A study of QBF Merge Resolution and MaxSAT Resolution

By

Gaurav Sood

MATH10201604006

The Institute of Mathematical Sciences, Chennai

A thesis submitted to the

Board of Studies in Mathematical Sciences

In partial fulfillment of requirements

for the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

March, 2023

Homi Bhabha National Institute

Recommendations of the Viva Voce Committee

As members of the Viva Voce Committee, we certify that we have read the dissertation prepared by Gaurav Sood entitled "A study of QBF Merge Resolution and MaxSAT Resolution" and recommend that it may be accepted as fulfilling the thesis requirement for the award of Degree of Doctor of Philosophy.

Date: March 7, 2023

Chairman - Venkatesh Raman

Date: March 7, 2023

Guide/Convenor - Meena Mahajan

Date: March 7, 2023

Examiner - Prahladh Harsha

Date: March 7, 2023

Member 1 - V. Arvind

Date: March 7, 2023

Member 2 - Vikram Sharma

Date: March 7, 2023

Member 3 - Prajakta Nimbhorkar

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to HBNI.

I hereby certify that I have read this thesis prepared under my direction and recommend that it may be accepted as fulfilling the thesis requirement.

Date: March 7, 2023

Place: Chennai Guide

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at Homi Bhabha National Institute (HBNI) and is deposited in the Library to be made available to borrowers under rules of the HBNI.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the Competent Authority of HBNI when in his or her judgement the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

Gaurav Sood

DECLARATION

I hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.

Gaurav Sood

LIST OF PUBLICATIONS ARISING FROM THE THESIS

Journal

1. MaxSAT Resolution and Subcube Sums, Yuval Filmus, Meena Mahajan, Gaurav Sood and Marc Vinyals, ACM Transactions on Computational Logic, 2023, Vol. 24(1), pp. 8:1–8:27.

Conferences

1. MaxSAT Resolution and Subcube Sums, Yuval Filmus, Meena Mahajan, Gaurav Sood and Marc Vinyals, In 23rd International Conference on Theory and Applications of Satisfiability Testing (SAT 2020), Lecture Notes in Computer Science (LNCS), Vol. 12178, pp. 295–311.
2. Hard QBFs for Merge Resolution, Olaf Beyersdorff, Joshua Blinkhorn, Meena Mahajan, Tomáš Peitl and Gaurav Sood, In 40th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2020), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 182, pp. 12.1–12.15.
3. QBF Merge Resolution is powerful but unnatural, Meena Mahajan and Gaurav Sood, In 23rd International Conference on Theory and Applications of Satisfiability Testing (SAT 2020), Leibniz International Proceedings in Informatics (LIPIcs), vol. 236, pp. 22.1–22.19.

Gaurav Sood

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor Prof. Meena Mahajan for her guidance and support. I thank her for patiently listening to my half-baked ideas and giving valuable feedback. Her questions always clarified my thoughts and made the ideas more precise.

I would also like to thank my co-authors Yuval Filmus, Marc Vinyals Olaf, Beyersdorff, Joshua Blinkhorn and Tomáš Peitl for their deep insights.

I would also like to thank the members of this committee for the feedback. Their suggestions have greatly improved this thesis.

I am grateful to my M.Tech advisor Dr. K. Murali Krishnan for his guidance. Even after completing my masters, he has been a mentor to me providing guidance and encouragement.

I want to thank my friends Abhranil, Ashwin, Yogesh, Subhankar, Semanti, Ujjal, Sujoy, Ajjath and Ramit for making this journey more enjoyable.

Finally, I would like to thank my family for unwavering support and encouragement during this journey.

Contents

Summary

In this thesis, we study the proof complexity of two proof systems: (i) Merge Resolution proof system for Quantified Boolean Formulas (QBFs), and (ii) MaxSAT Resolution proof system for certifying unsatisfiability.

Merge Resolution

Merge Resolution (M-Res) is a proof system for Quantified Boolean Formulas (QBFs), proposed in [18]. The original motivation was to overcome the limitations encountered in long-distance Q-Resolution proof system (LD-Q-Res), where the syntactic side-conditions, while prohibiting all unsound resolutions, also end up prohibiting some sound resolutions. However, while the advantage of M-Res over many other resolution-based QBF proof systems was already demonstrated, a comparison with LD-Q-Res itself had remained open. Here, we settle this question. We show that M-Res has an exponential advantage over not only LD-Q-Res, but even over LQU$^+$-Res and IRM, the most powerful among currently known resolution-based QBF proof systems.

We also show the first exponential lower bound for M-Res, thereby uncovering its limitations. Combining this lower bound with upper bounds for M-Res in [18] (for QU-Res and $\mathrm{CP} + \forall\mathrm{Red}$) and those in this thesis (for LQU-Res and LQU$^+$-Res), we conclude that these four proof systems are incomparable with M-Res.

Our proof method reveals two additional and curious features about M-Res:

(i) M-Res is not closed under restrictions, and is hence not a natural proof system, and (ii) weakening axiom clauses with existential variables provably yields an exponential advantage over M-Res without weakening. We further show that in the context of regular derivations, weakening axiom clauses with universal variables provably yields an exponential advantage over M-Res without weakening. These results suggest that M-Res is better used with weakening, though whether M-Res with weakening is closed under restrictions remains open. We note that even with weakening, M-Res continues to be simulated by eFrege + $\forall$red (the simulation of ordinary M-Res was shown recently in [30]).

MaxSAT Resolution

MaxSAT Resolution (MaxRes) is a proof system for the MaxSAT problem, proposed in [28, 53]. We study the proof complexity of this system. In particular, we compare it with standard proof systems. To have a fair comparison with proof systems which only certify unsatisfiability (instead of the MaxSAT value), we use MaxRes for certifying unsatisfiability.

We show that MaxRes 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. 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.

List of Figures

Chapter 1

Introduction

Computational complexity theory aims to classify computational problems by their intrinsic difficulty. Computational problems are placed into buckets, called complexity classes, and the goal is to find the relationships among these classes. The most well-known are the classes P and NP. Class P is the set of problems solvable by deterministic Turing machines in polynomial time. On the other hand, class NP is the set of problems solvable by non-deterministic Turing machines in polynomial time. The question about their relationship — whether P is a proper subset of NP — is the most important question in the field. In other words, the question asks whether there exists a problem in NP which requires super-polynomial time on deterministic Turing machines. To solve this question, we will have to prove a super-polynomial 'lower bound' on the worst-case runtime of every deterministic Turing machine which solves the problem.

Despite continued effort for nearly half a century, such a lower bound remains elusive. This failure has led to a less ambitious but more realistic goal — to prove lower bounds on computational models weaker than Turing machines. To be precise, given a weak computational model, the goal is show that some problem in NP requires super-polynomial resources when solved on this computational model.

Examples of such models include branching programs and various restrictions of uniform circuits, for example (uniform versions of) Boolean formulas, monotone circuits, depth-restricted circuits and arithmetic circuits.

Since this sub-area of complexity theory focuses on 'concrete' computational models in contrast to the all-encompassing Turing machine model, it is called *concrete complexity theory.* There has been some progress, for instance exponential size lower bounds have been proven for monotone [1, 66] and constant depth circuits [45, 76], even if they are non-uniform. However, most questions still remain open — the best lower bound for Boolean formulas is $\Omega(n^3)$ [44] and for arithmetic circuits is $\Omega(n \log n)$ [9].

1.1 Proof complexity

One area within concrete complexity is proof complexity. The objects of study are proof systems and the relevant measures of complexity are size, width, space, etc. required for proving (or refuting) statements in these proof systems.

A proof system consists of a set of formulas (called axioms) assumed to be true and a set of rules (called inference rules) that can be used to derive new formulas from the axioms and the formulas already derived. Traditionally, mathematicians have asked whether every true statement has a proof, in a suitably general proof system. If this is true, can such a proof be discovered by a mechanical process (i.e. a Turing machine) in a finite amount of time? The answer to both of these questions is 'No' — the first is from the famous incompleteness theorem proven by Kurt Gödel, and the second is from the construction by Alan Turing of undecidable and non-enumerable sets.

But both these results rely on the fact that the variables in the formulas take values from an infinite domain, for example the set of real numbers or integers. However, we can restrict the domain to be finite, for example a bounded set of integers or the Boolean set $\{\mathrm{True}, \mathrm{False}\}$. With this restriction, the answer to both of the above questions becomes 'Yes' — we can check all possible assignments to the variables to decide whether the formula is True. In addition, the evaluation of the formula for all possible assignments forms (a very long) proof that the formula is true/false.

Historically, mathematicians have viewed the finite case as trivial. On the advent of computers, people wanted to solve such formulas using computers. It was soon realized that the above proof is impractical. For instance, for formulas in propositional logic, the variables take values in $\{\mathrm{True}, \mathrm{False}\}$. Even though there exists a proof (list out the evaluation of all combinations of variable assignments), such a proof is not very useful — it is of length 2^n, which is very large even for $n = 50$.

This led to the following question: does every unsatisfiable propositional formula have polynomial-size proofs of unsatisfiability? (Note that satisfiable propositional formulas have a linear-size proof of satisfiability i.e. the satisfying assignment.) This is the NP vs coNP question.

Like the P vs NP question, the NP vs coNP question has also proven difficult to answer. Like the P vs NP question, this has also led to a less ambitious program — for concrete proof systems, prove that there exists a true (resp. false) formula family which requires super-polynomial size proofs (resp. refutations) in that proof system.

Many refutational systems have been studied in the literature — for example, Resolution, Cutting Planes and the Frege system etc. Exponential lower bounds have been proven for Resolution [43] and Cutting Planes [64]. But the Frege system has resisted all such attempts.

1.1.1 Relation to solving

In the last two decades, many heuristic-based solvers have been built for testing whether a propositional formula is satisfiable (which is called the SAT problem). Even though this problem is NP-complete, these solvers perform extremely well on industrial SAT instances. Since many of these solvers can be modeled by the proof system resolution, lower bounds on the size of resolution refutations imply runtime lower bounds on the solvers.

In fact, one reason that proof complexity is interesting is that most standard proof systems capture some natural approach of solving SAT. Because of this, lower bounds for concrete proof systems are also very useful. This is in contrast to lower bounds for other restricted models like restricted circuits where they are just stepping stones, and may not be interesting results in themselves.

Since many proof systems capture natural ways of solving SAT, it is useful to compare the powers and limitations of different proof systems. This, in turn, tells us about the powers and limitations of different ways of solving SAT.

1.1.2 Beyond SAT

With SAT solvers performing so well, the community has set sights on solving harder problems. These include Quantified Boolean formulas (QBFs) and the Maximum Satisfiability problem (MaxSAT).

Quantified Boolean formulas

Quantified Boolean Formulas (QBFs) are a generalization of propositional formulas, in the sense that some of the variables are quantified universally. This allows a more natural and succinct encoding of many constraints. As a result, QBF solving has many more practical applications. However, it is PSPACE-complete [70] and hence believed to be much harder to solve than SAT.

Many QBF solvers are built by adapting the resolution-based SAT solvers to make them work for QBFs. As a result, many of the QBF solvers can be modeled by some modification of resolution. This has led to a variety of resolution-based proof systems for QBFs. QBF proof complexity mainly focuses on comparing the powers and limitations of these QBF proof systems.

Maximum Satisfiability

The Maximum Satisfiability problem asks for the maximum number of clauses of a CNF that can be satisfied simultaneously (i.e. given a CNF formula, the problem asks for a number k such that k clauses can be simultaneously satisfied but $k+1$ clauses cannot be satisfied). 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 is potentially harder since it is hard for both NP and coNP.

Many MaxSAT solvers work by making repeated queries to SAT solvers. In this thesis, we will study a different approach — a proof system for MaxSAT, called MaxSAT Resolution [28, 53].

1.1.3 Formal definitions

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 $\neg x$).

For set Z of variables, let $\langle Z \rangle$ denote the set of all total assignments to variables in Z. For a (multi-) set F of clauses, $\mathrm{viol}_F \colon \langle Z \rangle \to \{0\} \cup \mathbb{N}$ 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.

For a formula Φ and a partial assignment ρ to some of its variables, $\Phi\restriction_\rho$ denotes the restricted formula resulting from setting the specified variables according to ρ.

Quantified Boolean Formulas

A *Quantified Boolean Formula* (QBF) in *prenex conjunctive normal form* (p-cnf), denoted $\Phi = \mathcal{Q}.\phi$, consists of two parts: (i) a quantifier prefix $\mathcal{Q} = Q_1 Z_1, Q_2 Z_2, \ldots, Q_n Z_n$ where the Z_i are pairwise disjoint sets of variables, each $Q_i \in \{\exists, \forall\}$, and $Q_i \neq Q_{i+1}$; and (ii) a conjunction of clauses ϕ with variables in $Z = Z_1 \cup \cdots \cup Z_n$. In this thesis, when we say QBF, we mean a p-cnf QBF.

The set of existential (resp. universal) variables of Φ, denoted X (resp. U), is the union of Z_i for which $Q_i = \exists$ (resp. $Q_i = \forall$). The quantifier prefix defines a left/right ordering relation on the set of variables. This relation, denoted $z <_Q z'$, is defined as follows: $z <_Q z'$ holds if $z \in Z_i$, $z' \in Z_j$, and $i < j$. For $u \in U$, the set of existential variables left of u is $L_Q(u) \coloneqq \{x \in X \mid x <_Q u\}$.

A *strategy* h for a QBF Φ is a set $\{h^u \mid u \in U\}$ of functions $h^u \colon \langle L_Q(u) \rangle \to \{0, 1\}$ (for each $\alpha \in \langle X \rangle$, $h^u(\alpha\restriction_{L_Q(u)})$ and $h(\alpha)$ should be interpreted as a Boolean assignment to the variable u and the variable set U respectively). The strategy h is called a *winning strategy* (also called a countermodel) if, for each $\alpha \in \langle X \rangle$, the restriction of ϕ by the assignment $(\alpha, h(\alpha))$ is false. A QBF is false if it has a countermodel, and otherwise it is true [23, Sec. 31.2].

The semantics of QBFs is also explained by a *two-player evaluation game* played on a QBF. In a run of the game, two players, the existential and the universal player, assign values to the variables in the order of quantification in the prefix. The existential player wins if the assignment so constructed satisfies all the clauses of ϕ; otherwise the universal player wins. Assigning values according to a countermodel guarantees that the universal player wins no matter how the existential player plays; hence the term "winning strategy" [23, Sec. 31.2].

Proof systems

We will now define proof systems and proof complexity concepts more formally.

An alphabet Σ is a finite set of symbols. A language over alphabet Σ is a subset of Σ^* (here Σ^* is the set of strings of any length over alphabet Σ).

Definition 1.1.1 ([52, Def. 1.5.1])**.** A proof system P for a language L over alphabet Σ is a binary relation $P \subseteq L \times \Sigma^*$ satisfying the following:

1. P is computable in polynomial-time.
2. Soundness: For any $\alpha, \pi \in \Sigma^*$, if $P(\alpha, \pi)$ holds, then $\alpha \in L$.
3. Completeness: For any $\alpha \in L$, there is $\pi \in \Sigma^*$ such that $P(\alpha, \pi)$ holds.

If $P(\alpha, \pi)$ holds, we call π a P-proof of α.

As an example, for propositional formulas, L can be the set of satisfiable (or unsatisfiable) formulas. For quantified Boolean formulas (QBFs), L can be the set of true (or false) QBFs. If L is the set of unsatisfiable or false formulas, then π is called a refutation, and such a proof system is sometimes also called a refutational system.

For $\alpha \in L$, the size of the smallest P-proof, denoted $\mathrm{size}_P(\alpha)$, is defined as follows: $\mathrm{size}_P(\alpha) = \min\{|\pi| \mid P(\alpha, \pi) \text{ holds}\}$[52, Sec. 1.5]. A proof system P is called polynomially bounded (p-bounded) if every $\alpha \in L$ has a polynomial-size P-proof [52, Sec. 1.5].

Theorem 1.1.2 (The Cook–Reckhow theorem [33],[52, Thm. 1.5.2])**.** *A p-bounded proof system exists for unsatisfiable propositional formulas if and only if* $\mathsf{NP} = \mathsf{coNP}$.

So, if we can prove that no p-bounded proof system for unsatisfiable propositional formulas exists, then $\mathsf{NP} \neq \mathsf{coNP}$. This gives an approach for solving the NP vs coNP problem.

We will also be interested in comparing different proof systems. The next definition is motivated by this.

Definition 1.1.3 ([52, Def. 1.5.4].)**.** Let P and P' be proof systems for language L. We say that proof system P' simulates proof system P if there is a computable function f satisfying the following two properties: (i) for all $\alpha, \pi \in \Sigma^*$, if $P(\alpha, \pi)$ holds then $P'(\alpha, f(\pi))$ also holds; and (ii) for all $\pi \in \Sigma^*$, $|f(\pi)|$ is polynomial in $|\pi|$. If, furthermore, f is computable in polynomial-time, then we say that P' polynomially simulates (p-simulates) P

Let us now discuss a concrete proof system for unsatisfiable propositional formulas This proof system, called resolution [34, 35], is the most well-studied proof system.

We first define the resolution rule:

$$\frac{x \vee A \qquad \overline{x} \vee B}{A \vee B}$$

Here variable x is called the resolution pivot.

A resolution refutation of a false CNF formula F is a sequence of clauses $C_1, \ldots, C_t$ such that $C_t = \Box$ (i.e. the empty clause), and each C_i satisfies one of the following:

- C_i is in F
- there exist $j, k < i$, and there exist clauses A, B such that $C_j = x \vee A$, $C_k = \overline{x} \vee B$, and $C_i = A \vee B$.

Notice that a resolution refutation is a sequence of clauses. Such systems are called line-based systems. Many proof systems that we will encounter in this thesis will be line-based systems (however the lines may be more complex than clauses).

A proof in a line-based system can be viewed as a directed acyclic graph which has lines as nodes, and directed edge from line L_i to L_j if L_i is used in the step deriving L_j. *Tree-like resolution* (TreeRes) is the fragment of resolution which only allows those refutations in which the underlying graph is a tree [50, Sec. 18.1]. *Regular resolution* is the fragment of resolution with the following restriction: on every source-to-sink path, each variable can be used as pivot at most once [50, Sec. 18.2].

For a formula Φ and a partial assignment ρ to some of its variables, $\Phi\lceil_\rho$ denotes the restricted formula resulting from setting the specified variables according to ρ.

Definition 1.1.4. A propositional (resp. QBF) proof system P is *closed under restrictions* if for every unsatisfiable formula (resp. false QBF) Φ and every partial assignment ρ to some variables (resp. existential variables), the size of the smallest P-refutation of $\Phi\lceil_\rho$ is at most polynomial in the size of the smallest P-refutation of Φ.

Definition 1.1.5 ([10])**.** A proof system is *natural* if it is closed under restrictions.

1.2 Merge Resolution: A proof system for QBFs

Many of the currently known QBF proof systems are built on the resolution proof system [24, 67]. Broadly speaking, resolution has been adapted to handle the universal variables in QBFs in two intrinsically different ways. The first is an *expansion-based approach:* universal variables are eliminated at the outset by implicitly expanding the universal quantifiers into conjunctions, creating annotated copies of existential variables. The systems $\forall\mathrm{Exp} + \mathrm{Res}$, IR, and IRM [21, 49] are of this type. The second is a *reduction-rule approach:* under certain conditions, resolution may be blocked, and also under certain conditions, universal variables can be deleted from clauses. The conditions are formulated to preserve soundness, ensuring that if a QBF is true, then so is the QBF resulting from adding a derived clause. The systems Q-Res, QU-Res, $\mathrm{CP} + \forall\mathrm{Red}$ [22, 51, 74] are of this type.

A central role in QBF proof complexity is played by the *two-player evaluation game* on QBFs, and the existence of winning strategies for the universal player in false QBFs. For many QBF resolution systems, such strategies were used to construct proofs and demonstrate completeness, and soundness was demonstrated by extracting such strategies from proofs [7, 21, 36]. The *strategy extraction* procedures build partial strategies at each line of the proof, with the strategies at the final line forming a complete countermodel. These extraction procedures are based on the fact that in each application of a rule in the proof system, any winning strategies of the existential player are not destroyed.

In the systems Q-Res [51] and QU-Res [74], the soundness of the resolution rule is ensured by enforcing a very simple side-condition: variables other than the pivot cannot appear in both polarities in the antecedents. It was observed early on that this is often too restrictive. The *long-distance resolution proof system* LD-Q-Res [7, 77] arose from efforts to have less restrictive but still sound rules. In this system, a universal variable could appear in both polarities, provided it was to the right of the pivot in the quantifier prefix. Conventionally, $u \vee \overline{u}$ is abbreviated as u^* and called a merged literal. The following is an LD-Q-Res refutation of the QBF $\exists x \forall u.(x \vee u) \wedge (\overline{x} \vee \overline{u})$:

$$\dfrac{\dfrac{x \vee u \qquad \overline{x} \vee \overline{u}}{u^*}}{\square}$$

On the other hand, for the QBF $\forall u, \exists x.(u \vee x) \wedge (\overline{u} \vee \overline{x})$, the following is not a valid LD-Q-Res refutation:

$$\dfrac{\dfrac{u \vee x \qquad \overline{u} \vee \overline{x}}{u^*}}{\square}$$

The system LD-Q-Res, while provably better than Q-Res [36], is still needlessly restrictive in some situations. In particular, by checking a very simple syntactic prefix-ordering condition, it fails to exploit the fact that soundness is not lost even if universal variables to the left of the pivot are merged in both antecedents, provided the partial strategies built for them in both antecedents are identical. For example, for the QBF $\exists x \forall u, \exists t.(x \vee u \vee t) \wedge (\overline{x} \vee \overline{u} \vee t) \wedge (x \vee u \vee t) \wedge (\overline{x} \vee \overline{u} \vee t)$, the following refutation is not allowed in LD-Q-Res even though it would be sound in this case:

$$\dfrac{\dfrac{\dfrac{x \vee u \vee t \qquad \overline{x} \vee \overline{u} \vee t}{u^* \vee t} \qquad \dfrac{x \vee u \vee \overline{t} \qquad \overline{x} \vee \overline{u} \vee \overline{t}}{u^* \vee \overline{t}}}{u^*}}{\square}$$

A *new system Merge Resolution* (*M-Res*) was introduced three year ago precisely to address this point [18]. In M-Res, partial strategies are explicitly represented within the proof, in a particular representation format called merge maps – these are essentially deterministic branching programs (DBPs). In this format, isomorphism checking can be done efficiently, and this opens the way for enabling sound applications of resolution that would have been blocked in LD-Q-Res (and Q-Res). Returning to our previous example $\exists x \forall u, \exists t.(x \vee u \vee t) \wedge (\overline{x} \vee \overline{u} \vee t) \wedge (x \vee u \vee t) \wedge (\overline{x} \vee \overline{u} \vee t)$, following is an M-Res refutation.

$$\dfrac{\dfrac{x \vee t, \{u = 0\} \qquad \overline{x} \vee t, \{u = 1\}}{t, \{u = \text{if } x \textit{ is } 0 \text{ then } 0 \text{ else } 1\}} \qquad \dfrac{x \vee \overline{t}, \{u = 0\} \qquad \overline{x} \vee \overline{t}, \{u = 1\}}{\overline{t}, \{u = \text{if } x \textit{ is } 0 \text{ then } 0 \text{ else } 1\}}}{\square, \{u = \text{if } x \textit{ is } 0 \text{ then } 0 \text{ else } 1\}}$$

We explicitly represent a partial strategy for u in each line. In contrast to the disallowed LD-Q-Res refutation, here we can resolve the line "$t, \{u = \text{if } x \; is \; 0 \text{ then } 0 \text{ else } 1\}$" with the line "$\bar{t}, \{u = \text{if } x \; is \; 0 \text{ then } 0 \text{ else } 1\}$" because these partial strategies are isomorphic.

In [18], it was shown that M-Res brought a rich pay-off: there is a family of formulas, the SquaredEquality formulas, with short (linear-size) proofs in M-Res, even in its tree-like and regular versions, but requiring exponential size in Q-Res, QU-Res, CP + ∀Red, ∀Exp + Res, and IR. It is notable that the hardness of SquaredEquality in these systems stems from a certain semantic cost associated with these formulas and a corresponding lower bound [16, 17]. Thus the results of [18] show that such semantic costs are not a barrier for M-Res.

Our contributions

The authors of [18] did not show any advantage over LD-Q-Res — the system that M-Res was designed to improve. They only showed advantage over a restricted version of LD-Q-Res, the system reductionless LD-Q-Res. We show that M-Res is indeed quite powerful, answering one of the main questions left open in [18]. We show that there are formula families which have polynomial-size refutations in M-Res but require exponential-size refutationsin LD-Q-Res. In fact, we show that there are formula families having polynomial-size refutations in M-Res but requiring exponential-size refutations in the most powerful resolution-based QBF proof systems: reduction-based system LQU^+-Res and expansion-based system IRM.

We then show the limitations of M-Res. In particular, we show that KBKF-lq formula family requires exponential size refutations in M-Res. Combining this with results from [18], we conclude that M-Res is incomparable with QU-Res and CP + ∀Red. In addition, we show lower bounds for tree-like and regular M-Res

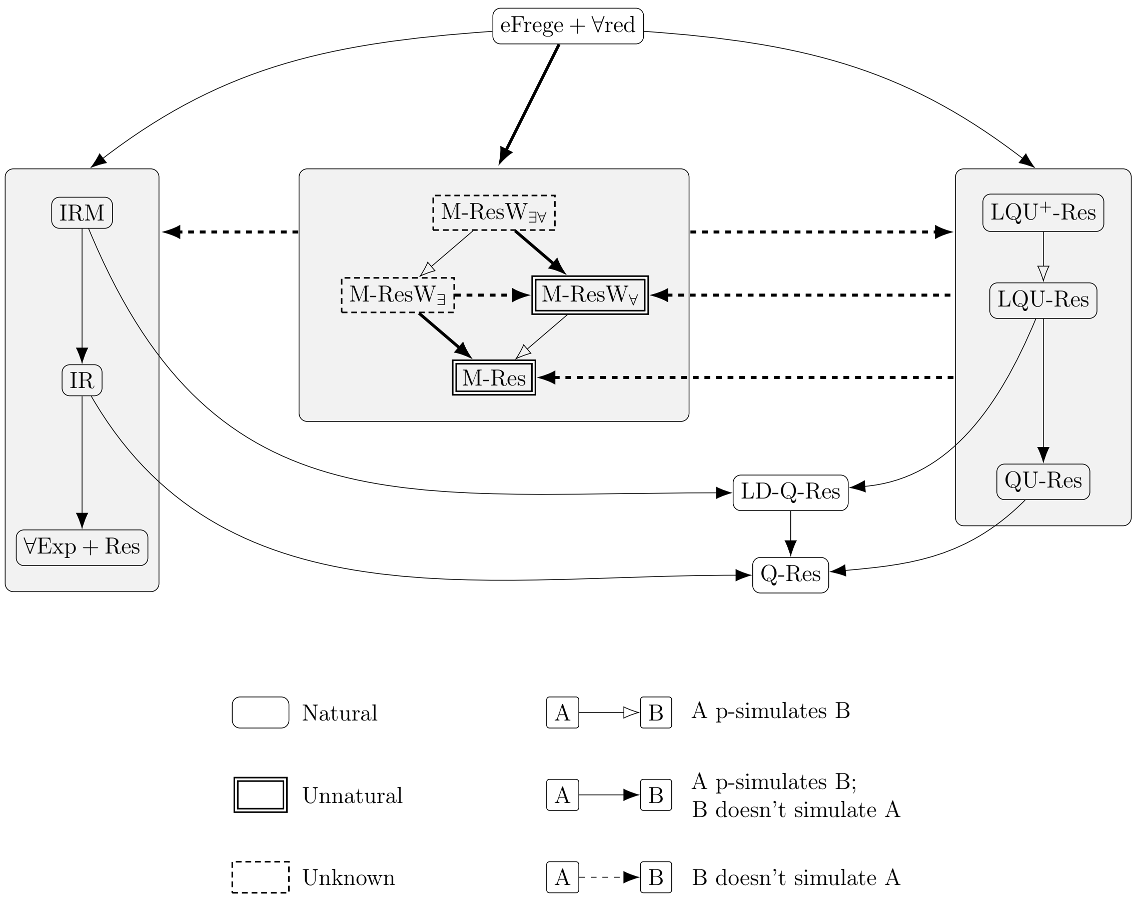


Figure 1.1: Relations among resolution-based QBF proof systems, with new results and observations highlighted using thicker lines. In addition, regular M-ResW$_\forall$ strictly p-simulates regular M-Res. (i) Lines from a big grey box mean that the line is from every proof system within the box. (ii) The missing relations follow from transitivity, otherwise the systems are incomparable.

which show that these systems are incomparable with Q-Res, QU-Res, CP + ∀Red, ∀Exp + Res and IR.

We then look at the role of the weakening rule when used with M-Res. Weakening is a rule that is sometimes augmented to resolution. This rule allows the derivation of $A \vee x$ from A, provided that A does not contain the literal $\overline{x}$. The weakening rule is mainly used to make resolution refutations more readable — it does not make them shorter [3]. The same holds for all other known resolution-based QBF proof systems. Here, we observe that weakening adds power to M-Res i.e. allowing weakening can make M-Res refutations exponentially shorter. We distinguish between two types of weakenings, namely existential clause weakening and strategy weakening. Both these weakenings were defined in the original paper [18] in which M-Res was introduced. However, these weakenings were used only for Dependency-QBFs (DQBFs); in that setting they are necessary for completeness. The potential use of weakening for QBFs was not explicitly addressed. Here, we show that existential clause weakening adds exponential power to M-Res. We do not know whether strategy weakening adds power to M-Res. However, we show that it does add exponential power to regular M-Res. At the same time, weakening of any or both types does not make M-Res unduly powerful; we show that eFrege + $\forall$red polynomially simulates (p-simulates) M-Res even with both types of weakenings added. This is proven by observing that the p-simulation of M-Res by eFrege + $\forall$red shown in [30] can very easily be extended to handle weakenings.

Another observation is that M-Res is not closed under restrictions. Closure under restrictions is a very important property of proof systems. For a (QBF) proof system, it means that restricting a false formula by a partial assignment to some of the (existential) variables does not make the formula much harder to refute. Note that a refutation of satisfiability of a formula implicitly encodes a refutation of satisfiability of all its restrictions, and it is reasonable to expect that such refutations can be extracted without paying too large a price. This is indeed the case for virtually all known proof systems to date. Many solvers work by setting some variables and simplifying the formula [59]. Without closure under restrictions, setting a bad variable may make the job of refuting the formula exponentially harder. Because of this reason, proofs systems which are closed under restrictions have been called *natural proof systems* [10]. We show that M-Res, with and without strategy weakening, is unnatural. We believe this would mean that it is hard to build QBF solvers based on it. On the other hand, we do not yet know whether it remains unnatural if existential clause weakening or both types of weakenings are added. We believe that this is the most important open question about M-Res — a negative answer can salvage it.

Our results are summarized in Figure 1.1.

1.3 MaxSAT Resolution

The MaxSAT Resolution proof system or more briefly MaxRes, was proposed as a proof system for the Maximum Satisfiability (MaxSAT problem) in [28, 53]. It operates on multi-sets of clauses, and uses the multi-output MaxSAT resolution (MaxRes) rule [28], defined as follows:

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

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. [28]

Since MaxRes maintains multi-sets of clauses and replaces used clauses, this suggests a “read-once”-like constraint [28]. However, this is not the case; read-once resolution is not even complete [47], 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 [61], 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 [4, 6, 14, 38]). Given this position in the ecosystem of simple proof systems, understanding its capabilities and limitations seems an interesting question.

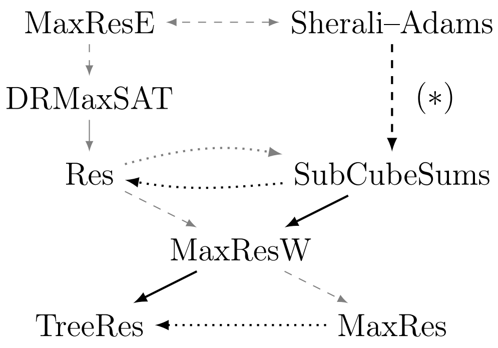
Our contributions

1. We observe that for certifying unsatisfiability, the proof system MaxResW p-simulates the tree-like fragment of Res, TreeRes (Lemma 6.2.1). 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 6.2.8). We exhibit a formula, which is a variant of the pebbling contradiction [13] on a pyramid graph, with short refutations in MaxRes (Lemma 6.2.2), and show that it requires exponential size in TreeRes (Lemma 6.2.7).

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 7.3.1). We also give a direct combinatorial proof that the pigeon-hole principle formulas have short SubCubeSums refutations (Theorem 7.3.5); this fact is implicit in a recent result from [54].

3. We show that the Tseitin contradiction on an odd-charged expander graph is hard for SubCubeSums (Theorem 7.4.2) and hence also hard for MaxResW. While this already follows from the fact that these formulas are hard for Sherali–Adams [4], 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



- $A \longrightarrow B$ denotes that A simulates B and B does not simulate A.
- $A \dashrightarrow B$ denotes that A simulates B.
  (∗) with caveats.
- $A \cdots\rightarrow B$ denotes that B does not simulate A.

Figure 1.2: Relation of MaxRes and MaxResW with other proof systems, with our results highlighted using black lines.

(Theorem 7.5.1). 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.

Recently, one of the open problems raised by us has been resolved in [37]; a lower bound for SubCubeSums size is shown for a formula that has short refutations in resolution. Also, in [39], 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.

The relations among these proof systems are summarized in Figure 1.2, which also includes two proof systems discussed in Related Work.

Related work

One reason why studying MaxRes is interesting is that it displays unexpected power after some preprocessing. As described in [46] (see also [58]), 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 [26], 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 [27, 54, 55, 68] study a generalization of the weighted version of MaxRes, under the 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 [54, 55, 68] and for certifying unsatisfiability in [27]. 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, [27] allows doing this. On the other hand, this is not allowed in [54, 55, 68] because this would make the system unsound for MaxSAT. With this added power the system in [27] is p-equivalent to another recently defined proof system called Circular Resolution [5]; hence by the results in [5], it is also p-equivalent to Sherali–Adams. Though most results in [54, 68] 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 [27]: the system is p-equivalent to Circular Resolution and Sherali–Adams. As can be seen from [27], 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 [55, 68] as MaxRes with a split rule, or ResS. It is shown in [54, 55, 68] 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 [42], 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 [41].

1.4 Organisation of the thesis

This thesis is divided into two parts:

Part 1 (Merge Resolution) We describe the Merge Resolution proof system in Chapter 2. In Chapter 3, we prove lower bounds for M-Res, thereby separating it from QU-Res and CP + $\forall$Red. In Chapter 4, we show the advantage of M-Res over other resolution-based QBF proof systems. Finally, in Chapter 5, we show that weakening adds power to M-Res. In the same chapter, we also show that M-Res is unnatural.

Part 2 (MaxSAT Resolution) In Chapter 6, we define the MaxRes proof system and compare it with Tree-like Resolution. In Chapter 7, we define the SubCubeSums proof system combinatorially, and formulate it as a restriction of the Sherali-Adams proof sytem. We then show its separation from Resolution, show that Tseitin contradictions are hard for it, and establish a lifting technique for proving lower bounds.

Part I

The Merge Resolution proof system

Chapter 2

Merge Resolution

2.1 Defining the proof system

The formal definition of the *Merge Resolution proof system*, denoted M-Res, is rather technical and can be found in [18]. Here we present a somewhat informal description.

First, we describe the *idea behind the proof system.* M-Res is a line-based proof system. Each line L has a clause C with only existential literals, and a partial strategy h^u for each universal variable u. The idea is to maintain the invariant that for each existential assignment α, if α falsifies C, then α extended by the partial universal assignment setting each u to $h^u(\alpha)$ falsifies at least one of the clauses used to derive L. Thus the set of functions $\{h^u\}$ gives a partial strategy that wins whenever the existential player plays from the set of assignments falsifying C. The goal is to derive a line with the empty clause; the corresponding strategy at that line will be a complete winning strategy, a countermodel. Along the way, resolution is used on the clauses. If the pivot is x, then for universal variables u right of x, the partial strategies can be combined with a branching decision on x. However, for u left of x, in the evaluation game, the value of u is already set when x is to be assigned. Thus already existing non-trivial partial strategies for u cannot be combined with a branching decision, and so this resolution step is blocked. However, if both the strategies are identical, or if one of them is trivial (unspecified), then the non-trivial strategy can be carried forward while maintaining the desired invariant. Checking whether strategies are identical can itself be hard, making verification of the proof difficult. In M-Res, this is handled by choosing a particular representation called merge maps, where isomorphism checks are easy.

Now we can describe the proof system itself. First we describe *merge maps*. Syntactically, these are deterministic branching programs, specified by a sequence of instructions of one of the following two forms:

- $\langle \text{line } \ell \rangle : b$ where $b \in \{*, 0, 1\}$.[1]

  Merge maps containing a single such instruction are called simple. In particular, if $b = *$, then they are called trivial.

- $\langle \text{line } \ell \rangle$: If $x = 0$ then go to $\langle \text{line } \ell_1 \rangle$ else go to $\langle \text{line } \ell_2 \rangle$, for some $\ell_1, \ell_2 < \ell$.

  In a merge map M for u, all queried variables x must precede u in the quantifier prefix.

  Merge maps with such instructions are called complex.

(All line numbers are natural numbers.) The merge map M^u computes a partial strategy for the universal variable u starting at the largest line number (the leading instruction) and following the instructions in the natural way. The value $*$ denotes an undefined value.

Two merge maps M_1, M_2 are said to be consistent, denoted $M_1 \bowtie M_2$, if for every line number i appearing in both M_1, M_2, the instructions with line number i are identical. Two merge maps M_1, M_2 are said to be isomorphic, denoted $M_1 \simeq M_2$, if there is a bijection between the line numbers in M_1 and M_2 that transforms M_1 to

[1] In [18], the notation used is $b \in \{*, u, \overline{u}\}$; $u, \overline{u}, *$ denote $u = 1, u = 0$, undefined respectively.

M_2 in the natural way.

For the remainder of this chapter let $\Phi = Q \cdot \phi$ be a QBF with existential variables X and universal variables U. The *proof system M-Res* has the following rules:

1. *Axiom:* For a clause A in the matrix ϕ, let C be the existential part of A. For each universal variable u, let b_u be the value u must take to falsify A; if $u \notin \text{var}(A)$, then $b_u = *$. For any natural number i, the line $(C, \{M^u : u \in U\})$ where each M^u is the simple merge map $\langle i \rangle : b_u$ can be derived in M-Res.

2. *Resolution:* From lines $L_a = (C_a, \{M^u_a : u \in U\})$ for $a \in \{0, 1\}$, in M-Res, the line $L = (C, \{M^u : u \in U\})$ can be derived, where for some $x \in X$,

   - $C = \text{Res}(C_0, C_1, x)$, and
   - for each $u \in U$,
     - either M^u_a is trivial and $M^u = M^u_{1-a}$ for some a, or
     - $M^u = M^u_0 \simeq M^u_1$, or
     - x precedes u and M^u has a leading instruction that builds the complex merge map $\textsf{If } x = 0 \textsf{ then } \langle M^u_0 \rangle \textsf{ else } \langle M^u_1 \rangle$.

A *refutation* is a derivation using these rules and ending in a line with the empty existential clause. The size of the refutation is the number of lines. We will denote refutations by the Greek letter Π.

2.2 An illustrative example

We reproduce from [18] a small example to illustrate how M-Res operates. The formulas to be refuted are the Equality formulas from [17], defined as follows: The *Equality family* is the QBF family whose nth instance has the prefix

$\exists x_1, \ldots, x_n, \forall u_1, \ldots, u_n, \exists t_1, \ldots, t_n$ and the following set of clauses

$\{x_i, u_i, t_i\}, \{\overline{x}_i, \overline{u}_i, t_i\}$ for $i \in [n]$, and $\{\bar{t}_1, \ldots, \bar{t}_n\}$.

In [18] (Example 3), linear-size reductionless LD-Q-Res refutations are described for these formulas, and later, M-Res is shown to simulate reductionless LD-Q-Res. Here, we directly present the implied linear-size M-Res refutations.

First, we download the axioms. Line 0 downloads the long clause $\{\bar{t}_1, \ldots, \bar{t}_n\}$, with all trivial merge maps. The next $2n$ lines download the short axiom clauses. Letting $i \in [n]$, we define these lines as follows:
Line $2i - 1$ is the clause $\{x_i, t_i\}$ with merge map 0 for u_i and all other merge maps are trivial.
Line $2i$ is the clause $\{\overline{x}_i, t_i\}$ with merge map 1 for u_i and all other merge maps are trivial.

For $i \in [n]$, line $2n + i$ is obtained by applying the merge resolution rule on lines $2i - 1$ and $2i$. This gives the clause $\{t_i\}$; the merge maps for $j \neq i$ are trivial, and the merge map for u_i has the instruction:
If $x_i = 0$ then go to $\langle$line $2i - 1\rangle$ else go to $\langle$line $2i\rangle$.

At line $3n + 1$, applying merge resolution on lines 0 and $2n + 1$, we obtain the clause $\{\bar{t}_2, \ldots, \bar{t}_n\}$. The merge map for u_1 is taken from line $2n + 1$, since at line 0 it is trivial.

Now for $i \in [2, n]$, line $3n + i$ is obtained by applying merge resolution on lines $2n + i$ and $3n + i - 1$. This gives the clause $\{\bar{t}_{i+1}, \ldots, \bar{t}_n\}$. The merge map for u_i is taken from line $2n + i$ since at line $3n + i - 1$ it is trivial. For $j < i$, the merge map for u_j is taken from line $3n + i - 1$ since at line $2n + i$ it is trivial. Effectively, at this line, for all $j \leq i$, the merge map for u_j is from line $2n + j$, and for all $j > i$, the merge map for u_j is trivial.

Line $4n$ derives the empty clause and the strategy computing, for each $i \in [n]$,

$u_i = x_i$. This completes the refutation and the example.

2.3 Properties

As shown in [18], the merge maps at the final line of a refutation compute a countermodel for the QBF. To establish this, some stronger properties of the derivation are established and will be useful to us. We restate the relevant properties here.

Lemma 2.3.1 (Extracted/adapted from [18] Section 4.3, (Proof of Lemma 21))**.** *Let $\Phi = Q \cdot \phi$ be a QBF with existential variables X and universal variables U. Let $\Pi \stackrel{def}{=} L_1, \dots, L_m$ be an M-Res refutation of Φ, where each $L_i = (C_i, \{M_i^u \mid u \in U\})$. Further, for each $i \in [m]$,*

- *let α_i be the minimal partial assignment falsifying C_i,*
- *let A_i be the set of assignments to X consistent with α_i,*
- *for each $u \in U$, let h_i^u be the function computed by M_i^u,*
- *for each $\alpha \in A_i$, let $h_i(\alpha)$ be the partial assignment which sets variable u to $h_i^u(\alpha\restriction_{LQ(u)})$ if $h_i^u(\alpha\restriction_{LQ(u)}) \neq *$, and leaves it unset otherwise.*

Then for each $\alpha \in A_i$, the (partial) assignment $(\alpha, h_i(\alpha))$ falsifies at least one clause of ϕ used in the sub-derivation of L_i.

Let G_Π be the derivation graph corresponding to Π (with edges directed from the antecedents to the consequent, hence from the axioms to the final line).

Proposition 2.3.2 ([18])**.** *Let $\Phi = Q \cdot \phi$ be a QBF with existential variables X and universal variables U. Let $\Pi \stackrel{def}{=} L_1, \dots, L_m$ be an M-Res refutation of Φ, where each $L_i = (C_i, \{M_i^u \mid u \in U\})$. Then, for all $u \in U$, M_m^u is isomorphic to a subgraph of G_Π (up to path contraction).*

Let S be a subset of the existential variables X of Φ. We say that an M-Res refutation of Φ is *S-regular* if for each $x \in S$, there is no leaf-to-root path that uses x as pivot more than once. An X-regular proof is simply called a *regular proof.* If G_Π is a tree, then we say that Π is a *tree-like proof.* Note that the refutation in Section 2.2 is both tree-like and regular.

Chapter 3

Lower bounds

In this chapter, we show lower bounds for M-Res, thereby uncovering its limitations. The lower bounds are either transferred from bounds from circuit complexity (for restricted versions of M-Res) or directly obtained by combinatorial arguments (for full M-Res). Our results imply that the M-Res approach is *largely orthogonal to other QBF resolution models* such as the QCDCL resolution systems QRes and QURes and the expansion systems $\forall$Exp + Res and IR.

(A) Lower bounds from circuit complexity for restricted versions of M-Res. Since the strategies are explicitly represented inside the proofs, computational hardness of strategies immediately translates to proof size lower bounds. While computational hardness of strategies is a known source of hardness in all reduction-based proof systems admitting efficient strategy extraction [19, 21], the computational model relevant for M-Res is one for which no unconditional lower bounds are known. For tree-like and regular M-Res, the relevant models are decision trees and read-once DBPs, where lower bounds are known. Using this approach, we show:

1. Tree-like M-Res is exponentially weaker than M-Res.

The QParity formulas witness the separation (Theorem 3.2.3) as their unique countermodel is the parity function which requires large decision trees.

2. Tree-like M-Res is incomparable with the dag-like and tree-like versions of Q-Res, QU-Res, $\mathsf{CP} + \forall\mathsf{Red}$, $\forall\mathsf{Exp} + \mathsf{Res}$ and IR.
   One direction was shown in [18] via the Equality formulas: these formulas are easy for tree-like M-Res but hard for dag-like Q-Res, QU-Res, $\mathsf{CP} + \forall\mathsf{Red}$, $\forall\mathsf{Exp} + \mathsf{Res}$, IR. The other direction is witnessed by the Completion Principle formulas, easy in tree-like versions of Q-Res and $\forall\mathsf{Exp} + \mathsf{Res}$ [48, 49], but exponentially hard for tree-like M-Res (Theorem 3.2.6). Unlike the QParity formulas, these formulas do not have unique countermodels. However, we show that every countermodel requires large decision-tree size, and hence obtain the lower bound for tree-like M-Res.

(B) Combinatorial lower bounds for full M-Res. Even when winning strategies are unique and easy to compute by DBPs, the formulas can be hard for M-Res. We establish such hardness in three cases, obtaining more incomparabilities.

1. The LQParity formulas, easy in $\forall\mathsf{Exp} + \mathsf{Res}$ [21], are exponentially hard for regular M-Res (Theorem 3.3.1). Hence regular M-Res is incomparable with $\forall\mathsf{Exp} + \mathsf{Res}$ and IR.

2. The Completion Principle formulas, easy in tree-like versions of Q-Res and $\forall\mathsf{Exp} + \mathsf{Res}$ [48, 49], are exponentially hard for regular M-Res (Theorem 3.3.6). Hence regular M-Res is incomparable with the dag-like and tree-like versions of Q-Res, QU-Res, $\mathsf{CP} + \forall\mathsf{Red}$, $\forall\mathsf{Exp} + \mathsf{Res}$ and IR.

3. The KBKF-lq formulas, easy in QU-Res [8], are exponentially hard for M-Res (Theorem 3.4.1). Hence M-Res is incomparable with QU-Res and $\mathsf{CP} + \forall\mathsf{Red}$.

The third hardness result above for the KBKF-lq formulas provides the first lower bound for the full system of M-Res, for which previously no lower bounds were known.

3.1 The formulas

We describe the formulas we will use throughout this chapter.

The QParity and LQParity formulas [21]. Let $\text{parity}^c(y_1, y_2, \ldots, y_k)$ be a shorthand for the following conjunction of clauses:
$\bigwedge_{S\subseteq[k],\,|S|\equiv 1\,(\text{mod }2)} ((\vee_{i\in S}\overline{y_i}) \vee (\vee_{i\notin S} y_i))$. Thus $\text{parity}^c(y_1, y_2, \ldots, y_k)$ is equal to 1 iff $y_1 + y_2 + \cdots + y_k \equiv 0 \pmod 2$. QParity_n is the QBF
$\exists x_1, \ldots, x_n, \forall z, \exists t_1, \ldots, t_n.\ \left(\bigwedge_{i\in[n+1]} \phi_n^i\right)$ where

$$\begin{aligned}
\phi_n^1 &= \text{parity}^c(x_1, t_1) \\
\phi_n^i &= \text{parity}^c(t_{i-1}, x_i, t_i), \qquad \forall i \in [2, n] \\
\phi_n^{n+1} &= (t_n \vee z) \wedge \left(\overline{t_n} \vee \overline{z}\right)
\end{aligned}$$

Intuitively, $\phi_n^1 \wedge \cdots \wedge \phi_n^i$, for $i \in [n]$, enforces that the constraint $x_1 + \cdots + x_i \equiv t_i \pmod 2$. Similarly, $\phi_n^1 \wedge \cdots \wedge \phi_n^{n+1}$ enforces the constraint $x_1 + \cdots + x_i \not\equiv z \pmod 2$. Since the value of z is set by the universal player after the existential player sets the values of $x_1, x_2, \ldots, x_n$, the universal player has a winning strategy. This means that the formula is false. Note that the only winning strategy for the universal player is to play z satisfying $z \equiv x_1 + \cdots + x_n \pmod 2$.

Similarly, let $\widehat{\text{parity}}^c(y_1, y_2, \ldots, y_k; z)$ abbreviate
$\bigwedge_{C\in\text{parity}^c(y_1,y_2,\ldots,y_k)} \left((C \vee z) \wedge (C \vee \overline{z})\right)$. LQParity_n is the QBF

$\exists x_1, \ldots, x_n, \forall z, \exists t_1, \ldots, t_n.\ \left(\bigwedge_{i \in [n+1]} \phi_n^i\right)$ where

$$\begin{aligned}
\phi_n^1 &= \widehat{\mathrm{parity}}^c(x_1, t_1; z) \\
\phi_n^i &= \widehat{\mathrm{parity}}^c(t_{i-1}, x_i, t_i; z), \qquad \forall i \in [2, n] \\
\phi_n^{n+1} &= (t_n \vee z) \wedge \left(\overline{t_n} \vee \overline{z}\right).
\end{aligned}$$

For both $\mathrm{QParity}_n$ and $\mathrm{LQParity}_n$, for $i, j \in [n+1], i \leq j$, we let $\phi_n^{[i,j]}$ denote $\bigwedge_{k \in [i,j]} \phi_n^k$. Also, $X = \{x_1, \ldots, x_n\}$ and $T = \{t_1, \ldots, t_n\}$.

Observation 3.1.1. *For both $QParity_n$ and $LQParity_n$:* (a) *for each $i \in [n]$, and each $C \in \phi_n^i$, $\{x_i, t_i\} \subseteq var(C)$; and* (b) *for each $i \in [n+1] \setminus \{1\}$, and each $C \in \phi_n^i$, $\{t_{i-1}\} \subseteq var(C)$.*

The Completion Principle formulas CR_n [49]. The QBF CR_n is defined as follows:

$$\mathrm{CR}_n = \underset{i,j \in [n]}{\exists} x_{ij}, \forall z, \underset{i \in [n]}{\exists} a_i, \underset{j \in [n]}{\exists} b_j.\ \left(\bigwedge_{i,j \in [n]} (A_{ij} \wedge B_{ij})\right) \wedge L_A \wedge L_B$$

where $A_{ij} = x_{ij} \vee z \vee a_i$, $B_{ij} = \overline{x_{ij}} \vee \overline{z} \vee b_j$, $L_A = \overline{a_1} \vee \cdots \vee \overline{a_n}$, and $L_B = \overline{b_1} \vee \cdots \vee \overline{b_n}$. Let X, A, B denote the variable sets $\{x_{ij} : i, j \in [n]\}$, $\{a_i : i \in [n]\}$, and $\{b_j : j \in [n]\}$. It is convenient to think of the X variables as arranged in an $n \times n$ matrix.

Intuitively, the formulas describe a completion game, played on a $2 \times n^2$ dimensional matrix whose $(i-1)n + j$-th column (for $1 \leq i, j \leq n$) is $\binom{a_i}{b_j}$. Explicitly, the matrix is the following:

$$\begin{pmatrix}
a_1 & \ldots & a_1 & a_2 & \ldots & a_2 & \ldots\ldots & a_n & \ldots & a_n \\
b_1 & \ldots & b_n & b_1 & \ldots & b_n & \ldots\ldots & b_1 & \ldots & b_n
\end{pmatrix}$$

The $\exists$-player first deletes exactly one cell per column and the $\forall$-player then chooses one row. The $\forall$-player wins if his row contains all of A or all of B (cf. [49]).

The KBKF-lq$[n]$ formulas [8]. Our last QBFs are a variant of the formulas introduced by Kleine Büning et al. [51], which in various versions appear prominently throughout the QBF literature [8, 17, 21, 36, 74]. For $n > 1$, the nth member of the KBKF-lq$[n]$ family consists of the prefix
$\exists d_1, e_1, \forall x_1, \exists d_2, e_2, \forall x_2, \ldots, \exists d_n, e_n, \forall x_n, \exists f_1, f_2, \ldots, f_n$ and clauses

$$\begin{array}{lll}
A_0 = \{\overline{d_1}, \overline{e_1}, \overline{f_1}, \ldots, \overline{f_n}\} & & \\
A_i^d = \{d_i, x_i, \overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\} & A_i^e = \{e_i, \overline{x_i}, \overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\} & \forall i \in [n-1] \\
A_n^d = \{d_n, x_n, \overline{f_1}, \ldots, \overline{f_n}\} & A_n^e = \{e_n, \overline{x_n}, \overline{f_1}, \ldots, \overline{f_n}\} & \\
B_i^0 = \{x_i, f_i, \overline{f_{i+1}}, \ldots \overline{f_n}\} & B_i^1 = \{\overline{x_i}, f_i, \overline{f_{i+1}}, \ldots \overline{f_n}\} & \forall i \in [n-1] \\
B_n^0 = \{x_n, f_n\} & B_n^1 = \{\overline{x_n}, f_n\} &
\end{array}$$

Note that the existential part of each clause in KBKF-lq$[n]$ is a Horn clause (at most one positive literal), and except A_0, is even strict Horn (exactly one positive literal).

We use the following shorthand notation. Sets of variables: $D = \{d_1, \ldots, d_n\}$, $E = \{e_1, \ldots, e_n\}$, $F = \{f_1, \ldots, f_n\}$, and $X = \{x_1, \ldots, x_n\}$. Sets of literals: For $Y \in \{D, E, X, F\}$, set $Y^1 = \{u \mid u \in Y\}$ and $Y^0 = \{\overline{u} \mid u \in Y\}$. Sets of clauses:

$$\begin{array}{lllllll}
\mathcal{A}_0 & = & \{A_0\} & & & & \\
\mathcal{A}_i & = & \{A_i^d, A_i^e\} \quad \forall i \in [n] & & \mathcal{B}_i & = & \{B_i^0, B_i^1\} \quad \forall i \in [n] \\
\mathcal{A}_{[i,j]} & = & \cup_{k \in [i,j]} \mathcal{A}_k \quad \forall i, j \in [0, n], i \leq j & & \mathcal{B}_{[i,j]} & = & \cup_{k \in [i,j]} \mathcal{B}_k \quad \forall i, j \in [n], i \leq j \\
\mathcal{A} & = & \mathcal{A}_{[0,n]} & & \mathcal{B} & = & \mathcal{B}_{[1,n]}
\end{array}$$

We use the following property of these formulas:

Proposition 3.1.2. *Let h be any countermodel for KBKF-lq$[n]$. Let α be any assignment to D, and β be any assignment to E.*

For each $i \in [n]$, if $\alpha_j \neq \beta_j$ for all $1 \leq j \leq i$, then $h^{x_i}\big((\alpha, \beta)\restriction_{LQ(x_i)}\big) = \alpha_i$.

In particular, if $\alpha_j \neq \beta_j$ for all $j \in [n]$, then the countermodel computes $h(\alpha, \beta) = \alpha$.

Proof. Let h be any countermodel for KBKF-lq$[n]$. For $i \in [n]$, let α^i be an assignment to $\{d_1, \dots, d_i\}$, and β^i be an assignment to $\{e_1, \dots, e_i\}$. For $j \leq i$, let α^i_j (resp. β^i_j) be the assignment to d_j (resp. e_j) set by the assignment α^i_j (resp. β^i_j). We will show that for each $i \in [n]$, if $\alpha^i_j \neq \beta^i_j$ for all $1 \leq j \leq i$, then $h^{x_i}(\alpha^i, \beta^i) = \alpha^i_i$. This implies the claimed result.

Fix some $i \in [n]$. Assume to the contrary that $\alpha^i_j \neq \beta^i_j$ for all $1 \leq j \leq i$ and $h^{x_i}(\alpha^i, \beta^i) \neq \alpha^i_i$. We will give a winning strategy for the existential player. Note that all clauses in $\mathcal{A}[0, i-1]$ are satisfied by the partial assignment (α^i, β^i). The existential player sets $d_j = e_j = 1$ for all $j > i$ and sets $f_j = 1$ for all $j \in [n]$. This satisfies all the remaining clauses, irrespective of the strategy of the universal player. Therefore the existential player wins. This contradicts the assumption that h is a countermodel for KBKF-lq$[n]$. $\square$

3.2 Transferring branching program lower bounds

The following lemma allows us to transfer lower bounds for decision trees (resp. read-once branching programs) to lower bounds for tree-like (resp. regular) Merge Resolution.

Lemma 3.2.1. *Let $\Phi = Q \cdot \phi$ be a QBF with existential variables X and universal variables U. Let $\Pi \stackrel{def}{=} L_1, \dots, L_m$ be an M-Res refutation of Φ, where each $L_i = (C_i, \{M^u_i \mid u \in U\})$. If Π is tree-like (resp. regular), then for all $u \in U$, M^u_m is a decision tree (resp. read-once branching program) with $\{M^u_m \mid u \in U\}$ computing a countermodel of Φ. Moreover, the size of Π is lower bounded by the size of M^u_m.*

Proof. It is an immediate consequence of Proposition 2.3.2. □

Tree-like Merge Resolution

For QParity$_n$ and LQParity$_n$, the only winning strategy for the universal player is to set z such that $z \equiv x_1 + x_2 + \cdots + x_n \pmod 2$.

Proposition 3.2.2 (Folklore)**.** *The decision-tree size complexity of the parity function is* 2^n*.*

From Lemma 2.3.1, Lemma 3.2.1, and Proposition 3.2.2, we obtain the desired lower bound.

Theorem 3.2.3. $size_{\text{M-ResTree}}(QParity_n) = 2^{\Omega(n)}$ *and* $size_{\text{M-ResTree}}(LQParity_n) = 2^{\Omega(n)}$*.*

Corollary 3.2.4. *Tree-like M-Res does not simulate regular M-Res and general M-Res.*

Proof. Theorem 3.2.3 shows that QParity requires exponential-size refutations in tree-like M-Res. It has polynomial-size refutations in reductionless LD-Q-Res [63] (in fact the refutation in [63] is regular). Since (regular) M-Res p-simulates (regular) reductionless LD-Q-Res, these formulas have polynomial-size refutations in regular M-Res also. The result follows. □

For the QBF CR$_n$, the winning strategy for the universal player (countermodel) is not unique. However, we show that all countermodels require large decision trees.

Lemma 3.2.5. *Every countermodel for* CR_n *has decision tree size complexity at least* 2^n*.*

Proof. We prove the size bound by showing that in every decision tree for every countermodel, all root-to-leaf paths query at least n variables, and hence the decision tree has at least 2^n nodes.

Assume to the contrary that some countermodel h is computed by a decision tree M that has a root-to-leaf path p querying less than n variables. Then there exist $k, \ell \in [n]$ such that no variable from Row k and no variable from Column ℓ is on this path. Let ρ_p be the minimal partial assignment that takes this path in M, and let ρ' be an arbitrary extension of ρ_p to variables in $\{x_{ij} \mid i \neq k, j \neq \ell\}$. Consider the following extension of ρ' to variables in $(X \setminus \{x_{k\ell}\}) \cup T$, giving assignment σ:
Set all variables in row k (other than $x_{k,\ell}$) to 1.
Set all variables in column ℓ (other than $x_{k,\ell}$) to 0.
Set a_k and b_ℓ to 0 and all other a_i, b_j variables to 1.

For $n \geq 2$, σ satisfies all the clauses of CR_n except $A_{k\ell}$ and $B_{k\ell}$, which get restricted to $x_{k\ell} \vee z$ and $\overline{x_{k\ell}} \vee \overline{z}$ respectively.

Let $\alpha_0 = \sigma \cup \{x_{k\ell} = 0\}$ and $\alpha_1 = \sigma \cup \{x_{k\ell} = 1\}$. Since both α_0 and α_1 extend ρ_p, they follow path p, therefore $h(\alpha_0) = h(\alpha_1)$. If $h(\alpha_0) = h(\alpha_1) = 0$, then $(\alpha_1, h(\alpha_1))$ satisfies all clauses of CR_n. On the other hand, if $h(\alpha_0) = h(\alpha_1) = 1$, then $(\alpha_0, h(\alpha_0))$ satisfies all clauses of CR_n. Thus in either case, h is not a countermodel for CR_n. $\square$

From Lemma 2.3.1, Lemma 3.2.1, and Lemma 3.2.5, we obtain the desired lower bound.

Theorem 3.2.6. $size_{\textit{M-ResTree}}(CR_n) = 2^{\Omega(n)}$.

Corollary 3.2.7. *Tree-Like M-Res is incomparable with the tree-like and general versions of Q-Res, QU-Res, CP + ∀Red, ∀Exp + Res, and IR.*

Proof. We showed in Theorem 3.2.6 that the Completion Principle CR_n requires exponential-size refutations in tree-like Merge Resolution. It has polynomial-size refutations in tree-like QRes [48] (and hence also in QU-Res and $\mathrm{CP} + \forall\mathrm{Red}$) and tree-like $\forall\mathrm{Exp} + \mathrm{Res}$ [49] (and hence also in IR). (While [49] does not explicitly mention tree-like proofs, the proof provided there for CR_n is tree-like.) On the other hand, the Equality formulas have polynomial-size tree-like M-Res refutations [18] but require exponential-size refutations in Q-Res, QU-Res, $\mathrm{CP} + \forall\mathrm{Red}$ [17], $\forall\mathrm{Exp} + \mathrm{Res}$, IR [16] (cf. [15] on how to apply the lower bound technique from [16] to the Equality formulas). □

Regular Merge Resolution

We now show how to lift lower bounds for any read-once branching program to those for regular M-Res. This follows the method used, for instance, in [21] (Section 4.1) and [63] (Section 6). Given a Boolean function f which requires exponential size read-once branching programs, we will construct a QBF formula such that the winning strategy for at least one of the universal variables is f. This will give the desired lower bound. We now describe how to construct such a QBF. Let $f\colon X \to \{0,1\}$ be a Boolean function, and let C_f be some Boolean circuit computing f. Let u be a variable such that $u \notin X$. We can use Tseitin transformation (see [71]) to construct a CNF formula $\phi(X, u, Y)$ such that $\exists Y.\phi(X, u, Y)$ is logically equivalent to $C_f(X) \neq u$. Using this, we construct the false QBF formula: $\Phi \coloneqq \exists X \forall u \exists Y.\phi(X, u, Y)$, which has the property that f is the unique winning strategy. Moreover, the size of Φ is polynomial in the size of C_f. Choosing a function f that can be computed by polynomial-size Boolean circuits but requires exponential-size read-once branching programs gives the desired lower bound. There are many such functions, for example see [25]. This gives us the desired lower bound.

3.3 Lower bounds for Regular Merge Resolution

In this section, we prove Regular M-Res lower bounds for formulas whose countermodels can be computed by polynomial-size read-once branching programs. That is, these lower bounds are *not* because of computational hardness of counter-models.

3.3.1 LQParity formulas

Our first result concerns the long-distance versions of the parity formulas [21] (cf. Section 3.1), which are known to be hard for LD-Q-Res. We establish that they are hard for regular Merge Resolution as well.

Theorem 3.3.1. $size_{\textit{M-ResReg}}(LQParity_n) = 2^{\Omega(n)}$.

This follows from a stronger result that we prove below: any T-regular refutation of $\mathrm{LQParity}_n$ in M-Res must have size $2^{\Omega(n)}$ (Theorem 3.3.5).

The proof proceeds as follows: Let Π be a T-regular M-Res refutation of $\mathrm{LQParity}_n$. Since every axiom has a variable from T while the final clause in Π is empty, there is a maximal "component" of the proof leading to and including the final line, where all clauses are T-free. The clauses in this component involve only the X variables. We show that the "boundary" of this component is large, by showing in Lemma 3.3.4 that each clause here must be wide. (This idea was used in [63] to show that CR is hard for reductionless LD-Q-Res.) To establish the width bound, we note that no lines have trivial strategies. Since the pivots at the boundary are variables from T, the merge maps incoming into each boundary resolution must be isomorphic. By carefully analysing which axiom clauses can and must be used to derive lines just above the boundary (Lemma 3.3.3), we conclude that the merge maps must be simple, yielding the lower bound. To fill in all the details, we first describe some properties (Lemma 3.3.2) of Π that will be used in obtaining this result.

The lines of Π will be denoted by L, L', L'' etc. For lines L and L' the respective clause, merge map and the function computed by the merge map will be denoted by C, M, h and C', M', h' respectively. Let G_Π be the derivation graph corresponding to Π (with edges directed from the antecedents to the consequent, hence from the axioms to the final line). We will refer to the nodes of this graph by the corresponding line. For $L, L' \in \Pi$, we will say $L \rightsquigarrow L'$ if there is a path from L to L' in G_Π.

For a line $L \in \Pi$, let Π_L be the minimal sub-derivation of L, and let G_{Π_L} be the corresponding subgraph of G_Π with sink L. Define UsedConstraints$(\Pi_L) = \{\phi_n^i \mid i \in [n+1], \text{leaves}(G_{\Pi_L}) \cap \phi_n^i \neq \emptyset\}$, and $\textsc{Uci}(\Pi_L) = \{i \in [n+1] \mid \phi_n^i \in \text{UsedConstraints}(\Pi_L)\}$. (Uci stands for UsedConstraintsIndex.) Note that for any leaf L, $\textsc{Uci}(\Pi_L)$ is a singleton.

Define $\mathcal{S}'$ to be the set of those lines in Π where the clause part has no T variable and furthermore there is a path in G_Π from the line to the final empty clause via lines where all the clauses also have no T variables. Let $\mathcal{S}$ denote the set of leaves in the subgraph of G_Π restricted to $\mathcal{S}'$; these are lines that are in $\mathcal{S}'$ but their parents are not in $\mathcal{S}'$. Note that no leaf of Π is in $\mathcal{S}'$ because all leaves of G_Π contain a variable in T.

Lemma 3.3.2. *Let $L = (C, M)$ be a line of Π. Then $\textsc{Uci}(\Pi_L)$ is an interval $[i, j]$ for some $1 \leq i \leq j \leq n+1$. Furthermore, (below i, j refer to the endpoints of this interval)*

1. *For all $k \in [i, j-1]$, $t_k \notin var(C)$.*
2. *If $i > 1$, then $t_{i-1} \in var(C)$.*
3. *If $j \leq n$, then $t_j \in var(C)$.*

4. *$|var(C) \cap T| = 1$ iff $[i,j]$ contains exactly one of $1, n+1$.*

   $var(C) \cap T = \emptyset$ iff $[i,j] = [1, n+1]$.

5. *For all $k \in [i,j] \cap [1,n]$, $x_k \in var(C) \cup var(M)$.*

Proof. Let $I = \text{Uci}(\Pi_L)$. Assume, to the contrary, that I is not an interval; for some $k \in [2,n]$, I contains an index $i < k$ and an index $j > k$, but does not contain k. Let L' be the first line in Π such that $\text{Uci}(\Pi_{L'})$ intersects both $[1, k-1]$ and $[k+1, n+1]$. Since leaves have singleton Uci sets, L' is not a leaf. Say $L' = \text{Res}(L'', L''', v)$. Assume that $\text{Uci}(\Pi_{L''}) \subseteq [1, k-1]$ and $\text{Uci}(\Pi_{L'''}) \subseteq [k+1, n+1]$; the argument for the other case is identical. So $v \in \text{var}_\exists(\text{UsedConstraints}(\Pi_{L''})) \subseteq \text{var}_\exists(\phi_n^{[1,k-1]})$, and $v \in \text{var}_\exists(\text{UsedConstraints}(\Pi_{L'''})) \subseteq \text{var}_\exists(\phi_n^{[k+1,n+1]})$. But $\text{var}_\exists(\phi_n^{[1,k-1]})$ and $\text{var}_\exists(\phi_n^{[k+1,n+1]})$ are disjoint, a contradiction.

Fixing i, j so that $I = \text{Uci}(\Pi_L) = [i,j]$, we now prove the remaining statements in the Lemma.

1. Fix any $k \in [i, j-1]$. Note that $\{k, k+1\} \subseteq \text{Uci}(\Pi_L)$. Let L' be the first line in Π_L such that $\{k, k+1\} \subseteq \text{Uci}(\Pi_{L'})$. Say L' is obtained as $\text{Res}(L'', L''', v)$. Assume that $\text{Uci}(\Pi_{L''})$ contributes k and $\text{Uci}(\Pi_{L''})$ contributes $k+1$; the other case is symmetric. Since $\text{Uci}(\Pi_{L''})$ must also be an interval, and since it contains k but not $k+1$, $\text{Uci}(\Pi_{L''}) \subseteq [1,k] \cap \text{Uci}(\Pi_L) = [i,k]$. Similarly, $\text{Uci}(\Pi_{L'''}) \subseteq [k+1, j]$. The pivot variable v must thus belong to both $\phi_n^{[i,k]}$ and $\phi_n^{[k+1,j]}$; the only such existential variable is t_k. Hence each t_k is used as a pivot in Π_L.

   Since Π is T-regular, and since t_k is used as a pivot to derive L' inside Π_L, it cannot reappear in any line on any path from (including) L' to the final clause. Hence it does not appear in L.

2. Let $i > 1$. By Observation 3.1.1, t_{i-1} appears in at least one axiom used in Π_L.

Assume to the contrary that $t_{i-1} \notin \text{var}(C)$. Let ρ_C be the minimal partial assignment falsifying C. By assumption, ρ_C does not set t_{i-1}, and by Item 1 above, ρ_C does not set any variable t_k with $i \leq k < j$. Extend ρ_C arbitrarily to all unassigned variables in $(X \cup T) \setminus \{t_{i-1}, \ldots, t_{j-1}\}$ to get ρ_1. Since the merge map M does not depend on variables in T, the partial assignment ρ_1 is sufficient to evaluate M and h. Define the value y as follows:

$$y = \begin{cases} \rho_1(t_j) & \text{if } j \leq n \\ h(\rho_1) & \text{if } j = n+1 \end{cases}$$

For $b \in \{0, 1\}$, let ρ_1^b denote the extension of ρ_1 by $t_{i-1} = b$. Exactly one of ρ_1^0, ρ_1^1 satisfies the equation $t_{i-1} + x_i + x_{i+1} + \ldots + x_j + y \equiv 0 \bmod 2$; let this extension be ρ_2. Then there is a unique extension α of ρ_2 to $X \cup T$ such that

- if $j \leq n$, then α satisfies the existential part of all clauses in $\phi_n^{[i,j]}$;
- if $j = n+1$, then $(\alpha, h(\rho_1))$ satisfies all clauses in $\phi_n^{[i,j]}$. (That is, assigning $X \cup T$ according to α and assigning z the value $h(\rho_1)$ satisfies $\phi_n^{[i,j]}$.)

(To find α, work backwards from y to determine the appropriate values of $t_{j-1}, t_{j-2}, \ldots, t_i$ to satisfy $\phi_n^j, \phi_n^{j-1}, \ldots, \phi_n^i$.)

Note that $h(\rho_1) = h(\rho_2) = h(\alpha)$. So $(\alpha, h(\alpha))$ falsifies C (since it extends ρ_C) and satisfies all axiom clauses used to derive L. This contradicts Lemma 2.3.1.

3. Let $j \leq n$. Assume to the contrary that $t_j \notin \text{var}(C)$. The argument is identical to that in Item 2 (only the indices differ): ρ_C falsifies C; ρ_1 extends it arbitrarily to all unassigned variables in $(X \cup T) \setminus \{t_i, \ldots, t_j\}$; ρ_2 is the extension of ρ_1 obtained by setting t_j so as to satisfy the equation $t_{i-1} + x_i + x_{i+1} + \ldots + x_j + t_j \equiv 0 \bmod 2$; (Here, if $i = 1$, discard t_0 from the equation; i.e. assume $t_0 = 0$); α is the unique extension of ρ_2 to $X \cup T$

satisfying $\phi_n^{[i,j]}$ (To obtain α, work forwards obtaining $t_i, t_{i+1}, \ldots, t_{j-1}$). Now $(\alpha, h(\alpha))$ contradicts Lemma 2.3.1.

4. Since $\textsc{Uci}(\Pi_L) = [i,j]$, variables t_k for $k \notin [i-1,j]$ do not appear in any of the used axioms (Observation 3.1.1) and hence do not appear in C. By the preceding three items, $\mathrm{var}(C) \cap T$ does not include any t_k with $k \in [i, j-1]$, includes t_{i-1} whenever $i > 1$, and includes t_j whenever $j < n+1$. The claim follows.

5. Assume to the contrary that for some $k \in [i,j]$, $x_k \notin \mathrm{var}(C) \cup \mathrm{var}(M)$. The argument is similar to that in Item 2: ρ_C falsifies C; ρ_1 extends it arbitrarily to all unassigned variables in $(X \setminus \{x_k\}) \cup (T \setminus \{t_i, \ldots, t_{j-1}\})$; y is the value of t_j if $j \leq n$ and the value of h otherwise (since $x_k \notin \mathrm{var}(M)$, ρ_1 is sufficient to evaluate h); ρ_2 is the extension of ρ_1 obtained by setting x_k so as to satisfy the equation $t_{i-1} + x_i + x_{i+1} + \ldots + x_j + y \equiv 0 \bmod 2$; (Here, if $i = 1$, discard t_0 from the equation; i.e. assume $t_0 = 0$); α is the unique extension of ρ_2 to $X \cup T$ satisfying $\phi_n^{[i,j]}$ (To obtain α, work forwards from t_i towards t_{j-1}). Now $(\alpha, h(\alpha))$ contradicts Lemma 2.3.1.

□

Lemma 3.3.3. *Let $L \in \mathcal{S}$ be derived in Π as $L = Res(L', L'', t_k)$. Then $\textsc{Uci}(\Pi_L) = [1, n+1]$, and $\textsc{Uci}(\Pi_{L'}), \textsc{Uci}(\Pi_{L''})$ partition $[1, n+1]$ into $[1,k], [k+1, n+1]$.*

Proof. Since $L \in \mathcal{S}$, L has no variable from T. By Lemma 3.3.2(4), $\textsc{Uci}(\Pi_L) = [1, n+1]$.

Since $L = \mathrm{Res}(L', L'', t_k)$, we have $\mathrm{var}(C') \cap T = \mathrm{var}(C'') \cap T = \{t_k\}$. By Lemma 3.3.2(2,3,4), $\textsc{Uci}(\Pi_{L'}), \textsc{Uci}(\Pi_{L''}) \in \{[1,k], [k+1, n+1]\}$.

If both $\textsc{Uci}(\Pi_{L'}), \textsc{Uci}(\Pi_{L''})$ equal $[k+1, n+1]$, then $\textsc{Uci}(\Pi_L) = [k+1, n+1]$, contradicting $\textsc{Uci}(\Pi_l) = [1, n+1]$.

If both $\textsc{Uci}(\Pi_{L'}), \textsc{Uci}(\Pi_{L''})$ equal $[1, k]$, then $\textsc{Uci}(\Pi_L) = [1, k]$. Since t_k is a pivot variable, $k \leq n$, contradicting $\textsc{Uci}(\Pi_l) = [1, n+1]$.

Hence one each of $\textsc{Uci}(\Pi_{L'}), \textsc{Uci}(\Pi_{L''})$ equals $[1, k]$ and $[k+1, n+1]$ as claimed. □

Lemma 3.3.4. *For all $L \in \mathcal{S}$, width$(C) = n$.*

Proof. Let $L \in \mathcal{S}$ be derived in Π as $L = \text{Res}(L', L'', t_k)$. Since all axioms create non-trivial strategies, neither M' nor M'' equals $*$. By the rules of M-Res, $M' = M'' = M \neq *$. We will show that in fact M must be a constant strategy, $M \in \{0, 1\}$.

By definition of $\mathcal{S}$, $\text{var}(C) \cap T = \emptyset$, and hence $\text{var}(C') \cap T = \text{var}(C'') \cap T = \{t_k\}$. By Lemma 3.3.3, $\textsc{Uci}(\Pi_L) = [1, n+1]$ is partitioned by $\textsc{Uci}(\Pi_{L'})$ and $\textsc{Uci}(\Pi_{L''})$ into $[1, k], [k+1, n+1]$.

Assume $\textsc{Uci}(\Pi_{L'}) = [1, k]$, $\textsc{Uci}(\Pi_{L''}) = [k+1, n+1]$; the argument in the other case is identical. Then $\text{var}(M) = \text{var}(M') \subseteq \text{var}(\phi^{[1,k]}) \cap X = \{x_1, \ldots, x_k\}$, and $\text{var}(M) = \text{var}(M'') \subseteq \text{var}(\phi^{[k+1,n+1]}) \cap X = \{x_{k+1}, \ldots, x_n\}$. The only way both these conditions can be satisfied is if $\text{var}(M) = \emptyset$; that is, M is a constant strategy.

Since $\textsc{Uci}(\Pi_L) = [1, n+1]$ and $\text{var}(M) = \emptyset$, Lemma 3.3.2(5) implies that $X \subseteq \text{var}(C)$. Therefore $\text{width}(C) = n$. □

Theorem 3.3.5. *Every T-regular refutation of LQParity$_n$ in M-Res has size $2^{\Omega(n)}$.*

Proof. Let Π be a T-regular refutation of LQParity_n in M-Res. Let $\mathcal{S}', \mathcal{S}$ be as defined just before Lemma 3.3.2. By definition, for each $L = (C, M) \in S'$, $\text{var}(C) \subseteq X$. Let $\widehat{\Pi} = \{C \mid L = (C, M) \in S'\}$. Then $\widehat{\Pi}$ contains a propositional resolution refutation of $\mathcal{C} = \{C \mid L = (C, M) \in S\}$. Therefore $\mathcal{C}$ is an unsatisfiable

CNF formula over the n variables in X. By Lemma 3.3.4, each clause in $\mathcal{C}$ has width n and so is falsified by exactly one assignment. Therefore, to ensure that each of the 2^n assignments falsifies some clause, (at least) 2^n clauses are required. Therefore $|\mathcal{C}| \geqslant 2^n$. Hence $|\Pi| \geqslant 2^n$. □

3.3.2 Completion Principle formulas

Our second hardness result for regular Merge Resolution is for the completion principle formulas, introduced in [49] (cf. Section 3.1).

Theorem 3.3.6. *Every $(A \cup B)$-regular refutation of CR_n in M-Res has size 2^{n-1}.*

The proof proceeds as follows: Let Π be a $(A \cup B)$-regular M-Res refutation of CR_n. Since every axiom has a variable from $A \cup B$ while the final clause in Π is empty, there is a maximal "component" of the proof leading to and including the final line, where all clauses are $(A \cup B)$-free. The clauses in this component involve only the X variables. We show that the "boundary" of this component is large, by showing in Lemma 3.3.7 that each clause here must be wide. (This idea was used in [63] to show that CR is hard for reductionless LD-Q-Res.)

To establish the width bound, we first note that except for the axioms L_A, L_B, no lines have trivial strategies. Since the pivots at the boundary are variables from $A \cup B$, which are all to the right of z, the merge maps incoming into each boundary resolution must be isomorphic. By analysing what axiom clauses cannot be used to derive lines just above the boundary, we show that many variables are absent in the corresponding merge maps, and invoking soundness of M-Res, we show that they must then be present in the boundary clause, making it wide.

Proof. (of Theorem 3.3.6) The statement of theorem is trivially true for $n = 1$. We prove it for $n \geq 2$.

Let Π be an $(A \cup B)$-regular refutation of CR_n (for $n \geq 2$) in M-Res. Define $\mathcal{S}'$ to be the set of those lines in Π where the clause part has no variable from $A \cup B$, and furthermore there is a path in G_Π from the line to the final empty clause via lines where all the clauses also have no variables from $A \cup B$. Let $\mathcal{S}$ denote the set of leaves in the subgraph of G_Π restricted to $\mathcal{S}'$; these are lines that are in $\mathcal{S}'$ but their parents are not in $\mathcal{S}'$. Note that no leaf of Π is in $\mathcal{S}'$ because all leaves of G_Π contain a variable in $A \cup B$.

By definition, for each $L = (C, M^z) \in S'$, $\mathrm{var}(C) \subseteq X$. The sub-derivation $\widehat{\Pi} = \{C \mid \exists L = (C, M^z) \in S'\}$ contains a propositional resolution refutation of the conjunction of clauses $F = \{C \mid \exists L = (C, M^z) \in S\}$. Hence F is an unsatisfiable CNF formula over the n^2 variables in X. We show below, in Lemma 3.3.7, that each clause in F has width at least $n-1$. Hence it is falsified by at most $2^{n^2-(n-1)}$ assignments. Therefore, to ensure that each of the 2^{n^2} assignments falsifies some clause, at least 2^{n-1} clauses are required. Therefore $|F| \geqslant 2^{n-1}$. Hence $|\Pi| = 2^{\Omega(n)}$. □

Lemma 3.3.7. *For all $L = (C, M^z) \in S$, width$(C) \geq n - 1$.*

Proof. Since $\mathrm{var}(C) \cap (A \cup B) = \emptyset$, L is not a leaf of Π. Say $L = \mathrm{Res}(L_1, L_2, v)$ where $L_1 = (C_1, M_1^z)$ and $L_2 = (C_2, M_2^z)$. Since $\mathrm{var}(C_1) \cap (A \cup B) \neq \emptyset$ and $\mathrm{var}(C_2) \cap (A \cup B) \neq \emptyset$, we have $v \in A \cup B$. Consider the case when $v \in A$; the argument for the case when $v \in B$ is symmetrically identical. Without loss of generality, assume that $v = a_n$; and $a_n \in C_1$ and $\overline{a_n} \in C_2$.

Since Π is $(A \cup B)$-regular, a_n does not occur as a pivot in the sub-derivation Π_{L_1}. Therefore $L_A \notin \mathrm{leaves}(G_{\Pi_{L_1}})$ (otherwise $\overline{a_n} \in C_1$, and therefore C_1 would be tautological clause, a contradiction). This implies that the sub-derivation Π_{L_1} cannot use any axiom that contains a positive A literal other than a_n, since such a literal would have to be eliminated by resolution before reaching C_1, requiring the corresponding negated literal, and L_A is the only axiom with negated literals from

A. That is, Π_{L_1} does not use any of the axioms A_{ij} for $i \in [n-1]$. The positive literal x_{ij} appears only in A_{ij}. Hence for $i \in [n-1]$, $j \in [n]$, x_{ij} is not a pivot in Π_{L_1} and hence does not appear in M_1^z. On the other hand, M_1^z is not trivial since some A_{nj} clause is used.

C_2 contains $\overline{a_n}$, but no other $\overline{a_i}$. So C_2 is not the axiom L_A. Hence M_2^z is not trivial.

Since the pivot a_n at the step obtaining line L is to the right of z, by the rules of M-Res, M_1^z and M_2^z are isomorphic. Hence for each $i \in [n-1]$, and each $j \in [n]$, $x_{ij} \notin \mathrm{var}(M_2^z)$. We claim the following:

Claim 3.3.8. Either for all $i \in [n-1]$, C_2 has a variable of the form x_{i*}, or for all $j \in [n]$, C_2 has a variable of the form x_{*j}.

In either case, C_2 has at least $n-1$ variables.

It remains to prove the claim.

Proof. (of Claim) We know that $\overline{a_n} \in C_2$, and for all $i \in [n-1]$, for all $j \in [n]$, $x_{ij} \notin \mathrm{var}(M_2^z)$. Aiming for contradiction, suppose that there exist $i \in [n-1]$ and $j \in [n]$ such that for all $\ell \in [n]$, $x_{i\ell} \notin \mathrm{var}(C_2)$, and for all $k \in [n]$, $\mathrm{var}(x_{kj}) \notin C_2$. Fix such an i, j.

Let ρ be the minimum partial assignment falsifying C_2. Then

- ρ sets $a_n = 1$, leaves all other variables in $A \cup B$ unset.
- ρ does not set any $x_{i\ell}$ or x_{kj}.

For $c \in \{0, 1\}$, extend ρ to α_c as follows: Set $a_i = 0, b_j = 0$, set all other unset variables from $A \cup B$ to 1. Set $x_{ij} = c$. All $x_{i\ell}$ other than x_{ij} set to 1. All x_{kj} other than x_{ij} set to 0. Set remaining variables arbitrarily (but in the same way in α_0 and α_1).

The common part of α_0 and α_1 satisfies all axiom clauses except A_{ij} and B_{ij}, and does not falsify any axiom. The extensions α_c satisfy one more axiom, and still do not falsify the remaining axiom (it has a universal literal z or $\overline{z}$). They both falsify C_2, since they extend ρ.

Since α_0 and α_1 agree everywhere except on x_{ij}, and since $x_{ij} \notin \mathrm{var}(M_2^z)$, it follows that $M_2^z(\alpha_0) = M_2^z(\alpha_1) = d$, say.

By Lemma 2.3.1, both (α_0, d) and (α_1, d) should falsify some axiom. However, $(\alpha_{\bar{d}}, d)$ actually satisfies all axioms, a contradiction. □

With the claim established, the proof of the lemma is complete. □

Corollary 3.3.9. *Regular M-Res is incomparable with the tree-like and general versions of Q-Res, QU-Res, CP + ∀Red, ∀Exp + Res, and IR.*

Proof. Let $S \in \{\text{Q-Res}, \text{QU-Res}, \text{CP} + \forall\text{Red}, \forall\text{Exp} + \text{Res}, \text{IR}\}$.

The CR_n formulas have polynomial-size refutations in tree-like S [48, 49] but require exponential-size refutations in regular M-Res (Theorem 3.3.6), so regular M-Res does not simulate tree-like or general versions of S.

The Equality formulas require exponential-size refutations in S [15, 16] but have polynomial-size refutations in regular M-Res [18], so S (and hence also tree-like S) does not simulate regular M-Res. □

3.4 A lower bound for Merge Resolution

In this section we turn towards the full system of Merge Resolution and consider the KBKF-lq formulas (cf. Section 3.1). Similarly as the LQParity formulas, these formulas were originally introduced as hard principles for LD-Q-Res [8]. Here we show that they are hard for the full system of Merge Resolution. This constitutes the first lower bound for unrestricted M-Res in the literature.

Theorem 3.4.1. *$size_{M\text{-}Res}$(KBKF-lq[n]) $= 2^{\Omega(n)}$.*

Proof idea We will show that, in any M-Res refutation of the KBKF-lq formulas, the literals over the variables in $F = \{f_1, f_2, \ldots, f_n\}$ must be removed before the strategies become 'very complex'. From this we conclude that there must be exponentially many lines.

To argue that literals over F must be removed before the strategies become 'very complex', we look at the form of the lines containing literals over F. If any such line has a 'very complex' strategy (by which we mean that for some $i \in [n]$, u_i depends on either d_i or e_i), then the literals over F cannot be removed from the clause.

Elaborating on the roadmap of the argument: Let Π be an M-Res refutation of KBKF-lq[n]. Each line in Π has the form $L = (C, M^{x_1}, \ldots, M^{x_n})$ where C is a clause over D, E, F, and each M^{x_i} is a merge map computing a strategy for x_i.

Define $\mathcal{S}'$ to be the set of those lines in Π where the clause part has no F variable and furthermore the line has a path in G_Π to the final empty clause via lines where all the clauses also have no F variables. Let $\mathcal{S}$ denote the set of leaves in the subgraph of G_Π restricted to $\mathcal{S}'$; these are lines that are in $\mathcal{S}'$ but their parents are not in $\mathcal{S}'$. Note that by definition, for each $L = (C, \{M^{x_i} \mid i \in [n]\}) \in \mathcal{S}'$, $\text{var}(C) \subseteq D \cup E$. No line in $\mathcal{S}'$ (and in particular, no line in $\mathcal{S}$) is an axiom since all axiom clauses have variables from F.

Recall that the variables of KBKF-lq[n] can be naturally grouped based on the quantifier prefix: for $i \in [n]$, the ith group has d_i, e_i, x_i, and the $(n+1)$th group has the F variables. By construction, the merge map for x_i does not depend on variables in later groups, as is indeed required for a countermodel. We say that a merge map for x_i has self-dependence if it does depend on d_i and/or e_i.

We show that every merge map at every line in $\mathcal{S}'$ is non-trivial (Lemma 3.4.6). Further, we show that at every line on the boundary of $\mathcal{S}'$, i.e. in $\mathcal{S}$, no merge map has self-dependence (Lemma 3.4.7). Using this, we conclude that $\mathcal{S}$ must be exponentially large, since in every countermodel the strategy of each variable must have self-dependence (Proposition 3.1.2).

In order to show that lines in $\mathcal{S}$ do not have self-dependence, we first establish several properties of the sets of axiom clauses used in a sub-derivation (Lemma 3.4.2, Lemma 3.4.3, Lemma 3.4.4, Lemma 3.4.5).

Detailed proof For a line $L \in \Pi$, let Π_L be the minimal sub-derivation of L, and let G_{Π_L} be the corresponding subgraph of G_Π with sink L. Let $\textsc{Uci}(\Pi_L) = \{i \in [0, n] \mid \text{leaves}(G_{\Pi_L}) \cap \mathcal{A}_i \neq \emptyset\}$. (Uci stands for UsedConstraintsIndex). Note that we are only looking at the clauses in $\mathcal{A}$ to define Uci.

Lemma 3.4.2. *For every line $L = (C, \{M^{x_i} \mid i \in [n]\})$ of Π,*

1. $\textsc{Uci}(\Pi_L) = \emptyset$ *if and only if* $C \cap F^1 \neq \emptyset$ *if and only if* $|C \cap F^1| = 1$.
2. $\textsc{Uci}(\Pi_L) \neq \emptyset$ *if and only if* $C \cap F^1 = \emptyset$.

Proof. Since the existential part of each clause in KBKF-lq$[n]$ is a Horn clause, and since the resolvent of Horn clauses is also Horn, $|C \cap F^1| \leq 1$ for each line of Π. It thus suffices to prove that $\forall L \in \Pi, \quad \textsc{Uci}(\Pi_L) = \emptyset \iff C \cap F^1 \neq \emptyset$.

($\Rightarrow$): For an arbitrary line $L \in \Pi$, suppose $\textsc{Uci}(\Pi_L) = \emptyset$, so L is derived from $\mathcal{B}$. Since $\text{var}_\exists(\mathcal{B}) = F$, $\text{var}(C) \subseteq F$. The existential part of these clauses is strict Horn, and the resolvent of strict Horn clauses is also strict Horn, so C is strict Horn. So $C \cap F^1 \neq \emptyset$.

($\Leftarrow$): The statement $C \cap F^1 \neq \emptyset \Rightarrow \textsc{Uci}(\Pi_L) = \emptyset$ holds at all axioms. Assume to the contrary that it does not hold everywhere in Π. Pick a highest L (closest to the axioms) for which this statement fails. That is, $C \cap F^1 \neq \emptyset$, and $\textsc{Uci}(\Pi_L) \neq \emptyset$. Let L', L'' be the parents of L in Π; by choice of L, both L' and L'' satisfy the statement. Let f_j be the positive literal in C (unique, because C is Horn). Without loss of generality, $f_j \in C'$. Since L' satisfies the statement, $\textsc{Uci}(\Pi_{L'}) = \emptyset$. So $\mathrm{var}(C') \subseteq F$, and since C' is Horn, $C' \setminus \{f_j\} \subseteq F^0$. Since $f_j \in C$, the pivot at this step is not f_j, so it must be an f_k for some $\overline{f_k} \in C'$. So $f_k \in C''$. Since L'' satisfies the statement, $\textsc{Uci}(\Pi_{L''}) = \emptyset$. But then $\textsc{Uci}(\Pi_L) = \textsc{Uci}(\Pi_{L'}) \cup \textsc{Uci}(\Pi_{L''}) = \emptyset$, contradicting our choice of L. Hence our assumption was wrong, and the statement holds for all L in Π. □

Lemma 3.4.3. *A line $L = (C, \{M^{x_i} \mid i \in [n]\})$ of Π with $\textsc{Uci}(\Pi_L) = \emptyset$ has these properties:*

1. *var$(C) \subseteq F$; for all $i \in [n]$, $M^{x_i} \in \{*, 0, 1\}$;*
2. *For some $j \in [n]$, $f_j \in C$ and $M^{x_j} \in \{0, 1\}$;*
3. *For $1 \leq i < j$, $f_i \notin$ var(C) and $M^{x_i} = *$;*
4. *For $j < i \leq n$, if $f_i \notin$ var(C), then $M^{x_j} \in \{0, 1\}$.*

Proof.

1. Since $\textsc{Uci}(\Pi_L) = \emptyset$, $\mathrm{var}(C) \subseteq \mathrm{var}_\exists(\mathcal{B}) = F$.

   All pivots in Π_L are from F, and all universal variables are left of F in the quantifier prefix. So no step in Π_L can use the merge operation to update merge maps; all steps in Π_L use only the select operation, which does not create any branching.

2. By Lemma 3.4.2, $|C \cap F^1| = 1$, so there is a unique j with the literal $f_j \in C$. This literal appears only in the clauses of $\mathcal{B}_j$, both of which create a non-trivial strategy for x_j. So $M^{x_j} \neq *$. By (Item 1) proven above, $M^{x_j} \in \{0, 1\}$.

3. Let k be the least index such that Π_L uses an axiom from $\mathcal{B}_k$. Since the positive literal f_j is in C and appears only in $\mathcal{B}_j$, $k \le j$. Assume $k < j$. The axiom from $\mathcal{B}_k$ introduces the positive literal f_k into Π_L, and by choice of k, no axiom in Π_L has the literal $\overline{f_k}$. Hence f_k cannot be removed by resolution, and so $f_k \in C$, contradicting the fact that C is Horn. So in fact $k = j$. This means that no axiom introduces the variables f_i, $i < j$, into Π_L, so $f_i \notin \text{var}(C)$. Furthermore, amongst all the axioms in $\mathcal{B}$, only the axioms in $\mathcal{B}_i$ have a non-trivial merge map for x_i. Hence for $i < j$, no non-trivial merge map for x_i is created.

4. Since $f_j \in C$, Π_L uses an axiom from $\mathcal{B}_j$. This axiom introduces the literals $\overline{f_i}$, for $j < i \le n$, into Π_L.

   If $\overline{f_i}$ is removed (by resolution) in Π_L, then an axiom from $\mathcal{B}_i$ must be used to introduce the positive literal f_i. This axiom created a non-trivial merge map for x_i, so the merge map for x_i at L is also non-trivial.

$\square$

Lemma 3.4.4. *Let $L = (C, \{M^{x_i} \mid i \in [n]\})$ be a line of Π with* $\textsc{Uci}(\Pi_L) \neq \emptyset$. *Then* $\textsc{Uci}(\Pi_L)$ *is an interval $[a, b]$ for some $0 \le a \le b \le n$. Furthermore, (in the items below, a, b refer to the endpoints of this interval), it has the following properties:*

1. *For $k \in [n] \cap [a, b]$, $M^{x_k} \neq *$.*

2. *If $a \ge 1$, then $|\{d_a, e_a\} \cap C| = 1$. If $a = 0$, then C does not have any positive literal.*

3. *If $b < n$, then $\overline{d_{b+1}}, \overline{e_{b+1}} \in C$.*

4. *For all $k \in [n] \setminus [a, b]$, (i) $d_k, e_k \notin var(M^{x_k})$, and (ii) if $M^{x_k} = *$ then $\overline{f_k} \in C$.*

Proof. Assume to the contrary that $\textsc{Uci}(\Pi_L)$ is not an interval. Then there exist $0 \le a < c < b \le n$ such that $a, b \in \textsc{Uci}(\Pi_L)$ but $c \notin \textsc{Uci}(\Pi_L)$. Let L_1 be the first line in Π_L such that $\textsc{Uci}(\Pi_{L_1})$ intersects both $[0, c-1]$ and $[c+1, n]$ (note that L_1 exists). Since leaves have singleton $\textsc{Uci}$ sets, L_1 is not a leaf. Say $L_1 = \text{Res}(L_2, L_3, v)$. By our choice of L_1, exactly one each of $\textsc{Uci}(\Pi_{L_2})$ and $\textsc{Uci}(\Pi_{L_3})$ is a non-empty subset of $[0, c-1]$ and of $[c+1, n]$. So $v \in \text{var}_\exists(\mathcal{A}_{[0,c-1]})$ and $v \in \text{var}_\exists(\mathcal{A}_{[c+1,n]})$. But $\text{var}_\exists(\mathcal{A}_{[0,c-1]}) \cap \text{var}_\exists(\mathcal{A}_{[c+1,n]}) = F$, and by Lemma 3.4.2, both C_2 and C_3 contain variables of F only in negated form. So no variable from F can be a resolution pivot, a contradiction. It follows that $\textsc{Uci}(\Pi_L)$ is an interval.

1. For $k \in [n] \cap [a, b]$, some axiom from $\mathcal{A}_k$ has been used to derive L. Both these axioms create non-trivial strategies for x_k. Subsequent M-Res steps cannot make a non-trivial strategy trivial.

2. Consider first the case $a \geq 1$. Since C is a Horn clause, C can contain at most one of the literals d_a, e_a.

   Since $a \in \textsc{Uci}(\Pi_L)$, at least one of A_a^d, A_a^e appears in leaves(Π_L), so at least one of the literals d_a, e_a is introduced into Π_L. Since A_{a-1}^d and A_{a-1}^e are the only axioms that contain $\overline{d_a}$ or $\overline{d_a}$, and since neither of these is used in Π_L, therefore the positive literals d_a, e_a, if introduced, cannot be removed through resolution. Hence at least one of them is in C. It follows that C has exactly one of d_a, e_a.

   If $a = 0$, Π_L uses the clause A_0 which has only negative literals. The resolvent of such a clause and a Horn clause also has only negative literals. Following the sequence of resolutions on the path from a leaf using A_0 to C shows that C has only negative literals.

3. Since $b < n$ and $b \in \textsc{Uci}(\Pi_L)$, some clause from $\mathcal{A}_b$ is used in Π_L and introduces the literals $\overline{d_{b+1}}, \overline{e_{b+1}}$ into Π_L. Since $b + 1 \notin \textsc{Uci}(\Pi_L)$, no leaf of Π_L contains the positive literals d_{b+1}, e_{b+1}. So $\overline{d_{b+1}}$ and $\overline{e_{b+1}}$ cannot be removed through resolution.

4. For $k > b$, no leaf in Π_L contains the positive literals d_k, e_k. For $k < a$, no leaf in Π_L contains the negative literals $\overline{d_k}, \overline{e_k}$. Thus, for $k \notin [a, b]$, the variables d_k, e_k are not used as resolution pivots anywhere in Π_L, and hence are not queried in any of the merge maps.

   Each negative literal $\overline{f_k}$ is present in every clause of $\mathcal{A}$, and hence is introduced into Π_L. If $M^{x_k} = *$, then $B_k^0, B_k^1 \notin \text{leaves}(\Pi_L)$ (both of them have non-trivial merge maps for x_k). Since these are the only clauses with the positive literal f_k, the literal $\overline{f_k}$ cannot be removed in Π_L; hence $\overline{f_k} \in C$.

□

Lemma 3.4.5. *For any line $L = (C, \{M^{x_i} \mid i \in [n]\})$ in Π, and any $k \in [n]$, if $\{d_k, e_k\} \cap var(M^{x_k}) \neq \emptyset$, then $\text{UCI}(\Pi_L) = [a, n]$ for some $a \leq k - 1$.*

Proof. Since $\{d_k, e_k\} \cap \text{var}(M^{x_k}) \neq \emptyset$, either d_k or e_k must be used as a pivot in Π_L, and hence must appear in both polarities in Π_L. The variables d_k, e_k appear positively only in $\mathcal{A}_k$, and negatively only in $\mathcal{A}_{k-1}$. Hence $a \leq k - 1$.

Suppose $b < n$. By Lemma 3.4.4 (3), both $\overline{d_{b+1}}$ and $\overline{e_{b+1}}$ are in C. Consider any path ρ in Π from L to the final line $L_\Box$. At every line on this path, the merge map for x_k queries at least one of d_k, e_k since it is at least as complex as the merge map M^{x_k}. Along this path, both d_{b+1} and e_{b+1} must appear as pivots, since the negated literals are eventually removed. Pick the first such step on ρ, and assume without loss of generality that the pivot is d_{b+1} (the other case is symmetric). So $\overline{d_{b+1}}$ is present in the line, say L_1, on ρ, and d_{b+1} is present in the clause L_2 with which it is resolved to obtain $L_3 = \text{Res}(L_2, L_1, d_{b+1})$ on ρ. By Lemma 3.4.4 (2), $\text{UCI}(\Pi_{L_2}) = [b + 1, b']$ for some $b' \geq b + 1$. Hence by Lemma 3.4.4 (4), $d_k, e_k \notin \text{var}((M_2)^{x_k})$. However, $\{d_k, e_k\} \cap \text{var}((M_1)^{x_k}) \neq \emptyset$. Since this resolution on d_{b+1} is not blocked, it must be the case that $(M_2)^{x_k} = *$. Hence, by Lemma 3.4.4 (4), $\overline{f_k} \in C_2$ and so $\overline{f_k} \in C_3$. To remove this literal, at some later point along ρ, f_k must appear as pivot. However, at that point, the line from ρ has a complex merge map for x_k, while the line with the positive literal f_k has a non-trivial constant merge map (by Lemma 3.4.3 (2)). Hence the resolution on f_k is blocked, a contradiction.

It follows that $b = n$. □

Lemma 3.4.6. *For all $L \in \mathcal{S}'$, for all $k \in [n]$, $M^{x_k} \neq *$.*

Proof. Consider a line $L = (C, \{M^{x_i} \mid i \in [n]\}) \in \mathcal{S}'$. Since $L \in \mathcal{S}'$, $\text{var}(C) \cap F = \emptyset$, so $C \cap F^1 = \emptyset$. By Lemma 3.4.2, $\textsc{Uci}(\Pi_L) \neq \emptyset$. Since every clause in $\mathcal{A}$ contains all literals in F^0, for each $k \in [n]$, Π_L has a leaf where the clause contains $\overline{f_k}$. This literal is removed in deriving L, so Π_L also has a leaf where the clause contains the positive literal f_k. That is, it uses an axiom from $\mathcal{B}_k$; this leaf has a non-trivial merge map for x_k. Since a step in M-Res cannot make a non-trivial merge map trivial, the merge map for x_k at L is non-trivial. □

Lemma 3.4.7. *For all $L \in \mathcal{S}$, for all $k \in [n]$, $d_k, e_k \notin var(M^{x_k})$.*

Proof. Consider a line $L \in \mathcal{S}$; $L = (C, \{M^{x_i} \mid i \in [n]\})$. Assume to the contrary that for some $k \in [n]$, $\{d_k, e_k\} \cap \text{var}(M^{x_k}) \neq \emptyset$.

Line L is obtained by performing resolution on two non-$\mathcal{S}'$ clauses with a pivot from F. Let $L = \text{Res}(L', L'', f_\ell)$ for some $\ell \in [n]$; $f_\ell \in C'$ and $\overline{f_\ell} \in C''$. Since L has no variable in F, f_ℓ is the only variable from F in $\text{var}(C')$ and $\text{var}(C'')$.

Since C' has the literal $f_\ell \in F^1$, by Lemma 3.4.2, $\textsc{Uci}(\Pi_{L'}) = \emptyset$ and L' is derived exclusively from $\mathcal{B}$. Since $D \cup E$ and $\text{var}(\mathcal{B})$ are disjoint, all the merge maps in L' have no variable from $D \cup E$. So M^{x_k} gets its $D \cup E$ variables from $(M'')^{x_k}$. Since this does not block the resolution step, $(M')^{x_k}$ must be trivial and $M^{x_k} = (M'')^{x_k}$. Since $\text{var}(C') \cap F = f_\ell$, by Lemma 3.4.3 (2),(3),(4), $k < \ell$.

The line L'' has no literal from F^1, so by Lemma 3.4.2, $\textsc{Uci}(\Pi_{L''}) \neq \emptyset$. It has a merge map for x_k involving at least one of d_k, e_k, so by Lemma 3.4.5,

$\textsc{Uci}(\Pi_{L''}) = [a, n]$ for some $a \le k-1$. Thus we have $a \le k-1 < k < \ell \le n$.

Consider the resolution of L' with L''. By Lemma 3.4.3 (2), $(M')^{x_\ell} \in \{0,1\}$, and by Lemma 3.4.4 (1), $(M'')^{x_\ell} \neq *$. To enable this resolution, $(M'')^{x_\ell} = (M')^{x_\ell}$. The clauses A_ℓ^d and A_ℓ^e give rise to different constant strategies for x_ℓ. So the derivation of L'' uses exactly one of these two clauses. Assume it uses A_ℓ^d; the other case is symmetric. Since $a < \ell$, the derivation of L'' uses a clause from $A_{\ell-1}$, introducing literals $\overline{d_\ell}$ and $\overline{e_\ell}$. Since the only clause containing positive literal e_ℓ is not used, $\overline{e_\ell}$ survives in C''. Going from L'' to L removes only $\overline{f_\ell}$, so $\overline{e_\ell} \in C$.

To summarize, at this stage we know that $L \in \mathcal{S}$, $\overline{e_\ell} \in C$, $\{d_k, e_k\} \cap \mathrm{var}(M^{x_k}) \neq \emptyset$, $M^{x_\ell} \in \{0,1\}$ and $1 \le k < \ell \le n$.

Fix any path ρ in G_Π from L to $L_\Box$. Along this path, e_ℓ appears as the pivot somewhere, since the literal $\overline{e_\ell}$ is eventually removed. Consider the resolution step at that point, say $C_1 = \mathrm{Res}(C_2, C_3, e_\ell)$, with C_3 being the clause at the line on ρ. At the corresponding line L_3, the strategies are at least as complex as those at L. Hence $\mathrm{var}(M_3^{x_k}) \cap \{d_k, e_k\} \neq \emptyset$. On the other hand, C_2 has the positive literal e_ℓ. By Lemma 3.4.4, for the corresponding line L_2, $\textsc{Uci}(\Pi_{L_2}) = [\ell, c]$ for some $c \ge \ell$. Since $k < \ell$, by Lemma 3.4.4, $\{d_k, e_k\} \cap \mathrm{var}(M_2^{x_k}) = \emptyset$. However, the path from L_2 to L_1 and thence to $L_\Box$ along ρ witnesses that $L_2 \in \mathcal{S}'$, so by Lemma 3.4.6, $(M_2)^{x_k} \neq *$. Thus $M_2^{x_k}$ and $M_3^{x_k}$ are non-trivial but not isomorphic, and this blocks the resolution on e_ℓ.

Thus our assumption that $\{d_k, e_k\} \cap \mathrm{var}(M^{x_k}) \neq \emptyset$ must be false. The lemma is proved. $\square$

Proof. (of Theorem 3.4.1) Let Π be a refutation of KBKF-lq$[n]$ in M-Res. Let $\mathcal{S}', \mathcal{S}$ be as defined in the beginning of this section. Let the final line of Π be $L_\Box = (\Box, \{s^{x_i} \mid i \in [n]\})$, and for $i \in [n]$, let h_i be the functions computed by the merge map s^{x_i}. By soundness of M-Res, the functions $\{h_i\}_{i \in [n]}$ form a countermodel for KBKF-lq$[n]$.

For each $a \in \{0,1\}^n$, consider the assignment α to the variables of $D \cup E$ where $d_i = a_i$, $e_i = \overline{a_i}$. Call such an assignment an anti-symmetric assignment. Given such an assignment, walk from $L_\Box$ towards the leaves of Π as far as is possible while maintaining the following invariant at each line $L = (C, \{M^{x_i} \mid i \in [n]\})$ along the way:

1. α falsifies C, and

2. for each $i \in [n]$, $h_i(\alpha) = M^{x_i}(\alpha)$.

Clearly this invariant is initially true at $L_\Box$, which is in $\mathcal{S}'$. If we are currently at a line $L \in \mathcal{S}'$ where the invariant is true, and if $L \notin \mathcal{S}$, then L is obtained from lines L', L''. The resolution pivot in this step is not in F, since that would put L in $\mathcal{S}$. So both L' and L'' are in $\mathcal{S}'$, and the pivot is in $D \cup E$. Let the pivot be in $\{d_\ell, e_\ell\}$ for some $\ell \in [n]$. Depending on the pivot value, exactly one of C', C'' is falsified by α; say C' is falsified. By Lemma 3.4.6, for each $i \in [n]$, both $(M')^{x_i}$ and $(M'')^{x_i}$ are non-trivial. By definition of the M-Res rule,

- For $i < \ell$, $(M')^{x_i}$ and $(M'')^{x_i}$ are isomorphic (otherwise the resolution is blocked), and $M^{x_i} = (M')^{x_i} = (M'')^{x_i}$.
- For $i \geq \ell$, there are two possibilities:
  (1) $(M')^{x_i}$ and $(M'')^{x_i}$ are isomorphic, and $M^{x_i} = (M')^{x_i}$.
  (2) M^{x_i} is a merge of $(M')^{x_i}$ and $(M'')^{x_i}$ with the pivot variable queried. By definition of the merge operation, since C' is falsified by α,
  $M^{x_i}(\alpha) = (M')^{x_i}(\alpha)$.

Thus in all cases, for each i, $h_i(\alpha) = M^{x_i}(\alpha) = (M')^{x_i}(\alpha)$. Hence L' satisfies the invariant.

We have shown that as long as we have not encountered a line in $\mathcal{S}$, we can move further. We continue the walk until a line in $\mathcal{S}$ is reached. We denote the line so reached by $P(\alpha)$. Thus P defines a map from anti-symmetric assignments to $\mathcal{S}$.

Suppose $P(\alpha) = P(\beta) = (C, \{M^{x_i} \mid i \in [n]\})$ for two distinct anti-symmetric assignments obtained from $a, b \in \{0,1\}^n$ respectively. Let j be the least index in $[n]$ where $a_j \neq b_j$. By Lemma 3.4.7, M^{x_j} depends only on $\{d_i, e_i \mid i < j\}$, and α, β agree on these variables. Thus we get the equalities
$a_j = h_j(\alpha) = M^{x_j}(\alpha) = M^{x_j}(\beta) = h_j(\beta) = b_j$, where the first and last equalities follow from Proposition 3.1.2, the third equality from Lemma 3.4.7 and choice of j, and the second and fourth equalities by the invariant satisfied at $P(\alpha)$ and $P(\beta)$ respectively. This contradicts $a_j \neq b_j$.

We have established that the map P is one-to-one. Hence, $\mathcal{S}$ has at least as many lines as anti-symmetric assignments, so $|\Pi| \geq |\mathcal{S}| \geq 2^n$. □

Corollary 3.4.8. *M-Res is incomparable with QU-Res and CP + ∀Red.*

Proof. Theorem 3.4.1 shows that the KBKF-lq[n] formula requires exponential-size refutations in M-Res. It has polynomial-size refutations in QU-Res [8], and also in CP + ∀Red (since CP + ∀Red simulates QU-Res [22]). The other direction follows from the Equality formulas, as already mentioned in the proofs of
Corollary 3.2.7, Corollary 3.3.9. □

Chapter 4

Power of Merge Resolution

In this chapter, we will show that M-Res has exponential advantage over the two most powerful resolution-based QBF proof systems, reduction-based system LQU^+-Res and expansion-based system IRM. This is shown using modifications of two well-known formula families: KBKF-lq [8] which was shown hard for M-Res in the previous chapter, and QUParity [21] which we believe is also hard. The main observation is that the reason making these formulas hard for M-Res is the mismatch of partial strategies at some point in the refutation. This mismatch can be eliminated if the formulas are modified appropriately. The resultant formulas, called KBKF-lq-split and MParity, have polynomial-size refutations in M-Res but require exponential-size refutations in IRM and LQU^+-Res respectively.

4.1 Advantage over IRM

To show that M-Res is not simulated by IRM, we use the KBKF-lq formula family from the previous chapter. We reproduce the definition of this family below and then define two further variants that will be useful for our purpose.

KBKF-lq$[n]$ is the QBF with the quantifier prefix $\exists d_1, e_1, \forall x_1, \ldots, \exists d_n, e_n, \forall x_n,$

$\exists f_1, \ldots, f_n$ and with the following clauses:

$$
\begin{array}{lll}
A_0 = \{\overline{d_1}, \overline{e_1}, \overline{f_1}, \ldots, \overline{f_n}\} & & \\
A_i^d = \{d_i, x_i, \overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\} & A_i^e = \{e_i, \overline{x_i}, \overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\} & \forall i \in [n-1] \\
A_n^d = \{d_n, x_n, \overline{f_1}, \ldots, \overline{f_n}\} & A_n^e = \{e_n, \overline{x_n}, \overline{f_1}, \ldots, \overline{f_n}\} & \\
B_i^0 = \{x_i, f_i, \overline{f_{i+1}}, \ldots \overline{f_n}\} & B_i^1 = \{\overline{x_i}, f_i, \overline{f_{i+1}}, \ldots \overline{f_n}\} & \forall i \in [n-1] \\
B_n^0 = \{x_n, f_n\} & B_n^1 = \{\overline{x_n}, f_n\} &
\end{array}
$$

We now define two new formula families: KBKF-lq-weak and KBKF-lq-split.

KBKF-lq-weak$[n]$ has the same quantifier prefix as KBKF, and all the A-clauses of KBKF-lq, but it has the following clauses instead of B_i^0 and B_i^1:

$$
\left.\begin{array}{ll}
\text{weak-B}_i^0 & = d_i \vee B_i^0 \\
\text{weak-B}_i^1 & = \overline{d_i} \vee B_i^1
\end{array}\right\} \quad \forall i \in [n]
$$

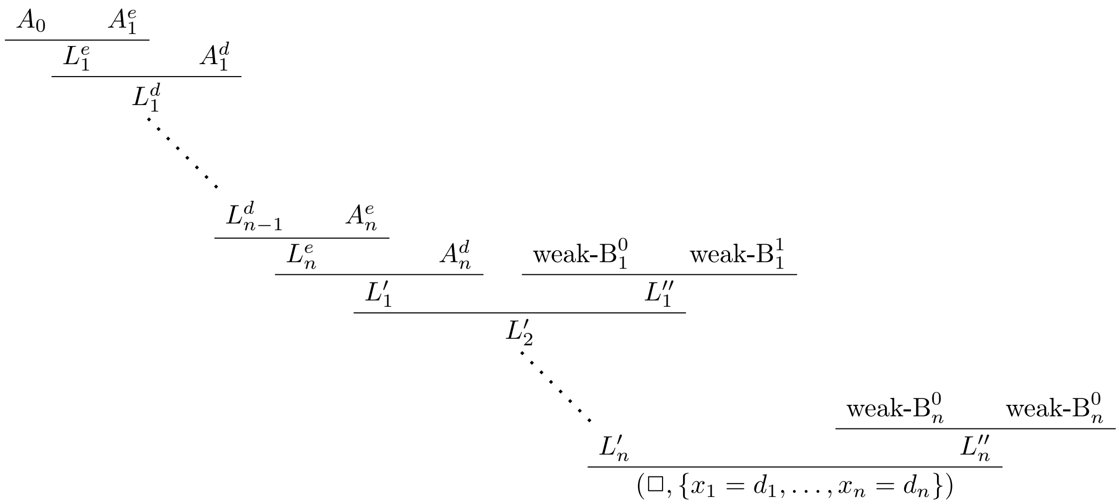
KBKF-lq-split$[n]$ has all variables of KBKF-lq and one new variable t quantified existentially in the first block, so the quantifier prefix for this formula is $\exists t, \exists d_1, e_1, \forall x_1, \ldots, \exists d_n, e_n, \forall x_n, \exists f_1, \ldots, f_n$. It has all the A-clauses of KBKF-lq, but the following clauses instead of B_i^0 and B_i^1:

$$
\left.\begin{array}{ll}
\text{split-B}_i^0 & = t \vee B_i^0 \\
\text{split-B}_i^1 & = t \vee B_i^1 \\
T_i^0 & = \{\bar{t}, d_i\} \\
T_i^1 & = \{\bar{t}, \overline{d_i}\}
\end{array}\right\} \quad \forall i \in [n]
$$

Lemma 4.1.1. *KBKF-lq-weak has polynomial-size M-Res refutations.*

Proof. Let L_i'' denote the M-Res-resolvent of weak-B_i^0 and weak-B_i^1. It has only one non-trivial merge-map, setting $x_i = d_i$. Starting with A_0, resolve in sequence with A_1^e, A_1^d, A_2^e, A_2^d, and so on up to A_n^e, A_n^d to derive the line with all negated f literals and merge-maps computing $x_i = d_i$ for each i. Now sequentially resolve this with L_1'', L_2'', up to L_n'' to obtain the empty clause. It can be verified that none of these resolutions are blocked, and the final merge-maps compute the winning strategy $x_i = d_i$ for each i.

The refutation is pictorially depicted below. The abbreviations A_0, A_i^d, weak-B_i^0 etc. will denote the clause, merge-map pair corresponding to the respective axioms.



Here $L_i^e, L_i^d, L_i', L_i''$, for all $i \in [n]$, are the following lines, with only non-trivial merge-maps written explicitly:

- $L_1^e = \left(\left\{\overline{d_1}, \overline{d_2}, \overline{e_2}, \overline{f_1}, \ldots, \overline{f_n}\right\}, \{x_1 = 1\}\right)$
- $L_i^e = \left(\left\{\overline{d_i}, \overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\right\}, \{x_1 = d_1, \ldots, x_{i-1} = d_{i-1}, x_i = 1\}\right)$ for all $i \in [2, n-1]$
- $L_n^e = \left(\left\{\overline{d_n}, \overline{f_1}, \ldots, \overline{f_n}\right\}, \{x_1 = d_1, \ldots, x_{n-1} = d_{n-1}, x_n = 1\}\right)$
- $L_i^d = \left(\left\{\overline{d_{i+1}}, \overline{e_{i+1}}, \overline{f_1}, \ldots, \overline{f_n}\right\}, \{x_1 = d_1, \ldots, x_i = d_i\}\right)$ for all $i \in [n-1]$
- $L_i' = \left(\left\{\overline{f_i}, \ldots, \overline{f_n}\right\}, \{x_1 = d_1, \ldots, x_n = d_n\}\right)$ for all $i \in [n]$

- $L''_i = \left(\left\{f_i, \overline{f_{i+1}}, \ldots \overline{f_n}\right\}, \{x_i = d_i\}\right)$ for all $i \in [n-1]$
- $L''_n = (\{f_n\}, \{x_n = d_n\})$ □

Lemma 4.1.2. *KBKF-lq-split has polynomial-size M-Res refutations.*

Proof. For each $i \in [n]$ and $k \in \{0, 1\}$, resolving split-B_i^k and T_i^k yields weak-B_i^k. This gives us the KBKF-lq-weak formula family which, as shown in Lemma 4.1.1, has polynomial-size M-Res refutations. □

Theorem 4.1.3. *IRM does not simulate M-Res.*

Proof. The KBKF-lq-split formula family witnesses the separation. By Lemma 4.1.2, it has polynomial size M-Res refutations. Restricting it by setting $t = 0$ gives the family KBKF-lq, which requires exponential size to refute in IRM, [21]. Since IRM is closed under restrictions (Lemma 11 in [21]), KBKF-lq-split also requires exponential size to refute in IRM. □

4.2 Advantage over LQU$^+$-Res

To show that LQU$^+$-Res does not simulate M-Res, we define a new formula family called MParity. This family is a modification of the QUParity formula family [21], which is a variant of the QParity and LQParity families from the previous chapter.

We reproduce the definition of LQParity from the previous chapter, informally describe the variant QUParity, and then define our new variant MParity. Let $\widehat{\text{parity}}^c(y_1, y_2, \ldots, y_k, z)$ abbreviate $\bigwedge_{C \in \text{parity}^c(y_1, y_2, \ldots, y_k)} \left((C \vee z) \wedge (C \vee \overline{z})\right)$.

LQParity_n is the QBF $\exists x_1, \ldots, x_n, \forall z, \exists t_1, \ldots, t_n. \left(\bigwedge_{i \in [n+1]} \phi_n^i \right)$ where

$$
\begin{aligned}
\phi_n^1 &= \widehat{\text{parity}}^c(x_1, t_1, z) \\
\phi_n^i &= \widehat{\text{parity}}^c(t_{i-1}, x_i, t_i, z), \qquad \forall i \in [2, n] \\
\phi_n^{n+1} &= (t_n \vee z) \wedge \left(\overline{t_n} \vee \overline{z}\right).
\end{aligned}
$$

QUParity is obtained from LQParity by duplicating the universal variable. That is, the block $\forall z$ is replaced with the block $\forall z_1, z_2$. Each clause of the form $C \cup \{z\}$ in LQParity is replaced with the clause $C \cup \{z_1, z_2\}$, and each clause of the form $C \cup \{\overline{z}\}$ is replaced with the clause $C \cup \{\overline{z_1}, \overline{z_2}\}$.

We will be inspired by the short LD-Q-Res refutation of QParity (from [31, p. 54]). This refutation relies on the fact that most axioms of QParity do not have universal variable z. This enables steps in which a merged literal z^* is present in one antecedent but there is no literal over z in the other antecedent. LQParity is created from QParity by replacing each clause C not containing z by two clauses $C \vee z$ and $C \vee \overline{z}$. Since, every axiom of LQParity (and hence also each derived clause) now has a literal over z, we can no longer resolve clauses containing the merged literal z^* with any other clause. This forbids the creation of merged literals, which in turn, forbids all possible short refutations. The same problem seems to occur in M-Res also — though M-Res allows resolution steps if the merge-maps are isomorphic, we do not know of any way of making them isomorphic. This leads us to define the new variant MParity. We notice that if the formula family is modified appropriately, we can indeed make the merge-maps isomorphic, and additionally throwing in the modifications of LQParity and QUParity does not destroy this feature. This leads us to define the modified family MParity.

Definition 4.2.1. $\mathrm{MParity}_n$ is the following QBF:

$$\underset{i,j\in[n]}{\exists} a_{i,j}, \exists x_1, \ldots, x_n, \forall z_1, z_2, \exists t_1, \ldots, t_n. \left(\bigwedge_{i\in[n+1]} \psi_i \right)$$

where each ψ_i contains the following clauses:

- For $i = 1$, for all $C \in \mathrm{parity}^c(x_1, t_1)$, the clauses
  $A^0_{1,C} = C \cup \{z_1, z_2, a_{1,n}\}$ and $A^1_{1,C} = C \cup \{\overline{z_1}, \overline{z_2}, a_{1,n}\}$
- For all $i \in [2, n-1]$, for all $C \in \mathrm{parity}^c(t_{i-1}, x_i, t_i)$, the clauses
  $A^0_{i,C} = C \cup \{z_1, z_2, a_{i,n}\}$ and $A^1_{i,C} = C \cup \{\overline{z_1}, \overline{z_2}, a_{i,n}\}$.
- For $i = n$, for all $C \in \mathrm{parity}^c(t_{n-1}, x_n, t_n)$, the clauses
  $A^0_{i,C} = C \cup \{z_1, z_2\}$ and $A^1_{i,C} = C \cup \{\overline{z_1}, \overline{z_2}\}$.
- For $i = n+1$, the clauses $\{t_n, z_1, z_2\}$ and $\{\overline{t_n}, \overline{z_1}, \overline{z_2}\}$.
- For all $i \in [n-1]$, the following clauses:

$$B^0_{i,j} = \{\overline{a_{i,j}}, x_j, a_{i,j-1}\}, \quad B^1_{i,j} = \{\overline{a_{i,j}}, \overline{x_j}, a_{i,j-1}\} \qquad \forall j \in \{n, n-1, \ldots, i+2\}$$
$$B^0_{i,i+1} = \{\overline{a_{i,i+1}}, x_{i+1}\}, \quad B^1_{i,i+1} = \{\overline{a_{i,i+1}}, \overline{x_{i+1}}\}$$

We can adapt the LD-Q-Res refutation of QParity to an M-Res refutation of MParity. We describe below exactly how this is achieved. The proof has two stages. In the first stage, the a variables are eliminated. The role of these $a_{i,j}$ variables and the B-clauses is to build up complex merge-maps meeting the isomorphism condition, so that subsequent resolution steps are enabled. In the second phase, the LD-Q-Res refutation of QParity is mimicked, eliminating the t variables.

(In the proofs below, notice that each line contains a single merge-map. This is done because the merge-maps for z_1 and z_2 in every line are same. So, we write them only once to save space.)

For $i \in [n+1]$, let g_i be the function $\oplus_{j \geq i} x_j$, and let h_i denote its complement. (The parity of an empty set of variables is 0; thus $g_{n+1} = 0$ and $h_{n+1} = 1$.) Let M_i^1 (resp. M_i^0) be the smallest merge-map which queries variables in the order $x_i, \ldots, x_n$ and computes the function g_i (resp. h_i). Note that both these branching programs have $2(n-i)+1$ internal nodes and two leaf nodes labelled 0 and 1.

The main idea is to replace the constant merge-maps in the axioms of $A_{i,C}^0$ and $A_{i,C}^1$ by the merge-maps M_{i+1}^0 and M_{i+1}^1 — the clause, merge-map pairs so generated will be denoted by $\widetilde{\psi_i}$ (and are defined below). These merge-maps will allow us to pass the isomorphism checks later in the proofs.

For $i \in [n]$, let $\widetilde{\psi_i}$ be the following sets of clause, merge-map pairs:

$$\widetilde{\psi_i} = \left\{ \left(C, M_{i+1}^b\right) \mid C \in \text{parity}_n^c(t_{i-1}, x_i, t_i), b \in \{0,1\} \right\} \qquad \forall i \in [2, n]$$
$$\widetilde{\psi_1} = \left\{ \left(C, M_2^b\right) \mid C \in \text{parity}_n^c(x_1, t_1), b \in \{0,1\} \right\}$$

Lemma 4.2.2. *For all $i \in [n]$, $\psi_i \vdash_{\text{M-Res}} \widetilde{\psi_i}$. Moreover the size of these derivations is polynomial in n.*

Proof. At $i = n$, $\widetilde{\psi_n}$ is the same as ψ_n so there is nothing to prove.

Consider now an $i \in [n-1]$. For each $b \in \{0,1\}$ and each $C \in \text{parity}^c(t_{i-1}, x_i, t_i)$ (if $i = 1$, omit t_{i-1}), the clause $A_{i,C}^b \in \psi_i$ yields the line $(C \cup \{a_{i,n}\}, M_{n+1}^{1-b})$. Resolving each of these with each of $B_{i,n}^d$ for $d \in \{0,1\}$, we obtain four clauses that can be resolved in two pairs to produce the lines $(C \cup \{a_{i,n-1}\}, M_n^b)$. Repeating this process successively for $j = n, n-1, \ldots, i+2$, using the clause pairs $B_{i,j}^d$ with the previously derived clauses, we can obtain each $(C \cup \{a_{i,j}\}, M_{j+1}^b)$. In each stage, the index j of the variable $a_{i,j}$ present in the clause decreases, while the merge-map accounts for one more variable. Finally, when we use the clause pairs $B_{i,i+1}^d$, the $a_{i,i+1}$ variable is eliminated, variables $x_{i+1}, \ldots, x_n$ are accounted for in the merge-map, and we obtain the lines (C, M_{i+1}^b), corresponding to the clauses in $\widetilde{\psi_i}$.

The derivation at one stage is as shown below.

$$\dfrac{\dfrac{\left(C\cup\{a_{i,j}\},M^1_{j+1}\right) \qquad \overbrace{\left(\{\overline{a_{i,j}},x_j,a_{i,j-1}\},*\right)}^{B^0_{i,j}}}{\left(C\cup\{x_j,a_{i,j-1}\},M^1_{j+1}\right)} \qquad \dfrac{\left(C\cup\{a_{i,j}\},M^0_{j+1}\right) \qquad \overbrace{\left(\{\overline{a_{i,j}},\overline{x_j},a_{i,j-1}\},*\right)}^{B^1_{i,j}}}{\left(C\cup\{\overline{x_j},a_{i,j-1}\},M^0_{j+1}\right)}}{\left(C\cup\{a_{i,j-1}\},M^1_j\right)}$$

$$\dfrac{\dfrac{\left(C\cup\{a_{i,j}\},M^1_{j+1}\right) \qquad \overbrace{\left(\{\overline{a_{i,j}},\overline{x_j},a_{i,j-1}\},*\right)}^{B^1_{i,j}}}{\left(C\cup\{\overline{x_j},a_{i,j-1}\},M^1_{j+1}\right)} \qquad \dfrac{\left(C\cup\{a_{i,j}\},M^0_{j+1}\right) \qquad \overbrace{\left(\{\overline{a_{i,j}},x_j,a_{i,j-1}\},*\right)}^{B^0_{i,j}}}{\left(C\cup\{x_j,a_{i,j-1}\},M^0_{j+1}\right)}}{\left(C\cup\{a_{i,j-1}\},M^0_j\right)}$$

$\square$

In the second phase, we successively eliminate the t variables in stages.

Lemma 4.2.3. *The following derivations can be done in M-Res in size polynomial in n:*

1. *For $i = n, n-1, \ldots, 2$, the following:*
   $\left(\{t_i\},M^1_{i+1}\right), \left(\{\overline{t_i}\},M^0_{i+1}\right), \widetilde{\psi_i} \vdash \left(\{t_{i-1}\},M^1_i\right), \left(\{\overline{t_{i-1}}\},M^0_i\right)$.
2. $\left(\{t_1\},M^1_2\right), \left(\{\overline{t_1}\},M^0_2\right), \widetilde{\psi_1} \vdash (\square, M^1_1)$.

Proof. For $i \geq 2$, the derivation is as follows:

$$\dfrac{\dfrac{\left(\{t_{i-1},x_i,\overline{t_i}\},M^1_{i+1}\right) \qquad \left(\{t_i\},M^1_{i+1}\right)}{\left(\{t_{i-1},x_i\},M^1_{i+1}\right)} \qquad \dfrac{\left(\{t_{i-1},\overline{x_i},t_i\},M^0_{i+1}\right) \qquad \left(\{\overline{t_i}\},M^0_{i+1}\right)}{\left(\{t_{i-1}\overline{x_i}\},M^0_{i+1}\right)}}{\left(\{t_{i-1}\},M^1_i\right)}$$

$$\dfrac{\dfrac{\left(\{\overline{t_{i-1}},\overline{x_i},\overline{t_i}\},M^1_{i+1}\right) \qquad \left(\{t_i\},M^1_{i+1}\right)}{\left(\{\overline{t_{i-1}},\overline{x_i}\},M^1_{i+1}\right)} \qquad \dfrac{\left(\{\overline{t_{i-1}},x_i,t_i\},M^0_{i+1}\right) \qquad \left(\{\overline{t_i}\},M^0_{i+1}\right)}{\left(\{\overline{t_{i-1}},x_i\},M^0_{i+1}\right)}}{\left(\{\overline{t_{i-1}}\},M^0_i\right)}$$

The derivation at the last stage is as follows:

$$\dfrac{\dfrac{\left(\{x_1,\overline{t_1}\},M^1_2\right) \qquad \left(\{t_1\},M^1_2\right)}{\left(\{x_1\},M^1_2\right)} \qquad \dfrac{\left(\{\overline{x_1},t_1\},M^0_2\right) \qquad \left(\{\overline{t_1}\},M^0_2\right)}{\left(\{\overline{x_1}\},M^0_2\right)}}{\left(\square,M^1_1\right)}$$

$\square$

We can now conclude the following:

Lemma 4.2.4. *MParity has polynomial size M-Res refutations.*

Proof. We first use Lemma 4.2.2 to derive all the $\widetilde{\psi_i}$. Next, we start with $\left(\{t_n\}, M^1_{n+1}\right)$ and $\left(\{\overline{t_n}\}, M^0_{n+1}\right)$, the lines corresponding to the clauses in ψ_{n+1}. From these lines and $\widetilde{\psi_n}$, we derive $(\{t_{n-1}\}, M^1_n)$ and $\left(\{\overline{t_{n-1}}\}, M^0_n\right)$, using Lemma 4.2.3. We continue in this manner deriving $\left(\{t_i\}, M^1_{i+1}\right)$ and $\left(\{\overline{t_i}\}, M^0_{i+1}\right)$ for $i = n-2, n-3, \ldots, 1$. From $(\{t_1\}, M^1_2)$ and $\left(\{\overline{t_1}\}, M^0_2\right)$, we derive $(\square, M^1_1)$ using $\widetilde{\psi_1}$ using Lemma 4.2.3. $\square$

Theorem 4.2.5. *LD-Q-Res does not p-simulate M-Res. Moreover, LQU-Res and LQU$^+$-Res are incomparable with M-Res.*

Proof. We showed in Lemma 4.2.4 that the MParity formulas have polynomial size M-Res refutations. We will now show that MParity requires exponential size LQU$^+$-Res refutations. We first note that QUParity requires exponential size LQU$^+$-Res refutations [21]. We further note that LQU$^+$-Res is closed under restrictions (Proposition 2 in [8]). Since restricting the MParity formulas by setting $a_{i,j} = 0$, for all $i, j \in [n]$, gives the QUParity formulas, we conclude that MParity requires exponential size LQU$^+$-Res refutations. Therefore LQU$^+$-Res does not simulate M-Res. Since LQU$^+$-Res p-simulates LD-Q-Res and LQU-Res, these two systems also do not simulate M-Res.

In Theorem 3.4.1, it is shown that M-Res does not simulate QU-Res. (The separating formula is in fact KBKF-lq.) Since LQU-Res and LQU$^+$-Res p-simulate QU-Res [8] and the simulation order is transitive, it follows that M-Res does not simulate LQU-Res and LQU$^+$-Res.

Hence LQU-Res and LQU$^+$-Res are incomparable with M-Res. $\square$

Remark 4.2.6. *In these proofs, note that the hardness for LQU^+-Res and IRM was proven using restrictions. But the same did not apply to M-Res — a restricted formula being hard for M-Res does not mean that the original formula is also hard. This means that M-Res is not closed under restrictions, and is hence unnatural.*

Remark 4.2.7. *Another observation is that the clauses of the KBKF-lq-weak formula family are weakenings of the clauses of KBKF-lq. Since KBKF-lq requires exponential-size M-Res refutations but KBKF-lq-weak has polynomial-size M-Res refutations, we conclude that weakening adds power to M-Res.*

The next chapter further explores weakenings and restrictions in M-Res.

Chapter 5

Role of weakenings, and unnaturalness

At the end of the last chapter, we saw how weakening can add power to M-Res. We also saw that M-Res is an unnatural proof system. In this chapter, we will study these two aspects in detail.

5.1 Weakenings

Let $(C, \{M^u \mid u \in U\})$ be a line. Then it can be weakened in two different ways [18]:

- Existential clause weakening: $C \vee x$ can be derived from C, provided it does not contain the literal $\overline{x}$. The merge-maps remain the same. Similarly, $C \vee \overline{x}$ can be derived if $x \notin C$.
- Strategy weakening: A trivial merge-map $(*)$ can be replaced by a constant merge-map (0 or 1). The existential clause remains the same.

Adding these weakenings to M-Res gives the following three proof systems:

- M-Res with existential clause weakening (M-ResW$_\exists$),
- M-Res with strategy weakening (M-ResW$_\forall$), and
- M-Res with both existential clause and strategy weakening (M-ResW$_{\exists\forall}$).

In the remainder of this section, we will study the relation among these systems.

First, we note that existential clause weakening adds exponential power.

Theorem 5.1.1. *M-ResW$_\exists$ is strictly stronger than M-Res.*

Proof. Since M-ResW$_\exists$ is a generalization of M-Res, M-ResW$_\exists$ p-simulates M-Res.

The KBKF-lq formulas can be transformed into the KBKF-lq-weak formulas in M-ResW$_\exists$ using a linear number of applications of the existential weakening rule. The transformed KBKF-lq-weak formulas have polynomial size M-Res (and hence M-ResW$_\exists$) refutations, Lemma 4.1.1. Thus the KBKF-lq formulas have polynomial size M-ResW$_\exists$ refutations. Since the KBKF-lq formulas require exponential size M-Res refutations (Theorem 3.4.1), we get the desired separation. □

Next we observe that a lower bound for M-Res in Theorem 3.4.1 can be lifted to M-ResW$_\forall$.

Lemma 5.1.2. *KBKF-lq requires exponential size refutations in M-ResW$_\forall$.*

Proof. We observe that the M-Res lower bound for KBKF-lq in Theorem 3.4.1 works with a minor modification. In Lemma 3.4.3, item 3 says that $M^{x_i} = *$. However a weaker condition $M^{x_i} \in \{*, 0, 1\}$ is sufficient for the lower bound. With this modification, we observe that the remaining argument carries over, and hence the lower bound also works for M-ResW$_\forall$. □

This tells us that strategy weakening is not as powerful as existential weakening.

Theorem 5.1.3. *M-ResW$_\forall$ does not simulate M-ResW$_\exists$; and M-ResW$_{\exists\forall}$ is strictly stronger than M-ResW$_\forall$.*

Proof. We showed that the KBKF-lq formulas require exponential size refutations in M-ResW$_\forall$ (Lemma 5.1.2) but have polynomial size refutations in M-ResW$_\exists$ and M-ResW$_{\exists\forall}$ (proof of Theorem 5.1.1). Therefore M-ResW$_\forall$ does not simulate M-ResW$_\exists$ and M-ResW$_{\exists\forall}$. Since M-ResW$_{\exists\forall}$ p-simulates M-ResW$_\forall$, M-ResW$_{\exists\forall}$ is strictly stronger than M-ResW$_\forall$. □

The next logical question is whether strategy weakening adds power to M-Res. We do not know the answer. However, we can answer this for the regular versions of these systems.

Theorem 5.1.4. *Regular M-ResW$_\forall$ is strictly stronger than regular M-Res.*

To prove this theorem, we will use a variant of the Squared-Equality (Eq^2) formula family, called Squared-Equality-with-Holes (H-Eq$^2(n)$). Squared-Equality, defined in [18], is a two-dimensional version of the Equality formula family [17], and has short regular tree-like M-Res refutations. It was used to show that the systems Q-Res, QU-Res, reductionless LD-Q-Res, $\forall$Exp + Res, IR and CP + $\forall$Red do not p-simulate M-Res. We recall its definition below:

Definition 5.1.5. *Squared-Equality* (Eq$^2(n)$) is the following QBF family:

$$\underset{i\in[n]}{\exists} x_i, y_i, \underset{j\in[n]}{\forall} u_j, v_j, \underset{i,j\in[n]}{\exists} t_{i,j}. \left(\bigwedge_{i,j\in[n]} A_{i,j}\right) \wedge B$$

where

- $B = \vee_{i,j\in[n]}\overline{t_{i,j}}$,

- For $i, j \in [n]$, $A_{i,j}$ contains the following four clauses:

$$x_i \vee y_j \vee u_i \vee v_j \vee t_{i,j}, \qquad x_i \vee \overline{y_j} \vee u_i \vee \overline{v_j} \vee t_{i,j},$$
$$\overline{x_i} \vee y_j \vee \overline{u_i} \vee v_j \vee t_{i,j}, \qquad \overline{x_i} \vee \overline{y_j} \vee \overline{u_i} \vee \overline{v_j} \vee t_{i,j}$$

Inspired by the lower bound for Eq^2 for reductionless LD-Q-Res (Theorem 28 in [18]), we now define the variant, H-$\mathrm{Eq}^2(n)$, and show that it is hard for regular M-Res (using somewhat similar arguments) but becomes easy for regular M-ResW$_\forall$. The variant identifies regions in the $[n] \times [n]$ grid, and changes the clause sets $A_{i,j}$ depending on the region that (i, j) belongs to. We can use any partition of $[n] \times [n]$ into two regions R_0, R_1 such that each region has at least one position in each row and at least one position in each column; call such a partition a *covering partition*. One possible choice for R_0 and R_1 is the following:
$R_0 = ([1, n/2] \times [1, n/2]) \cup ([n/2 + 1, n] \times [n/2 + 1, n])$ and
$R_1 = ([1, n/2] \times [n/2 + 1, n]) \cup ([n/2 + 1, n] \times [1, n/2])$. We will call R_0 and R_1 two regions of the matrix.

Definition 5.1.6. Let R_0, R_1 be a covering partition of $[n] \times [n]$.

Squared-Equality-with-Holes (H-$\mathrm{Eq}^2(n)(R_0, R_1)$) is the following QBF family:

$$\underset{i\in[n]}{\exists} x_i, y_i, \underset{j\in[n]}{\forall} u_j, v_j, \underset{i,j\in[n]}{\exists} t_{i,j}. \left(\bigwedge_{i,j\in[n]} A_{i,j} \right) \wedge B$$

where

- $B = \vee_{i,j\in[n]} \overline{t_{i,j}}$,
- For $(i, j) \in R_0$, $A_{i,j}$ contains the following four clauses:

$$x_i \vee y_j \vee u_i \vee v_j \vee t_{i,j}, \qquad x_i \vee \overline{y_j} \vee u_i \vee t_{i,j},$$
$$\overline{x_i} \vee y_j \vee v_j \vee t_{i,j}, \qquad \overline{x_i} \vee \overline{y_j} \vee t_{i,j}$$

- For $(i,j) \in R_1$, $A_{i,j}$ contains the following four clauses:

$$x_i \vee y_j \vee t_{i,j}, \qquad x_i \vee \overline{y_j} \vee \overline{v_j} \vee t_{i,j},$$

$$\overline{x_i} \vee y_j \vee \overline{u_i} \vee t_{i,j}, \qquad \overline{x_i} \vee \overline{y_j} \vee \overline{u_i} \vee \overline{v_j} \vee t_{i,j}$$

(We do not always specify the regions explicitly but merely say H-Eq^2.)

Lemma 5.1.7. *H-Eq$^2(n)$ requires exponential size refutations in regular M-Res.*

Before proving this, we show how to obtain Theorem 5.1.4.

Proof of Theorem 5.1.4. Since regular M-ResW$_\forall$ is a generalization of regular M-Res, it p-simulates regular M-Res.

Using strategy weakening, we can get Eq^2 from H-Eq^2 in linear number of steps. Since Eq^2 has polynomial-size refutations in regular M-Res, we get polynomial-size refutations for H-Eq^2 in regular M-ResW$_\forall$. On the other hand, Lemma 5.1.7 gives an exponential lower bound for H-Eq^2 in regular M-Res. Therefore regular M-ResW$_\forall$ is strictly stronger than regular M-Res. □

It remains to prove Lemma 5.1.7. This is a fairly involved proof, but in broad outline and in many details it is similar to the lower bound for Eq^2 in reductionless LD-Q-Res ([18]).

The size bound is trivially true for $n = 1$, so we assume that $n > 1$. Let Π be a Regular M-Res refutation of H-Eq$^2(n)$. Since a tautological clause cannot occur in a regular M-Res refutation, we assume that Π does not have a line whose clause part is tautological.

Let us first fix some notation. Let $X = \{x_1, \ldots, x_n\}$, $Y = \{y_1, \ldots, y_n\}$, $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$, and $T = \{t_{i,j} \mid i, j \in [n]\}$. For lines L_1, L_2, etc., the respective clauses and merge-maps will be denoted by C_1, C_2 and M_1, M_2 etc.

For a line L in Π, Π_L denotes the sub-derivation of Π ending in L. Viewing Π as a directed acyclic graph, we can talk of leaves and paths in Π. For a line L of Π, let $\textsc{Uci}(L) = \{(i,j) \mid A_{i,j} \cap \text{leaves}(\Pi_L) \neq \emptyset\}$.

We first show some structural properties about Π. The first property excludes using many axioms in certain derivations.

Lemma 5.1.8. *For line $L = (C, M)$ of Π, and $i, j \in [n]$, if $t_{i,j} \in C$, then $\textsc{Uci}(L) = \{(i,j)\}$.*

Proof. Since the literal $t_{i,j}$ only occurs in clauses in $A_{i,j}$, so $\text{leaves}(L) \cap A_{i,j} \neq \emptyset$, hence $\textsc{Uci}(L) \supseteq \{(i,j)\}$.

Now suppose $|\textsc{Uci}(L)| > 1$. Let (i', j') be an arbitrary element of $\textsc{Uci}(L)$ distinct from (i,j). Pick a leaf of Π_L using a clause in $A_{i',j'}$, and let ρ be a path from this leaf to L and then to the final line of Π. Both $t_{i,j}$ and $t_{i',j'}$ are necessarily used as pivots on this path. Assume that $t_{i,j}$ is used as a pivot later (closer to the final line) than $t_{i',j'}$; the other case is symmetric. Let $L_c = \text{Res}(L_a, L_b, t_{i',j'})$ and $L_f = \text{Res}(L_d, L_e, t_{i,j})$ respectively be the positions where $t_{i',j'}$ and $t_{i,j}$ are used as resolution pivots on this path (here L_a and L_d are the lines of path p, hence $t_{i',j'} \in C_a$ and $t_{i,j} \in C_d$). Then C_b has the negated literal $\overline{t_{i',j'}}$; hence $B \in \text{leaves}(L_b)$. Since $\overline{t_{i,j}} \in B$ but $\overline{t_{i,j}} \notin L_d$, $t_{i,j}$ is used as a resolution pivot in the derivation Π_{L_d}. This contradicts the fact that Π is regular. $\square$

The next property is the heart of the proof, and shows that paths with B at the leaf must have a suitable wide clause.

Lemma 5.1.9. *On every path from $\left(\vee_{i,j\in[n]}\overline{t_{i,j}}, \{*, \cdots, *\}\right)$ (the line for axiom clause B) to the final line, there exists a line $L = (C, M)$ such that either $X \subseteq var(C)$ or $Y \subseteq var(C)$.*

Proof. With each line $L_l = (C_l, M_l)$ in Π, we associate an $n \times n$ matrix N_l in which

$N_l[i,j] = 1$ if $\overline{t_{i,j}} \in C_l$ and $N_l[i,j] = 0$ otherwise.

Let $p = L_1, \ldots, L_k$ be a path from $\left(\vee_{i,j\in[n]}\overline{t_{i,j}}, \{*, \cdots, *\}\right)$ to the final line in Π. Since Π is regular, each $\overline{t_{i,j}}$ is resolved away exactly once, so no clause on p has any positive $t_{i,j}$ literal. Let l be the least integer such that N_l has a 0 in each row or a 0 in each column. Note that $l \geq 2$ since N_1 has no zeros. Consider the case that N_l has a 0 in each row; the argument for the other case is identical. We will show in this case that $X \subseteq \mathrm{var}(C_l)$. We will use the following claim:

Claim 5.1.10. In each row of N_l, there is a 0 and a 1 such that the 0 and 1 are in different regions (i.e. one is in R_0 and the other in R_1).

We proceed assuming the claim. We want to prove that $X \subseteq var(C_l)$. Suppose, to the contrary, there exists $i \in [n]$ such that $x_i \notin \mathrm{var}(C_l)$. We know that there exist $j_1, j_2 \in [n]$ such that $N_l[i,j_1] = 0$ and $N_l[i,j_2] = 1$; and either $(i,j_1) \in R_0$ and $(i,j_2) \in R_1$, or $(i,j_1) \in R_1$ and $(i,j_2) \in R_0$. Without loss of generality, we may assume that $(i,j_1) \in R_0$ and $(i,j_2) \in R_1$.

We know that on path p, there is a resolution with pivot t_{i,j_1} before L_l and a resolution with pivot t_{i,j_2} after L_l. Let the former resolution be $L_c = \mathrm{Res}(L_a, L_b, t_{i,j_1})$ where L_b is on path p, and let the latter resolution be $L_f = \mathrm{Res}(L_d, L_e, t_{i,j_2})$ where L_e is on path p. Since Π is a regular refutation, $t_{i,j_1} \in C_a$, $\overline{t_{i,j_1}} \in C_b$ and $t_{i,j_2} \in C_d$, $\overline{t_{i,j_2}} \in C_e$. Thus along path p these lines appear in the relative order $B, L_b, L_c, L_l, L_e, L_f, \Box$.

Claim 5.1.11. $\overline{x_i} \in C_c$.

Proof. By Lemma 5.1.8, $\textsc{Uci}(L_d) = \{(i,j_2)\}$, or equivalently $\mathrm{leaves}(L_d) \subseteq A_{i,j_2}$. Since $(i,j_2) \in R_1$, no clause in A_{i,j_2} has literal u_i. Hence $M_d^{u_i} \in \{*, 1\}$. Furthermore, if $M_d^{u_i} = *$, then $x_i \in C_d$. Since the pivot for resolving L_d and L_e is t_{i,j_2}, this would imply that $x_i \in C_f$.

By a similar argument, we can conclude that (i) leaves$(L_a) \subseteq A_{i,j_1}$, (ii) $M_a^{u_i} \in \{*, 0\}$, and (iii) if $M_a^{u_i} = *$, then $\overline{x_i} \in C_c$.

If $M_d^{u_i} = *$ and $M_a^{u_i} = *$, then $x_i \in C_f$ and $\overline{x_i} \in C_c$. So x_i must be used twice as pivot, contradicting regularity.

If $M_d^{u_i} = *$ and $M_a^{u_i} = 0$, then $x_i \in C_f$ and Π_{L_a} uses some clause containing x_i to make the merge-map for u_i non-trivial. Thus $x_i \in \Pi_{L_a}$, $x_i \notin L_l$ by assumption, $x_i \in L_f$. Hence x_i is used twice as pivot, contradicting regularity.

Hence $M_d^{u_i} = 1$. Since the resolution at line L_f is not blocked, $M_e^{u_i} \in \{*, 1\}$. But L_e is derived after, and using, L_a. Since merge-maps don't get simpler along a path, $M_a^{u_i} \in \{*, 1\}$. It follows that $M_a^{u_i} = *$. Hence $\overline{x_i} \in C_c$. □

Since $\overline{x_i} \notin C_l$, x_i has been used as a resolution pivot between L_c and L_l on path p. Let $L_w = \text{Res}(L_u, L_v, x_i)$ be the position on path p where x_i is used as pivot (since the refutation is regular, such a position is unique). Let L_v be the line on path p. By regularity of the refutation, $x_i \in L_u$ and $\overline{x_i} \in L_v$.

As observed at the outset, L_w is on path p and so does not contain a positive t literal. Since C_w is obtained via pivot x_i, this implies that C_u also does not contain a positive t literal. Since all axioms contain at least one t variable but only B contains negated t literals, so $B \in$ leaves(L_u).

Let q be a path thats starts from a leaf using B, passes through L_u to L_w, and then continues along path p to the final clause. Since the refutation is regular, $N_v = N_u = N_w$. Hence $N_v[i, j_1] = 0$ i.e. $\overline{t_{i,j_1}} \notin C_v$. This implies that t_{i,j_1} is used as resolution pivot before L_v on path q.

We already know that t_{i,j_2} is used as a pivot after line L_l on path p, and hence on path q. Arguing analogous to Claim 5.1.11 for path p but with respect to path q, we observe that $\overline{x_i}$ belongs to at least one leaf of L_u. Since $x_i \in C_u$ and since the refutation is regular, x_i is not used as a resolution pivot before C_u on path q. This implies that $\overline{x_u} \in C_u$. We already know that $x_i \in C_u$, since it contributed the pivot at L_w. This means that C_u is a tautological clause, a contradiction.

It remains to prove Claim 5.1.10.

Proof of Claim 5.1.10. We already know that N_l has a 0 in each row. We will first prove that N_l also has a 1 in each row. Aiming for contradiction, suppose that N_l has a full 0 row r. Since $l \geq 2$, N_{l-1} exists. Note that, by definition of resolution, there can be at most one element that changes from 1 in N_{l-1} to 0 in N_l. Since N_{l-1} does not have a 0 in every column, it does not contain a full 0 row. Hence, the unique element that changed from 1 in N_{l-1} to 0 in N_l must be in row r. Thus all other rows of N_{l-1} already contain the one 0 of that row in N_l. Since $n \geq 2$, N_{l-1} also has at least one 0 in row r; thus N_{l-1} has a 0 in each row, contradicting the minimality of l.

Since R_0 and R_1 form a covering partition, it cannot be the case that all the 0s and 1s of any row are in the same region R_b; that would imply that R_{1-b} does not cover the row. □

With the claim proven, the proof of Lemma 5.1.9 is now complete. □

We can finally prove Lemma 5.1.7. This part is identical to the corresponding part of the proof of Theorem 28 in [18]; we include it here for completeness.

Proof of Lemma 5.1.7. For each $a = (a_1, \ldots, a_n) \in \{0,1\}^n$, consider the assignment σ_a to the existential variables which sets $x_i = y_i = a_i$ for all $i \in [n]$, and $t_{i,j} = 1$ for all $i, j \in [n]$. Call such an assignment a symmetric assignment. Given a symmetric assignment σ_a, walk from the final line of Π towards the leaves maintaining the following invariant: for each line $L = (C, \{M^u \mid u \in U \cup V\})$, σ_a falsifies C. Let p_a be the path followed. By Lemma 5.1.9, this path will contain a line

$L = (C, \{M^u \mid u \in U \cup V\})$ such that either $X \subseteq var(C)$ or $Y \subseteq var(C)$. Let us define a function f from symmetric assignments to the lines of Π as follows: $f(a) = (C, \{M^u \mid u \in U \cup V\})$ is the last line (i.e. nearest to the leaves) on p_a such that either $X \subseteq var(C)$ or $Y \subseteq var(C)$. Note that, for any line L of Π, there can be at most one symmetric assignment a such that $f(a) = L$. This means that there are at least 2^n lines in Π. This gives the desired lower bound. $\square$

5.2 Simulation by eFrege + $\forall$red

It was recently shown that eFrege + $\forall$red p-simulates all known resolution-based QBF proof systems; in particular, it p-simulates M-Res [30]. We observe that this p-simulation can be extended in a straightforward manner to handle both the weakenings in M-Res. Hence we obtain a p-simulation of M-ResW$_\exists$, M-ResW$_\forall$ and M-ResW$_{\exists\forall}$ by eFrege + $\forall$red.

Theorem 5.2.1. *eFrege + $\forall$red strictly p-simulates the proof systems M-ResW$_\exists$, M-ResW$_\forall$ and M-ResW$_{\exists\forall}$.*

Proof. The separation follows from the separation of the propositional proof systems resolution and eFrege [72]. We prove the p-simulation below.

It suffices to prove that eFrege + $\forall$red p-simulates M-ResW$_{\exists\forall}$. The proof is essentially same as that of the p-simulation of M-Res in [30], but with two additional cases for the two weakenings. So, we will briefly describe that proof and then describe the required modifications.

Let Π be an M-ResW$_{\exists\forall}$ refutation Π of a QBF Φ. The last line of this refutation gives a winning strategy for the universal player; let us call this strategy S. We will first prove that there is a short eFrege derivation $\Phi \vdash \neg S$. Then, as mentioned in [30], the technique of [20, 29] can be used to derive the empty clause from $\neg S$ using universal reduction.

We will now describe an eFrege derivation $\Phi \vdash \neg S$. Let $L_i = (C_i, \{M_i^u \mid u \in U\})$ be the i$^{\text{th}}$ line of Π. We create new extension variables: $s_{i,j}^u$ is the variable for the j$^{\text{th}}$ node of M_i^u. If node j is a leaf of M_i^u labeled by constant c, then $s_{i,j}^u$ is defined to be c. Otherwise, if $M_i^u(j) = (x, a, b)$, then $s_{i,j}^u$ is defined as $s_{i,j}^u \triangleq \left(x \wedge s_{i,a}^u\right) \vee \left(\overline{x} \wedge s_{i,b}^u\right)$. The extension variables for u will be to its left in the quantifier prefix.

We will prove that for each line L_i of Π, we can derive the formula $F_i \triangleq \wedge_{u \in U_i}(u \leftrightarrow s_{i,r(u,i)}^u) \rightarrow C_i$; where $r(u, i)$ is the index of the root of merge-map M_i^u, and U_i is the set of universal variables for which M_i^u is non-trivial.

Our proof will proceed by induction on the lines of the refutation.

The base case is when L_i is an axiom; and the inductive step will have three cases depending on which rule is used to derive L_i: (i) resolution, (ii) existential clause weakening, or (iii) strategy weakening. The proof for the base case and the resolution step case is as given in [30]. We give proofs for the other two cases below:

- Existential clause weakening: Let line $L_b = (C_b, \{M_b^u \mid u \in U\})$ be derived from line $L_a = (C_a, \{M_a^u \mid u \in U\})$ using existential clause weakening. Then $C_b = C_a \vee x$ for some existential literal x such that $\overline{x} \notin C_a$, and $M_b^u = M_a^u$ for all $u \in U$. By the induction hypothesis, we have derived the formula $F_a \triangleq \wedge_{u \in U_a}(u \leftrightarrow s_{a,r(u,a)}^u) \rightarrow C_a$. We have to derive the formula $F_b \triangleq \wedge_{u \in U_b}(u \leftrightarrow s_{b,r(u,b)}^u) \rightarrow C_b = \wedge_{u \in U_b}(u \leftrightarrow s_{b,r(u,b)}^u) \rightarrow C_a \vee x$. Since $M_b^u = M_a^u$ for each u, there is a short eFrege + $\forall$red derivation of the formula $s_{a,j}^u \leftrightarrow s_{b,j}^u$ for each $u \in U_i$, and each node j of M_a^u. This allows us to replace variable $s_{a,j}^u$ by $s_{b,j}^u$ in F_a. As a result, we get the formula $F_b' \triangleq \wedge_{u \in U_b}(u \leftrightarrow s_{b,r(u,b)}^u) \rightarrow C_a$. Now, using an inference of the form $p \rightarrow q \vdash p \rightarrow q \vee r$, we obtain the formula F_b.

- Strategy weakening: Let line $L_b = (C_b, \{M_b^u \mid u \in U\})$ be derived from line

$L_a = (C_a, \{M_a^u \mid u \in U\})$ using strategy weakening for a variable v. Then $C_b = C_a$, $M_b^u = M_a^u$ for all $u \in U \setminus \{v\}$, and $M_a^v = *$, M_b^v is a constant, say d. Similar to the above case, we start with the inductively obtained F_a and replace each $s_{a,j}^u$ with $s_{b,j}^u$ to obtain a formula $F_b' \triangleq \wedge_{u \in U_b \setminus \{v\}}(u \leftrightarrow s_{b,r(u,b)}^u) \rightarrow C_b$. With a final inference of the form $p \rightarrow q \vdash p \wedge r \rightarrow q$, we can then add $(v \leftrightarrow s_{b,r(v,b)}^v)$ to the conjunction to obtain F_b. □

5.3 Unnaturalness

In this section, we observe that M-Res and M-ResW$_\forall$ are unnatural proof systems, i.e. they are not closed under restrictions.

Theorem 5.3.1. *M-Res and M-ResW$_\forall$ are unnatural proof systems.*

Proof. The KBKF-lq-split formula family has polynomial-size refutations in M-Res (and M-ResW$_\forall$), as seen in Lemma 4.1.2. The restriction of this family obtained by setting $t = 0$ is exactly the KBKF-lq formula family, which, as shown in Lemma 5.1.2, is exponentially hard for M-ResW$_\forall$ and hence also for M-Res. □

We believe that the unnaturalness of M-Res would have significant consequence on its practicality. Most SAT solvers work by setting some variables and simplifying the formula. If a simplified formula is harder to refute than the original formula, it would make the job of such solvers harder. So, a solver based on an unnatural proof system like M-Res would not perform very-well in practice.

Part II

The MaxSAT Resolution proof system

Chapter 6

The MaxRes proof system

6.1 Defining the proof system

The MaxSAT resolution (MaxRes) proof system operates on multi-sets of clauses, and uses the multi-output MaxSAT resolution (MaxRes) rule [28], defined as follows:

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

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 $\text{viol}_{F'}$ 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 $\Box$. 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 [28, 53], 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 $\Box$ is derived.

6.2 Comparison of MaxSAT resolution and Tree-like resolution

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 chapter, 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.

6.2.1 Simulation

Lemma 6.2.1. *For every unsatisfiable CNF F, $size_{MaxResW}(F) \leq 2 \cdot size_{TreeRes}(F)$.*

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 [73]; 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 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 6.1. Following the

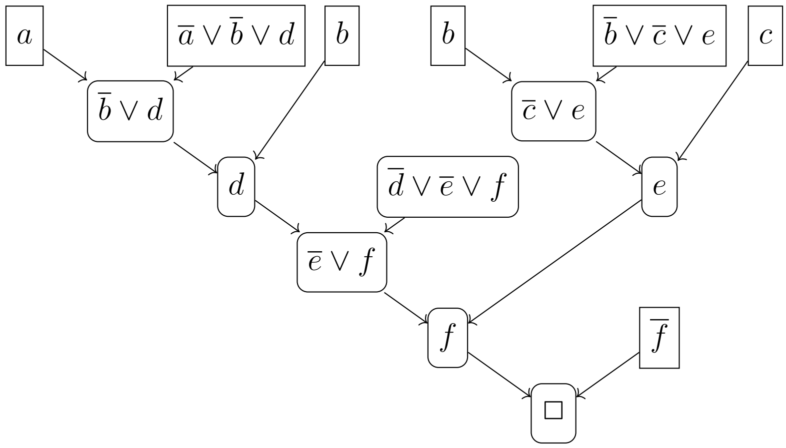


Figure 6.1: A tree-like resolution proof

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.

6.2.2 Separation

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 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 PebHint(G) composed with the OR function.[1] 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.

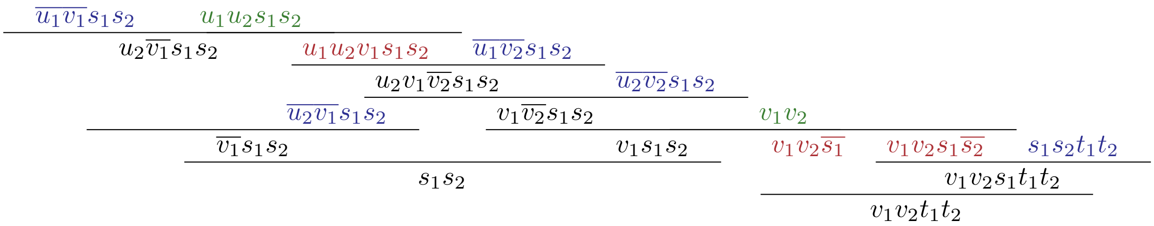
Lemma 6.2.2. *The* $\mathrm{PebHint}(P_h) \circ \mathrm{OR}$ *formulas have* $\Theta(n)$ *size MaxRes refutations.*

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

[1]The hints are added to make the Pebbling formulas easier to refute in MaxRes. We believe that, without the hints, these formulas require exponential size MaxRes refutations.

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 [12], but with some extra care to respect the hints. As in [12] 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 6.2.3 ([32]). $\mathrm{bpeb}(P_h) = h + 1$.

Lemma 6.2.4 ([12]). *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 [12] 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 [57], that is the minimum over all decision trees computing f of the maximum of 1-answers that the decision tree receives.[2]

Lemma 6.2.5. *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 6.2.4, $\mathrm{bpeb}(R_i \to w_i) \leq \max(p_0, p_1 + 1)$. 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.

[2]Essentially the same notion of one-sided query complexity is used in [62] under the name *positive depth*.

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 [65] 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 6.2.6 ([65]). *If Delayer can win p points, then all TreeRes proofs require size at least 2^p.*

Lemma 6.2.7. $F \circ \mathrm{OR}$ *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 6.2.6. □

The formulas $\mathrm{PebHint}(P_n) \circ \mathrm{OR}$ are easy to refute in MaxRes (Lemma 6.2.2), but from Lemmas 6.2.3, 6.2.5 and 6.2.7, they are exponentially hard for TreeRes. Hence,

Theorem 6.2.8. *TreeRes does not simulate MaxResW and MaxRes.*

Note that $\mathrm{DT}_1(f) \leq \mathrm{DT}(f)$ for any relation f, therefore Lemma 6.2.5 also holds for the standard measure of query complexity. The reason behind using one-sided query complexity is Lemma 6.2.7, 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 6.2.7 if we composed our formula with $\oplus$ instead of OR, but that would make the upper bound in Lemma 6.2.2 more intricate.

Chapter 7

The SubCubeSums proof system

7.1 Defining the proof system

With each application of the MaxRes rule, the number of clauses falsified by every assignment remains the same. This stronger invariant (compared to resolution) can be used to prove lower bounds for MaxRes and separate it from resolution. This is the motivation behind the new proof system called SubCubeSums.

We define the system first combinatorially, then through an algebraic framework, and define various measures in both settings. Then we show how it relates to MaxRes.

We then explore the power and limitations of the SubCubeSums proof system. On the one hand we show (Theorem 7.3.1) 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 7.4.2). 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 7.5.1).

SubCubeSums: Combinatorial view

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 $\text{viol}_F(\alpha) = 1 + \text{viol}_G(\alpha)$ 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 [33], 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

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

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,

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

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 if and only if $Q(x)$ vanishes everywhere; i.e. $Q(x) = 0$ is a polynomial identity. (See for instance [50, 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 [2].)

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 [42], 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 unnegated 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 [4, 6, 38, 40, 56]. For our purpose, 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 $|F|$,

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

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

7.2 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 1.2. (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 7.2.1. *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 $\mathrm{O}(m + ns)$.

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 $\{\Box\} \cup G$ 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 7.3 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 7.2.2. *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 7.2.3. 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 $\big(1 + 3 \cdot (2^d - 1)\big) \cdot (|F| + |p_0| + 1)$. $\square$

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 $\sum_{j\in[n]} \#mon(r_j(1 - x_j - \overline{x_j})) \leq 3s + 3 \cdot (2^{d-1} - 1) \cdot 2s = 3 \cdot (2^d - 1) \cdot s$.

$\square$

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 7.2.4. *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 $c_\beta = (f - g)(\beta) \in \mathbb{R}^{\geq 0}$. $\square$

7.3 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 [54], but we provide a new combinatorial proof.

7.3.1 The Subset Cardinality formulas

The first separation is achieved using subset cardinality formulas [60, 69, 75]. 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 7.3.1. *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 [60, Theorem 6]. The upper bound is the following lemma.

Lemma 7.3.2. *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

$$\sum_{f\in F_u} f - \left(\lceil d(u)/2\rceil - \sum_{e\ni u} x_e\right) = \sum_j c_{u,j} h_j \tag{7.1}$$

and

$$\sum_{f\in F_v} f - \left(\lceil d(v)/2\rceil - \sum_{e\ni v} \overline{x_e}\right) = \sum_j c_{v,j} h_j \tag{7.2}$$

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 (7.1) and (7.2). The easiest way is to appeal to the implicational completeness of SubCubeSums, Proposition 7.2.4. We continue deriving equation (7.1), 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 $f(x) = \binom{d-k+s}{d-k+1} = \frac{(d-k+s) \cdot \cdots \cdot s}{(d-k+1) \cdot \cdots \cdot 1} \geq s$.

We proved that $f \geq g$, therefore by Proposition 7.2.4 we can write $f - g$ as a sum of subcubes of size at most $2^d = \mathrm{O}(1)$.

Equation (7.2) 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 7.2.2 implies that combinatorial and algebraic size are at most a constant factor apart, hence the proof also has algebraic size $\mathrm{O}(n)$. $\square$

In proving the upper bound in Lemma 7.3.2, we invoked implicational completeness from Proposition 7.2.4. However, in our case the numbers are small enough that we can show how to derive equation (7.1) explicitly, by solving the appropriate LP, and without relying on Proposition 7.2.4. As a curiosity, and in preparation for the combinatorial proof, we display them next. We have

$$\begin{aligned} \text{(7.3)} \qquad & \overline{x_{1,2,3}} + \overline{x_{1,2,4}} + \overline{x_{1,3,4}} + \overline{x_{2,3,4}} - (2 - x_1 - x_2 - x_3 - x_4) = \\ & 2x_1x_2x_3x_4 + x_1x_2x_3\overline{x_4} + x_1x_2\overline{x_3}x_4 + x_1\overline{x_2}x_3x_4 + \overline{x_1}x_2x_3x_4 + 2\overline{x_1}\overline{x_2}\overline{x_3}\overline{x_4} \end{aligned}$$

and

$$\overline{x_{1,2,3}} + \overline{x_{1,2,4}} + \overline{x_{1,2,5}} + \overline{x_{1,3,4}} + \overline{x_{1,3,5}} + \overline{x_{1,4,5}} + \overline{x_{2,3,4}} + \overline{x_{2,3,5}} + \overline{x_{2,4,5}} + \overline{x_{3,4,5}} - (3 - x_1 - x_2 - x_3 - x_4 - x_5) = 2x_1x_2x_3x_4x_5 + x_1x_2x_3x_4\overline{x_5} + x_1x_2x_3\overline{x_4}x_5 + x_1x_2\overline{x_3}x_4x_5 + x_1\overline{x_2}x_3x_4x_5 + \overline{x_1}x_2x_3x_4x_5 + 2\overline{x_1x_2x_3x_4}x_5 + 2\overline{x_1x_2x_3}x_4\overline{x_5} + 2\overline{x_1x_2}x_3\overline{x_4x_5} + 2\overline{x_1}x_2\overline{x_3x_4x_5} + 2x_1\overline{x_2x_3x_4x_5} + 7\overline{x_1x_2x_3x_4x_5} \tag{7.4}$$

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

$$\forall \alpha \in \{0,1\}^{|E(G)|} : \mathrm{viol}_F(\alpha) = 1 + \mathrm{viol}_H(\alpha). \tag{7.5}$$

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 7.1.

Towards proving Equation (7.5), we introduce clause multisets f'_w and h'_w, described in Table 7.2. (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

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 7.1: 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.

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 7.2: 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.

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.

The multisets f'_w and h'_w are related to the multisets f_w and h_w by Equation (7.6) below, which can be verified by inspection (see Equation (7.3) and Equation (7.4) for an example).

$$\forall \alpha \in \{0,1\}^{E(G)}; \forall w \in W : \mathrm{viol}_{f_w}(\alpha) + \mathrm{viol}_{f'_w}(\alpha) = \mathrm{viol}_{h_w}(\alpha) + \mathrm{viol}_{h'_w}(\alpha). \tag{7.6}$$

Hence

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

7.3.2 The Pigeonhole Principle formulas

Recall the definition of the Pigeonhole Principle (PHP) formulas:

Definition 7.3.3 (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 ([43]).

In [54] 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^δ:

Definition 7.3.4 (PHP^δ from Theorem 5 of [54])**.** The clause set PHP^δ 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

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

- 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

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

Theorem 7.3.5 (implicit in [54] Theorem 5)**.** $\mathrm{viol}_{PHP^\delta} = \mathrm{viol}_{PHP} - 1$.

In the proof of Theorem 5 in [54], 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 [54] 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 7.3.6. *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. $\Box$

In [54] 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 7.3.7. *For any $A \in \{0,1\}^{(m+1)\times m}$, with row sums $r_i = \sum_j A_{i,j}$ and column sums $c_j = \sum_i A_{i,j}$,*

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

We can improve Corollary 7.3.6 to a stronger claim about algebraic size.

Corollary 7.3.8. *PHP_m has a refutation in SubCubeSums with algebraic size polynomial in m.*

Proof. Viewing the SubCubeSums proof in Corollary 7.3.6 from the algebraic viewpoint, the degree of the proof is linear. However, the negative degree is 3. So we can still use Proposition 7.2.2 to conclude that there is a refutation with algebraic size $\mathrm{O}(m^4)$. □

7.4 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 [72], 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 7.3.1, this lower bound cannot be inferred from hardness for Res.

We will use these standard facts:

Fact 7.4.1. *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 7.4.2. *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 7.4.1, X_i is empty for even i. We focus on X_1, X_3, and X_5 for the lower bound.

Let $\mathcal{C}$ be a SubCubeSums refutation of F, that is, $\mathrm{viol}_{\mathcal{C}} = \mathrm{viol}_F - 1 = g$. Define a matrix M with rows indexed by assignments to variables and columns indexed by clauses/cubes of $\mathcal{C}$, 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 $X_3 \times \mathcal{C}$ has $2|X_3|$ 1s, and the submatrix $X_5 \times \mathcal{C}$ 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 $X_3 \times \mathcal{C}$ come from light clauses (Lemma 7.4.3 below), and that a light clause can contribute only an exponentially small fraction of the 1s in $X_3 \times \mathcal{C}$ (Lemma 7.4.4 below). It then follows that $\mathcal{C}$ must have exponentially many light clauses.

For a clause $C \in \mathcal{C}$, 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 7.4.3.

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

Lemma 7.4.4. *For a light clause $C \in \mathcal{C}$,*

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

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

Proof. (of Theorem 7.4.2, assuming Lemma 7.4.3, Lemma 7.4.4)

$$
\begin{aligned}
2|X_3| &= (\text{number of 1s in } X_3 \times \mathcal{C}) \\
&\leq 10 \times (\text{number of 1s in } X_3 \times \mathcal{C} \text{ contributed by light} \\
&\quad \text{clauses}) && \text{(by Lemma 7.4.3)} \\
&\leq 10 \times (\text{number of light clauses}) \\
&\quad \times (\text{max number of 1s contributed by a light clause}) \\
&\leq 10 \times |\mathcal{C}| \times \frac{3|X_3|}{2^{n(0.1c-1)}} && \text{(by Lemma 7.4.4)}
\end{aligned}
$$

Hence $|\mathcal{C}| \geq \dfrac{2^{n(0.1c-1)}}{15} = 2^{\Omega(n)}$.

□

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

Proposition 7.4.5. *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 7.4.1(2)). □

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

Proof. (of Lemma 7.4.3) Consider the following probability distribution μ on $\mathcal{C}$:

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

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

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

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

Claim 7.4.6.

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

Proof. (of claim)

$$\begin{aligned}
&\mathbb{E}_{C\sim\mu}[\text{rel-density}(C)] \\
&\quad = \sum_{C\in\mathcal{C}:\mu(C)\neq 0} \mu(C)\frac{|C \cap X_5|}{|C \cap X_3|} \\
&\quad = \sum_{C\in\mathcal{C}:\mu(C)\neq 0} \frac{|C \cap X_5|}{2|X_3|} && \text{(each row in } X_3 \times \mathcal{C} \text{ has exactly 2 1s)} \\
&\quad = \frac{1}{2|X_3|} \sum_{C\in\mathcal{C}:\mu(C)\neq 0} |C \cap X_5| \leq \frac{4|X_5|}{2|X_3|} && \text{(each row in } X_5 \times \mathcal{C} \text{ has exactly 4 1s)} \\
&\quad = \frac{2\binom{n}{5}}{\binom{n}{3}} && \text{(by Proposition 7.4.5)} \\
&\quad \leq \frac{n^2}{10}.
\end{aligned}$$

□

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

Now we need to show that light clauses cannot contribute many 1s, Lemma 7.4.4. 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 7.4.7 below. Then we will show that if C is light, then it is wide; Lemma 7.4.8. Putting these together will prove Lemma 7.4.4.

To state Lemma 7.4.7,Lemma 7.4.8 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 $\text{viol}_{\mathcal{C}} = \text{viol}_F - 1$, 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 7.4.5 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 7.4.7. *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

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

□

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

Lemma 7.4.8. *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

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

By the c-expansion property of G, $Q1 \ge cn_i$.

If $n_i > n/2$, it still cannot be too large because C is light. Recall

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

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 $n_i \le 5n/6$. Thus even when $n_i > n/2$, we can conclude that $n_i/5 \le n/6 \le n - n_i < n/2$. By expansion of $V \setminus V_i$, we have $Q2 \ge c(n - n_i) \ge cn_i/5$.

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

Hence $w(C) \ge cn/10$ as claimed. □

Now we have all that is needed to prove Lemma 7.4.4.

Proof. (of Lemma 7.4.4) 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.

$$
\begin{aligned}
|C \cap X_3| &= n_1 n_2 n_3 2^{m-w(C)-n+k} && \text{(by Lemma 7.4.7)} \\
&= \frac{n_1 n_2 n_3 2^{m-w(C)-n+k}}{\binom{n}{3} 2^{m-n+1}} \times |X_3| && \text{(by Proposition 7.4.5)} \\
&= \frac{n_1 n_2 n_3}{\binom{n}{3}} 2^{k-w(C)-1} \times |X_3| \\
&\leq 6 \times 2^{n-w(C)-1} \times |X_3| = 3 \cdot 2^{n-w(C)} \cdot |X_3| \\
&\leq 3 \cdot 2^{n-cn/10} \cdot |X_3| && \text{(by Lemma 7.4.8)} \\
&= \frac{3|X_3|}{2^{n(0.1c-1)}} && \text{as claimed.}
\end{aligned}
$$

This completes the proof of Theorem 7.4.2. □

Remark As mentioned earlier, 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 $\mathrm{viol}_{\mathcal{C}} = \mathrm{viol}_F - 1$, and thus we assume that the axiom polynomials from F are multiplied only by -1.

7.5 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 [11]), which also consists of applying a random restriction to a formula composed with parity.

Theorem 7.5.1. *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 7.5.2. $\mathrm{viol}_F(\alpha_1 \oplus \alpha_2) = \mathrm{viol}_{F\circ\oplus}(\alpha_1, \alpha_2)$.

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

Corollary 7.5.3. $\mathrm{viol}_{F\circ\oplus} - 1 = ((\mathrm{viol}_F) \circ \oplus) - 1 = (\mathrm{viol}_F - 1) \circ \oplus$.

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 7.5.4. *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 7.5.1.

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 7.5.3, there is an integral conical junta for $(\mathrm{viol}_F - 1) \circ \oplus$ of size s. By Lemma 7.5.4 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 [41], 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.

Chapter 8

Conclusion

We studied two proof systems: (i) Merge Resolution (M-Res) for QBFs, and (ii) MaxSAT resolution (MaxRes) for certifying unsatisfiability.

Merge Resolution

M-Res was introduced in [18] to overcome the weakness of LD-Q-Res. It was shown that M-Res has advantages over many proof systems, but the advantage over LD-Q-Res was not demonstrated. We have filled this gap — we have shown that M-Res has advantages over not only LD-Q-Res, but also over more powerful systems, LQU$^+$-Res and IRM. We have also looked at the role of weakening — that it adds power to M-Res.

We then proved some lower bounds for M-Res, highlighting its limitations. We then showed a more fundamental limitation of M-Res, that M-Res with and without strategy weakening is unnatural. We believe that this makes it useless in practice. For the system to still be useful in practice, one will have to prove that it can be made natural by adding existential weakening or both weakenings. This, in our opinion, is the most important open question about M-Res.

MaxSAT Resolution

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 thesis is whether our conjecture that pebbling contradictions composed with XOR separate Res and SubCubeSums holds. (Very recently, in [37], 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.

Bibliography

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

[2] Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009.

[3] Albert Atserias. On sufficient conditions for unsatisfiability of random formulas. *J. ACM*, 51(2):281–311, 2004.

[4] 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.

[5] 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.

[6] Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. *ACM Transactions on Computational Logic*, 17(3):19:1–19:30, May 2016. Preliminary version in *CCC '14*.

[7] Valeriy Balabanov and Jie-Hong R. Jiang. Unified QBF certification and its applications. *Form. Methods Syst. Des.*, 41(1):45–65, August 2012.

[8] Valeriy Balabanov, Magdalena Widl, and Jie-Hong R. Jiang. QBF resolution systems and their proof complexities. In Carsten Sinz and Uwe Egly, editors, *Theory and Applications of Satisfiability Testing - SAT 2014 - 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings*, volume 8561 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2014.

[9] Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theor. Comput. Sci.*, 22:317–330, 1983.

[10] Paul Beame, Henry A. Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res.*, 22:319–351, 2004.

[11] Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, May 2009. Preliminary version in *STOC '02*.

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

[13] 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*.

[14] 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.

[15] Olaf Beyersdorff and Joshua Blinkhorn. Formulas with large weight: a new technique for genuine QBF lower bounds. *Electron. Colloquium Comput. Complex.*, 24:32, 2017.

[16] Olaf Beyersdorff and Joshua Blinkhorn. Lower bound techniques for QBF expansion. *Theory Comput. Syst.*, 64(3):400–421, 2020.

[17] Olaf Beyersdorff, Joshua Blinkhorn, and Luke Hinde. Size, cost, and capacity: A semantic technique for hard random QBFs. *Log. Methods Comput. Sci.*, 15(1), 2019.

[18] Olaf Beyersdorff, Joshua Blinkhorn, and Meena Mahajan. Building strategies into QBF proofs. *J. Autom. Reason.*, 65(1):125–154, 2021. Preliminary version in the proceedings of the 36th International Symposium on Theoretical Aspects of Computer Science (STACS 2019).

[19] Olaf Beyersdorff, Ilario Bonacina, and Leroy Chew. Lower bounds: From circuits to QBF proof systems. In Madhu Sudan, editor, *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, Cambridge, MA, USA, January 14-16, 2016*, pages 249–260. ACM, 2016.

[20] Olaf Beyersdorff, Ilario Bonacina, Leroy Chew, and Ján Pich. Frege systems for quantified Boolean logic. *J. ACM*, 67(2):9:1–9:36, 2020.

[21] Olaf Beyersdorff, Leroy Chew, and Mikolás Janota. New resolution-based QBF calculi and their proof complexity. *ACM Trans. Comput. Theory*, 11(4):26:1–26:42, 2019.

[22] Olaf Beyersdorff, Leroy Chew, Meena Mahajan, and Anil Shukla. Understanding cutting planes for QBFs. *Inf. Comput.*, 262:141–161, 2018.

[23] Olaf Beyersdorff, Mikoláš Janota, Florian Lonsing, and Martina Seidl. Quantified Boolean formulas. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, pages 1177–1221. IOS Press, Netherlands, 2nd edition, May 2021.

[24] A. Blake. *Canonical expressions in Boolean algebra*. PhD thesis, University of Chicago, 1937.

[25] Beate Bollig. A very simple function that requires exponential size nondeterministic graph-driven read-once branching programs. *Inf. Process. Lett.*, 86(3):143–148, 2003.

[26] 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.

[27] 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.

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

[29] Leroy Chew. Hardness and optimality in QBF proof systems modulo NP. In Chu-Min Li and Felip Manyà, editors, *Theory and Applications of Satisfiability Testing - SAT 2021 - 24th International Conference, Barcelona, Spain, July 5-9, 2021, Proceedings*, volume 12831 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 2021.

[30] Leroy Chew and Friedrich Slivovsky. Towards uniform certification in QBF. In Petra Berenbrink and Benjamin Monmege, editors, *39th International Symposium on Theoretical Aspects of Computer Science, STACS 2022, March 15-18, 2022, Marseille, France (Virtual Conference)*, volume 219 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[31] Leroy Nicholas Chew. *QBF proof complexity.* PhD thesis, University of Leeds, 2017.

[32] 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*.

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

[34] Martin Davis, George Logemann, and Donald W. Loveland. A machine program for theorem-proving. *Commun. ACM*, 5(7):394–397, 1962.

[35] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.

[36] Uwe Egly, Florian Lonsing, and Magdalena Widl. Long-distance resolution: Proof generation and strategy extraction in search-based QBF solving. In Kenneth L. McMillan, Aart Middeldorp, and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - 19th International Conference, LPAR-19, Stellenbosch, South Africa, December 14-19, 2013. Proceedings*, volume 8312 of *Lecture Notes in Computer Science*, pages 291–308. Springer, 2013.

[37] Noah Fleming, Mika Göös, Stefan Grosser, and Robert Robere. On semi-algebraic proofs and algorithms. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 69:1–69:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[38] 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.

[39] Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Separations in proof complexity and TFNP. *CoRR*, abs/2205.02168, 2022.

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

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

[42] 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*.

[43] Amin Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[44] Johan Håstad. The shrinkage exponent of de Morgan formulas is 2. *SIAM J. Comput.*, 27(1):48–64, 1998.

[45] Johan Torkel Håstad. *Computational limitations for small-depth circuits.* MIT press, 1987.

[46] 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.

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

[48] Mikolás Janota. On Q-resolution and CDCL QBF solving. In Nadia Creignou and Daniel Le Berre, editors, *Theory and Applications of Satisfiability Testing - SAT 2016 - 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings*, volume 9710 of *Lecture Notes in Computer Science*, pages 402–418. Springer, 2016.

[49] Mikolás Janota and João Marques-Silva. Expansion-based QBF solving versus Q-resolution. *Theor. Comput. Sci.*, 577:25–42, 2015.

[50] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.

[51] Hans Kleine Büning, Marek Karpinski, and Andreas Flögel. Resolution for quantified Boolean formulas. *Inf. Comput.*, 117(1):12–18, 1995.

[52] Jan Krajíček. *Proof complexity*, volume 170. Cambridge University Press, 2019.

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

[54] 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.

[55] 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.

[56] Massimo Lauria and Jakob Nordström. Tight size-degree bounds for sums-of-squares proofs. *Computational Complexity*, 26(3):911–948, December 2017. Preliminary version in *CCC '15*.

[57] 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.

[58] 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.

[59] Joao Marques-Silva, Ines Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, pages 133–182. IOS Press, Netherlands, 2nd edition, May 2021.

[60] 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.

[61] 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.

[62] Theodoros Papamakarios and Alexander A. Razborov. Space characterizations of complexity measures and size-space trade-offs in propositional proof systems. *Electron. Colloquium Comput. Complex.*, page 74, 2021.

[63] Tomás Peitl, Friedrich Slivovsky, and Stefan Szeider. Proof complexity of fragments of long-distance Q-resolution. In Mikolás Janota and Inês Lynce, editors, *Theory and Applications of Satisfiability Testing - SAT 2019 - 22nd*

International Conference, SAT, Proceedings, volume 11628 of *Lecture Notes in Computer Science*, pages 319–335. Springer, 2019.

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

[65] 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.

[66] Alexander A Razborov. Lower bounds for the monotone complexity of some Boolean functions. In *Soviet Math. Dokl.*, volume 31, pages 354–357, 1985.

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

[68] Emma Rollon and Javier Larrosa. Proof Complexity for the Maximum Satisfiability Problem and its Use in SAT Refutations. *Journal of Logic and Computation*, March 2022.

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

[70] Larry J. Stockmeyer and Albert R. Meyer. Word problems requiring exponential time: Preliminary report. In Alfred V. Aho, Allan Borodin, Robert L. Constable, Robert W. Floyd, Michael A. Harrison, Richard M. Karp, and H. Raymond Strong, editors, *Proceedings of the 5th Annual ACM Symposium on Theory of Computing, April 30 - May 2, 1973, Austin, Texas, USA*, pages 1–9. ACM, 1973.

[71] G. S. Tseitin. On the complexity of derivation in propositional calculus. In Jörg H. Siekmann and Graham Wrightson, editors, *Automation of Reasoning:*

2: Classical Papers on Computational Logic 1967–1970, pages 466–483. Springer Berlin Heidelberg, Berlin, Heidelberg, 1983.

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

[73] Alasdair Urquhart. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 1(4):425–467, 1995.

[74] Allen Van Gelder. Contributions to the theory of practical quantified Boolean formula solving. In *Proc. Principles and Practice of Constraint Programming (CP'12)*, pages 647–663, 2012.

[75] 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.

[76] Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 1–10. IEEE Computer Society, 1985.

[77] Lintao Zhang and Sharad Malik. Conflict driven learning in a quantified Boolean satisfiability solver. In *IEEE/ACM International Conference on Computer-aided Design, ICCAD 2002*, pages 442–449, 2002.