the full monomial size of the smallest corresponding Sherali–Adams proof, including Boolean axioms. The relation between SubCubeSums size and extended size is analogous to that between Sherali–Adams monomial size excluding Boolean axioms and full monomial size: in both cases we do not know if the measures are equivalent, but we can prove upper bounds for the larger measure and lower bounds for the smaller measure.

Strictly speaking we do not know if Sherali–Adams simulates SubCubeSums; hence the caveat in Figure 1. However, upper bounds on SubCubeSums extended size imply upper bounds on Sherali–Adams size, while known lower bounds on Sherali–Adams size excluding Boolean axioms imply lower bounds on SubCubeSums size. Hence for all practical purposes we can think as if it did.

The following proposition shows why this restriction of Sherali–Adams remains complete.

Proposition 1. *SubCubeSums p-simulates MaxResW.*

For any unsatisfiable formula with n variables and m clauses, a MaxResW refutation of size s can be converted to a SubCubeSums proof with extended size $O(mn+s)$.

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. A MaxRes step increases it by at most $n-2$.) The subcubes falsifying the clauses in G give a SubCubeSums proof.

The simulation still holds if we measure extended size. To see that, observe that given a monomial m and a set of literals $A = a_1, \ldots, a_s$ we can weaken m as

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

and that 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

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

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

In Section 4.1 we establish size upper bounds in SubCubeSums for certain formulas. To show that these size upper bounds also apply to extended 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. *A SubCubeSums proof of size s and positive (negative) degree d has extended size* $\mathrm{O}(2^d(|F|+s))$.

Proof. We prove the following claim.

Claim. Let p be a polynomial that

1. is multilinear, on $2n$ variables $\{x_i, \overline{x_i} \mid j \in [n]\}$,
2. has $\#mon(p) = s$ monomials,
3. has positive (negative) degree d, and
4. satisfies $p \equiv 0$ modulo the Boolean axioms.

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 $p + q = 0$.

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 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 $\big(1 + 3 \cdot (2^d - 1)\big) \cdot (|F| + |p_0| + 1)$. ☐

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. Modulo Boolean axioms, $p' \equiv 0$, since both $p \equiv 0$ and $q'' \equiv 0$. 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 Boolean x, $f(x) \geq g(x)$.

Proposition 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/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 4. *For every unsatisfiable CNF F, $size(F \vdash_{MaxResW} \Box) \leq 2size(F \vdash_{TreeRes} \Box)$.*

Proof. Let T be a tree-like derivation of $\Box$ from F of size s. Without loss of generality, we may assume that T is regular; 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 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. $\Box$

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.

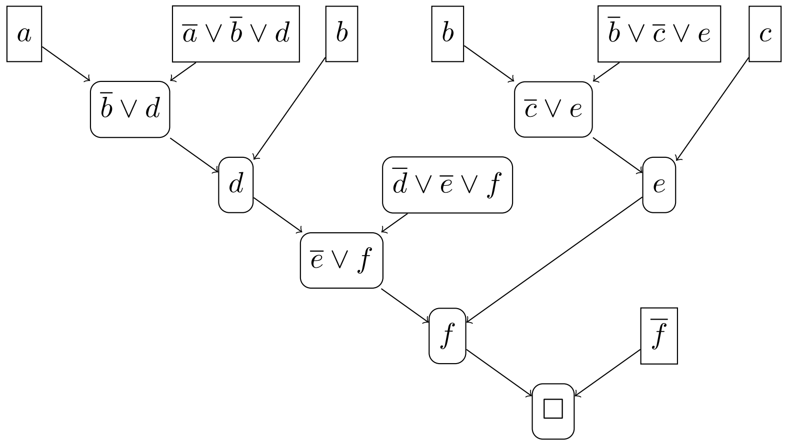


Fig. 2. A tree-like resolution proof

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 $((u_1 \vee u_2) \wedge (v_1 \vee v_2)) \rightarrow (w_1 \vee w_2)$, 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 $u_1 \vee u_2 \vee v_1 \vee v_2$.

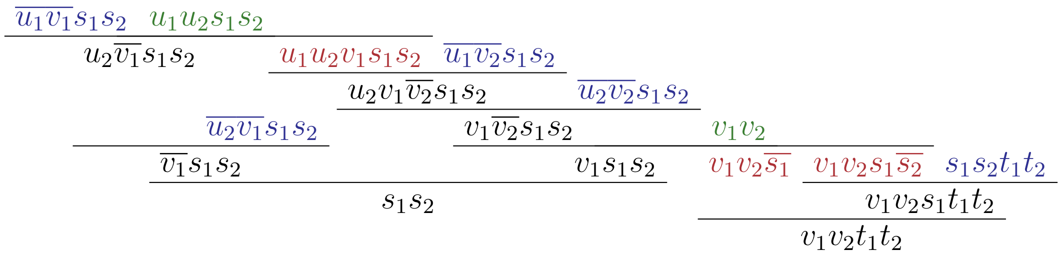
Note that the first three types of clauses are also present in standard composed pebbling formulas, while the last type are the hints.

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 5. *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.



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 $\Theta(n)$. $\square$

We can prove a tree-like lower bound along the lines of [BIW04], but with some extra care to respect the hints. As in [BIW04] 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 6 ([Coo74]). $\mathrm{bpeb}(P_h) = h + 1$.

Lemma 7 ([BIW04]). *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 [BIW04] and, instead of directly translating a proof to a pebbling strategy, we go through query complexity. 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 [LM19], that is the minimum over all decision trees computing f of the maximum of 1-answers that the decision tree receives.

Lemma 8. *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 $\text{bpeb}(G) - \text{bpeb}(R_i \to w_i)$.

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 $p_0 = \text{bpeb}(R_i \to v)$ and $p_1 = \text{bpeb}(R_i \cup \{v\} \to w_i)$. By Lemma 7, $\text{bpeb}(R_i \to w_i) \leq \max(p_0, p_1 + 1)$. 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 $p_0 \geq \text{bpeb}(R_i \to w_i)$, item 2 is also preserved. Otherwise we answer 1 and since $p_1 \geq \text{bpeb}(R_i \to w_i) - 1$ 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 $\text{bpeb}(R_t \to w_t) = 1$. By item 2 of the invariant the number of 1 answers is at least $\text{bpeb}(G) - 1$. □

To complete the lower bound we use the Pudlák–Impagliazzo Prover–Delayer game [PI00] 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 9 ([PI00]). *If Delayer can win p points, then all TreeRes proofs require size at least 2^p.*

Lemma 10. $F \circ \text{OR}$ *requires size* $\exp(\Omega(\text{DT}_1(\text{Search}(F))))$ *in tree-like resolution.*

Proof. We use a strategy for the 1-query game of $\text{Search}(F)$ to ensure that Delayer gets $\text{DT}_1(F)$ 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 9. □

The formulas $\text{PebHint}(P_n) \circ \text{OR}$ are easy to refute in MaxRes (Lemma 5), but from Lemmas 6,8, and 10, they are exponentially hard for TreeRes. Hence,

Theorem 11. *TreeRes does not simulate MaxResW and MaxRes.*

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 12) 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 20). 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 21).

4.1 Res does not simulate SubCubeSums

We now show that Res does not simulate SubCubeSums.

Theorem 12. *There are formulas that have SubCubeSums proofs of size* $\mathrm{O}(n)$ *but require resolution length* $\exp(\Omega(n))$.

The first separation is achieved using subset cardinality formulas [Spe10,VS10,MN14]. 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 $I \in \binom{E(v)}{\lfloor d(v)/2 \rfloor + 1}$.

The lower bound requires G to be an expander, and is proven in [MN14, Theorem 6]. The upper bound is the following lemma.

Lemma 13. *Subset cardinality formulas have SubCubeSums proofs of (extended) size* $\mathrm{O}(n)$.

To obtain the size upper bound, it is convenient to use the Sherali–Adams 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 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 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 $f(x) = \binom{d-k+s}{d-k+1} = \frac{(d-k+s)\cdots\cdots s}{(d-k+1)\cdots\cdots 1} \geq s$.

We proved that $f \geq g$, therefore by Proposition 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 size.

Since the proof has constant degree, Proposition 2 implies that size and extended size are at most a constant factor apart, hence the proof also has extended size $\mathrm{O}(n)$. $\square$

In proving the upper bound in Lemma 13, we invoked implicational completeness from Proposition 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 3. As a curiosity, and in preparation for the combinatorial proof, we display them next. We have

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

and

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

Direct combinatorial SubCubeSums proof. 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 clauses f_w and h_w are described in the table below. The table also has two more clause multisets f'_w and h'_w; these are not part of the SubCubeSums proof but are used to prove that Equation 5 is indeed satisfied.

The entries in the table give the multiplicity of the clause in the clause sets:

Vertex Type / Clause	$w \in U$; degree 4	$w \in U$; degree 5	$w \in V$; degree 4	$w \in V$; degree 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
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
$\bigvee_{e \in E_w} x_e$	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

Equation 6 below can now 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

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

The Pigeonhole Principle formulas are easy for SubCubeSums. Recall the definition of the pigeonhole principle PHP formulas:

Definition 14 (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 $1 \le i < i' \le m+1$.*

In [LR20a] 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 15 (PHP$^\delta$ from Theorem 5 of [LR20a]). *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*

- *$H1^\delta_j$ has a single clause encoding that hole j is not empty.*

$$H1^\delta_j = \left\{ \bigvee_{i=1}^{m+1} x_{i,j} \right\}.$$

- *$H2^\delta_j$ is a set of clauses encoding that no hole has more than two pigeons. It is the set*

$$H2^\delta_j = \left\{ \neg x_{i,j} \vee \left(\bigvee_{i<\ell<k} x_{\ell,j} \right) \vee \neg x_{k,j} \vee \neg x_{i',j} \;\middle|\; 1 \le i < k < i' \le m+1 \right\}.$$

Theorem 16 (implicit in [LR20a] Theorem 5). $\mathrm{viol}_{PHP^\delta} = \mathrm{viol}_{PHP} - 1$.

In the proof of Theorem 5 in [LR20a], a MaxResE derivation transforming PHP to $\mathrm{PHP}^\delta \cup \{\Box\}$ 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 $(\Box, -1), (x, 1), (\neg x, 1)$ 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 [LR20a] adds m such triples. It uses the weighted-viol-preserving rules of MaxResE to transform $\mathrm{PHP}_m \cup \{(\Box, -m)\} \cup \{x_{1,j}, \neg x_{1,j} \mid j \in [m]\}$ to $\mathrm{PHP}^\delta \cup \{\Box\}$. 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 17. *PHP_m has a short (poly-size) refutation in SubCubeSums.*

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

In [LR20a] 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 $\{\Box\} \cup \mathrm{PHP}^\delta$ 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 18. *For any $A \in \{0,1\}^{(m+1)\times m}$, with row sums $r_i = \sum_j A_{i,j}$ and column sums $c_j = \sum_i A_{i,j}$,*

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

We can improve Corollary 17 to a stronger claim about extended size.

Corollary 19. *PHP_m has a refutation in SubCubeSums with polynomial extended size.*

Proof. Viewing the SubCubeSums proof in Corollary 17 from the algebraic viewpoint, the degree of the proof is linear. However, the negative degree is 3. So we can still use Proposition 2 to conclude that there is a refutation with extended 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.

These formulas are exponentially hard for Res [Urq87], and hence are also hard for MaxResW. We now show that they are also hard for SubCubeSums. By Theorem 12, this lower bound cannot be inferred from hardness for Res.

We will use some standard facts: For connected graph G, over $\mathbb{F}_2$, if $\sum_i b_i \equiv 1 \bmod 2$, then the equations $IX = b$ have no solution, and if $\sum_i b_i \equiv 0 \bmod 2$, then $IX = b$ has exactly 2^{m-n+1} solutions. 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 20. *Tseitin contradictions on odd-charged expanders require exponential size SubCubeSums refutations.*

Proof. Fix a graph G that is a d-regular c-expander on n vertices, where n is odd; $m = dn/2$. Let b be the all-1s vector. The Tseitin contradiction F has $n2^{d-1}$ clauses. By the facts mentioned above, for all $a \in \{0,1\}^m$, $\mathrm{viol}_F(a)$ is odd. So viol_F partitions $\{0,1\}^m$ into $X_1, X_3, \ldots, X_{N-1}$, where $X_i = \mathrm{viol}_F^{-1}(i)$.

Let $\mathcal{C}$ be a SubCubeSums refutation of F, that is, $\mathrm{viol}_{\mathcal{C}} = \mathrm{viol}_F - 1 = g$, say. For a cube C, define $N_i(C) = |C \cap X_i|$. Then for all $C \in \mathcal{C}$, $N_1(C) = 0$, and so C is partitioned by X_i, $i \geq 3$. Let $\mathcal{C}'$ be those cubes of $\mathcal{C}$ that have a non-empty part in X_3. We will show that $\mathcal{C}'$ is large. In fact, we will show that for a suitable S, the set $\mathcal{C}'' \subseteq \mathcal{C}'$ of cubes with $|C \cap X_5| \leq S|C \cap X_3|$ is large.

Defining the probability distribution μ on $\mathcal{C}'$ as

$$\mu(C) = \frac{|C \cap X_3|}{\sum_{D \in \mathcal{C}'} |D \cap X_3|} = \frac{N_3(C)}{\sum_{D \in \mathcal{C}'} N_3(D)} \quad ,$$

$$|\mathcal{C}'| = \sum_{C \in \mathcal{C}'} 1 = \mathop{\mathbb{E}}_{C \sim \mu}\left[\frac{1}{\mu(C)}\right] \geq \underbrace{\mathop{\mathbb{E}}_{C \sim \mu}\left[\frac{1}{\mu(C)} \middle| \frac{|C \cap X_5|}{|C \cap X_3|} \leq S\right]}_{A} \cdot \underbrace{\Pr_{\mu}\left[\frac{|C \cap X_5|}{|C \cap X_3|} \leq S\right]}_{B} \tag{7}$$

We want to choose a good value for S so that A is very large and B is sufficiently large, $\Theta(1)$. To see what will be a good value for S, we estimate the expected value of $\frac{|C \cap X_5|}{|C \cap X_3|}$ and then use Markov's inequality. For this, we should understand the sets X_3, X_5 better. These set sizes are known precisely: for each odd i, $|X_i| = \binom{n}{i} 2^{m-n+1}$. (An assignment lies in i cubes of f, each cube corresponds to a distinct vertex because the 2^{d-1} cubes corresponding to a 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 $Ix = b'$.)

Now let us consider $C \cap X_3$ and $C \cap X_5$ for a $C \in \mathcal{C}'$ (that is, we know $C \cap X_3 \neq \emptyset$ and $C \cap X_1 = \emptyset$). 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 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 if $\sum_{j \in V_i} b'(i)_j \equiv 1 \bmod 2$, even otherwise. Let $|V_i| = n_i$ and $|E_i| = m_i$. Any a' falsifies an odd/even number of equations in an odd/even component.

For $a' \in C \cap X_3$, it must falsify three equations overall, so G_C must have either one or three odd components. If it has only one odd component, then there is another assignment in C falsifying just one equation (from this odd component), so $C \cap X_1 \neq \emptyset$, a contradiction. Hence G_C has exactly three odd components, with vertex sets V_1, V_2, V_3, and overall $k \geq 3$ components. An $a \in C \cap X_3$ falsifies exactly one equation in $I(1), I(2), I(3)$. 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

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

Hence we have, for $C \in \mathcal{C}'$, $\dfrac{|C \cap X_5|}{|C \cap X_3|} \geq \dfrac{1}{3} \displaystyle\sum_{i=1}^{k} \binom{n_i - 1}{2}$.

This alone does not tell us enough unless we can say something about the n_i's. But we can deduce more by using the definition of μ, and the following fact: Since $g = \mathrm{viol}_F - 1$, an assignment in X_3 belongs to exactly two cubes in $\mathcal{C}$, and by definition these cubes are in $\mathcal{C}'$. Similarly, an assignment in X_5 belongs to exactly four cubes in $\mathcal{C}$, not all of which may be in $\mathcal{C}'$. Hence

$$\sum_{C \in \mathcal{C}'} |C \cap X_3| = 2|X_3| = 2\binom{n}{3} 2^{m - n + 1}$$

$$\sum_{C \in \mathcal{C}'} |C \cap X_5| \leq 4|X_5| = 4\binom{n}{5} 2^{m - n + 1}$$

$$\mu(C) = \frac{|C \cap X_3|}{2|X_3|}$$

Now we can estimate the average:

$$\mathop{\mathbb{E}}_{\mu}\left[\frac{|C \cap X_5|}{|C \cap X_3|}\right] = \sum_{C \in \mathcal{C}'} \mu(C) \frac{|C \cap X_5|}{|C \cap X_3|} = \sum_{C \in \mathcal{C}'} \frac{|C \cap X_5|}{2|X_3|} \leq \frac{4|X_5|}{2|X_3|} \leq \frac{n^2}{10}$$

Choosing $S = n^2/9$, and using Markov's inequality, we get

$$B = \Pr_{\mu}\left[\frac{|C \cap X_5|}{|C \cap X_3|} \leq S = \frac{n^2}{9}\right] \geq 1/10$$

Now we show that conditioned on $\frac{|C \cap X_5|}{|C \cap X_3|} \leq S$, the average value of $\frac{1}{\mu(C)}$ is large.

$$\frac{1}{\mu(C)} = \frac{2|X_3|}{|C \cap X_3|} = \frac{2\binom{n}{3}2^{m-n+1}}{n_1 n_2 n_3 2^{m-w(C)-n+k}} = \frac{2\binom{n}{3}2^{w(C)+1-k}}{n_1 n_2 n_3} \geq \frac{2^{w(C)+1-n}}{3}$$

So we must show that $w(C)$ must be large. 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 of the conditioning. Recall

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

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 our chosen $S = \frac{n^2}{9}$. 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 $Q2 \geq c(n - n_i) \geq cn_i/5$.

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

Choose c-expanders where c ensures $w(C) + 1 - n = \Omega(n)$. (Any constant $c > 10$.) Going back to our calculation of A from Equation 7),

$$A = \mathop{\mathbb{E}}_{C \sim \mu}\left[\frac{1}{\mu(C)}\middle|\frac{|C \cap X_5|}{|C \cap X_3|} \leq S\right] \geq \mathop{\mathbb{E}}_{C \sim \mu}\left[\frac{2^{w(C)+1-n}}{3}\middle|\frac{|C \cap X_5|}{|C \cap X_3|} \leq S\right] = 2^{\Omega(n)}$$

for suitable $c > 10$. Thus $|\mathcal{C}| \geq |\mathcal{C}'| \geq A \cdot B \geq 2^{\Omega(n)} \cdot (1/10)$. □

4.3 Lifting degree lower bounds to size

We describe a general technique to lift lower bounds on conical junta degree to size lower bounds for SubCubeSums.

Theorem 21. *Let d be the minimum degree of a SubCubeSums refutation of an unsatisfiable CNF formula F. Then every SubCubeSums refutation of $F \circ \oplus$ has 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 22. $\mathrm{viol}_F(\alpha_1 \oplus \alpha_2) = \mathrm{viol}_{F\circ\oplus}(\alpha_1, \alpha_2)$.

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 22 may not be true for gadgets other than $\oplus$.

Corollary 23. $\mathrm{viol}_{F\circ\oplus} - 1 = ((\mathrm{viol}_F) \circ \oplus) - 1 = (\mathrm{viol}_F - 1) \circ \oplus$.

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 24. *If $f \circ \oplus_2$ 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 $f \circ \oplus_2$. 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 21.

Proof. We prove the contrapositive: if $F \circ \oplus$ has a SubCubeSums proof of 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 23, there is an integral conical junta for $(\mathrm{viol}_F - 1) \circ \oplus$ of size s. By Lemma 24 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 [GJW18], yields an exponential lower bound for the SubCubeSums and MaxResW refutation size for Tseitin contradictions.

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 21 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(W) 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 open problem is whether our conjecture that pebbling contradictions composed with XOR separate Res and SubCubeSums holds. It also 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

AH19. Albert Atserias and Tuomas Hakoniemi. Size-degree trade-offs for sums-of-squares and Positivstellensatz proofs. In *Proceedings of the 34th Computational Complexity Conference (CCC '19)*, pages 24:1–24:20, July 2019.

AL19. Albert Atserias and Massimo Lauria. Circular (yet sound) proofs. In *Proceedings of the 22nd International Conference on Theory and Applications of Satisfiability Testing (SAT '19)*, pages 1–18, July 2019.

ALN14. Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. In *Proceedings of the 29th Annual IEEE Conference on Computational Complexity (CCC '14)*, pages 286–297, June 2014.

BBI$^+$18. Maria Luisa Bonet, Sam Buss, Alexey Ignatiev, João Marques-Silva, and António Morgado. MaxSAT resolution with the dual rail encoding. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence, (AAAI '18)*, pages 6565–6572, 2018.

Ber18. Christoph Berkholz. 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)*, volume 96 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:14, February 2018.

BIW04. Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of tree-like and general resolution. *Combinatorica*, 24(4):585–603, September 2004.

BL20. Maria Luisa Bonet and Jordi Levy. Equivalence between systems stronger than resolution. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, pages 166–181. Springer International Publishing, 2020.

Bla37. Archie Blake. *Canonical expressions in Boolean algebra.* PhD thesis, University of Chicago, 1937.

BLM07. María Luisa Bonet, Jordi Levy, and Felip Manyà. Resolution for Max-SAT. *Artificial Intelligence*, 171(8):606 – 618, 2007.

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

Coo74. Stephen A. Cook. An observation on time-storage trade off. *Journal of Computer and System Sciences*, 9(3):308–316, 1974. Preliminary version in *STOC '73*.

CR79. Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1979.

FKP19. Noah Fleming, Pravesh Kothari, and Toniann Pitassi. Semialgebraic proofs and efficient algorithm design. *Foundations and Trends in Theoretical Computer Science*, 14(1-2):1–221, 2019.

FMSV20. Yuval Filmus, Meena Mahajan, Gaurav Sood, and Marc Vinyals. MaxSAT Resolution and Subcube Sums. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, pages 295–311. Springer International Publishing, 2020.

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

GLM$^+$16. Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, October 2016. Preliminary version in *STOC '15*.

IM95. Kazuo Iwama and Eiji Miyano. Intractability of read-once resolution. In *Structure in Complexity Theory Conference*, pages 29–36. IEEE Computer Society, 1995.

IMM17. Alexey Ignatiev, António Morgado, and Joao Marques-Silva. 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)*, pages 164–183, 2017.

LHdG08. Javier Larrosa, Federico Heras, and Simon de Givry. A logical approach to efficient Max-SAT solving. *Artifical Intelligence*, 172(2-3):204–233, 2008.

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

LR20a. Javier Larrosa and Emma Rollon. Augmenting the power of (partial) MaxSAT resolution with extension. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 2020.

LR20b. Javier Larrosa and Emma Rollon. Towards a better understanding of (partial weighted) maxsat proof systems. In Luca Pulina and Martina Seidl, editors, *Theory and Applications of Satisfiability Testing – SAT 2020*, pages 218–232. Springer International Publishing, 2020.

MIM17. Joao Marques-Silva, Alexey Ignatiev, and António Morgado. Horn maximum satisfiability: Reductions, algorithms and applications. In *18th EPIA Conference on Artificial Intelligence*, pages 681–694, 2017.

MN14. Mladen Mikša and Jakob Nordström. Long proofs of (seemingly) simple formulas. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14)*, pages 121–137, July 2014.

NB14. Nina Narodytska and Fahiem Bacchus. Maximum satisfiability using core-guided MaxSAT resolution. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 2717–2723, 2014.

PI00. Pavel Pudlák and Russell Impagliazzo. 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)*, pages 128–136, January 2000.

Rob65. John Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12:23–41, 1965.

Spe10. Ivor Spence. sgen1: A generator of small but difficult satisfiability benchmarks. *Journal of Experimental Algorithmics*, 15:1.2:1–1.2:15, March 2010.

Urq87. Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, January 1987.

VS10. Allen Van Gelder and Ivor Spence. Zero-one designs produce small hard SAT instances. In *Proceedings of the 13th International Conference on Theory and Applications of Satisfiability Testing (SAT '10)*, pages 388–397, July 2010.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il