The Proof Complexity of Integer Programming

by

Noah Fleming

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy

Graduate Department of Computer Science
University of Toronto

The Proof Complexity of Integer Programming

Noah Fleming
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
2021

Abstract

Proof complexity provides a promising approach aimed at resolving the P versus NP question by establishing the nonexistence of propositional proof systems which admit short proofs of every tautology. Proof complexity suggests an incremental program of proving lower bounds on the size of proofs in successively stronger proof systems as a means of building towards a resolution to the central question. The complexity of specific proof systems is also deeply connected to the analysis of practical algorithms. A proof system corresponds to a family of efficient algorithms. Thus, lower bounds for a proof system reveals the limitations inherent to the associated algorithms.

In this thesis we develop and study proof systems which formalize modern algorithms for integer programming. Our contributions are as follows.

A major conjecture in proof complexity is that random k-CNF formulas are hard to prove in *every* proof system. This conjecture has been confirmed for a number of proof systems [4, 40, 137, 139]. One notable exception is the Cutting Planes system, which formalizes classical algorithms for integer programming. In this thesis we confirm this conjecture for Cutting Planes.

Modern algorithms for integer programming, known as *branch-and-cut*, improve upon Cutting Planes by combining it with a branch-and-bound procedure. We introduce *Stabbing Planes* as an elegant proof system which formalizes branch-and-cut algorithms. In doing so, we show that Stabbing Planes can simulate Cutting Planes and surprisingly, that a partial converse exists as well. Using this converse we are able to obtain exponential lower bounds on branch-and-cut, as well as exhibit small Cutting Planes proofs of *any* systems of linear equations over a prime finite field, generalizing [48].

One striking feature of these Cutting Planes proofs of linear equations is that their *depth* far exceeds worst-case. This suggests that these formulas may yield a *supercritical* size/depth tradeoff for Cutting Planes — formulas for which short proofs require depth beyond worst-case. We make two contributions towards this. First, we develop a *geometric* technique for proving depth lower bounds and apply it to a "semantic" generalization of Cutting Planes. Second, generalizing [132], we establish supercritical size/depth tradeoffs for Cutting Planes.

Acknowledgements

It is my extreme good fortune to have had Toni Pitassi as an advisor. Toni is a truly exceptional researcher, with excellent intuition and seemingly boundless knowledge of mathematics, and it has been a real privilege to have worked with her for these past years. Toni, thank you for all of your guidance, your contagious enthusiasm, and your tireless work at advancing my career. I am "stoked" to have had you as an advisor; thanks for making these years so much fun! We still have to get our tattoo $\triangle$.

Next on the list is Robert Robere. I have so much to thank you for by now, Robert, that I don't know where to begin. You've been there from day one of my Ph.D. and I have learned a tremendous amount from you. Thank you for helping me get the hang of research, for helping me get settled in Toronto, for the most tactful retelling of "the Germany story", and for all of the great memories (the accidental break in, Denis' tight fits, our IAS Kubrick movie series, etc.). Most of all, thanks for being an excellent collaborator, mentor, and one of my closest friends.

I should also thank the architect behind all of this, Antonina Kolokolova. Thank you for sharing your enthusiasm for research with me, for inspiring me to pursue a Ph.D. in complexity and for putting me on the (renowned) Newfoundland-to-UofT pipeline. I am extremely grateful for all of your support, for being a fantastic collaborator, and I look forward to many more years of collaboration at Memorial.

A very special thank you goes to Yuichi Yoshida for advising me during my internship at the National Institute of Informatics, Toyko. Yuichi is a great person on top of being a fantastic collaborator and it was a pleasure getting to know him and explore fascinating area of property testing under his guidance.

I'd also like to thank my collaborators for some excellent research discussions over the years — (in no particular order) Denis Pankratov, Paul Beame, Russell Impagliazzo, Vijay Ganesh, Avi Wigderson, Mika Göös, Li-Yang Tan, Marc Vinyals, Pravesh Kothari, Chunxiao Li, Ian Mertz, James Cook, Morgan Shirley, Yuval Efron, Henry Yuen, and Stefan Grosser. It's been a privilege to have worked with such brilliant people. I would also like to thank Jakob Nordström for all of his help. Finally, a big thank you to my committee members Ben Rossman, Sasho Nikolov, and my external examiner Albert Atserias for excellent questions and ideas.

There is also more to life than research, and I'd like to thank everyone who has made these past years some of the best — the Wyatt Bros. (Wyatt and Keenan) for being my oldest and closest friends; the 709 Bustaz (and in particular Leo, JSN, JQ, Simon, and Jesse); the lab (Spooner, Robert, Lalla, Atiyeh, Lindsay (honourary), Colleen, Ian, Morgan, Lily, Deeksha, Greg, ...) for always being able to brighten up those windowless offices; Tyrone for his exceptional taste in office decor; Chris and Jeanie (and Mabel) for showing us all of the best spots in Clarenville. Finally, I would also like to thank my parents, grandparents, and sister for their constant love, support and encouragement, without which none of this would have been possible.

Above all, Emma, thank you for sticking with me through this adventure. You've given me more love and support than anyone deserves, and I couldn't have done this without you.

Contents

Chapter 1

Introduction

Given a tautological statement — a theorem — what is the shortest proof of its validity in some proof system? This is an intrinsic question of logic and its *propositional* version is deeply linked to fundamental questions in computational complexity.

Understanding the length of proofs in every propositional proof system is connected to resolving the P versus NP question, a question of profound importance to computer science and mathematics. As formalized by Cook and Reckhow [43], a propositional proof system is exactly an NP-verifier for the coNP-complete language of *propositional tautologies*. It follows that there does not exists a propositional proof system which admits short (polynomial-size) proofs of every tautology if and only if NP is different from its complement, coNP — and therefore $\mathsf{P} \neq \mathsf{NP}$. Thus, *proof complexity* provides an approach aimed at ultimately proving $\mathsf{P} \neq \mathsf{NP}$ by leveraging tools from both mathematical logic and computational complexity.

In their seminal work on this subject, Cook and Reckhow proposed an incremental approach — which has come to be known as *Cook's program* — of systematically proving lower bounds on the length of proofs in increasingly more powerful proof systems as a means of building towards a resolution to the central question. Since its inception, this program has produced celebrated lower bounds for many natural proof systems (see Figure 1.1). As well, deep connections with other areas of theoretical computer science have emerged, which have led to breakthroughs in areas such as circuit complexity, combinatorics, and the design and analysis of efficient algorithms.

In order to tackle the NP vs. coNP question, one needs to first formulate a candidate family of "hard" formulas; i.e., a family of propositional tautologies which is believed to be hard to prove in *every* proof system. While it is a folklore conjecture that almost every formula should be hard to prove in any proof system, it has been challenging to find *concrete* tautologies which do not have short proofs in strong proof systems such as Frege systems (see e.g., [27, 101] for excellent expositions on this topic). Because of this, the principal family of formulas which are conjectured to be hard for every proof system are *uniformly random CNF formulas*. This has given rise to a line of work that seeks to prove lower bounds on the complexity of proving random formulas in restricted proof systems.

While lower bounds on every proof system would imply $\mathsf{P} \neq \mathsf{NP}$, the length of proofs in particular proof systems is intimately tied to understanding large classes of practical algorithms for solving NP-hard problems. A proof system, in a specific formal sense, corresponds to a family of efficient, provably correct algorithms. Thus proving lower bounds on the length of proofs in this proof system rules out large classes of algorithms for solving NP-hard optimization problems. For example, the resolution proof system captures

the reasoning employed in practical SAT solvers, while lower bounds on the Cutting Planes proof system imply lower bounds on the runtime of an important class of integer programming algorithms. As well, broad classes of linear and semi-definite programs can be viewed as generic translations of Sherali-Adams and Sum-of-Squares proofs of the existence of a solution into algorithms for finding a solution. Proving lower bounds on the length of proofs in these systems has ruled out many of the most promising algorithms for NP-hard problems. Conversely, short proofs in certain proof systems such as Sum-of-Squares have given rise to state-of-the-art algorithms for a wide range of exact and approximate optimization problems.

In this thesis we develop and study proof systems which formalize modern algorithms for solving NP-hard optimization problems using the framework of *integer programming*. By extending deep connections with circuit and communication complexity, we establish strong lower bounds on the size of proofs of random CNF formulas in these proof systems. As well, we prove tradeoffs between the size and the depth of proofs which go far beyond the worst-case.

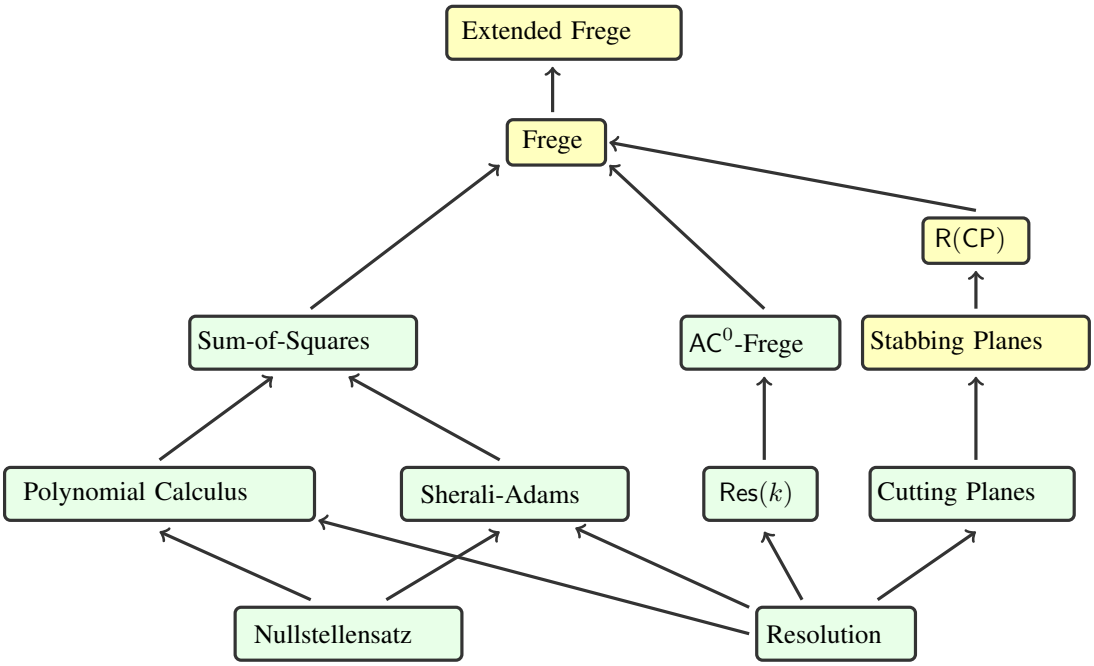
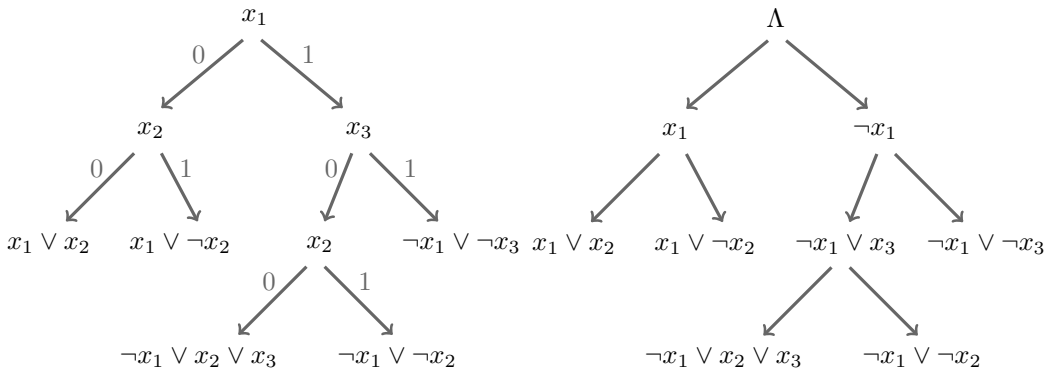


Figure 1.1: A selection of standard proof systems. There is an arrow from a proof system $\mathcal{P}_1$ to $\mathcal{P}_2$ if $\mathcal{P}_2$ p-simulates $\mathcal{P}_1$. Proof systems in green have established exponential lower bounds, while those in yellow do not.

1.1 Proof Complexity and Algorithm Design and Analysis.

Over the past several decades proof complexity has emerged as a systematic way to analyze and develop algorithms for exact and approximate problems. There are two high level themes underlying the connection between proofs and algorithms.

The first theme is that lower bounds on the length of proofs in certain proof systems imply lower bounds on the runtime of a broad class of related algorithms. For simplicity, we will describe this theme for exact algorithms solving the boolean satisfiability problem, SAT, however this framework can be readily extended to approximation algorithms and to other NP-hard problems. Since any polynomial-time algorithm that solves SAT must classify all *unsatisfiable* boolean formulas, it follows that the complexity of the SAT problem is



(a) DPLL (b) tree-like resolution

Figure 1.2: DPLL and tree-like resolution refutations of the CNF formula $(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_3)$.

intimately connected with the study of *refuting unsatisfiable formulas*, which by negation is equivalent the study of *proving tautogies*. More precisely, suppose that we have a correct algorithm A — meaning that it is guaranteed to output the correct answer — for solving SAT (or some other NP-hard problem). Since the algorithm is correct we can view the computation of the algorithm, given an unsatisfiable formula F as input, as a proof of the unsatisfiability of F. Furthermore, if A runs efficiently on F, then A provides us with a short proof that F is unsatisfiable. This proof belongs to some proof system, which we will denote by $\mathcal{P}_A$. It follows that super-polynomial lower bounds on $\mathcal{P}_A$ imply that A cannot be a polynomial time algorithm for SAT.

A simple example of this phenomenon is the *Davis-Putnam-Logemann-Loveland* (*DPLL*) algorithm [53, 54] which underlies modern algorithms for solving instances of SAT that occur in practice. DPLL is the standard backtracking search algorithm for deciding satisfiability of a CNF formula, F. It proceeds by choosing a variable x_i (according to some heuristic) and then recursing on F with x_i fixed to 1 and F with x_i fixed to 0. If ever a clause of F becomes falsified, we halt that recursive branch and label it with the falsified clause. Whenever F is *unsatisfiable*, the transcript of the algorithm procedures a decision tree over the underlying variables, where each leaf of the tree is labelled with some clause $C_i \in F$ such that the partial truth assignment of the variables queried along the path to that leaf falsifies C_i (see Figure 1.2a). Such a decision tree, in turn, is actually a *tree-like resolution* proof of C.

The *resolution* proof system is one of the most well-studied proof systems. It proves the validity of a tautology by refuting its negation. Because every formula can be efficiently encoded in conjunctive normal form (CNF), we may assume that the input is an unsatisfiable set of clauses $F = \{C_i\}_{i \in [m]}$. From these clauses, one can derive new clauses using the following rule.

- *Resolution Rule*. From clauses C_j containing a variable x and C_k containing $\neg x$, deduce the clause $(C_j \setminus x) \vee (C_k \setminus \neg x)$.

To refute a formula in resolution, one derives the empty clause Λ which is falsified by every truth assignment. A resolution proof is *tree-like* if every clause not in F can be used at most once before it must be re-derived (see Figure 1.2b). As shown in Figure 1.2, the transcript of the DPLL algorithm on an unsatisfiable formula F is exactly a tree-like resolution refutation of F, up to a relabelling of the nodes and edges. Therefore, running the DPLL algorithm on an unsatisfiable formula yields a tree-like Resolution refutation.

The theme of using proof complexity for algorithm analysis has been successfully applied to a number of other prominent classes of algorithms, including: modern *conflict driven clause learning* algorithms for solving SAT [92, 115, 142], linear and semi-definite programming algorithms such as the lift-and-project hierarchies of *Lovász-Schrijver* [107], *Sherali-Adams* [141], and *Lasserre* [105,122], and the classic Gomory-Chvátal *cutting planes* algorithms for integer programming [37,76].

The second theme of this connection is that upper bounds in certain proof systems can automatically give rise to efficient algorithms. A proof system is said to be *automatable* if there is an algorithm that can produce proofs in that system efficiently in the size of the shortest proof. An early example of this phenomenon was observed by Beame and Pitassi [17], building on [41], who showed that any resolution proof of size s could be found in time $2^{O(\sqrt{n \log s})}$. Another highly influential example is the *Sum-of-Squares* proof system in which efficient proofs give rise to a broad class semi-definite programs via its connection to the Lasserre hierarchy. The past decade has seen a flurry of work designing upper bounds in Sum-of-Squares as a means of creating efficient algorithms for many unsupervised learning and optimization problems. This theme will play only a minor role in this thesis, and we refer the interested reader to [67] for a survey.

1.2 The Proof Complexity of Integer Programming

This thesis studies proof systems which formalize algorithms for *integer programming*. Recall that an integer programming problem is one where we are given a polytope P, represented by system of integer linear inequalities $Ax \leq b$, and a vector $c \in \mathbb{R}^n$, and our goal is to find a point $x \in P \cap \mathbb{Z}^n$ such that x maximizes $c \cdot x$.[1] Integer programming represents one of the most prominent approaches for solving NP-hard optimization problems. In spite of the fact that integer programming is NP-complete, there has been extraordinary success in designing efficient algorithms which are able to produce good approximate or exact solutions to typical instances of integer programming that occur in practice [42].

The classical approach for solving an integer program — pioneered by Gomory [76] — is to introduce *cutting planes* to P. A *Chvátal-Gomory (CG) cutting plane* for P is any inequality of the form $ax \leq \lfloor b \rfloor$, where a is an integral vector, and b is rational, and *every* point of P is satisfies $ax \leq b$. See Figure 1.3 for an illustration. By the integrality of a, it follows that CG cutting planes *preserve* the integral points of P while potentially *removing* non-integral points. Cutting planes algorithms proceed by heuristically choosing "good" cutting planes to refine P until an exact or approximate solution can be found by a linear programming solver.

[1] We note that all of the ideas in this thesis can also be extended in a straightforward manner to *mixed* integer programming.

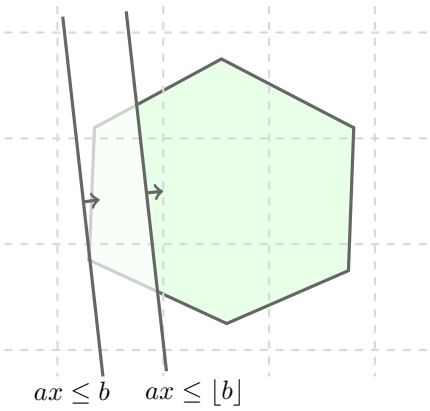


Figure 1.3: A CG cut $ax \leq \lfloor b \rfloor$, the shift of a valid halfspace $ax \leq b$ to the nearest integer point. The grid intersection points represent integer points.

These algorithms can naturally be formalized as the *Cutting Planes proof system*, originally proposed by Chvátal [37]. A Cutting Planes (CP) refutation of a polytope P is a sequence of polytopes $P_1, \ldots, P_s$ such that P_s is the empty polytope $\emptyset$, and each polytope P_i is derived by intersecting P_{i-1} with a CG cutting plane for P_{i-1}. Since CG cutting planes preserve integral points, such a refutation is only possible if P does not contain any integral points (i.e., $P \cap \mathbb{Z}^n = \emptyset$).

Cutting Planes has grown to be an influential proof system in proof complexity. The first exponential lower bounds on Cutting Planes refutations were established by Pudlák [127], building on the work of Bonet, Pitassi, and Raz [30] who established the same lower bounds for CP* — Cutting Planes proofs in which the coefficients of each cutting plane are bounded. Both of these lower bounds are instantiations of the method of *feasible interpolation*, a general technique formalized by Krajíček [98]. They proceed by first showing that short refutations of certain *split formulas* F can be transformed into small circuits computing an associated *monotone function* I_F:

- [30]: Any size s CP* refutations of F imply a size $\mathsf{poly}(s)$ *monotone circuit* — circuits operating with $\wedge$ and $\vee$ gates — computing I_F.
- [127]: Any size s CP refutation of F gives rise to a size $\mathsf{poly}(s)$ *monotone real circuit* — circuits in which each gate is a fanin-2 monotone function $g : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ — computing I_F.

The lower bounds on proof size follow by modifying the celebrated monotone circuit lower bounds due to Razborov [131] to apply to monotone real circuits.

Despite the success of feasible interpolation, it is limited to proving lower bounds on split formulas. Prior to this thesis, the only family of formulas which were known to be hard for Cutting Planes were the clique-coclique formulas [30, 127] and the broken mosquito screen formulas [83] and it was a long-standing open problem to establish lower bounds on the size of Cutting Planes proofs of formulas which are typically studied, in particular *random CNF formulas* [18] and the *Tseitin formulas* [45].

Beyond Cutting Planes. While Cutting Planes has become a influential proof system in proof complexity, the original cutting planes algorithms suffered from numerical instabilities, as well as difficulties in finding good heuristics for the next cutting planes to introduce [76]. Modern integer programming algorithms improve on the classical cutting planes by combining it with a *branch-and-bound procedure* [10, 104], giving

rise to an algorithmic paradigm known as *branch-and-cut* [121]. These algorithms solve an integer programming problem by repeating the following two procedures: (i) *branching*: the current polytope P is split into sub-polytopes $P_1, \ldots, P_k$ containing the integer points of P, (ii) *cutting*: additional cutting planes are added to further refine the polytopes. In practice, branching typically done by choosing a variable x_i and branching on all possible integer values that it can take. More general branching has also been considered, such as branching on the hamming weight of a subset of coordinates [65], or more general linear inequalities [1,2,94,102,112,120]. However, the additional strength provided by the branching procedure means that these algorithms are no longer captured by the Cutting Planes proof system, and the extensive work that has been done on Cutting Planes does not apply to modern integer programming algorithms.

1.3 Contributions

Chapter 3: Random $\Theta(\log n)$-CNFs are Hard for Cutting Planes

Random SAT formulas represent the principal family of formulas that is conjectured to be hard to refute in *every* proof system. The most well-studied model is the *random k-SAT model* $\mathcal{F}(m, n, k)$, where a k-CNF formula on m clauses and n variables is chosen uniformly at random. *Feige's Hypothesis* [63] links the complexity of refuting random k-CNF formulas to worst-case inapproximability, and average-case hardness of circuit lower bounds [136].

Beginning with the seminal work of Chvátal and Szemerédi [40], a line of work has sought to establish lower bounds on the complexity of refuting random k-CNF formulas in restricted proof systems. Such lower bounds have been established for a number of proof systems, including resolution [16, 40], k-DNF resolution [3, 8, 88], the Polynomial Calculus [4, 21], and Sum-of-Squares [137]. However, it has remained a long-standing open problem to establish lower bounds on random k-CNF formulas for Cutting Planes [18].

The first major contribution of this thesis is an exponential lower bound on the complexity of refuting random CNF formulas in Cutting Planes. This was also established independently and concurrently by Hrubeš and Pudlák [86].

Theorem 1.3.1. *There exists constants c, d such that the following holds. Let n be a sufficiently large positive integer, $k = c \log n$ and $m = n2^{dk}$. Then with high probability, any Cutting Planes refutation of a random k-CNF formula $F \sim \mathcal{F}(m, n, k)$ requires $2^{\tilde{\Omega}(n)}$ lines* [2].

While lower bounds were previously known for Cutting Planes, the technique used to establish these bounds (the method of feasible interpolation) was only applicable to the restricted class of *split formulas*, which does not contain random k-CNF formulas. Because of this, it has remained an important open problem to develop a lower bound technique for Cutting Planes that can be applied to more general classes of formulas. To prove Theorem 1.3.1, we significantly generalize Pudlák's feasible interpolation for Cutting Planes [127] so that it can be applied to *any* unsatisfiable CNF formula, rather than only split formulas. We show that for any unsatisfiable CNF formula F, if there is a Cutting Planes refutation of F, then there is a monotone real circuit of the same size computing an associated monotone function $\mathrm{mCSP\text{-}SAT}_F$. In fact, we establish a more general connection that holds not only for Cutting Planes, but for the stronger RCC_1 proof system (defined in Section 3.2). The next theorem characterizes the size of RCC_1 for any unsatisfiable CNF formula by the size of monotone real circuits for computing $\mathrm{mCSP\text{-}SAT}_F$.

[2] The notation $\tilde{\Omega}$ ignores factors of $\log n$.

Theorem 1.3.2. *Let F be any unsatisfiable CNF formula. There is an RCC_1 refutation of F of size s if and only if there is a monotone real circuit with $\mathsf{poly}(s)$ gates computing $\mathrm{mCSP\text{-}SAT}_F$.*

To obtain Theorem 1.3.1 from Theorem 1.3.2, we prove lower bounds on the size of monotone real circuits computing the $\mathrm{mCSP\text{-}SAT}_F$ instance obtained from a random k-CNF formula F. To do so, we would like to employ standard techniques for proving monotone real circuit lower bounds [23, 83, 93]. However, we are unable to apply these techniques to $\mathrm{mCSP\text{-}SAT}_F$ directly. Instead, we carefully design a reduction to a more well-structured instance of $\mathrm{mCSP\text{-}SAT}$ which is amenable to these lower bound techniques.

This chapter is based on the following publication:

[68]: Noah Fleming, Denis Pankratov, Toniann Pitassi, and Robert Robere. Random $\Theta(\log n)$-CNFs are hard for cutting planes. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 109–120, 2017.

Chapter 4: Stabbing Planes

As mentioned in the introduction, modern algorithms for integer programming are primarily based on the branch-and-cut paradigm which combines cutting planes deductions with a branch-and-bound procedure. As a result, modern integer programming algorithms are not modelled by the Cutting Planes proof system.

As the second major contribution of this thesis, we introduce and develop the *Stabbing Planes* proof system in order to model practical branch-and-cut algorithms. Stabbing Planes refutations are simple: given a polytope P containing no integer points, a *Stabbing Planes refutation* of P proceeds by choosing an arbitrary integer vector a, an integer b, and branching on $ax \geq b$ and its *integer negation* $ax \leq b - 1$. That is, we replace P by $P \cap \{x \in \mathbb{R}^n : ax \geq b\}$ and $P \cap \{x \in \mathbb{R}^n : ax \leq b - 1\}$. Then we recurse on these two polytopes, continuing until all descendant polytope are empty (that is, they do not contain any *real* solutions). The majority of branching schemes used in practical branch-and-cut algorithms (including all of the concrete schemes mentioned before) are examples of this general branching rule.

Branch-and-cut allows for Stabbing Planes-style branching as well as Cutting Planes deductions, hence it is not immediate that Stabbing Planes formalizes branch-and-cut. We show that from the perspective of proof complexity, these additional cutting planes are superfluous: Stabbing Planes can efficiently simulate Cutting Planes. In fact, we *characterize* Cutting Planes as a non-trivial sub-system of Stabbing Planes. A Stabbing Planes query is *facelike* if one of the sets $P \cap \{x : ax \leq b - 1\}$ or $P \cap \{x : ax \geq b\}$ is either empty or a *face* of the polytope P, and a *Facelike Stabbing Planes* is a Stabbing Planes proof which uses only facelike queries.

Theorem 1.3.3. *The proof systems Cutting Planes and Facelike Stabbing Planes are polynomially equivalent.*

This is surprising in part because the graphs underlying Stabbing Planes proofs are trees, while the graphs corresponding to Cutting Planes proofs may be dags: intuitively, this means that Cutting Planes proofs can "re-use" their intermediate steps, while Stabbing Planes proofs cannot. One may object, that this comparison is not fair as Cutting Planes is "deductive", meaning that new lines are inferred from previous ones, while Stabbing Planes queries are independent of previous queries. However, we show that Stabbing Planes can be viewed as a deductive system; in fact, it is equivalent to a *tree-like* variant of the $\mathsf{R(CP)}$ proof system of Krajíček [99]. While Stabbing Planes is equivalent to a proof system which is already in the literature, we

show that the perspective offered by Stabbing Planes is quite valuable. Indeed, none of our results, including the simulation of CP by SP or the next theorem, were known for $\mathsf{treeR}(\mathsf{CP})$.

As a second demonstration of the remarkable power of Stabbing Planes, we show that it admits short (quasipolynomial size) refutations of *any* unsatisfiable system of linear equations over a prime finite field.

Theorem 1.3.4. *Let F be the CNF encoding of an unsatisfiable system of m linear equations over a prime finite field. There is a Stabbing Planes refutation of F of size $|F|^{O(\log m)}$ and depth $O(\log^2 |F|)$.*

This stands in contrast to many of the standard proof systems that have been considered in the literature, such as Sum-of-Squares and AC^0-Frege, for which unsatisfiable systems of linear equations over a prime finite field are the class of formulas that are hard to refute [20, 81, 84, 124].

These results suggest that Stabbing Planes might be significantly more powerful than Cutting Planes. However, the following simulation shows that unless the Stabbing Planes proof makes use of large coefficients there can only be at most a quasipolynomial separation between the two.

Theorem 1.3.5. *Let F be any unsatisfiable CNF formula on n variables and suppose that there is an SP^* refutation of F in size s and maximum coefficient size c. Then there is a Cutting Planes refutation of F of size $s(cn)^{\log s}$.*

As a consequence, of this simulation we obtain short Cutting Planes refutations of any unsatisfiable system of linear equations over a prime finite field.

Corollary 1.3.6. *Let F be the CNF encoding of an unsatisfiable system of m linear equations over a prime finite field. Then there is a Cutting Planes refutation of F of size $|F|^{O(\log m)}$.*

This is unexpected; until recently it was believed that Cutting Planes could not efficiently reason over prime finite fields. Indeed, it was a long-standing conjecture that the *Tseitin formulas* — canonical systems of linear equations over $\mathbb{F}_2$ — required exponential size Cutting Planes proofs. This was recently refuted by Dadush and Tiwari [48] who showed that Cutting Planes admits quasipolynomial size refutations of the Tseitin formulas.

Additionally, Theorem 1.3.5 allows us to lift much of the analysis that has been done on Cutting Planes to branch-and-cut algorithms, including the lower bound on random CNF formulas from Theorem 1.3.1. Let SP^* denote the family of Stabbing Planes proofs in which each coefficient has at most quasipolynomial (that is, $n^{\log^{O(1)} n}$) magnitude.

Theorem 1.3.7. *There exists a family of unsatisfiable CNF formulas $\{F_n\}$ for which any SP^* refutation of F requires size 2^{n^ε} for constant $\varepsilon > 0$.*

Finally, an important problem left open by this work is to prove super-polynomial size lower bounds for *unrestricted* Stabbing Planes proofs. As a step towards this, we show how to obtain near-maximal lower bounds on the *depth* of Stabbing Planes refutations — the length of the longest root-to-leaf path in the proof tree. This is a natural measure of a proof, which captures the degree to which it can be parallelized.

Theorem 1.3.8. *There exists a family of unsatisfiable CNF formulas $\{F_n\}$ for which any SP refutation requires depth $\Omega(n/\log^2 n)$*

We complement this lower bound with a number of results which rule out natural approaches for proving lower bounds on the size of unrestricted Stabbing Planes proofs. In particular, those that seek to leverage

this depth lower bound in order to prove size lower bounds. In doing so, we prove that real communication complexity protocols cannot be *balanced* and establish the first lower bound on the real communication complexity of the famous *set-disjointness* problem.

This chapter is based on the following publications:

[15]: Paul Beame, Noah Fleming, Russell Impagliazzo, Antonina Kolokolova, Denis Pankratov, Toniann Pitassi, and Robert Robere. Stabbing planes. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 10:1–10:20, 2018.

[66]: Noah Fleming, Mika Göös, Russell Impagliazzo, Toniann Pitassi, Robert Robere, Li-Yang Tan, and Avi Wigderson. On the Power and Limitations of Branch and Cut. Accepted to CCC 2021.

Chapter 5: Lowerbounds and Supercritical Tradeoffs for Depth

Depth captures the inherent sequentialism in proofs, and therefore algorithms which employ the reasoning used in those proofs. For Cutting Planes, depth is closely related to the Chátal rank of polytopes, which has been extensively studied in integer programming theory.

Our simulation of Facelike Stabbing Planes by Cutting Planes (Theorem 1.3.3) suggests an interesting interplay between the depth and size of Cutting Planes proofs. First, we note that any CNF formula on n variables has a (trivial) depth n and exponential size refutation in Cutting Planes. However, the simulation of Facelike Stabbing Planes by Cutting Planes causes an explosion in the depth. For example, this simulation converts depth $O(\log^2 n)$ Stabbing Planes proofs of the Tseitin formulas — certain systems of $\mathbb{F}_2$ linear equations — into *quasipolynomial* depth Cutting Planes proofs. This is quite unusual since simulations between proof systems typically preserve the structure of the proofs. Thus, this brings up the possibility that the Tseitin formulas yield a *supercritical* size/depth tradeoff for Cutting Planes — formulas for which short proofs require superlinear depth.

Conjecture 1.3.9. *There exists a family of unsatisfiable formulas $\{F_n\}$ such that F_n has quasipolynomial size* CP *proofs, but any quasipolynomial-size proof requires superlinear depth. Furthermore, $\{F_n\}$ can be taken to be the Tseitin formulas on some family of graphs.*

Besides this question being interesting in its own right, connections between proof complexity and other areas, such as circuit complexity, raises the possibility that techniques used to establish this conjecture could allow us to establish tradeoffs which go beyond critical thresholds in other areas as well.

While Cutting Planes can somewhat efficiently reason about systems of linear inequalities over prime finite fields, the quasipolynomial depth of these proofs makes them prohibitive to implement. This raises another important question: how much additional strength needs to be added to Cutting Planes before the proofs of the Tseitin formulas can be parallelized. A natural candidate is the *semantic Cutting Planes* system, which augments the CG cut rule by allowing for *any* sound deduction from two previously derived linear inequalities. Semantic Cutting Planes proofs are known to be significantly more powerful than Cutting Planes [64].

In this chapter we make progress towards resolving both of these questions. Our first contribution is a novel geometric argument for proving lower bounds on the depth of semantic Cutting Planes proofs. As a consequence, we are able to establish a maximal lower bound on the depth of refuting the Tseitin formulas.

Theorem 1.3.10 (Informal)**.** *For all sufficiently large n, there exists an unsatisfiable instance of the Tseitin formulas which requires semantic Cutting Planes refutations of depth $\Omega(n)$.*

All prior depth lower bounds for Cutting Planes proceed by either reducing to communication complexity, or reducing to lower bounds on Chvátal rank. Since there is an $O(\log n)$ upper bound on the Tseitin formulas in communication complexity, this approach will be unable to prove a supercritical tradeoff for the Tseitin formulas. Similarly, it is known that the Chvátal rank of any CNF formula is bounded above by n. Our technique does not appear to suffer from these inherent counter examples, and therefore we hope that it is a step towards resolving the conjecture for the Tseitin formulas.

Our second contribution is a partial resolution to Conjecture 1.3.9. We establish supercritical size/depth tradeoffs for resolution (Res), k-DNF resolution ($\mathsf{Res}(k)$), and Cutting Planes (CP).

Theorem 1.3.11. *For any constant $\varepsilon > 0$, positive integers k, n sufficiently large, $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$, and any arbitrary real parameter $1 \leq c < n^{\frac{1-\varepsilon}{2+\varepsilon}}$, there is a CNF formula F on n variables and $n^{O(c)}$ clauses such that*

- *There is a $\mathcal{P}$-refutation of F of size $n^{O(c)}$.*
- *If Π is a $\mathcal{P}$-refutation of F with $\mathsf{size}(\Pi) = 2^{o\left(n^{\frac{1-\varepsilon}{2+\varepsilon}}/c\right)}$ then*

$$\mathsf{depth}(\Pi) \log^2 \mathsf{size}(\Pi) = \Omega\left(\frac{n^{c/(2+\varepsilon)}}{c \log n}\right).$$

Varying the parameter c between $O(1)$ and n^{δ}, for some small constant δ, allows us to obtain a family of tradeoff results. In one extreme, when $c = O(1)$ we obtain a formula F which has refutations of size $\mathsf{poly}(n)$, however any proof of size $\ll 2^{n^{1-\varepsilon}}$ must have depth *polynomial* depth. In the other extreme, setting $c = n^{\delta}$ implies an *exponential* blowup in the depth. To prove this theorem, we give a simplified proof of a supercritical tradeoff between resolution size and width [132]. Combining this with known lifting theorems [72, 139] establishes the tradeoff.

This chapter is based on the following publications:

[66]: Noah Fleming, Mika Göös, Russell Impagliazzo, Toniann Pitassi, Robert Robere, Li-Yang Tan, and Avi Wigderson. On the Power and Limitations of Branch and Cut. In *36th Computational Complexity Conference, CCCC 2021 July 20-23, 2021. Toronto, Ontario, Canada,* pages 6:1–6:30, 2021.

[119]: Noah Fleming, Toniann Pitassi, Robert Robere. Deep Proofs. Unpublished 2021.

Chapter 2

Preliminaries

A *boolean function* is a mapping $f : \{0,1\}^n \to \{0,1\}$ of strings of bits to a bit. An *assignment* or *restriction* is a map $\rho : [n] \to \{0,1,*\}$ to variables which is interpreted as fixing the variables indexed by $[n] \setminus \rho^{-1}(*)$ and leaving the variables indexed by $\rho^{-1}(*)$ free. An assignment is *total* if $|\rho^{-1}(*)| = 0$ and is *partial* otherwise. From a function f and a restriction ρ we can obtain a new function $f \restriction \rho : \{0,1\}^{|\rho^{-1}(*)|} \to \{0,1\}$, where for any $\alpha \in \{0,1\}^{|\rho^{-1}(*)|}$ we define $f \restriction \rho(\alpha) = f(\alpha \circ \rho)$. For total assignments, we will abuse notation and denote them as strings of bits $\rho \in \{0,1\}^n$, in which case we will denote $f \restriction \rho$ by $f(\rho)$. An assignment ρ is an *accepting input* or *yes instance* of f if $f(\rho) = 1$, while if $f(\rho) = 0$ then we say that ρ is a *rejecting input* or *no instance*.

Any boolean function can be encoded as a propositional logic formula using connectives $\wedge, \vee, \neg$. For simplicity, we will be particularly interested in encodings in *conjunctive normal form* (CNF) as a conjunction $C_1 \wedge \ldots \wedge C_m$ of clauses $C_i = (\ell_1 \vee \ldots \vee \ell_k)$, where each *literal* ℓ_j is either x_j or $\neg x_j$. A k-CNF, or k-SAT formula is a CNF formula in which every clause has *width* at most k, meaning that it contains at most k literals. For example, the following is an encoding of an $\mathbb{F}_2$ linear equation as a 3-CNF formula:

$$x_1 \oplus x_2 \oplus x_3 = 1 \quad \Rightarrow \quad (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3).$$

Similarly, a formula can be encoded in *disjunctive normal form* (DNF) as a disjunction of $T_1 \vee \ldots \vee T_m$ of terms $T_i = (\ell_1 \wedge \ldots \ell_k)$, and a DNF is a k-DNF if every term contains at most k literals.

A formula F is *satisfiable* if there exists an assignment $\rho \in \{0,1\}^n$ such that $F(\rho) = 1$, and is *unsatisfiable* otherwise. It is a *tautology* if it is satisfied by every assignment.

2.1 Proof Complexity

Propositional proof complexity is concerned with the task of *proving* the validity of tautological formulas in certain *propositional proof systems* (henceforth proof systems)[1]. By negation, this is equivalent to the task of *refuting* unsatisfiable formulas, and we will focus on the latter task. Furthermore, since every formula can be efficiently encoded in CNF, we will restrict attention to CNF formulas without loss of generality. Let $\mathsf{Unsat} \subseteq \{0,1\}^*$ denote the language of all unsatisfiable CNF formulas.

Proof System. A proof system for Unsat is a function $\mathcal{P} : \{0,1\}^* \to \mathsf{Unsat}$ satisfying the following:

[1] See [67] or [18] for a more in-depth introduction to proof complexity.

- *Soundness*. For every $\Pi \in \{0,1\}^*$, $\mathcal{P}(\Pi) \in \mathsf{Unsat}$.
- *Completeness*. For every $F \in \mathsf{Unsat}$ there exists $\Pi \in \{0,1\}^*$ such that $\mathcal{P}(\Pi) = F$.
- *Verifiability*. $\mathcal{P}$ is polynomial-time computable.

A proof system can be thought of as an efficient algorithm that checks whether the input string Π is a valid $\mathcal{P}$-proof; if it is, then the proof system outputs the formula $F \in \mathsf{Unsat}$ that it is a proof of. Otherwise, if the input is not a legal proof, then it simply outputs some fixed unsatisfiable formula, such as $x \wedge \neg x$. Soundness guarantees that the proof system only produces proofs of $F \in \mathsf{Unsat}$, while completeness ensures that every $F \in \mathsf{Unsat}$ has a proof.

The complexity of a proof system $\mathcal{P}$ is measured by the size of its proofs. For a proof Π, $\mathsf{size}(\Pi)$ will denote the number of bits needed to write it down. The size of refuting some $F \in \mathsf{Unsat}$ is the size of the shortest $\mathcal{P}$-proof of F, which we denote by $\mathsf{size}_{\mathcal{P}}(F)$.

Resolution. A standard example of a proof system is *resolution*, which forms the basis for many well-known automated theorem provers and algorithms for SAT. A resolution refutation of an unsatisfiable CNF formula F is a sequence of clauses $\{C_i\}_{i\in[s]}$ such that the final clause C_s is the empty clause Λ, and every C_i is either a clause from F or is deduced by one of the following rules.

- *Resolution Rule*. From clauses C_j containing a variable x and C_k containing $\neg x$, deduce the clause $(C_j \setminus x) \vee (C_k \setminus \neg x)$.
- *Weakening*. From C_i deduce C_j such that $C_j \supseteq C_i$.

We will encode a resolution refutation by encoding each clause in the sequence; since each clause contains at most n literals, the number of clauses in the refutation is polynomially related to the bit-length encoding. Thus, without loss of generality we take the number of clauses to be the size of a resolution refutation. The size of refuting F, $\mathsf{size}_{\mathsf{Res}}(F)$, is the minimum size of any resolution refutation of F. Another measure of resolution complexity, which is closely related to size [20], is the maximum *width* of a proof Π — the maximum number of literals in any clause occurring in Π — denoted $\mathsf{width}(\Pi)$. The width of refuting F, denoted $\mathsf{width}_{\mathsf{Res}}(F)$ is the minimum width of any resolution refutation of F.

It is helpful to visualize a resolution proof as a directed acyclic graph (dag) where the nodes correspond to clauses C_i for $i \in [s]$ and there are directed edges from C_i and C_j to C_k if C_k was derived by resolving C_i and C_j. With this in mind, the *depth* of a resolution proof Π (denoted $\mathsf{depth}(\Pi)$) is the length of the longest root-to-leaf path in the proof dag. The *resolution depth* $\mathsf{depth}_{\mathsf{Res}}(F)$ of refuting F is the minimum depth of any resolution refutation of F. As well, we will say that a resolution proof is *tree-like* if its dag is a tree, and denote by $\mathsf{treeRes}$ the proof system which produces only tree-like resolution proofs.

Resolution is among the weakest proof systems studied in proof complexity. We compare the strength of proof systems using the notion of *polynomial simulation*.

Polynomial Simulation. For proof systems $\mathcal{P}_1$ and $\mathcal{P}_2$, we say that $\mathcal{P}_1$ polynomially simulates $\mathcal{P}_2$ if there exists a polynomial q such that for sufficiently large n and for all $F \in \mathsf{Unsat}$ with $|F| \geq n$, $\mathsf{size}_{\mathcal{P}_1}(F) \leq q(\mathsf{size}_{\mathcal{P}_2}(F))$.

In other words, $\mathcal{P}_1$ polynomially simulates $\mathcal{P}_2$ if the size of the shortest $\mathcal{P}_1$ proof of F is at most polynomially smaller than the size of the shortest $\mathcal{P}_2$ proof of F.

2.2 Proof Complexity of Integer Programming

A proof system can be defined more generally for any language $\mathcal{L} \subseteq \{0,1\}^*$. In this thesis, we will study proof systems for the language of unsatisfiable *integer programming problems*.

Let us first recall up some terminology. A *polyhedron* $P \subseteq [0,1]^n$ is a convex set formed by the intersection of a system of linear inequalities $Ax \leq b$; i.e., $P = \{x \in \mathbb{R}^n : Ax \leq b\}$. Throughout this thesis, we will restrict attention to rational polyhedrons; i.e., those with $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$. A polyhedron P is a *polytope* if it is bounded. A hyperplane $ax = b$ is *supporting* for P if $b = \max\{ax : x \in P\}$, and if $ax = b$ is a supporting hyperplane then the set $P \cap \{x \in \mathbb{R}^n : ax = b\}$ is called a *face* of P. An inequality $ax \leq b$ is *valid* for P if every point of P satisfies the inequality.

In an integer programming problem, we are given a polyhedron P and a vector $c \in \mathbb{Z}^n$, and our goal is to find a point $x \in \mathbb{Z}^n \cap P$ maximizing $c \cdot x$. An integer program is *unsatisfiable* if the polytope contains no integer points; i.e., $\mathcal{P} \cap \mathbb{Z}^n = \emptyset$. In this thesis, we will be interested in proof systems for refuting unsatisfiable integer programs. This language is coNP-complete, and therefore by standard reductions between this language and UNSAT, we can consider these systems as propositional proof systems. Formally, to relate these proof systems to proof systems for Unsat, we will use the following simple encoding of a CNF formula as a system of linear inequalities which preserves satisfiability.

Proposition 2.2.1. *Let $F = C_1 \wedge \ldots \wedge C_m$ be a CNF formula. Construct the system of inequalities $Ax \leq b$ as follows: first, for each variable x_i add the inequalities $0 \leq x_i$ and $x_i \leq 1$. Next, if $C_i := \bigwedge_{i \in S} x_i \wedge \bigwedge_{j \in T} \neg x_j$ is a clause in F then add the inequality*

$$\sum_{i \in P} x_i + \sum_{j \in N} (1 - x_j) \geq 1.$$

Then, $Ax \leq b$ is satisfiable if and only if F is.

Proof. Suppose that $\alpha \in \mathbb{Z}^n$ satisfies $Ax \leq b$. By the inequalities $x_i \geq 0$ and $x_i \leq 1$, $\alpha \in \{0,1\}^n$. Now, consider an inequality encoding a clause $C \in F$. Since α satisfies this inequality, it must set a variables x_i with $i \in P$ to 1, or x_i with $i \in N$ to 0. It follows that α satisfies C. The converse is similar. □

The canonical proof system for integer programming is the *Cutting Planes* system [37].

Cutting Planes. A *Cutting Planes* (CP) *proof* of an inequality $cx \leq d$ from a polytope P, defined by a system of integer linear inequalities $Ax \leq b$, is given by a sequence of inequalities $\{c_i x \leq d_i\}_{i \in [s]}$ such that $c_s = c$, $d_s = d$, and each inequality $c_i x \leq d_i$ is either a defining inequality of P, or is deduced from earlier inequalities in the sequence by applying one of the following two deduction rules:

- *Conic Combination.* From inequalities $c_1 x \leq d_1, c_2 x \leq d_2$, deduce any nonnegative linear combination of these two inequalities with integer coefficients.
- *Division.* From an inequality $cx \leq d$, if $\delta \in \mathbb{Z}$ with $\delta \geq 0$ divides all entries of c then deduce $(c/\delta)x \leq \lfloor d/\delta \rfloor$.

A Cutting Planes refutation of P is a derivation of the inequality $1 \leq 0$ from the inequalities defining P. Similarly, a Cutting Planes refutation of a CNF formula F is a refutation of the system of linear inequalities corresponding to F given by Proposition 2.2.1.

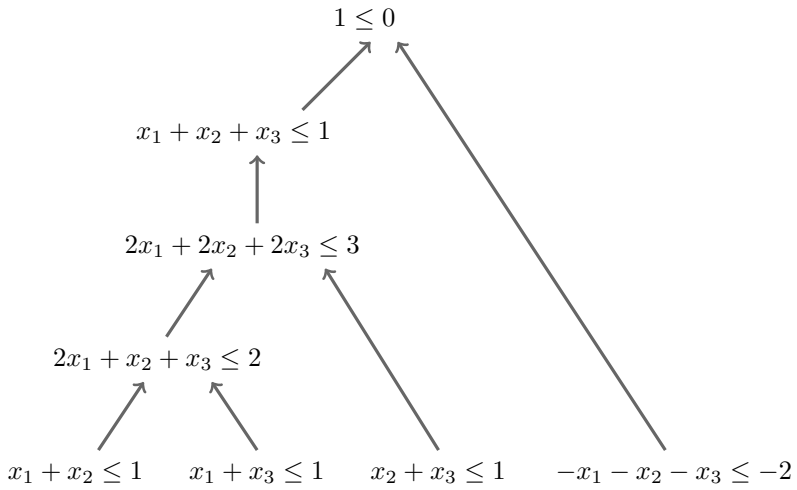


Figure 2.1: A Cutting Planes refutation of the inequalities labelling the leaves of the tree.

The *size* of a Cutting Planes proof Π, denoted $\mathsf{size}(\Pi)$ is the number of inequalities in it, and the size of refuting F, $\mathsf{size}_{\mathsf{CP}}(\Pi)$, is the minimum size of any CP refutation of F. It is known that for unsatisfiable CNF formulas that this measure is polynomially related to the length of the bit-encoding of the proof [45]).

As with resolution, it is natural to arrange Cutting Planes proofs into dags (see Figure 2.1). As well, a proof is *tree-like* if the underlying dag is a tree, and we denote by treeCP the restriction of Cutting Planes to tree-like proofs. With this dag in mind, we define the *depth* of a Cutting Planes refutation to be the longest root-to-leaf path in the proof, and let $\mathsf{depth}_{\mathsf{CP}}(F)$ be the minimum depth of any Cutting Planes refutation of F. It is known that *any* polytope contained in the unit cube $[0,1]^n$ has CP depth at most $O(n^2 \log n)$ [61]. However, for unsatisfiable CNF formulas, the CP-depth is at most n [25].

As stated in the introduction, an alternative characterization of Cutting Planes uses the notion of *Chvátal-Gomory cuts* (or simply *CG cuts*) [37, 45].

CG cut. Let P be a polytope, and let $cx \leq d$ be any valid inequality for P such that all coefficients in c are integers. The halfspace $\{x \in \mathbb{R}^n : cx \leq \lfloor d \rfloor\}$ is called a *CG cut* for P. We will abuse notation and also refer to the inequality $cx \leq \lfloor d \rfloor$ as a CG cut.

If $cx \leq \lfloor d \rfloor$ is a CG cut for the polytope P, then we can efficiently deduce a constraint which is at least as strong as $cx \leq \lfloor d \rfloor$ in Cutting Planes.

Proposition 2.2.2. *Let P be any polytope and $cx \leq \lfloor d \rfloor$ be a valid CG cut from P, then there is a CP derivation of $cx \leq \lfloor d \rfloor + \delta$ for some $\delta > 0$ in $O(n)$ steps. Conversely, any Cutting Planes deduction is a CG cut.*

The proof follows from Farkas' Lemma, which will be used throughout this thesis.

Farkas' Lemma. *Let $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$. Then exactly one of the following holds:*

(i) There exists $x \in \mathbb{R}^n$ such that $Ax \leq b$.

(ii) There exists $y \in \mathbb{Z}^m$ with $y \geq 0$ such that $y^\top A = 0$ and $y^\top b < 0$. Moreover, y has at most $n + 2$ non-zero coordinates.

The "moreover" part in (ii) follows from Carathéodory's Theorem. Proofs of both Farkas' Lemma and Carathéodory's Theorem can be found in [42].

Proof of Proposition 2.2.2. Let P be a polytope defined by a system of integer linear inequalities $Ax \geq b$, and assume for simplicity that P is non-empty. As well, let $cx \leq \lfloor d \rfloor$ be a CG cut from P. Then $cx \leq d$ is valid for P, and so $P' := P \cap \{x \in \mathbb{R}^n : -cx \leq -(d+\varepsilon)\}$ is empty for every $\varepsilon > 0$. By Farkas' Lemma there exists $y \in \mathbb{Z}^m, \tilde{y} \in \mathbb{Z}$ such that $y^\top A - \tilde{y}c = 0$ and $y^\top b - \tilde{y}(d+\varepsilon) < 0$. Furthermore, because P is non-empty, $\tilde{y} \neq 0$. It follows that $(y^\top/\tilde{y})A = c$ and $(y^\top/\tilde{y})b < d + \varepsilon$. Therefore, we can derive $cx \leq (y^\top/\tilde{y})b$ by $O(n)$ applications of the *conic combination* rule from P. Applying *division*, we have deduced $cx \leq \lfloor (y^\top/\tilde{y})b \rfloor \leq \lfloor d + \varepsilon \rfloor$. By choosing $0 < \varepsilon < 1$ so that $d + \varepsilon < \min(\lceil d \rceil, d+1)$, we have derived CG cut $cx \leq \lfloor (y^\top/\tilde{y})b \rfloor \leq \lfloor d \rfloor$.

Finally, note that the converse direction is immediate because the inequalities derived by conic combination are valid for P and those derived by division are CG cuts by definition. □

For algebraic proof systems, it is standard to use an asterisk to denote the restriction of that proof system to require that the coefficients have only quasipolynomial magnitude.

Bounded Weight Cutting Planes. A CP^* proof is a CP proof $\{c_i x \leq d_i\}_{i \in [s]}$ such that the *weight* $\|c_i\|_1$ of each inequality in the proof is quasipolynomially bounded, i.e., at most $n^{\log^{O(1)}(n)}$.

We will also be interested in the following *semantic* generalization of Cutting Planes.

Semantic Cutting Planes. The *semantic Cutting Planes* (denoted sCP or semantic CP) proof system is defined only for 0-1 integer programming problems (such as encodings of unsatisfiable CNF formulas) where you are asked to find a solution $x \in \{0,1\}^n$, rather than $\mathbb{Z}^n$. Semantic Cutting Planes is a strengthening of Cutting Planes proofs to allow *any deduction* that is sound over points in $\{0,1\}^n$ [30]. Like Cutting Planes, an sCP proof from a polytope P is given by a sequence of halfspaces $\{c_i x \leq d_i\}_{i \in [s]}$, such that each inequality is either in P or or is deduced from earlier inequalities by the following very powerful *semantic deduction rule*:

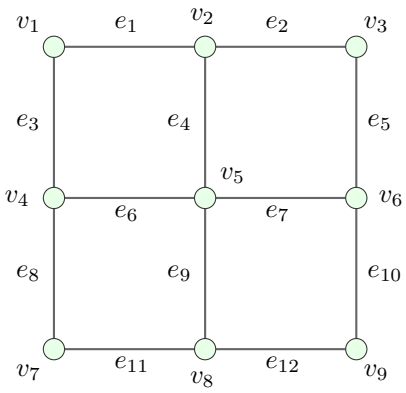
- *Semantic Deduction.* From $c_1 x \leq d_1$ and $c_2 x \leq d_2$ deduce $cx \leq d$ if every $\{0,1\}$-assignment satisfying both $c_1 x \leq d_1$ and $c_2 x \leq d_2$ also satisfies $cx \leq d$.

Analogous to CP, the size of refuting F in sCP is the minimum number lines of any sCP refutation of F, and the depth $\mathsf{depth}_{\mathsf{sCP}}(F)$ is the minimum depth of any sCP refutation of F.

Filmus et al. [64] showed that sCP is extremely strong: there are instances for which any refutation in CP requires exponential size, and yet these instances admit polynomial-size refutation in sCP. In fact, they show that this deduction rule is so strong that it is only polynomial-time verifiable if $\mathsf{P} = \mathsf{NP}$. Therefore, sCP is not a proof system as defined above. Even so, exponential lower bounds on the size of sCP proofs have been established [30, 64].

2.3 Hard Formulas

To prove lower bounds on a proof system, we first require a candidate family of hard formulas. In this subsection we will give a brief overview of the two main formulas that we will focus on in this thesis: random k-CNF formulas and the Tseitin formulas.



$$
\begin{aligned}
v_1 &: \ e_1 \oplus e_3 = 1 \\
v_2 &: \ e_1 \oplus e_2 \oplus e_4 = 1 \\
v_3 &: \ e_2 \oplus e_5 = 1 \\
v_4 &: \ e_3 \oplus e_6 \oplus e_8 = 1 \\
v_5 &: \ e_4 \oplus e_6 \oplus e_7 \oplus e_9 = 1 \\
v_6 &: \ e_5 \oplus e_7 \oplus e_{10} = 1 \\
v_7 &: \ e_8 \oplus e_{11} = 1 \\
v_8 &: \ e_{11} \oplus e_9 \oplus e_{12} = 1 \\
v_9 &: \ e_{12} \oplus e_{10} = 1
\end{aligned}
$$

Figure 2.2: An unsatisfiable instance of the Tseitin formulas on a 3×3 grid graph with $l(v) = 1$ for all $v \in V$.

Random k-CNF Formulas. Let $\mathcal{F}(m, n, k)$ denote the distribution over k-CNF formulas sampled as follows: select m clauses uniformly at random from the set of all possible $2^k \binom{n}{k}$ clauses of width k over n variables. Denote by $F \sim \mathcal{F}(m, n, k)$ that the formula F is drawn from this distribution.

The *unsatisfiability* of $F \sim \mathcal{F}(m, n, k)$ is controlled by the *clause-density* $\Delta_k := m/n$. This control exhibits a threshold phenomenon: as the density increases, $F \sim \mathcal{F}(m, n, k)$ transitions from being being almost surely satisfiable to almost surely unsatisfiable. The density parameter also plays a role in lower bounds for refuting $F \sim \mathcal{F}(m, n, k)$: the hardest formulas are generated for Δ_k near this threshold. A more detailed discussion of these formulas can be found in Chapter 3.

The second family of formulas that will feature prominently in this thesis is the *Tseitin formulas*. For many algebraic proof systems, the Tseitin formulas, and more generally systems of linear equations over a prime finite field, are the canonical family of hard formulas [32, 80, 81, 137] (see [69] for a survey).

Tseitin Formulas. For any graph G and any $\{0, 1\}$-labelling ℓ of the vertices of G, the *Tseitin formula* of (G, ℓ) is the following system of $\mathbb{F}_2$-linear equations: for each edge e we introduce a variable x_e, and for each vertex v we have an equation

$$\bigoplus_{u:uv \in E} x_{uv} = \ell(v)$$

asserting that the sum of the edge variables incident with v must agree with its label $\ell(v)$ (note that such a system is unsatisfiable as long as $\sum_v \ell(v)$ is odd).

2.4 Communication Complexity and the CNF Search Problem

In order to prove lower bounds for a family of unsatisfiable formulas it has been fruitful to study the following associated search problem.

CNF Search Problem. Let $F = C_1 \wedge \ldots \wedge C_m$ be a CNF formula and let (X, Y) be any partition of its variables. The associated CNF search problem $\mathsf{Search}_F^{(X,Y)} \subseteq \{0,1\}^{|X|} \times \{0,1\}^{|Y|} \times [m]$ is defined as $(x, y, i) \in \mathsf{Search}_F^{(X,Y)}$ if and only if $C_i(x, y) = 0$.

Lovász et al. [108] introduced the CNF search problem as a proof complexity analogue of the celebrated Karchmer-Wigderson game [95] for circuit complexity. Following this, Impagliazzo et al. [89] observed that

lower bounds on the cost of solving the CNF search problem in certain models of communication complexity implied lower bounds on the size of tree-like CP refutations. The basic notion of communication is as follows; for a more detailed introduction we recommend the excellent book of Kushilevitz and Nisan [103].

Deterministic Communication. A deterministic communication protocol for a search problem $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ consists of two players, Alice and Bob. Alice receives an input $x \in \mathcal{X}$ and Bob receives $y \in \mathcal{Y}$; their aim is to agree on some $o \in \mathcal{O}$ for which $(x, y, o) \in \mathcal{S}$. To do so, they are allowed to communicate by sending messages to each other (in the form of a single bit) according to some *protocol*, which they agree on before seeing their inputs. The transcript of bits sent by the two players on some input (x, y) is known as the *history* of the protocol. We can model this combinatorially: every step in the communication protocol is associated with *rectangle* of inputs $\mathcal{X}' \times \mathcal{Y}' \subseteq \mathcal{X} \times \mathcal{Y}$ consistent with the communication thus far; $\mathcal{X}'$ models what Bob knows about Alice's input, and $\mathcal{Y}'$ models what Alice knows about Bob's. If Alice communicates a bit, then this partitions $\mathcal{X}'$ into $\mathcal{X}'_0$ and $\mathcal{X}'_1$ corresponding to whether the bit Alice sent was 0 or 1; similarly, if Bob communicated then this partitions $\mathcal{Y}'$ into $\mathcal{Y}'_0$ and $\mathcal{Y}'_1$. The communication ends when $\mathcal{X}' \times \mathcal{Y}'$ is *monochromatic*, meaning that there is some $o \in \mathcal{O}$ such that $(x, y, o) \in \mathcal{S}$ for every $(x, y) \in \mathcal{X}' \times \mathcal{Y}'$.

Thus, we can define a *communication protocol* formally as a full binary tree where every node is labelled with a rectangle (the root is labelled with $\mathcal{X} \times \mathcal{Y}$), the leaves are labelled with outputs $o \in \mathcal{O}$ which are monochromatic for their rectangles, and the non-leaf nodes are labelled with a player, and have two outgoing edges labelled 0 and 1. If a node with rectangle $\mathcal{X}' \times \mathcal{Y}'$ is owned by Alice then the rectangles of the two children of the node partition $\mathcal{X}' \times \mathcal{Y}'$ into $\mathcal{X}'_0 \times \mathcal{Y}'$ and $\mathcal{X}'_1 \times \mathcal{Y}'$; similarly, if the node is owned by Bob then the children partition the rectangle along the $\mathcal{Y}$-inputs (see Figure 2.3). Every root-to-leaf path in this tree corresponds to a history of Alice and Bob's communication for some subset of inputs (x, y).

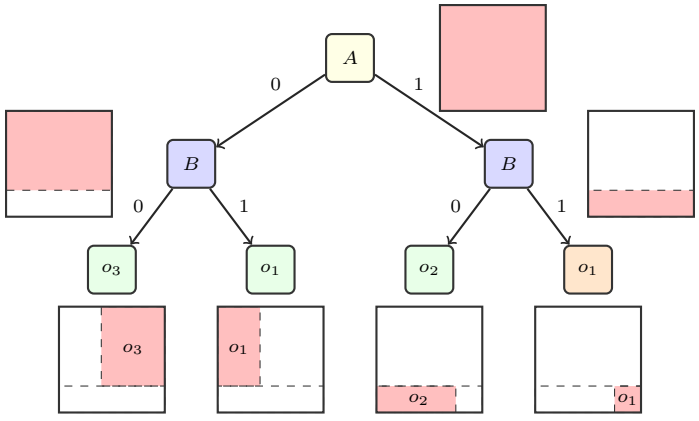


Figure 2.3: A communication protocol labelled with the corresponding rectangles. Alice speaks at the nodes labelled A and Bob speaks at the nodes labelled B. The outgoing edges of each node correspond to the bit sent by the player who owns that node. The leaf rectangles are labelled by their output o_i to denote that they are o_i-monochromatic.

The *(deterministic) communication complexity* of computing $\mathcal{S}$ is the minimum number of bits communicated, or *rounds* of communication, needed to solve $\mathcal{S}$ on any input $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Alternatively, it is the minimum depth of any communication protocol for $\mathcal{S}$.

Observe that for $x, y \in \{0, 1\}^n$, any integer linear inequality $ax + by \geq d$ can be evaluated in $w = \log \|a\|_1 + \log \|b\|_1$ bits by Alice communicating ax to Bob, and Bob responding with by. Thus, for example,

a Cutting Planes refutation of a CNF formula F of depth d in which the size of the coefficients in every inequality is bounded by 2^w gives rise to a $O(dw)$-round communication protocol for solving $\mathsf{Search}_F^{(X,Y)}$ for any partition (X, Y) of the variables. If the coefficients in the CP proof are not too large, then we can obtain depth lower lower bounds via lower bounds on the communication complexity of $\mathsf{Search}_F^{(X,Y)}$. Furthermore, by strengthening the underlying communication protocol we can simulate arbitrary linear inequalities; the following notion of a *real communication protocol* was introduced by Krajíček [100].

Real Communication. A *real communication protocol* for $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ generalizes the deterministic communication model. Now, the players Alice and Bob communicate via a "referee". In each round, Alice and Bob send real numbers r_A, r_B to the referee who responds with a single bit b which is 1 if $r_A > r_B$, and 0 otherwise. The *real communication complexity* of computing $\mathcal{S}$ is the number minimum number of rounds needed to communication needed to solve $\mathcal{S}$ on any input $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

Observe that any linear inequality $ax + by \geq d$ can be computed in a single round of real communication by Alice sending $d - ax$ to the referee and Bob sending by.

Chapter 3

Random CNF Formulas are Hard for Cutting Planes

3.1 Introduction

In this chapter we study the complexity of refuting *randomly generated* SAT instances. The most well-studied random SAT distribution is the *random k-SAT model* $\mathcal{F}(m, n, k)$ where a random k-CNF over n-variables is chosen by uniformly at random selecting m clauses from the set of all possible clauses on k distinct variables. This is an intrinsically natural distribution of instances similar to the Erdös-Rényi random graph model, and it is closely related to phase transitions and structural phenomenon in statistical physics (e.g., [96, 140]). As well, this model has close connections to complexity theory through *Feige's Hypothesis*: if $\mathcal{F}(m, n, k)$ is hard to refute on average for the "right" choice of m, n, k then worst-case inapproximability results follow for many NP-hard optimization problems [63], as well as average-case hardness of circuit lower bounds [136] and related problems [85].

In proof complexity, random k-CNF formulas make up the family of formulas which is most widely believed to be hard to refute in *every* propositional proof system, and this has already been confirmed for a number of weak proof systems. The first example was given by Chvátal and Szemerédi [40], who showed that random k-CNF instances require resolution refutations of size $\exp(\Omega(n))$ with high probability. Superpolynomial lower bounds for random k-CNF formulas have been shown for a number of other proof systems, including Polynomial Calculus [4, 21], $\mathsf{Res}(k)$ [3, 139], and Sum-of-Squares [137].

It is a well-known open problem to prove superpolynomial lower bounds on the size of Cutting Planes refutations of random k-CNF formulas (see, e.g., [18]), especially because superpolynomial lower bounds for other formulas have been shown [30, 127]. The main contribution of this chapter is the first lower bound on the size of refutations of random k-CNF formulas in Cutting Planes, provided k is large enough.

Theorem 1.3.1. *There exists constants c, d such that the following holds. Let n be a sufficiently large positive integer, $k = c \log n$ and $m = n2^{dk}$. Then with high probability, any Cutting Planes refutation of a random k-CNF formula $F \sim \mathcal{F}(m, n, k)$ requires $2^{\tilde{\Omega}(n)}$ lines* [1].

In fact, our exponential lower bounds even apply to some stronger proof systems than Cutting Planes — see Section 3.2 for details. This lower bound has been independently obtained by Pavel Hrubeš and Pavel

[1] The notation $\tilde{\Omega}$ ignores factors of $\log n$.

Pudlák [86] using similar techniques.

Our proof technique is a divergence from previous lower bounds on random k-CNF formulas. These have, with the exception of lower bounds for the Polynomial Calculus over $\mathbb{F}_2$ [4], proceeded by first showing lower bounds on systems of random $\mathbb{F}_2$-linear equations and then use formulas can be viewed as over-constrained versions of k-CNF formulas (detailed in Chapter 5). Indeed, this is the first lower bound on random k-CNF formulas for a proof system which can efficiently refute systems of linear equations over any rpime finite field, as we show in Chapter 4.

Proof Technique. To obtain our lower bounds we introduce a new technique for proving Cutting Planes lower bounds. Our technique is a significant generalization of the classic (and prior to this work, only) lower bound technique for Cutting Planes: the method of *feasible interpolation* [30, 98, 100, 127, 130]. As our technique generalizes it, let us first describe feasible interpolation. Suppose we are given an unsatisfiable CNF formula $F(x, y, z)$ on three sets of variables x, y, z of the following "split" form

$$F(x, y, z) = A(x, z) \wedge B(y, z),$$

where A and B are themselves CNF formulas. Then, given an assignment α to the z variables it follows that either $A(x, \alpha)$ is unsatisfiable or $B(y, \alpha)$ is unsatisfiable. A feasible interpolation argument shows that the complexity of *computing* the associated *interpolant function*

$$I_F(\alpha) = \begin{cases} 1 & \text{if } A(x, \alpha) \text{ is unsatisfiable} \\ 0 & \text{otherwise} \end{cases}$$

is a lower bound on the complexity of *refuting* F — or, said contrapositively, a proof system P has *feasible interpolation* if from a short P-refutation of $F(x, y, z)$ we can extract an efficient algorithm computing I_F in some model of computation. Feasible interpolation was introduced at this level of generality in the classic work of Krajíček [98] where it was shown, for example, that *resolution* has feasible interpolation by *monotone circuits* — lower bounds on the monotone circuit complexity of I_F can be used to show lower bounds on the size of resolution refutations of F (provided that the split formula F, and therefore I_F, is "monotone" in a certain technical sense).

Instances of Krajíček's general interpolation method have led to lower bounds for a number of proof systems where previously no lower bounds were known. First, Razborov [130] proved lower bounds for certain systems of Bounded Arithmetic from monotone circuit lower bounds. Following this, Bonet, Pitassi, and Raz [30] gave superpolynomial lower bounds for "low-weight" Cutting Planes (CP^*) proofs as well as for other proof systems such as CC-proofs where lines are computed by low-depth communication protocols [98]. In particular they proved that any size s and weight w Cutting Planes proof implies a size $\mathsf{poly}(s \log w)$ monotone circuit for separating the associated monotone interpolant, I_F. Then, by constructing a split formula whose interpolant corresponds to the clique function, they reduced lower bounds for CP^* proofs and CC proofs to the celebrated monotone circuit lower bounds for the clique function [131].

In [127], Pudlák proved the first exponential lower bounds for Cutting Planes proofs with *unbounded* weights. To do so, he first showed that small Cutting Planes refutations of monotone split formulas $F(x, y, z)$ imply small monotone *real* circuits computing the associated monotone interpolant I_F; thus, reducing lower the problem of obtaining lower bounds for Cutting Planes proofs of monotone split formulas to the proving lower bounds on monotone *real* circuits. Secondly, Pudlák strengthened Razborov's clique lower bound to

apply to the larger family of monotone real circuits. Lower bounds on monotone real circuits were also proved independently by Cook and Haken for the broken mosquito screen formulas [83]. Altogether, these imply exponential lower bounds on Cutting Planes proofs. Pudlák's result was later improved to hold for Cutting Planes proofs using *split cuts* by Dash [52], and then to the more general *semantic Cutting Planes* by Filmus, Hrubeš, and Lauria [64].

Despite the success of feasible interpolation, it limits the lower bounds to split formulas; in particular, at the time of this work, the only families of formulas that were known to be hard for (unrestricted) Cutting Planes were the clique-coclique formulas [30, 127] and the broken mosquito screen formulas [46].

To prove Theorem 1.3.1 we generalize Pudlák's feasible interpolation theorem for Cutting Planes so that it can be applied to *any* unsatisfiable CNF formula F. That is, we show that if there is a polynomial-size Cutting Planes refutation of *any* unsatisfiable CNF formula F then there is a polynomial-size monotone real circuit for computing a corresponding monotone (partial) function, $\mathrm{mCSP\text{-}SAT}_F$, which is a monotone encoding of the CSP-SAT problem whose definition depends on F. In fact, we provide a more general connection that holds not just for Cutting Planes, but for the stronger *real communication* proof system RCC_1 (defined in Subsection 3.4.1). The next theorem characterizes the size of RCC_1 refutations for any formula F by the size of monotone real circuits computing $\mathrm{mCSP\text{-}SAT}_F$.

Theorem 1.3.2. *Let F be any unsatisfiable CNF formula. There is an RCC_1 refutation of F of size s if and only if there is a monotone real circuit with $\mathsf{poly}(s)$ gates computing $\mathrm{mCSP\text{-}SAT}_F$.*

The proof of this theorem is inspired by the seminal Karchmer-Wigdreson connection between *circuit complex* and *communication complexity* [95], and generalizes several earlier results [30, 98, 130]. In more detail: Karchmer and Wigdreson proved that the *depth* of a boolean circuit computing a boolean function $f : \{0,1\}^n \to \{0,1\}$ is exactly the communication complexity of solving a certain relation — known as the *Karchmer-Wigderson game* (see Section 3.2). Razborov generalized this result, proving a non-trivial equivalence between the size of certain *dag-like* communication protocols and boolean circuit size [130].

Razborov's work played a key role in inspiring Krajíček's feasible interpolation result [98] discussed above. Using Razborov's equivalence, Krajíček generalized the result of Bonet, Pitassi, and Raz [30] to obtain a general interpolation theorem, showing that circuit lower bounds for computing interpolant functions imply lower bounds on the CC proof system mentioned above. In Section 3.4 we show that Krajíček's result can be generalized: the complexity of refuting *any* unsatisfiable CNF formula F in the CC proof system is actually *characterized* by the circuit complexity of the $\mathrm{mCSP\text{-}SAT}_F$ function. This observation is already strong enough to give lower bounds on CP^* proofs.

Theorem 1.3.2 provides a similar characterization, but this time for RCC_1 proofs and monotone real circuits. For this, Razborov's equivalence between dag-like communication protocols and boolean circuits is insufficient. We instead employ a recent (and beautiful) generalization of Razborov's result due to Hrubeš and Pudlák [87], which characterizes the size of monotone *real* circuits in terms of dag-like *real* communication protocols. Finally, to deduce Theorem 1.3.1 from Theorem 1.3.2, we need to prove lower bounds for monotone real circuits computing the $\mathrm{mCSP\text{-}SAT}$ problem obtained from a random k-CNF instance. To do so, we combine standard techniques for proving monotone circuit lower bounds (the symmetric method of approximations [23, 83, 93]) with a non-trivial reduction from random k-CNF instances to certain "balanced" random k-CNF instances. The theorem follows because RCC_1 proofs generalize Cutting Planes proofs.

As stated above, Hrubeš Pudlák have independently proved Theorem 1.3.2 using nearly identical techniques [86]. Given any unsatisfiable CNF formula F they show how to obtain a partial monotone boolean function which they call an *unsatisfiability certificate* for F (defined in Subsection 3.3.1). Then, they show

that the complexity of computing an unsatisfiability certificate by a monotone real circuit implies lower bounds for Cutting Planes by directly reducing these certificates to the feasible interpolation lower bounds. As boolean functions, the unsatisfiability certificates are exactly the same as our mCSP-SAT problem, which we show in Subsection 3.3.1. Their lower bounds for random k-CNF formulas are also obtained by using the symmetric method of approximations in a nearly identical proof to ours. Further, they used this technique to prove lower bounds for other problems: a generalization of the Pigeonhole Principle called the *Weak Bit Pigeonhole Principle*, and a function related to Feige's hypothesis.

It is natural to wonder whether or not this new lower bound technique can be extended to obtain lower bounds for k-CNF instances when k is bounded by a constant. By a careful balancing of parameters, our technique can be used to obtain superpolynomial lower bounds when $k \geq \log\log n$. However, when $k = \Theta(1)$ the method of approximations fails to give super polynomial lower bounds on the mCSP-SAT problem. Thus, it appears that we will not be able to push our lower bounds any further via this technique without improving the underlying monotone circuit lower bound techniques.

The Random k-SAT Model. The *unsatisfiability* of $F \sim \mathcal{F}(m, n, k)$ is controlled by the *clause-density* $\Delta_k := m/n$. For instance, it is easy to see that if $\Delta > 2k \ln 2$ then $F \sim \mathcal{F}(m, n, k)$ is unsatisfiable with high probability. The *satisfiability conjecture* states that this control exhibits a threshold phenomena: for all k there exists a fixed constant c_k such that random k-CNF formulas with density $\Delta_k > c_k$ are almost surely unsatisfiable, while those with density $\Delta_k < c_k$ are almost surely satisfiable. For $k = 2$, the conjecture was known to be true since the early 1990s [39, 55, 74]. In a recent breakthrough, this was resolved for large values of k by appealing to to arguments in statistical mechanics [59].

The density parameter also plays a crucial role in lower bounds for refuting $F \sim \mathcal{F}(m, n, k)$ in proof complexity. Our main theorem holds for $\Delta_k = \Theta(2^{(1+\tau)k})$ for some constant $\tau \in (0, 1)$. Furthermore, the interval for which our lower bounds hold seems to be relatively narrow (for instance, it seems impossible to choose $\tau \approx 0$ or $\tau \gtrapprox 1$). In contrast, the classic work of Chvátal and Szemerédi [40] show that for any fixed $\Delta_k > 2^k \ln 2$ there is a function f such that random k-CNF formulas with density Δ_k require resolution refutations of size $\exp(f(\Delta_k)n)$ with high probability. In their result, f decays doubly-exponentially as Δ_k increases, which makes their lower bound trivial when $m \geq n \log^{1/4} n$. Later lower bounds by Beame et al. [16] reduce the decay in f to polynomial in Δ_k and, in particular, show that a random k-CNF formula with at most $n^{(k+2)}/4$ clauses requires exponential size resolution refutations. Beame et al. also gave asymptotically matching tree-like resolution upper bounds of size $\exp(n/\Delta_k^{1/(k-2)})$. Similar dependencies on density exist in lower bounds for random k-CNF formulas in other proof systems, such as the Polynomial Calculus [21], k-DNF resolution [3], and Sum-of-Squares [137].

3.2 Monotone Circuits and Communication Proofs

If $x, y \in \mathbb{R}^n$ and for all i we have $x_i \leq y_i$ then we write $x \leq y$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is a *monotone* if $f(x) \leq f(y)$ whenever $x \leq y$. Note that if f is a monotone function and x is a yes instance and y is a no instance, then there must exist $i \in [n]$ such that $x_i > y_i$. This observation is formalized as the search problem known as the *monotone Karchmer-Wigderson Game*.

Monotone Karchmer-Wigderson Game. The monotone Karchmer-Wigderson game associated with a monotone function $f : \{0,1\}^n \to \{0,1\}$, is the search problem $\mathsf{mKW}_f \subseteq f^{-1}(0) \times f^{-1}(1) \times [n]$ defined as $(x, y, i) \in \mathsf{mKW}_f$ if and only if $x_i > y_i$.

A *partial Boolean function* is a Boolean function f defined on a subset of the domain $\{0,1\}^n$. We will denote a partial Boolean function by $f : \{0,1\}^n \to \{0,1,*\}$, where f is undefined on the inputs in $f^{-1}(*)$ (i.e., we do not care what the values of f are on these inputs). A partial Boolean function f is monotone if it can be extended to a total monotone Boolean function by specifying the values of f on $f^{-1}(*)$. That is, there exists a total monotone Boolean function g such that $f(x) = g(x)$ for all $x \in \{0,1\}^n \setminus f^{-1}(*)$.

Monotone Circuit. A monotone circuit is a fanin-2 Boolean circuit using only $\wedge$ and $\vee$ gates. That is, it is a directed acyclic graph (dag) with a single root node, one leaf for each of the inputs $z_1, \ldots, z_n$, as well as the constants 0 and 1. The root node of the dag is the *output* of the circuit, and a monotone circuit computes a monotone Boolean function f if the output of the circuit on input $z \in \{0,1\}^n$ matches the value $f(z)$. If f is partial, then we only require that the monotone circuit matches the value of $f(z)$ on all $z \in \{0,1\}^n \setminus f^{-1}(*)$. The *size* of a monotone circuit is the number of gates in the circuit.

Motivated by proof complexity, Pudlák introduced monotone *real* circuits.

Monotone Real Circuit. A monotone real circuit is a generalization of a monotone circuit in which every non-leaf node is labelled with a function $\phi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ which is monotone non-decreasing in its arguments.

One of the main results of this chapter is an *equivalence* between these models of monotone circuits and certain semantic proofs where the lines are computed by low-depth communication protocols. These were introduced by Krajíček in the study of feasible interpolation [98], and are defined next.

Communication Proofs. Let F be an unsatisfiable CNF formula and let (X, Y) be any partition of the variables of F. A *semantic refutation* is a sequence of lines $L_1, \ldots, L_s$ such that the fine line L_s is the 0 function, and each each line $L_i : \{0,1\}^n \to \{0,1\}$ is either a clause of F or is deduced by the following *semantic deduction rule*:

- *Semantic Deduction.* From L_i and L_j deduce L_k if $L_k(x) = 1$ for every $x \in \{0,1\}^n$ for which both $L_i(x) = 1$ and $L_j(x) = 1$.

A CC_d *refutation* of F with respect to the partition (X, Y) is a semantic refutation $L_1, \ldots, L_s$ such that each L_i can be computed by a d-round deterministic communication protocol with respect to the partition (X, Y). Similarly, an RCC_d *refutation* with respect to (X, Y) is a semantic refutation $L_1, \ldots, L_s$ such that every line L_i can be computed by a d-round real communication protocol.

An example of a CC proof is given below in Example 3.4.3.

Any linear inequality $ax + by \geq d$ whose *weight* $w := \|a\|_1 + \|b\|_1$ can be computed by a $O(\log w)$-round deterministic protocol, and a single round real communication protocol. It follows that a Cutting Planes refutation is also an RCC_1 refutation.

Proposition 3.2.1. *Let F be any unsatisfiable CNF formula and (X, Y) be any partition of its variables. Any CP refutation of F in which every inequality has at weight at most w is a $\mathsf{CC}_{O(\log w)}$-refutation; in particular, any CP^* refutation is a $\mathsf{CC}_{n^{O(\log n)}}$ refutation. Similarly, any CP refutation is an RCC_1 refutation.*

3.3 The Monotone CSP-SAT Function

In this section we introduce mCSP-SAT, which is a monotone variant of SAT that plays a central role in our results. Given any unsatisfiable CNF formula F and partition (X, Y) of F's variables we then show how to

produce a corresponding collection of instances of mCSP-SAT. More precisely: for each assignment $x \in \{0,1\}^{|X|}$ to the X variables we will obtain an accepting instance of mCSP-SAT, and for each assignment $y \in \{0,1\}^{|Y|}$ to the Y variables we will obtain a rejecting instance of mCSP-SAT. In the next section, we will show that separating these mCSP-SAT instances by a monotone boolean circuit is *equivalent* to refuting F in the CC proof system with respect to the partition (X, Y) (and we show a similar result for real circuits and RCC_1 refutations). The mCSP-SAT problem has appeared in many different guises in different works — the function essentially appears in the work of Raz and McKenzie [129] under a different name, and it has re-appeared in recent work on lifting theorems in communication complexity [33, 79].

In order to define mCSP-SAT we first introduce a very general form of the boolean constraint satisfaction problem.

Constraint Satisfaction Problem. A *constraint satisfaction problem* (CSP) $\mathcal{H}$ is defined as follows. Let $H = (L \cup R, E)$ be a bipartite graph and let $n = |R|$. The vertices in L represent the *constraints* of the CSP $\mathcal{H}$, and the vertices in R represent boolean valued *variables*. For each $i \in L$ we let $\mathsf{Vars}(i) \subseteq R$ denote the neighbourhood of i and we associate a boolean function $\mathrm{TT}_i : \{0,1\}^{|\mathsf{Vars}(i)|} \to \{0,1\}$ called the *truth table of* i that encodes the set of satisfying assignments to the ith constraint. The CSP $\mathcal{H}$ *accepts* an assignment $\alpha \in \{0,1\}^{|R|}$ if $\mathrm{TT}_i(\alpha \restriction \mathsf{Vars}(i)) = 1$ for all i, and it is *satisfiable* if it accepts some assignment.

To define a monotone variant of the CSP problem, we will use the fact that adding additional constraints to a CSP can only make it less satisfiable. The mCSP-SAT problem is defined by simply fixing the underlying constraint graph H and letting the input string specify each of the truth tables TT_i.

Monotone CSP-SAT. Let $H = (L \cup R, E)$ be a bipartite graph and let $N := \sum_{i \in L} 2^{|\mathsf{Vars}(i)|}$. The boolean function $\text{mCSP-SAT}_H : \{0,1\}^N \to \{0,1\}$ is defined as follows. An input $z \in \{0,1\}^N$ encodes a CSP $\mathcal{H}_z$ by specifying for each vertex $i \in L$ its truth table $\mathrm{TT}_i : \{0,1\}^{|\mathsf{Vars}(i)|} \to \{0,1\}$. For any $z \in \{0,1\}^N$, $\text{mCSP-SAT}_H(z) = 1$ if and only if the CSP $\mathcal{H}_z$ encoded by z is satisfiable.

Observe that this is a monotone Boolean function since for any $z, z' \in \{0,1\}^N$ with $z \leq z'$ (that is, $z_i \leq z'_i$ for every $i \in [N]$), any satisfying assignment for the CSP $\mathcal{H}_z$ is also a satisfying assignment for the CSP $\mathcal{H}_{z'}$. This is because z and z' both encode sets of truth tables, and so flipping any bit from 0 to 1 simply makes one of the constraints easier to satisfy. That is, we can always extend a partial mCSP-SAT_H instance to a total monotone Boolean function.

Let $F = C_1 \wedge \ldots \wedge C_m$ be an unsatisfiable k-CNF and let (X, Y) be any partition of the variables of F into two sets. Let $H = H(F, X)$ denote the constraint graph of F *restricted to the* X *variables*. That is, H has a left-vertex for each constraint of F and a right vertex for each $x \in X$, and there is an edge (C_i, x) if x appears (either positively or negatively) in C_i (see Example 3.3.1). Consider mCSP-SAT_H, which is a boolean function on N boolean variables. Define sets of accepting and rejecting instances of mCSP-SAT_H from F as follows.

Accepting Instances $\mathcal{A}$. For any $x \in \{0,1\}^{|X|}$ define $\mathcal{A}(x) \in \{0,1\}^N$ as follows. For each $i \in [m]$ and each $\alpha \in \{0,1\}^{\mathsf{Vars}(i)}$ set $TT_i(\alpha) = 1$ iff $x \restriction \mathsf{Vars}(i) = \alpha$.

Rejecting Instances $\mathcal{R}$. For any $y \in \{0,1\}^{|Y|}$ define $\mathcal{R}(y)$ as follows. For each $i \in [m]$ and each $\alpha \in \{0,1\}^{|\mathsf{Vars}(i)|}$ set $\mathrm{TT}_i(\alpha) = 1$ iff $C_i(\alpha, y) = 1$.

Note that if a clause C_i does not depend on any variable in X then we have a single variable TT_i, and $\mathcal{A}(x)$ sets $\mathrm{TT}_i = 1$ and $\mathcal{R}(y)$ sets $\mathrm{TT}_i = 1$ iff $C_i(y) = 1$. We will write mCSP-SAT_F to mean the

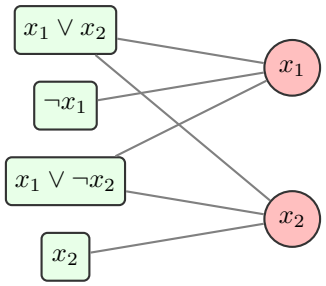
partial monotone boolean function corresponding to the above set of accepting and rejecting instances for the constraint graph $H = H(F, X)$.

Observe that accepting and rejecting inputs to $\mathrm{mCSP\text{-}SAT}_H$ have the following structure. The CSP $\mathcal{H}_{\mathcal{A}(x)}$ corresponding to $\mathcal{A}(x)$ has each truth table TT_i set to 0 everywhere except for exactly one 1 (except when C_i does not depend on X) corresponding to x, and it follows that $\mathcal{H}_{\mathcal{A}(x)}$ has x as its unique satisfying assignment. In particular, $\mathcal{H}_{\mathcal{A}(x)}$ is satisfiable and so it is an accepting instance of $\mathrm{mCSP\text{-}SAT}$. On the other hand, the CSP $\mathcal{H}_{\mathcal{R}(y)}$ corresponding to $\mathcal{R}(y)$ is exactly $F(x, y)$ (note the y variables are fixed); since F is an unsatisfiable CNF formula it follows that $\mathcal{H}_{\mathcal{R}(y)}$ is also unsatisfiable and so it is a rejecting instance of $\mathrm{mCSP\text{-}SAT}$. We give a detailed example next.

Example 3.3.1. Consider the unsatisfiable CNF formula

$$F := (x_1 \vee x_2 \vee y_1) \wedge (\neg x_1) \wedge (x_1 \vee \neg x_2) \wedge (x_2 \vee \neg y_1)$$

with the obvious partition into x– and y–variables. The underlying constraint graph of $\mathrm{mCSP\text{-}SAT}_F$ is depicted below — note that we only keep the x variables from the underlying CNF formula.



Consider the truth assignment $x = (1, 1)$ and $y = (1)$. The accepting input $\mathcal{A}(x)$ corresponding to x has $\mathrm{TT}_i(\alpha) = 1$ iff $\alpha = (1, 1)$; that is each constraint of TT_i is $x_1 \wedge x_2$. Note that this is satisfiable by the truth assignment $(1, 1)$. The rejecting input input $\mathcal{R}(y)$ is obtained by substituting $y = 1$ into each constraint of F and writing down the resulting truth table. That is, $\mathcal{R}(y)$ is given by the following truth tables $\mathrm{TT}_1 = 1$, $\mathrm{TT}_2 = \neg x_1$, $\mathrm{TT}_3 = x_1 \vee \neg x_2$, and $\mathrm{TT}_4 = x_2$; observe that these constraints are unsatisfiable.

3.3.1 The Relationship of Monotone CSPs to Unsatisfiability Certificates

Concurrently and independently to our work, Hrubeš and Pudlák [86] proved that Cutting Planes proofs could be reduced to monotone real circuits computing an associated function which they call an *unsatisfiability certificate*. In this section we explore how $\mathrm{mCSP\text{-}SAT}$ relates to unsatisfiability certificates. The content of this section will not be needed for the rest of this chapter and therefore, to maintain momentum, we recommend skipping this section on a first reading.

Unsatisfiability Certificate. Let $F := C_1 \wedge \ldots \wedge C_m$ be an unsatisfiable CNF formula and (X, Y) be a partition of its variables. The unsatisfiability certificate $\mathsf{cert}_F : \{0,1\}^m \to \{0,1\}$ is defined as follows:

$$\mathsf{cert}_F(z) := \begin{cases} 1 & \text{if the CNF formula } \{C_i^X : z_i = 0\} \text{ is satisfiable} \\ 0 & \text{if the CNF formula } \{C_i^Y : z_i = 1\} \text{ is satisfiable} \\ * & \text{otherwise,} \end{cases}$$

where C_i^X is the clause defined by removing all of the Y-variables from C_i, and similarly for C_i^Y.

Although cert_F is cosmetically dissimilar to $\mathrm{mCSP\text{-}SAT}$, the latter can actually be viewed as an extension of the former. Formally, we say that a (partial) function $g : \{0,1,*\}^m \to \{0,1\}$ is *embedded* in a function $f : \{0,1\}^N \to \{0,1\}$ if there is a projection $\pi : \mathsf{Vars}(f) \to \mathsf{Vars}(g)$ (i.e., a π^{-1} is a mapping of the variables of g to variables of f) such that for every $z \in \{0,1\}^n \setminus g^{-1}(*)$ there is $z^* \in \{0,1\}^N$ such that $\pi(z^*) = z$ and $g(z) = f(z^*)$. Thus, we can think of cert_F as being embedded within $\mathrm{mCSP\text{-}SAT}$.

Claim 3.3.2. For any unsatisfiable CNF formula F and partition (X, Y) of the variables, cert_F is embedded in $\mathrm{mCSP\text{-}SAT}_F$.

Proof. Let F be an unsatisfiable CNF formula and (X, Y) be any partition of its variables. First, define a projection $\pi : \mathsf{Vars}(\mathrm{mCSP\text{-}SAT}_F) \to \mathsf{Vars}(\mathsf{cert}_F)$ which maps the variable $\mathrm{TT}_i(\alpha)$, for which $C_i^X(\alpha) = 0$, of $\mathrm{mCSP\text{-}SAT}_F$ to the variable z_i of cert_F.

Let $z \in \{0,1\}^m \setminus \mathsf{cert}_F^{-1}(*)$ and define z^* as follows: for each $i \in [m]$, if $z_i = 1$ then z^* sets $\mathrm{TT}_i(\alpha)$ to 1 for all $\alpha \in \{0,1\}^{|\mathsf{Vars}(i)|}$ (i.e., the truth table of the ith constraint is identically 1). Otherwise, if $z_i = 0$ then z^* encodes the truth table of the clause C_i^X on the TT_i variables. That is, z^* sets $\mathrm{TT}_i(\alpha)$ to 1 if $C_i^X(\alpha) = 1$, and to 0 otherwise. Observe that $\pi(z^*) = z$.

We claim that for every $z \in \{0,1\}^m \setminus \mathsf{cert}_F^{-1}(*)$, $\mathsf{cert}_F(z) = \mathrm{mCSP\text{-}SAT}_F(z^*)$. Suppose that $\mathsf{cert}_F(z) = 1$, then $\{C_i^X : z_i = 0\}$ is satisfiable by some assignment $\gamma \in \{0,1\}^n$. We claim that γ is also a satisfying assignment for the CSP encoded by z^*. Indeed, every constraint of the CSP with index i for which $z_i = 0$ is satisfied by exactly the same assignments as C_i^X, and the constraints with indices i for which $z_i = 1$ are identically 1. Therefore, $\mathrm{mCSP\text{-}SAT}_F(z^*) = 1$. Otherwise, if $\mathsf{cert}_F(z) = 0$ then $\{C_i^Y : z_i = 1\}$ is satisfiable. Because F is unsatisfiable, this means that $\{C_i^X : z_i = 0\}$ must be unsatisfiable. It follows that the CSP encoded by z^* is unsatisfiable and $\mathrm{mCSP\text{-}SAT}_F(z^*) = 0$. □

To prove the lower bound on random CNF formulas, Hrubeš and Pudlák use the following sets of accepting and rejecting instances of cert_F:

- For every $x \in \{0,1\}^{|X|}$ define an accepting instance $\mathcal{U}(x) \in \{0,1\}^m$ by $\mathcal{U}_i(x) = 1$ iff $C_i^X(x) = 0$.
- For every $y \in \{0,1\}^{|Y|}$ define a rejecting instance $\mathcal{V}(y) \in \{0,1\}^m$ by $\mathcal{V}_i(y) = 0$ iff $C_i^Y(y) = 0$.

These can be seen as monotone projections of our accepting and rejecting instances. Indeed, for $i \in [m]$, let $\alpha \in \{0,1\}^{|\mathsf{Vars}(i)|}$ be the unique falsifying assignment to C_i^X. Then, for any $(x, y) \in \{0,1\}^{|X|} \times \{0,1\}^{|Y|}$ it holds that $\mathcal{A}(x)_{\mathrm{TT}_i(\alpha)} = \mathcal{U}(x)_i$ and $\mathcal{R}(y)_{\mathrm{TT}_i(\alpha)} = \mathcal{V}(y)_i$. Thus, the accepting and rejecting instances of Hrubeš and Pudlák can be obtained by projecting ours onto variables $\mathrm{TT}_i(\alpha)$. Similarly, our accepting instances $\mathcal{A}(x)$ can be obtained from theirs by setting $\mathrm{TT}_i(\alpha)$ to be $\mathcal{U}(x)_i$ and $\mathrm{TT}_i(\beta) = 0$ for $\beta \neq \alpha$. Our rejecting instances can be obtained similarly, where instead we set $\mathrm{TT}_i(\beta) = 1$ for $\beta \neq \alpha$.

3.4 Equating Monotone Circuits and Communication Proofs

In this section we prove the equivalence between CC_d-proofs and monotone circuits, as well as RCC_1-proofs and monotone *real* circuits.

Because every Cutting Planes line can be computed by a single-round real communication protocol (Proposition 3.2.1), the equivalence between RCC_1 proofs and monotone circuits implies that for any family of formulas F and for any partition of the underlying variables into (X, Y), a Cutting Planes refutation of F can be converted into a similar size monotone real circuit for separating the accepting and rejecting instances $\mathcal{A}(\{0,1\}^{|X|}), \mathcal{R}(\{0,1\}^{|Y|})$ of $\mathrm{mCSP\text{-}SAT}_F$. Similarly, because linear inequalities with coefficients

which are polynomially bounded in magnitude can be computed by $O(\log n)$-bit deterministic communication protocols (Proposition 3.2.1), the equivalence between CC_d proofs and monotone circuits implies that any Cutting Planes proof of F in which the coefficients are bounded in magnitude by $O(\log n)$ can be converted into into a similar sized monotone circuit separating $\mathcal{A}(\{0,1\}^{|X|})$ and $\mathcal{R}(\{0,1\}^{|Y|})$ of $\mathrm{mCSP\text{-}SAT}_F$. Thus, lower bounds on the size of monotone real circuits and monotone circuits give lower bounds on the size of Cutting Planes proofs and bounded-coefficient Cutting Planes respectively.

3.4.1 Monotone Circuits and CC Proofs

Our argument relating CC_d and monotone circuits is a direct generalization of the main theorem of Bonet, Pitassi, and Raz [30], which establishes the equivalence for the special case of the clique-coclique formulas. A similar argument of this type also appears in the work of Razborov [130]; Razborov's work was recently simplified by Sokolov [143].

Theorem 3.4.1. *Let F be an unsatisfiable CNF formula on n variables and let (X, Y) be any partition of the variables. For any positive integer d, there exists a CC_d-refutation of F with respect to the partition (X, Y) of size s, then there is a monotone circuit separating the accepting and rejecting inputs $\mathcal{A}(\{0,1\}^{|X|})$ and $\mathcal{R}(\{0,1\}^{|Y|})$ of size $O(2^{3d}s)$.*

The proof of this theorem can be seen as compresses two proofs into one: the first is a reduction from CC_d proofs to CC_1 proofs which incurs a 2^{3d} blowup in size, and the second is a conversion of CC_1 proofs into monotone circuits.

First, we give a sketch of the argument. From a CC_d-proof we will construct a monotone circuit inductively starting with the clauses of F and progressing to the final line. Roughly, for each line L in the proof we will construct a circuit $\mathcal{C}^L$ satisfying the following property: if L is falsified by an assignment (x, y) then $\mathcal{C}^L$ "separates" $\mathcal{A}(x)$ and $\mathcal{R}(y)$, meaning that $\mathcal{C}^L(\mathcal{A}(x)) = 1$ and $\mathcal{C}^L(\mathcal{R}(y)) = 0$. To construct $\mathcal{C}^L$ we will use the soundness of the proof. If L was derived from L' and L'' then, by induction, we will have constructed circuits $\mathcal{C}^{L'}$ and $\mathcal{C}^{L''}$. By soundness, every assignment (x, y) that falsifies L will falsify at least one of L' and L'', and so at least one of the corresponding circuits $\mathcal{C}^{L'}$ and $\mathcal{C}^{L''}$ will separate $\mathcal{A}(x)$ and $\mathcal{R}(y)$. Using this, we will construct $\mathcal{C}^L$ from the circuits $\mathcal{C}^{L'}$ and $\mathcal{C}^{L''}$. Once we arrive at the final line of the proof, because every truth assignment falsifies $0 \geq 1$, the corresponding circuit will separate $\mathcal{A}$ and $\mathcal{R}$.

More concretely, because each line in the CC_d-proof can be computed by a small communication protocol, this induces a partition of the communication matrix of L into at most 2^d monochromatic rectangles. Instead of constructing only a single circuit for each line L, we will actually construct one for every 0-monochromatic rectangle R of L (those containing inputs that falsify L), which will separate $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for every $(x, y) \in R$.

Proof. Let $F = C_1 \wedge \ldots \wedge C_m$ be an unsatisfiable CNF formula over variables (X, Y). Fix a CC_d refutation of F with s lines, where each line is either a clause of F, or follows semantically from two earlier lines. For every line L in the proof, because it can be computed by a d-round communication protocol, there are at most 2^d possible histories h, each with an associated monochromatic rectangle $R_L(h)$. Recall that each monochromatic rectangle is a subset of assignments that have the same evaluation under L. We call a history h *good* for L if $R_L(h)$ is 0-monochromatic. That is, a good history is one for which every assignment in the associated monochromatic rectangle falsifies L.

We build the circuit for $\mathrm{mCSP\text{-}SAT}_F$ separating $\mathcal{A}$ and $\mathcal{R}$ by induction on the lines in the proof. For each line L we construct a collection of circuits $\{\mathcal{C}_h^L\}$ for every good history h for L, such that the following hold:

(i) For a good history h for L, the circuit $\mathcal{C}_h^L$ correctly "separates" x and y for each $(x, y) \in R_L(h)$, meaning that $\mathcal{C}_h^L$ outputs 1 on $\mathcal{A}(x)$ and 0 on $\mathcal{R}(y)$.

(ii) If L is the tth line in the proof, then this collection of circuits for L (which may share internal gates) uses at most $2^{3d}t$ gates altogether.

Since every assignment falsifies the final line of the proof $0 \geq 1$, the associated monotone circuit will separate $\mathcal{A}$ from $\mathcal{R}$.

If L is a leaf of the proof then L is a clause C_i of F. The communication protocol for L is the following two bit protocol: Alice and Bob each send a 0 iff their respective inputs do not satisfy C_i. Thus, there is only a single good history $h = (0, 0)$ for L. Because C_i is a clause, there is exactly one pair of assignments $\alpha \in \{0,1\}^{\mathsf{Vars}(i)\cap X}$ and $\beta \in \{0,1\}^{\mathsf{Vars}(i)\cap Y}$ to the variables of C_i such that $C_i(\alpha, \beta) = 0$. Define the circuit $\mathcal{C}_h^L$ corresponding to line $L = C_i$ and good history $h = 00$ to be the variable $\mathrm{TT}_i(\alpha)$.

Suppose that L is derived from L_1 and L_2 and h is a good history for L. We will construct the circuit $\mathcal{C}_h^L$ using the circuits $\mathcal{C}_{h'}^{L_1}, \mathcal{C}_{h''}^{L_2}$ that we have constructed by induction, along with at most 2^{3d} additional gates. Let (x, y) be any assignment such that the protocol for L follows the history h. By the soundness of the proof, every assignment falsifying L also falsifies L_1 or L_2. That is, the 0-monochromatic rectangles of L are contained within the union of the 0-monochromatic rectangles of L_1 and L_2. It follows that (x, y) must lie within a 0-monochromatic rectangle of L_1 or L_2, and we can use the d-round communication protocols for L_1 and L_2 to find this rectangle. We will then convert this protocol into a circuit which decides which of the circuits $\mathcal{C}_{h'}^{L_1}, \mathcal{C}_{h''}^{L_2}$ will separate the pair of inputs $\mathcal{A}(x)$ and $\mathcal{R}(y)$.

Construct a *stacked* protocol tree, corresponding to first running the communication protocol for L_1 and then running the communication protocol for L_2. This will give us a height $2d$ binary tree, T, where the top part is the communication protocol tree for L_1, with protocol trees for L_2 attached to each of the leaves (Figure 3.1). Consider a path labelled h_1h_2 in T, where h_1 is the history from running L_1 and h_2 is the history from running L_2. Since $R_L(h)$ is 0-monochromatic (with respect to the communication matrix for L), by soundness, for every $(x', y') \in R_L(h)$ either $L_1(x', y') = 0$ or $L_2(x', y') = 0$. Because $R_{L_1}(h_1)$ and $R_{L_2}(h_2)$ are monochromatic rectangles, it follows that either

(i) the rectangle $R_{L_1}(h_1) \cap R_L(h)$ is non-empty and 0-monochromatic (with respect to the communication matrix of L_1), or

(ii) the rectangle $R_{L_2}(h_2) \cap R_L(h)$ is non-empty and 0-monochromatic (with respect to the communication matrix of L_2).

(iii) $R_{L_1}(h_1) \cap R_L(h) = \emptyset$ and $R_{L_2}(h_2) \cap R_L(h) = \emptyset$.

In the first case, we will label this leaf with $\mathcal{C}_{h_1}^{L_1}$ and otherwise we will label this leaf with $\mathcal{C}_{h_2}^{L_2}$. The nodes of type (iii) do not contribute to separating $\mathcal{A}$ from $\mathcal{R}$ and so we should remove them. We will do so by repeating the following process: choose the maximal sub-tree whose leaves are all of type (iii) and remove it from T.

Next, we convert this stacked protocol tree into a circuit. Label each internal node v of the stacked tree with a gate: if Alice speaks at v, then we label it with a $\vee$ gate, and otherwise if Bob speaks we label v with a

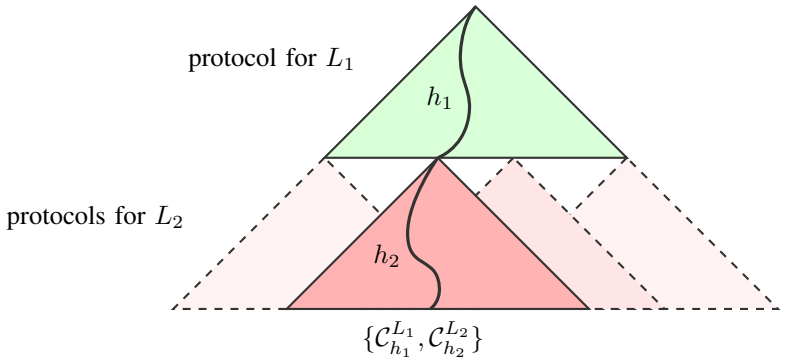


Figure 3.1: Stacked protocol tree T for L.

$\wedge$ gate. finally, we make all fanin-1 nodes that result from removing nodes (iii) into fanin-2 nodes by wiring them to a constant: if this node is labelled with a $\vee$ then create an additional incoming edge from the constant 0 and if it is labelled with a $\wedge$, create an additional incoming edge from the constant 1. Example 3.4.3 at the end of this subsection gives an example of this construction for a simple CC proof.

The monotone circuit for the history h that results from this process has 2^{2d} additional gates. There are at most 2^d histories for L and therefore, the circuits constructed for L have size at most 2^{3d} plus the size of the circuits for L_1 and L_2, which by induction is at most $2^{3d}(t-1)$ if L is the tth line in the proof. The correctness of this construction follows from the next claim.

Claim. The monotone circuit resulting from the above construction satisfies: for every line L in the proof, and each good history h for L, the circuit $\mathcal{C}^L_h$ will be correct for all $(x,y) \in R_L(h)$. That is, $\mathcal{C}^L_h(\mathcal{A}(x)) > \mathcal{C}^L_h(\mathcal{R}(y))$ for every $(x,y) \in R_L(h)$.

Proof of Claim. If L is a leaf of the proof then L is a clause C_i of F. As mentioned above, the communication protocol for C_i has exactly one 0-monochromatic rectangle, which is associated with the good history $h = (0,0)$. If $(x,y) \in R_L(h)$ then $C_i(x,y) = 0$ by definition. Let $\alpha = x \restriction \mathsf{Vars}(C_i)$. In our construction the circuit corresponding to $\mathcal{C}^L_h$ is labelled by the variable $\mathrm{TT}_i(\alpha)$, and it is easy to check that $\mathcal{A}(x)$ sets $\mathrm{TT}_i(\alpha)$ to true, and $\mathcal{R}(y)$ sets $\mathrm{TT}_i(\alpha)$ to false.

If L is not a leaf, then to prove the claim we will prove the following stronger statement by induction: for every line L derived from previous lines L_1 and L_2, and for each node v in the stacked protocol tree for L, corresponding to some (sub)history $h' = h_1h_2$, the subcircuit $\mathcal{C}^L_{h'}$ associated with v separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x,y) \in R_L(h) \cap R_{L_1}(h_1) \cap R_{L_2}(h_2)$. To see that this stronger claim implies our main claim, observe that once we reach $h' = \emptyset$, $C^L_{h'}$ will be correct on $(x,y) \in R_L(h) \cap R_{L_1}(h_1) \cap R_{L_2}(h_2) = R_L(h)$. This follows because if Alice and Bob haven't communicated (i.e., $h_i = \emptyset$) then their current rectangle is the entire communication matrix (i.e., $R_{L_i}(h_i) = \{0,1\}^{|X|} \times \{0,1\}^{|Y|}$).

For the base case, suppose that v is a leaf of the stacked protocol tree for L with history $h' = h_1h_2$. Then, either case (i), case (ii), or case (iii) above holds. In case (i) we labelled v by $\mathcal{C}^{L_1}_{h_1}$. Because $R_{L_1}(h_1) \cap R_L(h)$ is a 0-monochromatic rectangle and $R_{L_1}(h_1)$ is a monochromatic rectangle, it follows that $R_{L_1}(h_1)$ must be 0-monochromatic. By induction $\mathcal{C}^{L_1}_{h_1}$ is defined and separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x,y) \in R_{L_1}(h_1)$, and so it separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x,y) \in R_L(h) \cap R_{L_1}(h_1) \cap R_{L_2}(h_2)$. A similar argument holds in case (ii). For case (iii), $R_{L_1}(h_1) \cap R_L(h) = \emptyset$ and $R_{L_2}(h_2) \cap R_L(h) = \emptyset$ and so this case holds vacuously.

For the inductive step, let v be a non-leaf node in the protocol tree with history h'. Assume that Alice

owns v, and so v is labelled with an $\vee$ gate. The rectangle $R_L(h) \cap R_{L_1}(h_1) \cap R_{L_2}(h_2) = A \times B$ is partitioned into $A_0 \times B$ and $A_1 \times B$, where

- $A = A_0 \cup A_1$,
- $A_0 \times B$ is the rectangle with history $h'0$,
- $A_1 \times B$ is the rectangle with history $h'1$.

Fix any $(x, y) \in R_L(h) \cap R_{L_1}(h_1) \cap R_{L_2}(h_2)$. Since by induction $\mathcal{C}^L_{h'0}$ separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x, y) \in A_0 \times B$ and $\mathcal{C}^L_{h'1}$ separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x, y) \in A_1 \times B$, then it follows that $\mathcal{C}^L_{h'} = \mathcal{C}^L_{h'0} \vee \mathcal{C}^L_{h'1}$ separates $\mathcal{A}(x)$ and $\mathcal{R}(y)$ for all $(x, y) \in A \times B$. To see this, observe that if $x \in A_0$, then $\mathcal{C}^L_{h'0}(\mathcal{A}(x)) = 1$ and therefore

$$\mathcal{C}^L_{h'}(\mathcal{A}(x)) = \mathcal{C}^L_{h'0}(\mathcal{A}(x)) \vee \mathcal{C}^L_{h'1}(\mathcal{A}(x)) = 1.$$

The same applies when $x \in A_1$, as then $\mathcal{C}^L_{h'1}(\mathcal{A}(x)) = 1$. If $y \in B$ then both $\mathcal{C}^L_{h'0}(\mathcal{R}(y)) = \mathcal{C}^L_{h'1}(\mathcal{R}(y)) = 0$ and therefore

$$\mathcal{C}^L_{h'}(\mathcal{R}(y)) = \mathcal{C}^L_{h'0}(\mathcal{R}(y)) \vee \mathcal{C}^L_{h'1}(\mathcal{R}(y)) = 0.$$

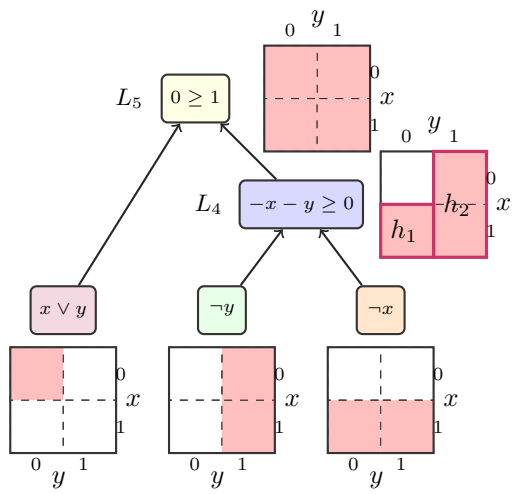
A similar argument holds in the case that v is an internal node of the protocol tree that is owned by Bob (and is therefore labelled with a $\wedge$ gate). □

Next, we establish the converse direction. Although this reduction from monotone circuits to CC_2 proofs is not necessary in order to establish our Cutting Planes lower bound, we believe the equivalence between monotone circuits and $\mathsf{CC}_{O(\log n)}$-proofs to be of independent interest.

Theorem 3.4.2. *If there is a monotone circuit of size s separating the accepting and rejecting inputs $\mathrm{mCSP\text{-}SAT}_F$, then there is a CC_2-refutation of F of size s with respect to the same variable partition.*

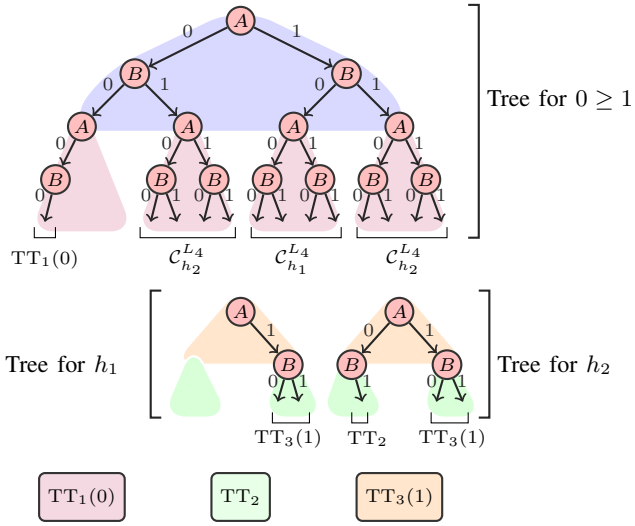
Proof. The topology of the proof will be in one-to-one correspondence with the directed acyclic graph of the monotone circuit. Relabel each input variable $\mathrm{TT}_i(\alpha)$ of the circuit with the corresponding clause C_i, and note that C_i is falsified by exactly those (x, y) for which the $\mathrm{TT}_i(\alpha)$ entry of $\mathcal{A}(x)$ is 1 and $\mathcal{R}(y)$ is 0. Relabel each internal gate v with the function corresponding to the following 2-bit protocol. On input x, Alice privately runs the circuit on input $\mathcal{A}(x)$ and sends the value A_v computed by the circuit at gate v to Bob. Analogously, Bob simulates the circuit privately on input $\mathcal{R}(y)$ and sends the value B_v computed at gate v to Alice. They output 0 if and only if $A_v = 1$ and $B_v = 0$. Since the root gate of the circuit is identically 1 on $\mathcal{A}(x)$ and 0 on $\mathcal{R}(y)$ for all x and y, the referee always outputs 0 at the last line of the refutation. It remains to verify that the refutation is sound. Let v be a node in the proof with children u_1 and u_2. If v is a $\vee$ gate, then $A_v = A_{u_1} \vee A_{u_2}$ and $B_v = B_{u_1} \vee B_{u_2}$. If the line at v is falsified on (x, y) then $A_v = 1$ and $B_v = 0$. Because v is a $\vee$ gate, it follows that either $A_{u_1} = 1$ or $A_{u_2} = 1$ and $B_{u_1} = B_{u_2} = 0$. Thus, the line at u_1 or the line at u_2 is falsified by (x, y). An analogous argument shows that the proof is sound when v is a $\vee$ gate. □

Example 3.4.3. As a simple example, we show how to convert a CC_2 proof of the CNF formula $F = (x \vee y) \wedge (\neg x) \wedge (\neg y)$ into a monotone circuit separating the sets $\mathcal{A}$ and $\mathcal{R}$ of $\mathrm{mCSP\text{-}SAT}_F$. The CC_2 proof of F that we will translate is the sequence of lines $(x \vee y), (\neg x), (\neg y), (-x - y \geq 0), (0 \geq 1)$. This is shown next, along with the communication matrices of each of the lines in the proof. The 0-monochromatic rectangles (good histories) are labelled in red.



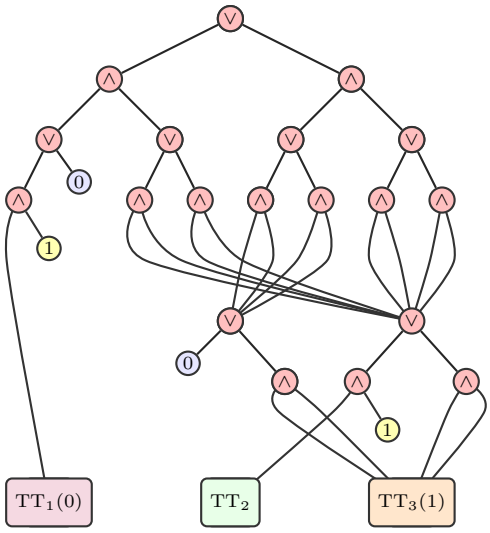
Observe that each of the lines in the proof has only a single 0-monochromatic rectangle (marked in red) and therefore only a single good history, except for the line $-x - y \geq 0$ which has two good histories, h_1 and h_2.

Next, we give stacked protocol trees for each of the good histories of each of the lines, after removing all subtrees whose leaves are all nodes of type (iii).



The line $L_4 := -x - y \geq 0$ has two good histories, and thus two stacked protocol trees, while the remaining lines have only a single good history. The colored areas behind each of the protocols match in color to the lines in the proof for which they are protocols for.

The monotone circuit that results from this transformation, along with the accepting inputs $\mathcal{A}$ and rejecting inputs $\mathcal{R}$, is given next.



	$\mathcal{A}(0)$	$\mathcal{A}(1)$	$\mathcal{R}(0)$	$\mathcal{R}(1)$
$\mathrm{TT}_1(0)$	1	0	0	1
$\mathrm{TT}_1(1)$	0	1	1	1
TT_2	1	1	1	0
$\mathrm{TT}_3(0)$	1	0	1	1
$\mathrm{TT}_3(1)$	0	1	0	0

3.4.2 Monotone Real Circuits and RCC Proofs

We now give a formal statement of Theorem 3.4.4.

Theorem 3.4.4. *Let F be an unsatisfiable CNF formula and let (X, Y) be any partition of the variables. There exists an RCC_1 refutation of F with respect to the partition (X, Y) if and only if there exists a monotone real circuit separating $\mathcal{A}(\{0,1\}^{|X|}$ from $\mathcal{R}(\{0,1\}^{|Y|})$ of the same size.*

We will split this theorem into two lemmas. Lemma 3.4.6 is the difficult direction, translating RCC_1 refutations of F into monotone real circuits for $\mathrm{mCSP\text{-}SAT}$. Lemma 3.4.7 shows a converse, and is a simple direct argument analogous to Theorem 3.4.2. As mentioned in Section 3.2, Lemma 3.4.6 follows from the proof of the following result of Hrubeš and Pudlák [86] relating real monotone circuits and certain "dag-like" real communication protocols (also known as triangle dags [72]).

Theorem 3.4.5 (Theorem 5 in [87])**.** *Let f be a monotone Boolean function. Given a dag-like real protocol P solving the monotone Karchmer-Wigderson game for f, there is a monotone real circuit of the same size computing f.*

The formal definition of dag-like real protocols will not be necessary for our technical results, and so we refer the interested reader to [143] for their definition. Our Lemma 3.4.6 states that from an RCC_1 refutation of an unsatisfiable formula F, we can construct a similarly-sized monotone real circuit for the function $\mathrm{mCSP\text{-}SAT}_F$.

Lemma 3.4.6. *Let F be an unsatisfiable CNF formula and let (X, Y) be any partition of the variables. If there is a RCC_1 refutation of F with respect to the partition (X, Y) of size s, then there is a real monotone circuit separating the accepting and rejecting inputs $\mathcal{A}(\{0,1\}^{|X|})$ and $\mathcal{R}(\{0,1\}^{|Y|})$ of $\mathrm{mCSP\text{-}SAT}_F$ with s gates.*

We will give a direct proof of Lemma 3.4.6 which is modelled on the proof of Theorem 3.4.5, but first let us sketch how Lemma 3.4.6 can be obtained using Theorem 3.4.5 as a black box. Let F be an unsatisfiable formula on n variables, let (X, Y) be any partition of these variables, and suppose that F has an RCC_1 refutation. The *search problem* associated with F and variable partition (X, Y) is the following two-party

communication problem: Alice receives an assignment to the variables in X, Bob receives an assignment to the variables in Y, and they want to find and output the index of a clause of F that is falsified by their joint assignment. From an RCC_1 refutation of F, one can extract a dag-like real protocol for solving this search problem; the proof follows standard ideas in the literature transforming communication lower bounds into proof length lower bounds. By combining the reductions appearing in [79, 129], the search problem associated with F is *equivalent* to the monotone Karchmer-Wigderson game associated with $\mathrm{mCSP\text{-}SAT}_F$. Thus, by the above theorem, $\mathrm{mCSP\text{-}SAT}_F$ also has a monotone real circuit of the same size as the refutation.

To prove the other direction of Theorem 3.4.4 (Lemma 3.4.7), we need to translate monotone real circuits computing $\mathrm{mCSP\text{-}SAT}_F$ into RCC_1 refutations for F. This follows by viewing the monotone real circuit as a dag-like real protocol for solving the monotone KW game associated with $\mathrm{mCSP\text{-}SAT}_F$, along with the equivalence between such protocols and dag-like real protocols solving the search problem associated with F; the latter is exactly an RCC_1 refutation of F.

We will now give self-contained proofs of Lemmas 3.4.6 and 3.4.7. The proofs are an adaptation of the argument in [87] to our setting, bypassing the intermediate communication protocols associated with F.

Proof of Lemma 3.4.6. Fix an RCC_1 refutation of F with respect to a partition (X, Y) of the variables. For every node v of the underlying directed acyclic graph of the refutation, associate two functions $A_v : \{0,1\}^{|X|} \to \mathbb{R}$ and $B_v : \{0,1\}^{|Y|} \to \mathbb{R}$ that Alice and Bob use to communicate with the referee at this node. Assume without loss of generality that $B_v(y) \geq 0$ for all y, and also that the referee returns 1 if and only if $A_v(x) > B_v(y)$.

Next, we convert the given proof into a real circuit separating $\mathcal{A}(\{0,1\}^{|X|}$ and $\mathcal{R}(\{0,1\}^{|Y|})$ as follows. The topology of the circuit will be in one-to-one correspondence with the topology of the dag, and we will label the nodes v of the dag by functions f_v as follows. If v is a leaf of the refutation, corresponding to a clause C_i, let α_i be the assignment to the X-variables that does not satisfy the X-part of C_i. We may assume that

$$A_v(x) = \mathcal{A}(x)_{\mathrm{TT}_i(\alpha_i)} \text{ and } B_v(y) = \mathcal{R}(y)_{\mathrm{TT}_i(\alpha_i)}. \tag{3.1}$$

That is, Alice and Bob both send their $\mathrm{TT}_i(\alpha_i)$-th bit of their inputs. Therefore, we label this leaf of the circuit with the input variable $f_v(z) := \mathrm{TT}_i(\alpha_i)(z)$.

For each internal node v with children u_1 and u_2 we associate v with the function f_v defined recursively as follows:

$$f_v(z) := \max_{x \in \{0,1\}^{|X|}} \{A_v(x) | f_{u_1}(z) \geq A_{u_1}(x) \wedge f_{u_2}(z) \geq A_{u_2}(x)\}.$$

We define $f_v(z)$ to be 0 if the set on the right-hand side is empty. We claim that these functions can be computed by monotone real gates and for every $x \in \{0,1\}^{|X|}$ and $y \in \{0,1\}^{|Y|}$ we have

$$f_v(\mathcal{A}(x)) \geq A_v(x) \text{ and } f_v(\mathcal{R}(y)) \leq B_v(y). \tag{3.2}$$

First, let's see how (3.2) implies that the constructed circuit separates $\mathcal{A}(\{0,1\}^{|X|})$ from $\mathcal{R}(\{0,1\}^{|Y|})$. Let r be the root node of the dag. Since we began with a valid RCC_1 refutation of F, for every $x \in \{0,1\}^{|X|}$ and $y \in \{0,1\}^{|Y|}$ we have that $A_r(x) > B_r(y)$. Therefore, $f_r(\mathcal{A}(x)) > f_r(\mathcal{R}(y))$ for all x and y. Finally, modifying f_r by composing it with an appropriately chosen threshold function gives us the separating circuit.

To see that f_v can be computed by a monotone real gate with inputs f_{u_1} and f_{u_2}, we show that f_v is monotone. To see this, observe that the value of f_v is determined by the values of f_{u_1} and f_{u_2}, and increasing the values of either f_{u_1} or f_{u_2} increases the feasible region of xs over which the maximum is taken in the definition of f_v.

It remains to show that $f_v(z)$ satisfies (3.2) for all nodes v in the dag. The base case is given by (3.1). Let v be an internal node and suppose that (3.2) holds for its children u_1 and u_2. In particular, for an arbitrary $x \in \{0,1\}^{|X|}$ we have $f_{u_1}(\mathcal{A}(x)) \geq A_{u_1}(x)$ and $f_{u_2}(\mathcal{A}(x)) \geq A_{u_2}(x)$. Thus, the region over which the maximum is taken in the definition of $f_v(\mathcal{A}(x))$ is taken is nonempty and contains x. It follows that $f_v(\mathcal{A}(x)) \geq A_v(x)$. Next, consider an arbitrary $y \in \{0,1\}^{|Y|}$ and assume for contradiction that $f_v(\mathcal{R}(y)) > B_v(y)$. Since. $B_v(y) \geq 0$, it follows that $f_v(\mathcal{R}(y)) = A_v(x)$ for some $x \in \{0,1\}^{|X|}$. Thus $A_v(x) > B_v(y)$, and by soundness of the refutation it follows that either $A_{u_1}(x) > B_{u_1}(y)$ or $A_{u_2}(x) > B_{u_2}(y)$. Assume without loss of generality that the first holds. Then, by the definition of $f_v(\mathcal{R}(y))$ we have that $f_{u_1}(\mathcal{R}(y)) \geq A_{u_1}(x) > B_{u_1}(y)$, which contradicts the inductive assumption. □

The above lemma proves the first part of Theorem 3.4.4. The next lemma proves the second part of the theorem.

Lemma 3.4.7. *Let F be an unsatisfiable CNF formula and let (X, Y) be an partition of the variables. A monotone real circuit separating the inputs $\mathcal{A}(\{0,1\}^{|X|})$ and $\mathcal{R}(\{0,1\}^{|Y|})$ of $\mathrm{mCSP\text{-}SAT}_F$ implies a RCC_1 refutation of F of the same size.*

Proof. The topology of the RCC_1 refutation that we construct will be in one-to-one correspondence with the monotone real circuit. Relabel each input variable $\mathrm{TT}_i(\alpha)$ of the circuit with the corresponding clause C_i. Relabel each internal gate v in the circuit by the function corresponding to the following RCC_1 protocol. On input x, Alice privately runs the circuit on $\mathcal{A}(x)$ and sends the value A_v computed by the circuit at gate v to the referee. On input y, Bob acts analogously — he simulates the circuit privately on input $\mathcal{R}(y)$ and sends the value B_v computed by the circuit at gate v to the referee. The referee outputs 0 if and only if $A_v > B_v$. Since the root gate of the circuit is identically 1 on $\mathcal{A}(x)$ and 0 on $\mathcal{R}(y)$, the referee always outputs 0 at the last line in the refutation. Thus, the only thing left to verify is that the refutation is sound. Let u_1 and u_2 be the children. Then, $A_v = f(A_{u_1}, A_{u_2})$ and $B_v = f(B_{u_1}, B_{u_2})$ for some monotone function f. By monotonicity, it follows that if $A_v > B_v$ then either $A_{u_1} > B_{u_1}$ or $A_{u_2} > B_{u_2}$. □

3.5 Lower Bounds for Random CNFs

In this section we prove Theorem 1.3.1. In particular, we prove lower bounds for RCC_1 refutations (and therefore Cutting Planes refutations) of uniformly random k-CNF formulas with sufficient clause density.

To prove Theorem 1.3.1 we will combine Theorem 1.3.2 with a lower bound on the monotone real circuit complexity of $\mathrm{mCSP\text{-}SAT}_F$ for a random CNF formula F. To prove this lower bound, we will use the well-known *method of symmetric approximations* [23, 83]. The following formalization of the method is exposited in [93]. First, we introduce some notation: If $A \subseteq \{0,1\}^N$, then for $r \in [N]$ and $b \in \{0,1\}$ define the filter (depicted in Figure 3.2)

$$\#_b(r, A) := \max_{I \subseteq [N] : |I| = r} |\{a \in A | \forall i \in I, a_i = b\}|.$$

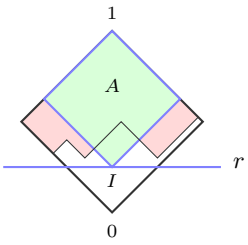


Figure 3.2: The action of $\#_1(r, A)$ on the unit cube. The red area represents the (monotone) set A, and the blue line represents the hamming weight r slice. The intersection of the green and red areas gives the value of $|\{a \in A | \forall i \in I, a_i = 1\}|$

Method of Symmetric Approximations (Theorem 19.9 in [93]). *Let $f : \{0,1\}^N \to \{0,1\}$ be a monotone Boolean function and let $1 \leq r, s \leq N$. Let $A \subseteq f^{-1}(1)$ and $R \subseteq f^{-1}(0)$. Then every monotone real circuit that outputs 1 on all inputs in A and 0 on all inputs in R has size at least*

$$\min\left\{\frac{|A| - (2s)\#_1(1, A)}{(2s)^{r+1}\#_1(r, A)}, \frac{|R|}{(2r)^{s+1}\#_0(s, R)}\right\}.$$

The proof of Theorem 1.3.1 is delayed until Subsection 3.5.2; to get a feeling for the argument, we first prove an easier lower bound for the simpler distribution of *balanced* random CNF formulas.

3.5.1 Balanced Random CNFs

Definition 3.5.1. Let $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$ be two disjoint sets of variables. Let $\mathcal{F}(m, n, k)^{\otimes 2}$ denote the following distribution over $2k$-CNF formulas: first, sample $F^1 = C_1^1 \wedge \ldots \wedge C_m^1$ from $\mathcal{F}(m, n, k)$ on the X variables, and $F^2 = C_1^2 \wedge \ldots \wedge C_m^2$ from $\mathcal{F}(m, n, k)$ on the Y variables independently. Then output

$$F = (C_1^1 \vee C_1^2) \wedge \ldots \wedge (C_m^1 \vee C_m^2).$$

This distribution shares the property with $\mathcal{F}(m, n, k)$ that dense enough formulas are unsatisfiable with high probability.

Lemma 3.5.2. *Let $c > 2/\log e$ and let n be any positive integer. If $k \in [n]$ and $m \geq cn2^{2k}$ then $F \sim \mathcal{F}(m, n, k)^{\otimes 2}$ is unsatisfiable with high probability.*

Proof. For a uniformly random assignment (x, y) to the variables of F, the probability that the ith clause is satisfied by the joint assignment is $1 - 1/2^{2k}$. Thus, the probability that *all* clauses are satisfied by the joint assignment is $(1 - 1/2^{2k})^m \leq 2^{-m/2^{2k}}$, since the clauses are sampled independently. By the union bound, the probability that some joint assignment satisfies the formula is at most $2^{2n}e^{-m/2^{2k}} = 2^{2n-(\log e)m/2^{2k}} \leq 2^{2n-(\log e)cn} \leq 2^{-\Omega(n)}$. □

The main theorem of this section is that $F \sim \mathcal{F}(m, n, k)^{\otimes 2}$ requires large RCC_1-proofs.

Theorem 3.5.3. *Let $k = 4 \log n$ and $m = cn2^{2k}$ where $c > 2/\log e$ is some constant. Let (X, Y) be the variable partition associated with $F \sim \mathcal{F}(m, n, k)^{\otimes 2}$. Then, for $F \sim \mathcal{F}(m, n, k)^{\otimes 2}$, any monotone real circuit separating $\mathcal{A}(\{0,1\}^{|X|})$ and $\mathcal{R}(\{0,1\}^{|Y|})$ has at least $2^{\tilde{\Omega}(n)}$ gates with high probability.*

As an immediate consequence of Theorem 3.5.3 and Theorem 1.3.2 we obtain a lower bound for Cutting Planes.

Corollary 3.5.4. *Let n be a sufficiently large positive integer, and let $k = 4\log n$ and $m = cn^9$, where $c > 2/\log e$ is some constant. With high probability, every RCC_1-refutation (and therefore, CP refutation) of $F \sim \mathcal{F}(m,n,k)^{\otimes 2}$ requires at least $2^{\tilde{\Omega}(n)}$ lines.*

The proof of Theorem 3.5.3 comes down to the fact that random k-CNF formulas are expanders. The next lemma records the expansion properties that we require; the proof is adapted from the monograph of Vadhan [145]. For a subset $S \subseteq F$ of clauses of a CNF formula F let $\mathsf{Vars}(S)$ denote the subset of variables that appear in the clauses S.

Lemma 3.5.5. *Let n be any sufficiently large positive integer. Let k, m be positive integers and $F \sim \mathcal{F}(m,n,k)$. Suppose that for some $0 < \delta < 1$ we have*

$$\log m \leq \delta \left(\frac{k}{2}\right) \log\left(\frac{k}{2}\right),$$

then every set $S \subseteq F$ of size $s \leq n/ek^2$ satisfies $|\mathsf{Vars}(S)| \geq ks/2$ with probability at least $1 - 2^{-(1-\delta)(ks/2)\log(k/2)}$.

Proof. Fix any set $S \subseteq F$ of size s. For each clause $C \in S$ sample the variables in C one at a time without replacement, and let $v_1, v_2, \ldots, v_{ks}$ denote the concatenation of the sequences of sampled variables for all $C \in S$. Say that a variable v_i is a *repeat* if it has already occurred among $v_1, \ldots, v_{i-1}$. In order for $|\,\mathsf{var}(S)| < ks/2$ the concatenated sequence must have at least $ks/2$ repeats. The probability that variable v_i is a repeat is at most $(i-1)/n \leq ks/n$, and this implies that

$$\Pr[|\mathsf{Vars}(S)| < ks/2] \leq \binom{ks}{ks/2}\left(\frac{ks}{n}\right)^{ks/2} \leq \left(\frac{2eks}{ks}\right)^{ks/2}\left(\frac{ks}{n}\right)^{ks/2} \leq \left(\frac{2}{k}\right)^{ks/2}$$

using standard bounds on binomial coefficients and the fact that $s \leq n/ek^2$. Thus

$$Pr[\exists S : |S| = s, |\mathsf{Vars}(S)| < ks/2] \leq m^s \left(\frac{2}{k}\right)^{ks/2},$$

and by assumption $\log m \leq \delta(k/2)\log(k/2)$, completing the proof of the lemma. □

Having established the expansion properties that we will need, we are ready to prove Theorem 3.5.3.

Proof of Theorem 3.5.3. We will apply the Method of Symmetric Approximations to $A := \mathcal{A}(\{0,1\}^{|X|})$ and $R := \mathcal{R}(\{0,1\}^{|Y|})$ (see Section 3.3) with $r = s = n/ek^2$ for $k = 4\log n$ and $m = cn2^{2k}$. Recall that $\mathcal{A}$ and $\mathcal{R}$ are functions mapping inputs to 1-inputs and 0-inputs of $\mathrm{mCSP\text{-}SAT}_F$ respectively. To finish the argument, we need to compute $|A|, \#_1(1,A), \#_1(r,A), |R|, \#_0(s,R)$.

First, bound the size of A and R. By definition of $\mathcal{A}(x)$, we set $\mathrm{TT}_i(\alpha) = 1$ if and only if $x \restriction \mathsf{Vars}(i) = \alpha$; thus, $\mathcal{A}(x) = \mathcal{A}(x')$ for some $x \neq x'$ only if there exists a variable in X that does not appear in any clause. However, it is easy to see that with high probability every variable in X participates in some clause, and thus $\mathcal{A}$ is one-to-one with high probability. Therefore, $|A| = 2^n$ with high probability.

Recall each 0-input $\mathcal{R}(y)$ of $\mathrm{mCSP\text{-}SAT}_F$ is obtained by applying an assignment y to the Y-variables of F and the writing out the truth tables of all of the clauses. The truth tables of the clauses that were satisfied by the Y-assignment are identically 1, and the truth tables of the clauses that were not satisfied by y contain

exactly one 0-entry, because each clause has a unique falsifying assignment. We call the set of clauses that were not satisfied by y the *profile* of y. The next lemma implies that the profiles of all Y-assignments are distinct with high probability.

Lemma 3.5.6. *Let n, m, k be positive integers. Sample $F \sim \mathcal{F}(m, n, k)$, and let $S \subseteq \{0,1\}^n$ be a collection of assignments. Let M be the following $|S| \times m$ matrix, with rows labelled by assignments $\alpha \in S$ and columns labelled by clauses of F. Namely, for any pair (α, i) set*

$$M[\alpha, i] = \begin{cases} 1 & \textit{if the } i\textit{th clause is } \text{not } \textit{satisfied by } \alpha, \\ 0 & \textit{otherwise.} \end{cases}$$

If $\log |S| < km/8n2^k$ then the rows of M are distinct with probability at least $1 - 2^{km/n2^k}$.

Proof. We will think of M as being generated column by column with each column sampled independently. Fix a pair of assignments α and $\hat{\alpha}$ such that $\alpha \neq \hat{\alpha}$. Let S be the set of indices on which the two assignments differ, i.e., $S = \{i : \alpha_i \neq \hat{\alpha}_i\}$, and set $s = |S|$. Letting C_i denote the ith clause, we have

$$\Pr[C_i \text{ not satisfied by } \hat{\alpha}, \text{ satisfied by } \alpha\,] = \frac{1}{2^k}\left(1 - \frac{\binom{n-s}{k}}{\binom{n}{k}}\right)$$

as $\hat{\alpha}$ must satisfy C_i and α must differ from $\hat{\alpha}$ on one of the indices in S. Continuing the calculation,

$$\frac{1}{2^k}\left(1 - \frac{\binom{n-s}{k}}{\binom{n}{k}}\right) \geq \frac{1}{2^k}\frac{\binom{n}{k} - \binom{n-1}{k}}{\binom{n}{k}} = \frac{1}{2^k}\frac{\binom{n-1}{k-1}}{\binom{n}{k}}.$$

Thus the probability that rows α and $\hat{\alpha}$ agree on column i is at most $1 - k/(2^k n)$. Since the columns are sampled independently, the probability that α and $\hat{\alpha}$ agree on all columns is at most

$$\left(1 - \frac{k}{n2^k}\right)^m \leq e^{-km/(n2^k)} \leq 2^{-5km/4n2^k},$$

since $\log e > 5/4$. By a union bound over ordered pairs of assignments in S, the probability that there exists a pair of rows that agree on all columns is at most

$$|S|^2 2^{-5km/4n2^k} \leq 2^{2\log|S| - 5km/4n2^k} \leq 2^{-km/n2^k}.$$

□

In our current setting we have $S = \{0,1\}^n$ and $km/n2^k \geq n \log n$. Thus, applying the previous lemma yields that all rows of M are distinct with high probability. Since each profile is distinct with high probability, this implies that $\mathcal{R}$ is one-to-one with high probability, and therefore $|R| = 2^n$. It remains to bound the terms $\#_1(1, A), \#_1(r, A)$, and $\#_0(s, R)$.

Bounding $\#_1(1, A)$. Fixing a single bit of the inputs in A to 1 is the same as selecting a vertex C in the bipartite constraint graph of F and an assignment α to the variables of C, and then setting $\mathrm{TT}_i(\alpha) = 1$. By the definition of $\mathcal{A}$, for any input $x \in \{0,1\}^n$, fixing this bit to 1 determines exactly k out of n variables of

x. Thus, the number of $x \in \{0,1\}^n$ that are consistent with this partial assignment is 2^{n-k}, and since $\mathcal{A}$ is one-to-one, we have $\#_1(1, A) = 2^{n-k}$.

Bounding $\#_1(r, A)$. This is similar to the previous bound, except that now we fix r bits of the truth table to 1. By the definition of $\mathcal{A}$, these bits must be chosen from r distinct truth tables in order to be consistent with any $x \in \{0,1\}^n$. With respect to the underlying CNF formula F, this corresponds to fixing an assignment to the set of variables appearing in an arbitrary set S of r clauses in F. By Lemma 3.5.6, with high probability we have $|\mathsf{Vars}(S)| \geq rk/2$. Thus fixing these r bits corresponds to setting at least $rk/2$ of the input variables participating in the constraints with determined truth tables. The number of x-inputs that are consistent with these indices fixed indices is therefore $\leq 2^{n-rk/2}$, and so $\#_1(r, A) \leq 2^{n-rk/2}$.

Bounding $\#_0(s, R)$. This case is symmetric to $\#_1(r, U)$. The result is that $\#_0(s, R) \leq 2^{n-2k/2}$.

Finally, observe that $(2s)\#_1(1, A) = (2s)2^n/n^4 \leq 2^{n-1}$. Putting this altogether, we can conclude that any monotone real circuit computing $\mathrm{mCSP\text{-}SAT}_F$ must have size at least

$$\frac{2^{n-1}}{(2s)^{s+1}2^{n-sk/2}} = 2^{sk/2-(s+1)\log(2s)-1} \geq 2^{s(k/2-2\log s)} \geq 2^{\tilde{\Omega}(n)},$$

where the final inequality follows because $s = n/ek^2$ and $k/4 \geq \log n$. □

3.5.2 Random CNFs

We show how to modify the argument from the previous section to apply to the usual distribution of random CNF formulas $\mathcal{F}(m, n, k)$, proving Theorem 1.3.1. Using the probabilistic method we find a partition of the variables of a random formula $F \sim \mathcal{F}(m, n, k)$ such that many of the clauses of F are balanced with respect to the partition. Ideally, every clause would be balanced, however this requirement turns out to be too strong — instead, we show that we can balance many of the clauses, and there exists a large collection of assignments that satisfies all of the *imbalanced* clauses. First, we introduce our notion of an "imbalanced" clause.

Definition 3.5.7. Fix $\varepsilon > 0$. Given a partition of n variables into X-variables and Y-variables, a k-clause is said to be *X-heavy* (resp., *Y-heavy*) if it contains more than $(1-\varepsilon)k$ X-variables (resp., Y-variables). A k-clause is called *balanced* if it is neither X-heavy nor Y-heavy.

Next, we define a *good partition* of the variables of a random CNF formula $F \sim \mathcal{F}(m, n, k)$. As discussed earlier, the notion of a good partition is supposed to help the rest of the proof in this section mimic the proof for balanced CNF formulas from the previous section. In particular, we will condition on a good partition and argue (in Lemma 3.5.9) using the Lovász Local Lemma that there exists a large collection of assignments satisfying all imbalanced clauses. Restricting attention to only these assignments will allow the proof to proceed as in the previous section. However, now we have a delicate balance of parameters. In particular, there is tension between Lemma 3.5.14 which requires m to be large, and the Lovász Local Lemma which requires m to be small. This is further complicated because we would like that all but a constant fraction of the assignments satisfy all imbalanced clauses (Lemma 3.5.9). Because of this we will need to set our parameters with precision.

In the following, let $H(\varepsilon) := -\varepsilon \log \varepsilon - (1-\varepsilon)\log(1-\varepsilon)$ denote the binary entropy function.

Definition 3.5.8. Let $\varepsilon = 1/50$ and let n be a sufficiently large positive integer. Let $k = 240 \log n$, $\tau = 1/16$ and $m = n2^{(1+\tau)k}$ $(= n^{256})$. For a CNF formula F, a partition (X, Y) of the variables is *good* for F if the following hold:

(i) The number of variables in X is $n/2 \pm o(n)$.

(ii) The number of X- and Y-heavy clauses are each upper bounded by $(3/2)n2^{(\tau+H(\varepsilon))k}$.

(iii) The functions $\mathcal{A}$ and $\mathcal{R}$ are both one-to-one on all assignments in $\{0,1\}^{|X|}$ and $\{0,1\}^{|Y|}$ respectively.

(iv) Each X-variable (Y-variable) occurs in at most $9k2^{(\tau+H(\varepsilon))k}$ X-heavy (Y-heavy) clauses respectively.

The next lemma says that a good partition exists with high probability. As the proof is nontrivial, we defer it to the end of this section.

Good Partition Lemma. *Let ε, n, k, and m be as above. Sample $F \sim \mathcal{F}(m, n, k)$ and partition the variables of F into two sets (X, Y) by including each variable in X with probability $1/2$, and in Y otherwise. Then, with probability $1 - o(1)$, (X, Y) is a good partition.*

Conditioning on a good partition (X, Y), it remain remains to show that there exists a large collection of assignments that satisfy all heavy clauses. The main tool in this proof is the Lovász Local Lemma.

Lovász Local Lemma (Theorem 5.1.1 in [6])**.** *Let $\mathcal{E} = \{E_1, \ldots, E_n\}$ be a finite set of events. For $E \in \mathcal{E}$ let $\Gamma(E)$ denote the set of events E_i on which E depends. If there is $q \in [0, 1)$ such that for all $E \in \mathcal{E}$ we have $\Pr[E] \leq q(1-q)^{|\Gamma(E)|}$, then the probability that none of the events E_i occur is at least $(1-q)^n$.*

The following lemma shows that for any partition (X, Y) satisfying the conditions of the Good Partition Lemma, there is a large collection of assignments satisfying all heavy clauses.

Lemma 3.5.9. *Let $F \sim \mathcal{F}(m, n, k)$ and let (X, Y) be a good partition for F. There exists a set Δ_X of $2^{|X|}/e^3$ assignments to the X-variables that satisfy all X-heavy clauses, and a set Δ_Y of $2^{|Y|}/e^3$ assignments to the Y-variables that satisfy all Y-heavy clauses.*

Proof. Consider a uniformly random assignment to the X-variables. Let E_i be the event that the ith X-heavy clause is not satisfied by the random assignment, and observe that $\Pr[E_i] \leq 2^{-(1-\varepsilon)k}$ since the clause is X-heavy. We aim to apply the Lovász Local Lemma to the events E_i.

We will continue to use the notation introduced in the definition of Good Partition Lemma, namely, $\varepsilon = 1/50$ and $\tau = 1/16$. As well, le $m_U = n2^{(\tau+H(\varepsilon)k}$. By property (ii) of the Good Partition Lemma, the number of events E_i is at most $(3/2)m_U$. By property (iv), for any event E_i, the number of events that share any X-variable with E_i is $|\Gamma(E_i)| \leq (9km_U/n)k$.

Setting $q = 2^{-\delta k}$ for $\delta = 1/15 + H(\varepsilon)$, for each E_i we have

$$\begin{aligned} q(1-q)^{|\Gamma(E_i)|} &\geq q\exp(-2q|\Gamma(E_i)|) = q\exp(-2 \cdot 2^{-\delta k}(9k^2/n)n2^{(\tau+H(\varepsilon))k}) \\ &= q\exp(-(18k^2)2^{-k/240}) \geq q/e \geq 2^{-(1-\varepsilon)k}, \end{aligned}$$

where we have used the fact that $e^{-2x} \leq 1-x$ for $x \in [0, 1/2]$ and that $-(18k^2)2^{-k/240} \geq -\mathsf{polylog}(n)/n \geq -1$ for sufficiently large n.

We have set q such that only a constant fraction of assignments will not satisfy all X-heavy clauses. To see this, observe that for our settings of τ, δ, and k,

$$qm_U = 2^{-\delta k}n2^{(\tau+H(\varepsilon))k} = n2^{-(\delta-(H(\varepsilon)+\tau))k} = n2^{-(1/15-1/16)240\log n} = 1.$$

Applying the Lovász Local Lemma we conclude that the probability that an assignment satisfies all X-heavy clauses is at least

$$(1-q)^{3m_U/2} \geq e^{-3qm_U} = e^{-3}.$$

Thus, the number of assignments to the X-variables satisfying all X-heavy clauses is at least $2^{|X|}/e^3$, and an identical calculation applied to the Y-variables by symmetry. □

With this lemma in place, we can proceed more or less as in the previous section. We perform the whole argument with respect to $A = \mathcal{A}(\Delta_X)$ and $R = \mathcal{R}(\Delta_Y)$, with Δ_X and Δ_Y chosen as in the previous lemma. This allows us to restrict our attention only to the balanced clauses, and the calculations from the previous section works *mutatis mutandis* since many clauses are balanced.

Theorem 3.5.10. *There is a constant $c > 0$ such that the following holds. Let $n \geq c$ be any positive integer and sample $F \sim \mathcal{F}(m, n, k)$ for $m = n2^{(1+1/16)k}$ and $k = 240 \log n$. With high probability there exists a partition (X, Y) of the variables of F such that any monotone real circuit computing $\mathrm{mCSP\text{-}SAT}_F$ requires at least $2^{\tilde{\Omega}(n)}$ gates.*

Proof. By the Good Partition Lemma we can find a good partition of the variables (X, Y), and let Δ_X, Δ_Y denote the set of assignments to the X- and Y-variables, respectively, given by Lemma 3.5.9. Let z be an input to $\mathrm{mCSP\text{-}SAT}_F$, and let z' be z restricted to truth tables corresponding to balanced clauses of F with respect to the partition (X, Y); it follows from the Good Partition Lemma that with high probability there are at least $m - 3m2^{-k/2} \geq m/2$ balanced clauses from n sufficiently large. Let $A = \{z' | z \in \mathcal{A}(\Delta_X)\}$ and $R = \{z' | z \in \mathcal{R}(\Delta_Y)\}$. Let $F' \subseteq F$ be the formula containing only balanced clauses of F, then we can think of z' as an input to $\mathrm{mCSP\text{-}SAT}_{F'}$.

Our aim will be to apply the Method of Symmetric Approximations to A and R, similar to what we did in the previous section. However, in order to do this we will have to show that the existence of a small monotone real circuit separating $\mathcal{A}(X)$ and $\mathcal{R}(Y)$ implies the existence of a small monotone real circuit that separates the truncated assignments A and R. The strategy of the proof is as follows: given a monotone real circuit C separating $\mathcal{A}(X)$ and $\mathcal{R}(Y)$ (and therefore $\mathcal{A}(\Delta_X)$ and $\mathcal{R}(\Delta_Y)$), we will apply a restriction ρ to C that fixes all of the input gates corresponding to the X- and Y-heavy clauses in such a way that the resulting circuit C_ρ separates A and R. Because F' is balanced, we can then perform the same argument for C_ρ with respect to $\mathcal{A}(\Delta_X)$ and $\mathcal{R}(\Delta_Y)$ as we did for balanced random CNFs in the previous section. A lower bound on the size of C_ρ then implies a lower bound on the size of the unrestricted circuit C.

We define the restriction ρ setting inputs (i.e. truth table entries) corresponding to unbalanced clauses as follows:

- Truth table entries corresponding to an X-heavy clause are all set to 1 except for the entry corresponding to the assignment that does not satisfy the clause.
- Truth table entries corresponding to a Y-heavy clause are all set to 1.

As we will see in the next claim, this restriction acts monotonically on the inputs in Δ_X and anti-monotonically on those in Δ_Y.

Claim 3.5.11. The circuit C_ρ obtained by applying the restriction ρ to C separates A and R.

Proof of Claim. Let $x \in \Delta_X$, and let $z = \mathcal{A}(x)$, then there is a corresponding $z' \in A$. Let $z' \circ \rho$ be the extension of z' by ρ to an input of $\mathrm{mCSP\text{-}SAT}_F$. Thus, C_ρ evaluated on z' is the same as the original circuit

C evaluated on $z' \circ \rho$. We claim that $z' \circ \rho \geq z$, i.e., $z' \circ \rho$ is z with some entries set to 1. To see this, observe that the truth table corresponding to every balanced clause is given the same assignment by z and $z' \circ \rho$. Clearly, for any Y-heavy clause C_i, the assignment given to TT_i by $z \circ \rho$ is at least the assignment given by z. Now, let C_i be an X-heavy clause, and recall that z is defined by setting $\mathrm{TT}_i(\alpha) = 1$ if and only if $x \restriction \mathsf{Vars}(i) = \alpha$. Let α' be the unique assignment to $\mathsf{Vars}(i)$ (the variables of C_i) that does not satisfy C_i. Because every assignment in Δ_X satisfies every X-heavy clause, it cannot be that $x \restriction \mathsf{Vars}(i) = \alpha'$, and so $\mathrm{TT}_i(\alpha') = 0$ in both z and $z' \circ \rho$. Therefore, $z' \circ \rho \geq z$. The original circuit C output 1 on z and therefore, by monotonicity, it also outputs 1 on $z' \circ \rho$. It follows that C_ρ outputs 1 on z'.

Next, let $y \in \Delta_Y$, let $z = \mathcal{R}(y)$, let $z' \in R$ be the input corresponding to z, and consider $z' \circ \rho$. We will argue that $z' \circ \rho \leq z$, i.e., $z' \circ \rho$ is z with some entries set to 0. Both z and $z' \circ \rho$ assign the same values to balanced clauses. Because every assignment in Δ_Y satisfies every Y-heavy clause, the truth tables corresponding to the Y-heavy clauses are identically 1 in both z and $z' \circ \rho$ by the definition of $\mathcal{R}$. The truth tables corresponding to the X-heavy clauses C_i are either the same in z as in $z' \circ \rho$ (if there exists $\alpha \in \{0,1\}^{|X|}$ such that $C_i(x, y) = 0$) or are identically 1 in z and contain a single 0-entry in ρ (if there is no such α). The original circuit C outputs 0 on z and therefore, by monotonicity, it also outputs 0 on $z' \circ \rho$. This completes the proof of the claim. □

The rest of the proof mirrors the proof of Theorem 3.5.3 with only minor changes. We will apply the Method of Symmetric Approximations to A and R, and count with respect to the balanced clauses. Because (X, Y) is a good partition, $\mathcal{A}$ and $\mathcal{R}$ are one-to-one on $\{0,1\}^{|X|}$ and $\{0,1\}^{|Y|}$ respectively, and are therefore one-to-one on Δ_X and Δ_Y. This implies that $|A| = |\Delta_X| = 2^{|X| - 3\log(e)}$ and $|R| = |\Delta_Y| = 2^{|Y| - 3\log(e)}$. It remains to bound $\#_1(1, A)$, $\#_1(r, A)$, and $\#_0(s, R)$. For this we will use the following immediately corollary of Lemma 3.5.5.

Lemma 3.5.12. *Let n be any sufficiently large integer, and k_0, m be positive integers. Let F be a CNF formula on m clauses, where each clause is sampled from $\mathcal{F}(1, n, k')$ for $k' \geq k_0$. Let $s \leq n/ek_0^2$ be a positive integer. If*

$$\log m \leq \delta \left(\frac{k_0}{2}\right) \log\left(\frac{k_0}{2}\right)$$

for some $0 < \delta < 1$, then for every $S \subseteq F$ of size s satisfies $|\mathsf{Vars}(S)| \geq k_0 s/2$ with probability at least $1 - 2^{-(1-\delta)(k_0 s/2)\log(k_0 s/2)}$.

This lemma follows immediately from the proof of Lemma 3.5.5 with $k_0 = k$ by noting that if each clause contains greater than k variables, then this can only increase the size of $\mathsf{Vars}(S)$.

Bounding $\#_1(r, A)$ and $\#_1(1, A)$. Fixing a single bit of an input in A to 1 is the same as selecting a balanced clause C_i in the constraint graph of F and an assignment α to the variables and setting $TT_i(\alpha) = 1$. Fixing this bit to 1 determines all variables in X that participate in this clause. By definition, each balanced clause contains at least $k_0 = k/50$ variables from X. Now, if we are to fix r truth table bits to 1, by the definition of $\mathcal{A}$, these bits must be chosen from r distinct truth tables in order to be consistent with any $x \in \{0,1\}^n$. We aim to apply Lemma 3.5.12 to show that every set of at most r clauses of F contain many variables. There are at least $m/2$ balanced clauses, and therefore

$$\log(m/2) = \log\left(n2^{(1+1/16)k-1}\right) = 256\log(n) - 1 \leq \gamma\left(\frac{k_0}{2}\right)\log\left(\frac{k_0}{2}\right),$$

for sufficiently large n and some constant $\gamma > 0$. Setting $r = s = n/(2ek_0^2)$, Lemma 3.5.12 implies that each collection S of r balanced clauses satisfies $|\mathsf{Vars}_X(S)| \geq k_0 r/2$ with high probability, where $\mathsf{Vars}_X(S)$ is the set of X variables that occur in clauses in S. Note that we can apply the argument from Lemma 3.5.12 because conditioned on containing some fixed number $k' \geq k/20 = k_0$ of X-variables, the X-part of a clause is distributed exactly according to $\mathcal{F}(1, |X|, k')$. Thus, fixing these r bits in the definition of $\#_1(r, A)$ corresponds to setting at least $k_0 r/2$ of the input variables that participate in the constraints with determined truth tables. The number of X-inputs that are consistent with these indices fixed is at most $2^{|X|-rk_0/2}$, and so $\#_1(r, A) \leq 2^{|X|-rk_0/2}$. By a similar argument, we have $\#_1(1, A) \leq 2^{|X|-k_0}$.

Bounding $\#_0(s, R)$. This case is similar to $\#_1(r, U)$ and we get $\#_0(s, R) \leq 2^{|Y|} - 2k_0/2$.

To conclude the theorem, we follow the calculation at the end of the proof of Theorem 3.5.3 using our new estimates. Note that our choice of $r = s = n/(2ek_0^2)$ implies that $2\log(2r) \leq 2\log n \leq k_0/2$ since $k_0 = k/15 > 4\log n$. Applying this, we have

$$(2s)\#_1(1, A) \leq 2^{\log(2r)+|X|-k_0} \leq 2^{|X|-(3/4)k_0}.$$

Altogether, this yields the following lower bound on the monotone real circuit size of $\mathrm{mCSP\text{-}SAT}_F$:

$$\begin{aligned}
\frac{|A| - (2s)\#_1(1, A)}{(2s)^{r+1}\#_1(r, U)} &\geq \frac{2^{|X|-3\log(e)-1}}{(2r)^{r+1}2^{|X|-rk_0/2}} \\
&\geq 2^{r(k_0/2-\log(2r))-\log(2r)-3\log(e)-1} \\
&\geq 2^{rk_0/4-\log(2r)-3\log(e)-1} \\
&\geq 2^{rk_0/4-\log(n)-3\log(e)-1} \\
&\geq 2^{\tilde{\Omega}(n)}.
\end{aligned}$$

□

As an immediate corollary of Theorem 3.5.10 together with Theorem 1.3.2, we can conclude the main theorem.

Corollary 3.5.13 (Theorem 1.3.1)**.** *There exists constants c, d such that the following holds. Let m be a sufficiently large positive integer, $k = c\log n$ and $m = n2^{dk}$. Then with high probability, any RCC_1 (and therefore CP) refutation of $F \sim \mathcal{F}(m, n, k)$ requires $2^{\tilde{\Omega}(n)}$ lines.*

3.5.3 Proof of the Good Partition Lemma

In this section we prove the Good Partition Lemma. To do so, we will make use of the following multiplicative Chernoff bound.

Multiplicative Chernoff Bound (Theorems 4.4 and 4.5 in [114])**.** *Suppose $E_1, \ldots, E_n$ are independent random variables taking values in $\{0, 1\}$. Let E denote their sum, and let $\mu = \mathbb{E}[E]$. Then*

- *For any $\delta \geq 0$,*

$$\Pr[E \geq (1 + \delta)\mu] \leq e^{-\delta\mu/3}.$$

- *For any* $0 \leq \delta \leq 1$
$$\Pr[E \leq (1-\delta)\mu] \leq e^{-\delta^2\mu/3}.$$

Before tackling the proof of the Good Partition Lemma, will first prove the following auxiliary lemma. In the case of *balanced* random CNF formulas, Lemma 3.5.6 allowed us to show that the function $\mathcal{A}$ is one-to-one. In order to handle random CNF formulas from the usual distribution, we must modify this lemma to take into account the fact that each clause may no longer contain an equal number of X and Y variables.

Lemma 3.5.14. *Let n, m, k be positive integers, and fix a set of variables $Z = \{z_1, \ldots, z_n\}$. Choose a partition $Z = (X, Y)$ by including each variable in X with probability $1/2$, and in Y otherwise. Sample m clauses independently as follows. Pick a uniformly random k-clause C over the Z-variables, then discard all X-literals from C; if all literals in C are discarded, then discard the entire clause. Let F' be the resulting formula, and let m' be the number of clauses in F'. Let S be a collection of assignments to the Y-variables, and let M be the $|S| \times m'$ matrix defined as follows: for any pair (α, i) with $\alpha \in S$ and $iin[m']$, let*

$$M[\alpha, i] = \begin{cases} 1 & \text{if the } i\text{th clause is not satisfied by } \alpha, \\ 0 & \text{otherwise.} \end{cases}$$

If $\log |S| < m/8n2^{k+3}$ then the rows of M are distinct with probability at least $1 - 2^{1-m/(n2^{k+3})}$.

To prove Lemma 3.5.14 we will use the following auxiliary lemma.

Lemma 3.5.15. *Let C be a k-clause over the Y-variables, sampled as in the statement of Lemma 3.5.14. Then,*

$$\Pr[C \text{ is empty}] = 1/2^k.$$

Proof Sketch. Since the variables of C do repeat, the following distributions on C are identical:

(i) Sample a random partition $Z = (X, Y)$ and then choose a uniformly random k-clause over (X, Y). Discard the X-variables.

(ii) Sample a uniformly random k-clause on the variables Z. Choose a partition $Z = (X, Y)$ by first partitioning the variables occurring in C by including each one in X with probability $1/2$ and in Y otherwise. Partition the remaining variables not occurring in C uniformly at random, and discard the X-variables from C.

In the latter interpretation it is easy to see that $|C|$ follows a binomial distribution with k trials and probability $1/2$ of success. In Subsection 3.5.4 we include a formal proof of this lemma that confirms that these distributions are identical. □

Proof of Lemma 3.5.14. We think of M as being generated column-by-column, with each column sampled independently as described in the statement of the lemma. Fix two assignments $\alpha, \hat{\alpha}$ such that $\alpha \neq \hat{\alpha}$. Let S be the set of indices on which they differ, i.e., $S := \{i : \alpha_i \neq \hat{\alpha}_i\}$, and let $s = |S|$. Let C_i be the ith clause in F' and w_i be its width; note that w_i is a random variable. Fix integers $t \leq k$ and $n' \leq n$. First, observe that conditioned on $w_i = t$ and $|Y| = n'$, the clause C_i is a uniformly random clause over the Y-variables of width t. Thus, if $t \geq 1$, we have

$$\Pr[C_i \text{ not sat by } \alpha, \text{ sat by } \hat{\alpha} | w_i = t, |Y| = n'] = \frac{1}{2^t}\left(1 - \frac{\binom{n'-s}{t}}{\binom{n'}{t}}\right).$$

This follows because $\hat{\alpha}$ falsifies C_i and α must differ from C_i on one of the indices in S. Continuing the calculation,

$$\frac{1}{2^t}\left(1-\frac{\binom{n'-s}{t}}{\binom{n'}{t}}\right) \geq \frac{1}{2^t}\left(\frac{\binom{n'}{t}-\binom{n'-1}{t}}{\binom{n'}{t}}\right) = \frac{1}{2^t}\frac{\binom{n'-1}{t-1}}{\binom{n'}{t}} = \frac{t}{2^t n'} \geq \frac{1}{2^k n},$$

where the final step holds because $t \leq k$, $n' \leq n$, and $t \geq 1$ since C_i is non-empty. Now, let E_t denote the event that C_i has width t for $0 \leq t \leq k$. Then

$$\begin{aligned}
\Pr[C_i \text{ not sat by } \alpha, \text{ sat by } \hat{\alpha}] &= \sum_{t=0}^{k} \Pr[C_i \text{ not sat by } \alpha, \text{ satisfied by } \hat{\alpha} \wedge E_t] \\
&= \sum_{t=0}^{k} \Pr[E_t] \Pr[C_i \text{ not sat by } \alpha, \text{ sat by } \hat{\alpha} | E_t] \\
&= \sum_{t=1}^{k} \Pr[E_t] \Pr[C_i \text{ not sat by } \alpha, \text{ sat by } \hat{\alpha} | E_t] \\
&\geq \frac{1-\Pr[E_0]}{2^k n} = \frac{1-1/2^k}{2^k n} \geq \frac{1}{2^{k+1} n}, \qquad (3.3)
\end{aligned}$$

where we have used the fact that the events $\{E_t\}$ partition the probability space, and the final equality follows from Lemma 3.5.15.

Next, we turn to bounding the probability that α and $\hat{\alpha}$ agree on all columns of M. Applying Lemma 3.5.15, $\mathbb{E}[m'] = (1-2^{-k})m$, where m' is the number of clauses in F'. Let E be the event that $m' \leq (1-\delta)\mathbb{E}[m']$, where δ is a parameter to be chosen later. By a Chernoff bound, $\Pr[E] \leq \exp(-\delta^2 \mathbb{E}[m']/3)$. Thus,

$$\begin{aligned}
\Pr[\alpha, \hat{\alpha} \text{ agree on all columns}] &\leq \Pr[E] + \Pr[\alpha, \hat{\alpha} \text{ agree on all columns} | \neg E] \\
&\leq \exp(-\delta^2 \mathbb{E}[m']/3] + \left(1 - \frac{1}{2^{k+1} n}\right)^{(1-\delta)\mathbb{E}[m']} \qquad \text{(By (3.3))} \\
&\leq \exp(-\delta^2 \mathbb{E}[m']/3) + \exp\left(-\frac{(1-\delta)\mathbb{E}[m']}{2^{k+1} n}\right) \\
&\leq \exp\left(-\frac{\delta^2(1-2^{-k})m}{3}\right) + \exp\left(-\frac{(1-\delta)(1-2^{-k})m}{2^{k+1} n}\right).
\end{aligned}$$

Setting $\delta = 1/(2^k-1)^{1/2}$, we get

$$\begin{aligned}
&\exp\left(-\frac{\delta^2(1-2^{-k})m}{3}\right) + \exp\left(-\frac{(1-\delta)(1-2^{-k})m}{2^{k+1} n}\right) \\
\leq &\exp\left(-\frac{m}{3 \cdot 2^k}\right) + \exp\left(-\frac{(1-\delta)(1-2^{-k})m}{2^{k+1} n}\right) \\
\leq &2\exp\left(-\frac{(1-\delta)(1-2^{-k})m}{2^{k+1} n}\right) \\
\leq &\exp\left(-\frac{m}{2^{k+3} n}\right),
\end{aligned}$$

where the second line uses the fact that a sum is at most twice the maximum, and the last line follows since $(1-\delta)(1-2^{-k}) \geq 1/4$ holds for sufficiently large n, by the definition of δ and that $k = 240 \log n$.

Thus, we can conclude that

$$\Pr[\alpha \text{ and } \hat{\alpha} \text{ agree on all columns}] \leq 2\exp\left(-\frac{m}{n2^{k+3}}\right) \leq 2 \cdot 2^{-5m/(4n2^{k+3})},$$

where the last inequality holds since $\log e > 5/4$. By a union bound over all pairs of assignments in S, the probability that there exists a pair of rows that agree on all columns is at most

$$2|S|^2 \cdot 2^{-5m/(4n2^{k+3})} \leq 2^{2\log|S|+1-5m/(4n2^{k+3})} \leq 2^{1-m/(n2^{k+3})}.$$

□

In what follows, we will use the following entropy bound on the binomial tail.

Entropy Bound on the Binomial Tail (Lemma 6.19 in [70]). *For any $0 < \varepsilon < 1/2$,*

$$\frac{2^{H(\varepsilon)n}}{\sqrt{8n\varepsilon(1-\varepsilon)}} \leq \sum_{j=0}^{\lfloor \varepsilon n \rfloor} \binom{n}{j} \leq 2^{H(\varepsilon)n},$$

where $H(\varepsilon) = -\varepsilon\log\varepsilon - (1-\varepsilon)\log(1-\varepsilon)$ is the binary entropy function.

We are now ready to prove the Good Partition Lemma, which we restate next for convenience.

Good Partition Lemma. *Let $\varepsilon = 1/50$, and let n be a sufficiently large positive integer. Let $k = 240\log n$, and let $m = n2^{(1+1/16)k}$ $(= n^{256})$. Sample $F \sim \mathcal{F}(m,n,k)$ and partition the variables of F into two sets (X,Y) by including each variable in X with probability $1/2$, and in Y otherwise. Then, with probability $1-o(1)$ (X,Y) is good partition. That is, the following holds:*

(i) The number of variables in X is $n/2 \pm o(n)$.

(ii) The number of X-heavy clauses and Y-heavy clauses are each upper bounded by $(3/2)n2^{(1/16+H(\varepsilon))k}$.

(iii) The functions $\mathcal{A}$ and $\mathcal{R}$ are both one-to-one on all assignments in $\{0,1\}^{|X|}$ and $\{0,1\}^{|Y|}$ respectively.

(iv) Each X-variable (Y-variable) occurs in at most $9k2^{(1/16+H(\varepsilon))k}$ X-heavy (Y-heavy) clauses respectively.

Proof. We will bound the probability that each event occurs and then conclude that they hold simultaneously with high probability by a union bound.

(i). First, note that $\mathbb{E}[|X|] = n/2$. Since each variable is placed in X independently with probability $1/2$, we have

$$\Pr[|X - n/2| > n^{2/3}] \leq 2\exp(-n^{1/3}/6)$$

by applying the Chernoff Bound.

(ii). For convenience, let $m = n2^{(1+\tau)k}$ where we set $\tau = 1/16$. For each clause C_i let T_i be the random variable indicating whether this clause is X-heavy. Using both inequalities from the Entropy Bound on the Binomial Tail we have

$$\Pr[T_i = 1] = \sum_{j=0}^{\varepsilon k} 2^{-k}\binom{k}{j} \leq 2^{(H(\varepsilon)-1)k}$$

and

$$\Pr[T_i = 1] = \sum_{j=0}^{\varepsilon k} 2^{-k}\binom{k}{j} \geq 2^{-k}\frac{2^{H(\varepsilon)k}}{\sqrt{8k\varepsilon(1-\varepsilon)}} > \frac{2^{(H(\varepsilon)-1)k}}{\sqrt{k}},$$

since $8\varepsilon(1-\varepsilon) < 1$ by our choice of ε. Let $T = \sum_{i=1}^{m} T_i$. Then the previous two bounds and linearity of expectation imply that

$$\frac{m2^{(H(\varepsilon)-1)k}}{\sqrt{k}} \leq \mathbb{E}[T] \leq m2^{(H(\varepsilon)-1)k}.$$

Denoting the lower and upper bounds by $m_L := m2^{(H(\varepsilon)-1)k}/\sqrt{k}$ and $m_U := m2^{(H(\varepsilon)-1)k} = n2^{(\tau+H(\varepsilon))k}$, by the Chernoff Bound we have

$$\Pr[T > 3m_U/2] \leq \Pr[T > 3\mathbb{E}[T]/2] \leq \exp(-\mathbb{E}[T]/12) \leq \exp(-m_L/12) \tag{3.4}$$

Thus, we have that $T < 3m_U/2$ with probability at least $1 - 2\exp(-m_L/12)$. The same conclusion holds for the Y-heavy clauses by symmetry. It follows by a union bound that the partition satisfies both of these properties simultaneously with high probability.

(iii). Recall that $\mathcal{A}(x)$ sets $\mathrm{TT}_i(\alpha) = 1$ if and only if $x \restriction \mathsf{Vars}(i) = \alpha$. Thus $\mathcal{A}(x) = \mathcal{A}(\hat{x})$ for some $x \neq \hat{x}$ only if there exists an X-variable that doesn't appear in any clause. The probability that *any* variable (in X or Y) does not occur in any clause is at most

$$n\left(\frac{\binom{n-1}{k}}{\binom{n}{k}}\right)^m = n\left(1 - \frac{k}{n}\right)^m \leq ne^{-km/n}.$$

Thus, the probability that every variable in X appears in some clause is at least $1 - ne^{-km/n}$.

Similarly, recall that $\mathcal{R}$ maps each assignment $y \in \{0,1\}^{|Y|}$ to the vector obtained by writing out the truth tables of each of the clauses of F under the assignment y to the Y-variables. The truth tables corresponding to clauses that were satisfied by y are identically 1, while the truth tables of the clauses that were not satisfied have exactly one 0-entry. Let $S = \{0,1\}^{|Y|}$ and observe that

$$m/8n2^{k+3} = 2^{k/16}/8^2 = n^{15}/8^2 > n \geq \log|S|.$$

Therefore, Lemma 3.5.14 implies that the set of clauses not satisfied by each assignment in S are distinct, and so $\mathcal{R}$ is one-to-one with probability at least $1 - 2^{1-m/n2^{k+3}} = 1 - 2^{1-n^{15}/8}$.

Altogether, by a union bound, the probability that $\mathcal{A}$ and $\mathcal{R}$ are one-to-one is at least $1 - 2^{1-n^{15}/8} - ne^{-km/n} = 1 - o(1)$.

(iv). We will use the same notation as in part (ii). We will prove this statement for the X-variables, the Y-variables will follow by symmetry. Furthermore, it is enough to prove this statement for the set of partitions $\mathcal{P} := \{(X,Y) : ||X| - n/2| \leq n^{2/3}\}$ because, by part (i), the probability of drawing $(X,Y) \notin \mathcal{P}$ is exponentially small.

Fix a partition $(X,Y) \in \mathcal{P}$. Let $x \in X$ be a fixed variable and let T_i be the indicator random variable (conditioned on this fixed partition) which is 1 if the variable in the ith X-heavy clause and 0 otherwise. Denote by $T := \sum_i T_i$ the total number of X-heavy clauses in which x occurs. We give an upper bound on T and then conclude the statement by a union bound over the variables in X. One would hope to apply

the usual Chernoff and Union bound combination, however in this case Z is a sum of a *random number* of random variables. Fortunately, we can sidestep this issue by conditioning on the number of X-heavy clauses, denoted H.

First, we show that the T_i variables are independent once we have conditioned on $H = h$. Consider the following method of sampling a random CNF formula subject to the partition (X, Y).

1. Independently for each $i \in [m]$, sample a number $v_i \in \{0, 1, \ldots, k\}$ where $v_i = \ell$ is chosen with probability
$$\frac{\binom{|X|}{\ell}\binom{|Y|}{k-\ell}}{\binom{n}{k}}.$$
2. Independently for each $i \in [m]$, sample a random clause by choosing a random set of v_i X-literals and a random set of $(k - v_i)$ Y-literals.

From the definition of the experiment it is clear that the variables T_j and $T_{j'}$ with $j \neq j'$ are independent, and will remain independent even after conditioning on any subset of h clauses being heavy.

To complete the proof we will need to the following two claims. The first claim shows that the bound on the number of X-heavy and Y-heavy clauses from part (ii) holds even when conditioning on a partition (X, Y). As the proof is similar to the argument from part (ii) we defer it to Subsection 3.5.4.

Claim 3.5.16. For any fixed $(X, Y) \in \mathcal{P}$, the number of X-heavy and Y-heavy clauses are each upper bounded by $3m_U/2$ and lower bounded by $m_L/2$, except with probability at most $\exp(-\Omega(m_L))$.

The second claim shows that when $m_L/2 \leq h \leq 3m_U/2$, the probability of T being large is small.

Claim 3.5.17. For any h such that $m_L/2 \leq h \leq 3m_U/2$,

$$\Pr[T > 9km_U/n | (X, Y), H = h] \leq \exp(-km_U/n).$$

Proof. Let

$$\mu_h := \mathbb{E}[T | H = h, (X, Y)] \text{ and } \delta_h := \frac{9|X|m_U}{hn} - 1.$$

Observe that

$$\frac{(1-\varepsilon)k}{|X|} \leq Pr[T_i = 1 | (X, Y), H = h] \leq \frac{k}{|X|}$$

because the ith heavy clause is generated by picking at most k and at least $(1 - \varepsilon)k$ variables from X. It follows that $h(1 - \varepsilon)k/|X| \leq \mu_t \leq hk/|X|$. By the Chernoff Bound we have

$$\begin{aligned}
\Pr[T > 9km_U/n | (X, Y), H = h] &\leq \Pr[T > (1 + \delta_h)\mu_h | (X, Y), H = h] \\
&\leq \exp(-\delta_h \mu_h/3) \\
&\leq \exp(-3(1 - \varepsilon)km_U/n + \mu_h/3) \\
&\leq \exp(-3(1 - \varepsilon)km_U/n + hk/3|X|) \\
&\leq \exp(-3(1 - \varepsilon)km_U/n + m_U k/2|X|).
\end{aligned}$$

Note that $|X| \geq n/2 - n^{2/3} \geq n/3$ for sufficiently large n. Continuing the calculation,

$$\begin{aligned}
\exp(-3(1 - \varepsilon)km_U/n + m_U k/2|X|) &\leq \exp(-3(1 - \varepsilon)km_U/n + 3m_U k/2n) \\
&\leq \exp(-km_U/n),
\end{aligned}$$

where the last step follows since we have set $\varepsilon = 1/50$.

□

Now, using the previously mentioned fact that the T_j's are independent conditioned on the value of H, together with Claim 3.5.17 and Claim 3.5.16, we can complete the proof. First, we bound the probability that there are many X-heavy clauses for a *fixed* partition $(X, Y) \in \mathcal{P}$. Let $\mathcal{H} := \{m_L/2 + 1, m_L/2 + 2, \ldots, 3m_U/2 - 1\}$, then

$$\begin{aligned}\Pr[T > 9km_U/n|(X,Y)] &\leq \Pr[H \notin \mathcal{H}|(X,Y)] + |\mathcal{H}| \max_{h\in\mathcal{H}} \Pr[T > 9km_U/n|H = h, (X,Y)] \\ &\leq \exp(-\Omega(m_L)) + m \cdot \exp(-km_U/n) \leq \exp(-\Omega(m_L)),\end{aligned}$$

where the last step holds for sufficiently large n since $k = O(\log n)$ and $m_L = m_U/\sqrt{k}$. Thus, we can take a union bound over all $x \in X$ and conclude that, for a fixed $(X, Y) \in \mathcal{P}$, the probability that there exists some X-variable that occurs in more than $9km_U/n$ heavy clauses is at most $n \exp(-\Omega(m_L))$.

We can now complete the proof of this part. Let B be the "bad" event that there is an X-variable that occurs in more than $9km_U/n$ X-heavy clauses. Then,

$$\begin{aligned}\Pr[B] &\leq \Pr[(X,Y) \notin \mathcal{P}] + 2^n \max_{(X,Y)\in\mathcal{P}} [B|(X,Y)] \\ &\leq 2\exp(-n^{1/3}/6) + n \cdot \exp(n \ln 2 - \Omega(m_L)) = o(1),\end{aligned}$$

where we have used the bound from part (i) and the fact that $m_L = \mathsf{poly}(n)$. By symmetry, the same bound holds for the Y-heavy clauses. Therefore, taking a union bound finishes the proof of this part.

Finally, taking a union bound over parts (i) – (iv) completes the proof of the Good Partition Lemma. □

3.5.4 Proof of Lemma 3.5.15 and Claim 3.5.16

We end by proving the remaining statements. We begin with a formal proof of Lemma 3.5.15, which is restated next for convenience.

Lemma 3.5.15. *Let C be a k-clause over the Y-variables, sampled as in the statement of Lemma 3.5.14. Then,*

$$\Pr[C \textit{ is empty}] = 1/2^k.$$

Proof. Observe that $|X|$ is a binomial random variable consisting of n trials with probability $p = 1/2$ of success, and so $\Pr[|X| = t] = \binom{n}{t} 2^{-n}$. Then

$$\begin{aligned}\Pr[C \text{ is empty}] &= \sum_{t=0}^{n} \Pr[|X| = t] \Pr[C \text{ is empty}||X| = t] \\ &= \sum_{t=0}^{n} \frac{\binom{n}{t}}{2^n} \cdot \frac{\binom{t}{k}}{\binom{n}{k}} \\ &= \frac{1}{2^n \binom{n}{k}} \sum_{t=k}^{n} \binom{n}{t}\binom{t}{k}\end{aligned}$$

where the change in indices follows since if $t < k$ then C can never be contained in X. This sum counts the number of ways to first choose a t-subset A of $[n]$, and then choose a k-subset B of A. Equivalently, we can

first choose the k-subset B of $[n]$, and then generate A by extending B to a t-subset. Thus

$$\begin{aligned}\frac{1}{2^n\binom{n}{k}}\sum_{t=k}^{n}\binom{n}{t}\binom{t}{k} &= \frac{1}{2^n\binom{n}{k}}\sum_{t=k}^{n}\binom{n}{k}\binom{n-k}{t-k}\\ &= \frac{1}{2^n}\sum_{t=k}^{n}\binom{n-k}{t-k}\\ &= \frac{2^{n-k}}{2^n} = \frac{1}{2^k}.\end{aligned}$$

□

Next, we prove Claim 3.5.16. We will use the same notation as in Good Partition Lemma; recall that $\varepsilon = 1/50$, $k = 240\log n$, and $m = n2^{(1+1/16)k}$. As well, $m_L := m2^{(H(\varepsilon)-1)k}/\sqrt{k}$ and $m_U := m2^{(H(\varepsilon)-1)k}$.

Claim 3.5.16. For any fixed $(X,Y) \in \mathcal{P}$, the number of X-heavy and Y-heavy clauses are each upper bounded by $3m_U/2$ and lower bounded by $m_L/2$, except with probability at most $\exp(-\Omega(m_L))$.

Proof. For each clause C_i let T_i be the random variable indicating whether this clause is X-heavy. Clearly the probability of a clause being X-heavy is maximized when $|X|$ is as large as possible. Since we are considering $|X| \in [n/2 - n^{2/3}, n/2 + n^{2/3}]$, it suffices to bound the probability of a clause being X-heavy for $|X| = n/2 + n^{2/3}$. Let $n' = n^{2/3}$ for convenience. We can bound the probability of a clause being X-heavy given (X,Y) as follows:

$$\begin{aligned}\Pr[T_i = 1|(X,Y)] &= \sum_{\ell=0}^{\epsilon k}\frac{\binom{|Y|}{\ell}\binom{|X|}{k-\ell}}{\binom{n}{k}}\\ &\le \sum_{\ell=0}^{\epsilon k}\binom{n/2-n'}{\ell}\binom{n/2+n'}{k-\ell}\frac{1}{\binom{n}{k}}\\ &= \sum_{\ell=0}^{\epsilon k}\frac{\binom{k}{\ell}}{2^k}\cdot\frac{(n/2-n')!}{(n/2-n'-\ell)!}\cdot\frac{(n/2+n')!}{(n/2+n'-k+\ell)!}\cdot\frac{2^k(n-k)!}{n!}.\end{aligned}$$

The expression $\sum_{\ell=0}^{\epsilon k}\binom{k}{\ell}/2^k$ is what we had before in the analysis of part (2). Thus, if we can bound the term $\frac{(n/2-n')!}{(n/2-n'-\ell)!}\cdot\frac{(n/2+n')!}{(n/2+n'-k+\ell)!}\cdot\frac{2^k(n-k)!}{n!}$ by a constant, we are done. This is maximized when $\ell = 0$, therefore it suffices to bound $\frac{(n/2+n')!}{(n/2+n'-k)!}\cdot\frac{2^k(n-k)!}{n!}$. For that we use the fact that there exist constants c_0 and c_1 such that[2] $c_0 n^{n+1/2}e^{-n} \le n! \le c_1 n^{n+1/2}e^{-n}$. Let $c = c_1^2/c_0^2$, then

$$\begin{aligned}\frac{(n/2+n')!}{(n/2+n'-k)!}\cdot\frac{2^k(n-k)!}{n!} &\le \frac{c(n/2+n')^{n/2+n'+1/2}e^{-n/2-n'}}{(n/2+n'-k)^{n/2+n'-k+1/2}e^{-n/2-n'+k}}\cdot\frac{2^k(n-k)^{n-k+1/2}e^{-n+k}}{n^{n+1/2}e^{-n}}\\ &= c\cdot\left(\frac{n/2+n'}{n/2+n'-k}\right)^{n/2+n'-k+1/2}\cdot\frac{2^k(n-k)^{n-k+1/2}(n/2+n')^k}{n^{n+1/2}}\\ &= c\left(1+\frac{k}{z_U}\right)^{z_U+1/2}\left(1-\frac{k}{n}\right)^{n-k+1/2}\left(1+\frac{2n'}{n}\right)^k\\ &\le c\cdot\exp(k+k/(2z_U))\exp(-k+k^2/n-k/(2n))\exp((2n'k)/n)\\ &\le c\cdot\exp(k/(2z_U)+k^2/n+(2n'k)/n) \le c\cdot\exp(3) = O(1),\end{aligned}$$

[2] More specifically, one can take $c_0 = \sqrt{2\pi}$ and $c_1 = e$.

where we have defined $z_U := n/2 + n' - k$. Therefore, $\Pr[T_i = 1|(X,Y)] \leq c_U \sum_{\ell=0}^{\varepsilon k} \binom{k}{\ell} 2^{-k}$ for some constant $c_U > 0$.

Lower bounding the probability of a clause being X-heavy can be done analogously. The probability of a clause being X-heavy is minimized when $|X|$ is as small as possible. Therefore,

$$\begin{aligned}\Pr[T_i = 1|(X,Y)] &\geq \sum_{\ell=0}^{\varepsilon k} \binom{n/2+n'}{\ell}\binom{n/2-n'}{k-\ell}\frac{1}{\binom{n}{k}} \\ &= \sum_{\ell=0}^{\varepsilon k} \frac{\binom{k}{\ell}}{2^k}\frac{(n/2+n')!}{(n/2+n'-\ell)!}\cdot\frac{(n/2-n)!}{(n/2-n'-k+\ell)!}\cdot\frac{2^k(n-k)!}{n!}\end{aligned}$$

The term $\frac{(n/2+n')!}{(n/2+n'-\ell)!}\cdot\frac{(n/2-n)!}{(n/2-n'-k+\ell)!}\cdot\frac{2^k(n-k)!}{n!}$ is minimized whenever $\ell = 0$, therefore it suffices to bound $\frac{(n/2-n')!}{(n/2-n'-k)!}\cdot\frac{2^k(n-k)!}{n!}$ from below by a constant. Using the same bound on $n!$ as above,

$$\begin{aligned}&\frac{(n/2-n')!}{(n/2-n'-k)!}\cdot\frac{2^k(n-k)!}{n!} \\ &\geq c^{-1}\cdot\left(\frac{n/2-n'}{n/2-n'-k}\right)^{n/2-n'-k+1/2}\cdot\frac{2^k(n-k)^{n-k+1/2}(n/2-n')^k}{n^{n+1/2}} \\ &= c^{-1}\cdot\left(1+\frac{k}{z_L}\right)^{z_L+1/2}\left(1-\frac{k}{n}\right)^{n-k+1/2}\left(1-\frac{2n'}{n}\right)^k \\ &\geq \frac{c^{-1}}{2^3}\exp\big((k/z_L)(z_L+1/2)\big)\exp\big((-k/n)(n-k+1/2)\big)\exp\big(-2n'k/nk\big) \\ &= \frac{c^{-1}}{2^3}\exp\big(k/(2z_L)+k^2/n-k/(2n)-2n'/n\big) = \Omega(1),\end{aligned}$$

where $z_L := n/2 - n' - k$. The third line follows from the fact that k/z_L, k/n, and $2n'k/n$ are all less than $1/2$ for sufficiently large n, and therefore we can use the inequality $(1+x) >= e^x/2$ when $|x| < 1/2$. Therefore, $\Pr[T_i = 1|(X,Y)] \geq \sum_{\ell=0}^{\varepsilon k} c_L\binom{k}{\ell}2^{-k}$ for some constant $0 < c_L < 1$.

The remainder of the proof is similar to the proof of property **(ii)** in the Good Partition Lemma. Using both of the inequalities in Entropy Bound on the Binomial Tail, we have

$$\begin{aligned}\Pr[T_i = 1|(X,Y)] &\leq c_U\sum_{\ell=1}^{\varepsilon k}\binom{k}{\ell}2^{-k} \leq c_U\cdot 2^{H(\varepsilon)k-k} \\ \Pr[T_i = 1|(X,Y)] &\geq c_L\sum_{\ell=1}^{\varepsilon k}\binom{k}{\ell}2^{-k} \geq c_L\cdot 2^{-k}\frac{2^{H(\varepsilon)k}}{\sqrt{8k\varepsilon(1-\varepsilon)}} \geq c_L\cdot\frac{2^{(H(\varepsilon)-1)k}}{\sqrt{k}},\end{aligned}$$

since $\sqrt{8\varepsilon(1-\varepsilon)} < 1$ for our choice of ε. Let $T := \sum_{i=1}^m T_i$. Then by linearity of expectation,

$$c_L\cdot m2^{(H(\varepsilon)-1)k}/\sqrt{k} \leq \mathbb{E}[T] \leq c_U\cdot m2^{H(\varepsilon)k-k} = c_U\cdot n2^{(\tau+H(\varepsilon))k},$$

where $\tau = 1/16$. Letting $m_L := m2^{(H(\varepsilon)-1)k}/\sqrt{k}$ and $m_U := n2^{(\tau+H(\varepsilon))k}$ as before, and define $\delta_U :=$

$3/2c_U - 1$. By the Chernoff bound, we have

$$\begin{aligned}\Pr[T > 3m_U/2|(X,Y)] &\leq \Pr[T > 3\mathbb{E}[T]/(2c_U)|(X,Y)] \\ &= \Pr[T > (1+\delta_U)\mathbb{E}[T]|(X,Y)] \\ &\leq \exp(-\delta_U^2\mathbb{E}[T]/3) \leq \exp(-\delta_U^2 m_L/3) = \exp(-\Omega(m_L)),\end{aligned}$$

where the final equality holds because δ_U is a constant. Similarly, for $\delta_L := 1 - 2/c_L$,

$$\begin{aligned}\Pr[T < m_L/2|(X,Y)] &\leq \Pr[T < (1-\delta_L)\mathbb{E}[T]|(X,Y)] \\ &\leq \exp(-\delta_L^2\mathbb{E}[T]/3) \leq \exp(-\delta_L^2 m_L/3) = \exp(-\Omega(m_L))\end{aligned}$$

Thus we have that $m_L/2 < T < 3m_U/2$ with probability at least $1 - 2\exp(-\Omega(m_L))$. Exactly the same conclusion holds via the same calculations for the Y-variables as well.

□

3.6 Conclusion

We end by discussing some questions left open by this work.

Problem 3.1. Lower Bounds for $O(1)$-Random CNFs. The main remaining question is to obtain Cutting Planes lower bounds on random k-CNF formulas when $k = \Theta(1)$. It seems likely that such lower bounds should hold even for RCC_1 proofs, however, as discussed in the introduction to this chapter it seems that current methods for proving monotone circuit lower bounds are incapable of obtaining strong lower bounds when k is constant.

Problem 3.2. More Robust Lower Bounds for Random CNFs. Our lower bound on random $\Theta(\log n)$-CNF formulas holds for clause density $\Delta_k = \Theta(2^{(1+\tau)k})$ for some $\tau \in (0,1)$. As discussed in the introduction, this interval is relatively narrow compared to lower bounds on random k-CNF formulas for other proof systems. Can we obtain a lower bounds on random $O(\log n)$-CNF formulas which hold for a more robust range of Δ_k?

Chapter 4

Stabbing Planes

4.1 Introduction

An effective method for analyzing classes of algorithms is to formalize the techniques used by the algorithms into a *formal proof system*, and then analyze the proof system instead. By doing this, we are able to hide many of the practical details of implementing these algorithms, while preserving the class of methods that the algorithms can feasibly employ. This allows us to factor understanding a given algorithm into two questions:

(i) How powerful is the proof system? For which inputs are there short proofs?

(ii) How close can actual implementations of the algorithm come to the ideal nondeterministic algorithm modelled by the proof system?

As an illustrative example of this approach, recall the *DPLL* algorithm [53,54], which forms the basis of modern conflict-driven clause learning algorithms for solving SAT. For a CNF formula F, the DPLL algorithm is the following recursive search for a satisfying assignment: choose a variable x_i (non-deterministically, or via some heuristic), and then recurse on the formulas $F \restriction (x_i = 0)$ and $F \restriction (x_i = 1)$. If at any point a satisfying assignment is found, the algorithm halts and outputs the assignment. Otherwise, if the current partial assignment falsifies some clause C of F, the recursive branch is terminated. If every recursive branch terminated with a falsified clause, then F is unsatisfiable and we can take the recursive tree as a proof of this fact; in fact, such a DPLL tree is equivalent to a *treelike resolution* refutation of F.

The approach of using proof complexity for algorithm analysis has been successfully employed to study many different families of algorithms, including

- *Conflict-driven clause-learning* algorithms for SAT [92, 115, 142], which can be formalized using *resolution* proofs [54].
- Optimization algorithms using *semidefinite programming* [73, 122], which can often be formalized using *Sums-of-Squares proofs* [12, 81].
- The *cutting planes* algorithms for integer programming [37, 76], which are formalized by *Cutting Planes proofs* [37, 38, 45].

We continue the study of formal proof systems corresponding to modern integer programming algorithms. Integer programming problems are an intrinsically natural formalization of NP-optimization problems, and

algorithms for solving them have had a profound effect throughout computer science and beyond. A classic approach for solving integer programming problems — pioneered by Gomory [76] — is to refine the polytope P by introducing *cutting planes*. These algorithms are formalized by the Cutting Planes proof system [45], and this connection has led to a number of lower bounds on the runtime of these algorithms [30,68,72,86,127], as well as related measures such as the Chvátal rank [31, 36, 79].

While Cutting Planes has grown to be an influential proof system in propositional proof complexity, the original cutting planes algorithms suffered from numerical instabilities, as well as difficulties in finding good heuristics for the next cutting planes to add [76]. Instead, modern algorithms for integer programming improve on the classical cutting planes method by combining it with a second technique, known as *branch-and-bound* [10, 104], resulting in a family of optimization algorithms broadly referred to as *branch-and-cut algorithms* [121]. These algorithms search for integer solutions in a polytope P by recursively repeating the following two procedures: First, P is split into smaller polytopes $P_1, \ldots, P_k$ such that $P \cap \mathbb{Z}^n \subseteq \bigcup_{i \in [k]} P_i$ (i.e. *branching*). Next, cutting planes deductions are made in order to further refine the branched polytopes (i.e. *cutting*). In practice, branching is usually performed by selecting a variable x_i and branching on all possible integer values of x_i; that is, recursing on $P \cap \{x_i = t\}$ for each feasible integer value t. More complicated branching schemes is also considered, such as branching on the hamming weight of subsets of variables [65], branching using basis-reduction techniques [1, 2, 82, 91, 102, 109], and more general linear inequalities [94, 112, 120] which fall under the purview of branching on *general disjunctions*[1]. While these branch-and-cut algorithms are much more efficient in practice than the classical cutting planes methods, they are no longer naturally modelled by Cutting Planes proofs.

In this work, we introduce *Stabbing Planes* as a proof system which formalize branch-and-cut algorithms. Intuitively, Stabbing Planes has the same branching structure as DPLL, but generalizes branching on single variables to branching on linear inequalities. We will formalize Stabbing Planes in stages as a generalization of DPLL. Recall that in the setting of integer programming we would like to prove the integer-infeasibility of a system of integer linear inequalities $Ax \geq b$ over real-valued variables; for simplicity of exposition, suppose that $Ax \geq b$ encodes a CNF formula F. We rephrase DPLL *geometrically* to the setting of integer linear programming as follows. Consider some DPLL refutation of F. Each time the DPLL tree branches on the $\{0, 1\}$-value of a variable x_i, instead branch on whether $x_i \leq 0$ or $x_i \geq 1$; because the encoding $Ax \geq b$ of F includes axioms $x_i \geq 0$ and $x_i \leq 1$ this is equivalent. After this replacement, each node v in the DPLL tree is naturally associated with a polytope P_v of points satisfying $Ax \geq b$ and each of the inequalities labelling the root-to-v path. Since we began with a DPLL refutation, it is clear that for any leaf ℓ, the polytope P_ℓ associated with the leaf is empty, as any $\mathbb{Z}^n$-valued point would have survived each of the inequalities queried on some path in the tree and thus would exist in one of the polytopes at the leaves.

The Stabbing Planes system is then the natural generalization of the previous object: at each step in the refutation an *arbitrary* integer linear form ax in the input variables is chosen and we recurse assuming that it is at least some integer b or at most its *integer negation* $b - 1$. Observe that because a and b are integral, any $x^* \in \mathbb{Z}^n$ will satisfy at least one of the inequalities $(ax \leq b - 1,\ ax \geq b)$, and so if the polytope at each leaf (again, obtained by intersecting the original system with the inequalities on the path to this leaf) is empty then we have certified that the original system has no integral solutions (an example is given in Figure 4.1). One of the major advantages of Stabbing Planes is its simplicity: refutations are decision trees that query integer linear inequalities.

The queries in a Stabbing Planes proof correspond to what is known as branching *general disjunctions*

[1] For far more in-depth discussions on branch-and-bound and branch-and-cut we refer the reader to [146], [42], and [7]

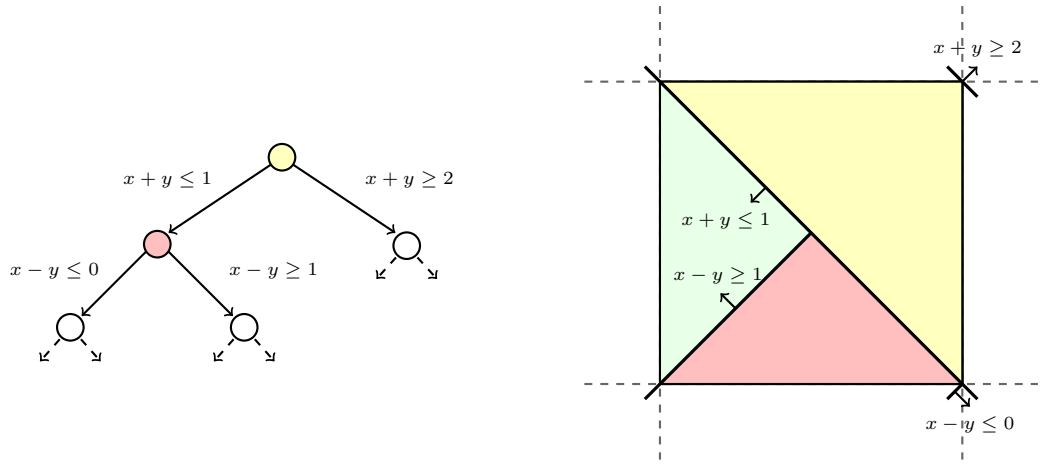


Figure 4.1: A partial stabbing planes proof (left) and its result on the unit square (right). The yellow and red areas are removed from the polytope (green), and we recurse on both sides.

or *split disjunctions* [44] in integer programming, and capture the majority of branching that is done in practice [42]. However, recall that branch-and-cut algorithms combine Stabbing Planes-style branching with additional cutting planes in order to refine the search space. Unlike Stabbing Planes proofs, Cutting Planes deductions can be *dag-like*, repeatedly reusing previously derived inequalities. Therefore it is not immediately clear that Stabbing Planes is able to fully formalize this part of branch-and-cut. Surprisingly, we show that these additional cutting planes are superfluous from the perspective of proof complexity: Stabbing Planes can efficiently simulate Cutting Planes deductions. Furthermore, this simulation was recently extended by [13] to show that Stabbing Planes can simulate *disjunctive cuts* which capture the vast majority of cutting planes used in practice, including *lift and project cuts* [11], *split cuts* [44], *Gomory mixed integer* cuts [75] and *MIR cuts*.

Beyond providing a theoretical model for branch and cut algorithms, we believe that the simplicity of Stabbing Planes proofs, as well as its closeness to DPLL, makes Stabbing Planes a better starting point for the development of search algorithms which operate over the reals, such as *pseudoboolean* SAT solvers, than established proof systems. Modern pseudoboolean solvers are built upon the Cutting Planes proof system, which suffers a complex deductive ruleset in which new constraints must be deduced directly from old constraints; this is in contrast with query-based proof systems such as Stabbing Planes. From the perspective of SAT solving, even though treeRes is equivalent to DPLL, it is the search point of view of DPLL that has led to major advances in SAT algorithms. A natural hypothesis is that it is much easier to invent useful heuristics in the language of query-based algorithms, as opposed to algorithms based on the deductive rules of resolution. Stabbing Planes offers similar benefits with respect to reasoning about inequalities. Furthermore, Stabbing Planes is a direct generalization of DPLL, and therefore we hope that the fine-tuned heuristics that have been developed for modern DPLL-based solvers can be lifted to algorithms based on Stabbing Planes.

Stabbing Planes in Proof Complexity

Despite its simplicity, Stabbing Planes proofs are remarkably powerful. As a motivating example, we give simple, short (quasipolynomial size) SP refutations of *any* unsatisfiable system of linear equations over a prime finite field. Systems of linear equations over a prime finite field, such as the *Tseitin formulas*, form one of the canonical families of hard examples for algebraic and semi-algebraic proof systems, including

Nullstellensatz [80], Polynomial Calculus [32], and Sum-of-Squares [81, 137].

Even so, Stabbing Planes is not so powerful that establishing lower bounds appears to be entirely out of reach. Letting SP^* be the restriction of SP that only allows branching on inequalities with coefficients of magnitude at most quasipolynomial in n, we prove the following lower bound.

Theorem 1.3.7. *There exists a family of unsatisfiable CNF formulas $\{F_n\}$ for which any SP^* refutation of F requires size 2^{n^ε} for constant $\varepsilon > 0$.*

To prove this we establish a general relationship between SP^* and CP. First, we characterize Cutting Planes as a non-trivial subsystem of Stabbing planes that we call *Facelike* Stabbing Planes. Then, we show that any SP^* proof can be made *facelike* with only a quasipolynomial blowup in the size. This is summarized by the next theorem.

Theorem 1.3.5. *Let F be any unsatisfiable CNF formula on n variables and suppose that there is an SP^* refutation of F in size s and maximum coefficient size c. Then there is a Cutting Planes refutation of F of size $s(cn)^{\log s}$.*

In fact, we establish a more general version of this theorem (Theorem 4.4.6) which allows us to quasipolynomially translate any SP^* refutation of a sufficiently bounded system of linear inequalities into CP. This allows for the significant analysis done on the size of Cutting Planes proofs to be lifted directly to branch-and-cut solvers.

As an application of the connection between SP^* and CP, we show that CP can quasipolynomially refute any unsatisfiable system of linear equations over a prime finite field.

Corollary 1.3.6. *Let F be the CNF encoding of an unsatisfiable system of m linear equations over a prime finite field. Then there is a Cutting Planes refutation of F of size $|F|^{O(\log m)}$.*

This generalizes the surprising result of Dadush and Tiwari [48] showing that CP has quasipolynomial size refutations of the Tseitin Formulas which had long conjectured to be hard to refute in CP [45].

As a second application of Theorem 1.3.5 we can conclude that SP^*, and therefore branch-and-cut, proofs cannot be found efficiently. Indeed, it follows almost immediately from the framework Göös et al. [78] that SP^* is not *automatable* unless $\mathsf{P} = \mathsf{NP}$.

We also explore the relationships between SP and other proof systems. This is summarized in Figure 4.2. Most notably, we show that SP is polynomially-equivalent to $\mathsf{treeR}(\mathsf{CP})$, the tree-like variant of Krajíček's $\mathsf{R}(\mathsf{CP})$ proof system [99]. This system can be thought of as a mutual generalization of resolution and Cutting Planes, in which the lines are disjunctions of integer linear inequalities, and we are allowed Cutting Planes rules on the inequalities, as well as resolution-style cuts on the disjunctions. Even though SP is equivalent to a system already in the literature, the new perspective provided by SP is enlightening. Indeed, none of the results in this chapter were known for $\mathsf{treeR}(\mathsf{CP})$, including the simulation of CP.

The remainder of this chapter tackles the problem of proving lower bounds on SP proofs with unbounded coefficients. Although we are unable to establish size lower bounds, we prove strong lower bounds on the *depth* of SP refutations, as well as explain why several natural approaches for proving size bounds fail. The depth of an SP proof — the longest root to leaf path in the proof — is a natural parameter that captures the parallelizability of these proofs, and is closely related to *rank* measures of polytope which have been studied extensively integer programming theory [36].

Theorem 1.3.8. *There exists a family of unsatisfiable CNF formulas $\{F_n\}$ for which any SP refutation requires depth $\Omega(n/\log^2 n)$*

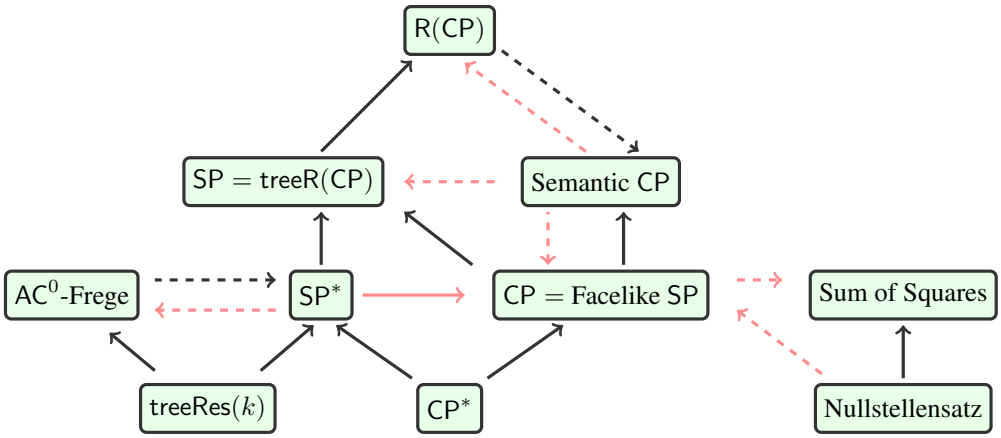


Figure 4.2: Known relationships between relevant proof systems. A solid black (red) arrow from proof system P_1 to P_2 indicates that P_2 can polynomially (quasipolynomially) simulate P_1. A black (red) dashed arrow from P_1 to P_2 indicates that P_2 cannot polynomially (quasipolynomially) simulate P_1.

The proof of this theorem proceeds by showing that shallow SP proofs give rise to short randomized and real communication protocols for the CNF search problem. One might hope that these depth bounds could be leveraged to prove size lower bounds. However, we show that straightforward approaches for doing this fail. First, we show that SP proofs cannot be *balanced* — SP proofs of size s do not imply the existence of proofs of size $\mathsf{poly}(s)$ and depth $\mathsf{polylog}(s)$.

While SP proofs cannot be balanced, the real communication protocols that result from SP proofs preserve the topology of the SP proof, and therefore size lower bounds on SP would follow by showing that real communication protocols can be balanced. There is a precedent for this: both deterministic and randomized communication protocols *can* be balanced, and lower bounds on treeCP proofs were obtained by exploiting this fact [89]. However, we show that real communication protocols cannot be balanced. Enroute we establish the first superlogarithmic lower bound on the real communication complexity of the set disjointness function, a function which has been central to many of the lower bounds which exploit communication complexity.

4.1.1 Related and Subsequent Work

Lower Bounds on Variable Branching. Lower bounds for a number of branch-and-cut algorithms using *variable-branching* — meaning that they branch on the integer value of single variables, rather than arbitrary inequalities (i.e., DPLL which branches on $x_i \in \mathbb{Z}$) — have been established. The first example of this was the lower bound of Jerslow [90] on the number of queries made by branch-and-bound algorithms with variable branching. Cook and Hartman proved exponential lower bounds on the number of operations required to solve certain travelling salesman instances by branch-and-cut algorithms which use variable branching and Chvátal-Gomory cuts [46]. However, their lower bound is exponential only in the number of variables, and not in the number of inequalities. The first truly exponential (in the encoding size of the instance) lower bound for branch-and-cut algorithms which use variable branching was established by Dash [51] who extending the lower bound of Pudlák [127] for Cutting Planes proofs.

Lower Bounds for treeR(CP). Lower bounds on certain restricted models of treeR(CP) were established in earlier works. Krajíček [99] proved superpolynomial lower bounds on the size of R(CP) proofs when both the *width* of the clauses, and the magnitude of the coefficients of every line are sufficiently bounded.

Concretely, let w and c be upper bounds on the width and coefficient size respectively; the lower bound that he obtained is $2^{n^{\Omega(1)}}/c^{w \log^2 n}$. Building on this work, Kojevnikov [97] showed that the dependence on the coefficient size for $\mathsf{treeR}(\mathsf{CP})$ proofs, obtaining a lower bound of $\exp(\Omega(\sqrt{n/w \log n}))$. In Subsection 4.5.1 we prove a size-preserving simulation of SP by $\mathsf{treeR}(\mathsf{CP})$ which translates depth d SP proofs into width d $\mathsf{treeR}(\mathsf{CP})$ proofs. Therefore, Kojevnikov's result implies a superpolynomial lower bound on the size of SP proofs of depth $o(n/\log n)$. In Subsection 4.6.1 we exhibit a formula for which any SP refutation requires depth $\Omega(n/\log^2 n)$, nearly matching the result of Kojevnikov.

Intermediate and Subsequent Work. Following [15], in which we gave quasipolynomial size SP refutations of the *Tseitin formulas* [15], Dadush and Tiwari [48] showed that these proofs could be translated into CP. This refuted a long-standing conjecture that CP requires exponential size refutations of these formulas [45]. In the same paper, they established a polynomial equivalence between the number of nodes and the *size* of SP proofs (i.e., the number of bits needed to express the proof).

As well, they consider SP in the context of *mixed integer programming* (MIP), and prove exponential lower bounds on SP in this setting. In this setting, you are given a polytope $P = \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} : Ax + By \geq b\}$, and you are searching for an integer solution to the x-variables and a real solution to the y-variables. That is, rather than proving that $P \cap \mathbb{Z}^{n_1+n_2} = \emptyset$, instead you would like to prove that $P \cap \mathbb{Z}^{n_1} \cap \mathbb{R}^{n_2} = \emptyset$. In this case, SP queries involving y-variables are disallowed, as this would not be sound. As shown by Dadush and Tiwari, this restriction turns out to be enough to obtain quite simple proofs of intractability. First, they prove that any SP refutation of a certain system of 2^n many inequalities (encoding the *complete unsatisfiable formula*) requires size $2^n/n$. Next, they show that this system of inequalities admits a $\mathsf{poly}(n)$-size MIP *extended formulation*. As SP cannot branch on the extension variables, refuting this extended formulation is identical refuting the complete unsatisfiable formula in SP. Dey, Dubey, and Molinaro [58] extended this technique to prove lower bounds for a number of MIP instances for packing, set cover, the Travelling Salesman problem, and the cross polytope, even when Gaussian noise is added to the coefficients. However, this technique crucially relies on the fact that for MIP problems, SP queries cannot involve the real-variables, and therefore it does not appear to be possible to extend this technique to prove lower bounds on pure integer programming problems (i.e. those with only integer-variables) such as standard encodings of CNF formulas.

Basu et al. [13] showed that SP can simulate *disjunctive cuts*, a result which we cover in more detail in Subsection 4.4.4. Furthermore, they give an integer programming instance that can be solved in size $O(1)$ in SP but requires $\mathsf{poly}(n)$ deductions using split cuts. As well, they explore the effect that *sparsity* —the number of non-zero coordinates in each query — has on branch-and-cut. Sparse queries can be thought of as an intermediate between full SP branching and variable branching. They provide an instance where the any branch-and-bound tree must be of exponential size if the sparsity is $o(n)$.

Recently, Dantchev et al. [49] introduced several novel techniques for proving lower bounds on SP proofs by exploiting their geometric structure. In particular, they make use of the fact that for a polytope P, every point $x^* \in P$ must be contained within some slab of the SP proof. This allowed them to establish linear lower bounds on the size of SP proofs of the *pigeonhole principle* as well as the *Tseitin formulas*; because SP proofs are binary trees, this leads to a depth $\Omega(\log n)$ lower bound for both formulas.

4.1.2 Organization

We begin by formally defining Stabbing Planes as a proof system in Section 4.2. The short Stabbing Planes refutations of linear equations over any prime finite field are given in Section 4.3. Section 4.4 explores how Stabbing Planes relates to Cutting Planes and is organized as follows: first, in Section 4.4 we characterize Cutting Planes as a sub-system of Stabbing Planes. In Subsection 4.4.1 we show that SP^* proofs can be converted into CP. Next, we explore whether a simulation of Cutting Planes by Stabbing Planes can preserve other parameters of the proof in Subsection 4.4.3. We end this section (in Subsection 4.4.4) we observe that Stabbing Planes can simulate most other types of cutting planes which are used in integer programming.

In Section 4.5 we explore how Stabbing Planes compares to other popular proof systems in the literature. In the final section (Section 4.6) we explore whether we can prove lower bounds on Stabbing Planes with unbounded coefficients. We prove unrestricted depth lower bounds in Subsection 4.6.1 and rule out several natural approaches that utilize communication complexity in Subsection 4.6.2.

4.2 The Stabbing Planes Proof System

We begin with a formal definition of the Stabbing Planes (SP) proof system, and then record some basic properties.

Stabbing Planes. Let $Ax \geq b$ be a system of linear inequalities with no integer solutions. A *Stabbing Planes (SP) refutation* of $Ax \geq b$ is a directed binary tree, T, where each edge is labelled with a linear integral inequality satisfying the following *consistency conditions*:

- *Internal Nodes.* For any internal node u of T, if the right outgoing edge of u is labelled with $cx \geq d$, then the left outgoing edge is labelled with its *integer negation* $cx \leq d-1$.
- *Leaves.* Each leaf node v of T is labelled with a conic combination of inequalities in F with inequalities along the path leading to v that yields $0 \geq 1$ (provided by Farkas' Lemma).

For an internal node u of T, the pair of inequalities $(cx \leq d-1,\ cx \geq d)$ is called the *query* corresponding to the node. Every node of T has a polytope P associated with it, where P is the polytope defined by the intersection of the inequalities in F together with the inequalities labelling the path from the root to this node. We will say that the polytope P *corresponds* to this node. That is, if P is the polytope corresponding to a node v which queries $(cx \leq d-1,\ cx \geq d)$, then the polytopes of the children of v are given by $P \cap \{x \in \mathbb{R}^n : cx \leq d-1\}$ and $P \cap \{x \in \mathbb{R}^n : cx \geq d\}$. For readability, we will use the abbreviation $P \cap \{cx \geq d\}$ for $P \cap \{x \in \mathbb{R}^n : cx \geq d\}$.

The *slab* corresponding to a query $(cx \leq d-1,\ cx \geq d)$ is $\{x \in \mathbb{R}^n \mid d-1 < cx < d\}$, which is the set of points ruled out by this query. The *width* of the slab is the minimum distance between $cx \leq d-1$ and $cx \geq d$, which is $1/\|c\|_2$. This gives an intuitive geometric interpretation of SP refutations: at each step we remove a slab from the polytope and recurse on the resulting polytopes on both sides of the slab. The aim is to recursively cover the polytope with slabs until every feasible point has been removed. An example of this can be seen in Figure 4.1, where the yellow and red areas are the slabs of the two queries.

The *size* of an SP refutation is the bit-length needed to encode a description of the entire proof tree, which, for CNF formulas as well as sufficiently bounded systems of inequalities, is polynomially equivalent to the number of queries in the refutation. In particular, Dadush and Tiwari [48] prove the following.

Proposition 4.2.1 (Corollary 1.2 in [48])**.** *Let $Ax \geq b$ be any system of linear inequalities with no integer solutions whose coefficients require ℓ bits to express, and let s be the number of nodes in an* SP *refutation of $Ax \geq b$. Then there exists an* SP *refutation of size $s\ell n^6$.*

As well, the *depth* (or *rank*) of the refutation is the depth of the binary tree. The depth of refuting an unsatisfiable (over $\mathbb{Z}^n$) system of linear inequalities $Ax \geq b$, denoted $\mathsf{depth}_{\mathsf{SP}}(Ax \geq b)$, is the minimum depth of any SP refutation of $Ax \geq b$. Observe that any unsatisfiable system of inequalities $Ax \geq b$ whose corresponding polytope is contained within the unit cube $[0,1]^n$ (this includes the encodings of all CNF formulas) has a trivial size 2^n and depth n SP refutation by branching on $(x_i \leq 0,\ x_i \geq 1)$ for every $i \in [n]$.

Next, we show that SP is indeed a proof system, as defined by Cook and Reckhow [43].

Proposition 4.2.2. *Stabbing Planes is sound, complete, and polynomially verifiable.*

Proof. Completeness follows immediately from the fact that SP simulates DPLL, which is itself a complete proof system. Soundness follows because each slab in an SP proof, corresponding to a query $(cx \leq d-1,\ cx \geq d)$, removes only non-integral points. Indeed, by the integrality of c and d, any $x \in \mathbb{Z}^n$ satisfies either $cx \leq d-1$ or $cx \geq d$. Finally, to see that SP proofs are polynomially verifiable, observe that we only need to verify that every query is of the form $(cx \leq d-1,\ cx \geq d)$ for integral c and d, and that each conic combination labelling the leaves evaluates to $0 \geq 1$. □

We will also be interested in the following natural restriction of Stabbing Planes.

Bounded-Weight Stabbing Planes. The *bounded-weight* Stabbing Planes proof system SP^* is the subsystem of Stabbing Planes obtained by requiring that all coefficients of the proofs to have magnitude at most quasipolynomial ($n^{\log^{O(1)} n}$) in the number of input variables.

4.3 Refutations of Linear Equations over Prime Finite Fields

Systems of linear equations over finite fields, and especially the *Tseitin formulas*, form some of the most prominent classes of hard formulas for many weak proof systems, including Resolution [144], Nullstellensatz [80], Polynomial Calculus [32], Sum-of-Squares [81, 137], and AC^0-Frege [20, 71, 84, 124], and were conjectured to be hard to prove in Cutting Planes [45]. As a motivating example, we exhibit simple, short Stabbing Planes refutations of *any* unsatisfiable system of linear equations over a prime finite field.

If $ax = b$ is a linear equation we say the *width* of the equation is the number of non-zero variables occurring in it. Any width-d linear equation over a prime finite field of size p, denoted $\mathbb{F}_p$, can be represented by a CNF formula with p^{d-1} width-d clauses — one ruling out each falsifying assignment. For a width-d system of m linear equations F over $\mathbb{F}_p$, we will denote by $|F| := mp^{d-1}$ the size of the CNF formula encoding F.

Theorem 4.3.1. *Let $F = \{f_1 = b_1, \ldots, f_m = b_m\}$ be a width-d, unsatisfiable set of linear equations over $\mathbb{F}_p$. There is an* SP *refutation of (the CNF encoding of) F in size $(mpd)^{O(\log m)} p^d = |F|^{O(\log m)}$ and depth $O(\log^2(mpd))$.*

First we sketch the idea for the Tseitin Formulas. Our SP proofs correspond to a branch decomposition procedure which is commonly used to solve SAT (see e.g. [5, 50, 57, 106]). Consider a cut $V = V_1 \cup V_2$ in G and observe that if we know the parity of the edge variables crossing the cut $\oplus_{uv \in V_1 \times V_2} x_{uv}$ then we know

that at least one of V_1 or V_2 is unsatisfiable depending on the parity of the sum of the cut variables. We can recurse on the unsatisfiable subset, decreasing the size of the instance by $1/2$. To determine the parity of the cut in SP, we simply branch on all possible values of the sum of the edges crossing the cut. Since there are n boolean variables (corresponding to the edges), each cut has at most $n+1$ possibilities for the sum, and if we maintain that the partition of the vertices defining the cut is balanced, then we will recurse at most $O(\log n)$ times.

Next, we describe how handle an arbitrary system F of m $\mathbb{F}_2$-linear equations. View the system F as a hypergraph over n vertices (corresponding to the variables) and with a d-edge for each equation. Partition the set of hyperedges into two sets $E = E_1 \cup E_2$ of roughly the same size, and consider the *cut* of vertices that belong to both an edge in E_1 and in E_2. Using the SP rule we branch on all possible values of the sum of the cut variables in order to isolate E_1 and E_2. Once we know this sum, we are guaranteed that either E_1 is unsatisfiable or E_2 is unsatisfiable. This allows us to recursively continue on the side of the cut (E_1 or E_2) that is unsatisfiable.

Over a prime finite field of size p the proof will proceed in much the same way. Instead of a subgraph, at each step we will maintain a subset of the equations $I \subseteq [m]$ such that $\{f_i = b_i\}_{i \in I}$ must contain a constraint that is violated by the SP queries made so far. We partition I into two sets I_1 and I_2 of roughly equal size and query the values a and b of $\sum_{i \in I_1} f_i$ and $\sum_{i \in I_2} f_i$. Because F is unsatisfiable, at least one of $a - \sum_{i \in I_1} b_i \not\equiv 0$ or $b - \sum_{i \in I_2} b_i \not\equiv 0$, meaning that that it is unsatisfiable, and we recurse on it.

In the following, we will let z stand for a vector of $\mathbb{F}_p$-valued variables z_i. When we discuss any form $f := az$ where $a \in \mathbb{F}_p^m$ and z is a vector of n variables z_i, we will implicitly associate it with the linear form $\sum_{i \in [n]} a_i (\sum_{j \in [\log p]} x_{i,j})$ where $x_{i,j}$ are the $\log p$ many boolean variables encoding z_i in the CNF encoding of F.

Proof of Theorem 4.3.1. Let $F = \{f_1 = b_1, \ldots, f_m = b_m\}$ be a system of unsatisfiable linear equations over $\mathbb{F}_p$, where each $f_i = a_i z$ for $a_i \in \mathbb{F}_p^n$, and $b_i \in \mathbb{F}_p$. Because F is unsatisfiable, there exists a $\mathbb{F}_p$ linear combination of the equations in F witnessing this; formally, there exists $\alpha \in \mathbb{F}_p^n$ such that $\sum_{i \in [m]} \alpha_i f_i \equiv 0 \mod p$, but $\sum_{i \in [m]} \alpha_i b_i \not\equiv 0 \mod p$.

Stabbing Planes will implement the following binary search procedure for a violated equation; we describe the procedure first, and then describe how to implement it in Stabbing Planes. In each round we maintain a subset $I \subseteq [m]$ and an integer k_I representing the value of $\sum_{i \in I} \alpha_i f_i$. Over the algorithm, we maintain the invariant that $k_I - \sum_{i \in I} \alpha_i b_i \not\equiv 0 \bmod p$, which implies that there must be a contradiction to F inside of the constraints $\{f_i = b_i\}_{i \in I}$.

Initially, $I = [m]$ and we obtain k_I by querying the value of the sum $\sum_{i \in [m]} \alpha_i f_i$. If $k_I \not\equiv 0 \mod p$ then this contradicts the fact that $\sum_{i \in I} \alpha_i f_i \equiv 0 \mod p$; thus, the invariant holds. Next, perform the following algorithm.

1. Choose a balanced partition $I = I_1 \cup I_2$ (so that $||I_1| - |I_2|| \leq 1$).
2. Query the value of $\sum_{i \in I_1} \alpha_i f_i$ and $\sum_{i \in I_2} \alpha_i f_i$; denote these values by a and b respectively.
3. If $a - \sum_{i \in I_1} \alpha_i b_i \not\equiv 0 \bmod p$ then recurse on I_1 with $k_{I_1} := a$. Otherwise, if $b - \sum_{i \in I_2} \alpha_i b_i \not\equiv 0 \bmod p$ then recurse on I_2 with $k_{I_2} := b$.

4. Otherwise (if $a - \sum_{i \in I_1} \alpha_i b_i \equiv b - \sum_{i \in I_2} \alpha_i b_i \equiv 0 \mod p$), then this contradicts the invariant:

$$\begin{aligned} 0 \not\equiv k_I - \sum_{i \in I} \alpha b_i &= \sum_{i \in I} \alpha_i (f_i - b_i) \\ &= \sum_{i \in I_1} \alpha_i (f_i - b_i) + \sum_{i \in I_2} \alpha_i (f_i - b_i) \\ &= (a - \sum_{i \in I_1} \alpha_i b_i) + (b - \sum_{i \in I_1} \alpha_i b_i) \equiv 0 \mod p. \end{aligned}$$

This recursion stops when $|I| = 1$, at which point we have an immediate contradiction between k_I and the single equation indexed by I.

It remains to implement this algorithm in SP. First, we need to show how to perform the queries in step 2. Querying the value of any sum $\sum_{i \in I} \alpha_i f_i$ can be done in a binary tree with at most p^2md leaves, one corresponding to every possible query outcome. Internally, this tree queries all possible integer values for this sum (e.g. $(\sum_{i \in I} \alpha_i f_i \leq 0,\ \sum_{i \in I} \alpha_i f_i \geq 1), (\sum_{i \in I} \alpha_i f_i \leq 1,\ \sum_{i \in I} \alpha_i f_i \geq 2), \ldots$). For the leaf where we have deduced $\sum_{i \in [m]} \alpha_i f_i \leq 0$ we use the fact that each variable is non-negative to deduce that $\sum_{i \in [m]} \alpha_i f_i \geq 0$ as well. Note that p^2md is an upper bound on this sum because there are m equations, each containing at most d variables, each taking value at most $(p-1)$ [2]. Thus, step 2 can be completed in $(p^2md)^2$ queries.

Finally, we show how to derive refutations in the following cases: (i) when we deduced that $\sum_{i \in [m]} \alpha_i f_i \not\equiv 0 \mod p$ at the beginning, (ii) in step 4, (iii) when $|I| = 1$.

(i) Suppose that we received the value $a \not\equiv 0 \mod p$ from querying $\sum_{i \in [m]} \alpha_i f_i$. Note that the coefficient of every variable in $\sum_{i \in [m]} \alpha_i f_i$ is a multiple of p. Query

$$\Big(\sum_{i \in [m]} \alpha_i f_i / p \leq \lceil a/p \rceil - 1,\ \sum_{i \in [m]} \alpha_i f_i / p \geq \lceil a/p \rceil \Big).$$

At the leaf that deduces $\sum_{i \in [m]} \alpha_i f_i / p \leq \lceil a/p \rceil - 1$, we can derive $0 \geq 1$ as a non-negative linear combination of this inequality together with $\sum_{i \in [m]} \alpha_i f_i \geq a$. Similarly, at the other leaf $\sum_{i \in [m]} \alpha_i f_i / p \geq \lceil a/p \rceil$ can be combined with $\sum_{i \in [m]} \alpha_i f_i \leq a$ to derive $0 \geq 1$.

(ii) Suppose that $a - \sum_{i \in I_1} \alpha_i b_i \equiv b - \sum_{i \in I_2} \alpha_i b_i \equiv 0 \mod p$. Then $0 \geq 1$ is derived by summing $\sum_{i \in I_1} \alpha_i f_i \geq a$, $\sum_{i \in I_2} \alpha_i f_i \geq b$ and $\sum_{i \in I} \alpha_i f_i \leq k_I$, all of which have already been deduced.

(iii) When $|I| = 1$ then we deduced that $a_I z = k_I$ for $k_I \not\equiv b_I \mod p$ and we would like to derive a contradiction using the axioms encoding $a_I z \equiv b_I$. These axioms are presented to SP as the linear-inequality encoding of a CNF formula, and while there are no integer solutions satisfying both these axioms and $a_I z = k_I$, there could in fact be *rational* solutions. To handle this, we simply force that each of the at most d variables in $a_I z$ takes an integer value by querying the value of each variable one by one. As there are at most d variables, each taking an integer value between 0 and $p - 1$, this can be done in a tree with at most p^d many leaves. At each leaf of this tree we deduce $0 \geq 1$ by a non-negative linear combination with the axioms, the integer-valued variables, and $a_I z \equiv b_I$.

[2] Note that instead of querying the value of $\sum_{i \in I} \alpha_i f_i$ we could have queried $\sum_{i \in I} \alpha_i f_i \pmod p$ to decrease the number of leaves to pmd.

The recursion terminates in at most $O(\log m)$ many rounds because the number of equations under consideration halves every time. Therefore, the size of this refutation is $(pmd)^{O(\log m)}p^d$. Note that by making each query in a balanced tree, this refutation can be carried out in depth $O(\log^2(mpd))$. □

4.4 The Relationship Between Stabbing Planes and Cutting Planes

It is an interesting question how Stabbing Planes compares to Cutting Planes, the main proof system based on ideas from integer programming. By contrasting the two systems we see three major differences:

- *Top-down vs. Bottom-up.* Stabbing Planes is a *top-down* proof system formed by performing queries on the polytope and recursing; while Cutting Planes is a *bottom-up* proof system, formed by deducing new inequalities from previously deduced ones.
- *Polytopes vs. Halfspaces.* Individual "lines" in a Stabbing Planes proof are polytopes, while individual "lines" in a Cutting Planes proof are halfspaces.
- *Tree-like vs. dag-like.* The graphs underlying Stabbing Planes proofs are trees, while the graphs underlying Cutting Planes proofs are general dags: intuitively, this means that Cutting Planes proofs can "re-use" their intermediate steps, while Stabbing Planes proofs cannot.

When taken together, these facts suggest that Stabbing Planes and Cutting Planes could be incomparable in power, as polytopes are more expressive than halfspaces, while dag-like proofs offer the power of line-reuse. Going against this intuition, we show next that Stabbing Planes and Cutting Planes are in fact very closely related.

4.4.1 Cutting Planes as a Subsystem of Stabbing Planes

In this section we show that SP can simulate CP proofs. In fact, we characterize CP as a nontrivial subsystem of SP, which we call *Facelike* SP.

Theorem 4.4.1. *The proof systems* CP *and Facelike* SP *are polynomially-equivalent.*

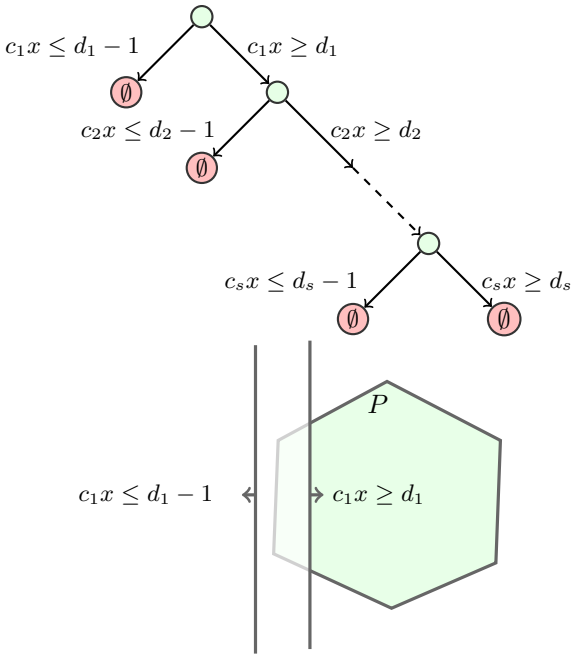
We begin by formally defining Facelike SP. Recall that for a polytope P and inequality $cx \geq d$ we denote $P \cap \{x \in \mathbb{R}^n : cx \geq d\}$ by $P \cap \{cx \geq d\}$.

Facelike Stabbing Planes. A Stabbing Planes query $(cx \leq d-1,\ cx \geq d)$ at a node P is *facelike* if one of the sets $P \cap \{cx \leq d-1\}$ and $P \cap \{cx \geq d\}$ is empty or is a face of P (see Figure 4.3b). An SP refutation is facelike if every query in the refutation is facelike.

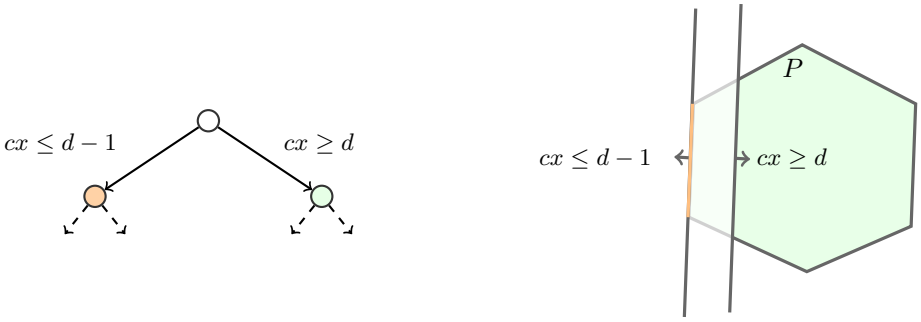
For polytopes P, P' we denote by $P \vdash P'$ that P' can be inferred from P by a facelike query; that is, there is $(cx \leq d-1,\ cx \geq d)$ such that $P \cap \{cx \leq d-1\} = \emptyset$ and $P \cap \{cx \geq d\} = P'$.

Enroute to proving Theorem 4.4.1, it will be convenient to introduce the following further restriction of Facelike Stabbing Planes.

Pathlike Stabbing Planes. A Stabbing Planes query $(cx \leq d-1, cx \geq d)$ at a node corresponding to a polytope P is *pathlike* if at least one of $P \cap \{cx \leq d-1\}$ and $P \cap \{cx \geq d\}$ is empty (see Figure 4.3a). A Pathlike SP refutation is one in which every query is pathlike.



(a) A Pathlike query. The polytope $P \cap \{c_1x \le d_1 - 1\} = \emptyset$, and $c_1x \ge d_1$ is a CG cut for P.



(b) A Facelike query. The polytope $P \cap \{cx \le d - 1\} = P \cap \{cx = d - 1\}$ marked in orange is a face of P.

Figure 4.3: Pathlike and Facelike SP queries on a polytope P. On the left are the proofs and on the right are the corresponding effects on the polytope.

The name "pathlike" comes from the fact that the underlying graph of a Pathlike Stabbing Planes proof is a path, since at most one child of every node has any children. In fact, we have already seen (nontrivial) Pathlike SP queries under another name: Chvátal-Gomory cuts.

Lemma 4.4.2. *Let P be a polytope and let $(cx \le d - 1,\ cx \ge d)$ be a pathlike Stabbing Planes query for P. Assume that $P \cap \{cx \le d - 1\} = \emptyset$ and that $P \cap \{cx \ge d\} \subsetneq P$. Then $cx \ge d$ is a CG cut for P.*

Proof. Since $cx \ge d$ is falsified by some point in P, it follows that there exists some $0 < \varepsilon < 1$ such that $cx \ge d - \varepsilon$ is valid for P — note that $\varepsilon < 1$ since otherwise $cx \le d - 1$ would not have empty intersection with P. This immediately implies that $cx \ge d$ is a CG cut for P. □

With this observation we can easily prove that Pathlike SP is equivalent to CP.

Lemma 4.4.3. *Pathlike* SP *is polynomially equivalent to* CP.

Proof. First, let $c_1 x \geq d_1, c_2 x \geq d_2, \ldots, c_s x \geq d_s$ be a CP refutation of an unsatisfiable (over $\mathbb{Z}^n$) system of linear inequalities $Ax \geq b$. Consider the sequence of polytopes $P_0 = \{Ax \geq b\}$ and $P_i = P_{i-1} \cap \{c_i x \geq d_i\}$. By inspecting the rules of CP, it can observed that $P_i \cap \{c_i x \leq d_i - 1\} = \emptyset$ and thus P_{i+1} can be deduced using one pathlike SP query from P_i for all $0 \leq i \leq s$.

Conversely, let P be any polytope and let $(cx \leq d-1,\ cx \geq d)$ be any pathlike SP query to P (so, suppose w.l.o.g. that the halfspace defined by $cx \leq d-1$ has empty intersection with P). By Lemma 4.4.2, $cx \geq d$ is a CG cut for P, and so can be deduced in Cutting Planes from the inequalities defining P in length $O(n)$ (see Proposition 2.2.2). Applying this to each query in the Pathlike SP proof yields the theorem. □

Next, we show how to simulate Facelike SP proofs by Pathlike SP proofs of comparable size. The proof of Lemma 4.4.5 is inspired by Dadush and Tiwari [48], and will use the following lemma due to Schrijver [138]. Recall that we write $P \vdash P'$ for polytopes P, P' to mean that P' can be obtained from P by a single facelike query from P.

Lemma 4.4.4. *Let P be a polytope defined by a system of integer linear inequalities and let F be a face of P. If F' is a polytope such that $F \vdash F'$ then there is a polytope P' such that $P \vdash P'$ and $P' \cap F \subseteq F'$.*

Proof. Let $P = \{Ax \leq b\}$ and $F = \{A_0 x = b_0, A_1 x \leq b_1\}$. Suppose $F' = F \cap \{cx \leq d\}$ is obtained by the Pathlike SP inference $cx \leq d$. Since $F \cap \{cx \geq d+1\}$ is empty, there is some $0 < \varepsilon < 1$ such that $cx \leq d + \varepsilon$ is valid for F. It follows that there are vectors y_0, y_1 such that $y_1 \geq 0$ and

$$\begin{aligned} y_0 A_0 + y_1 A_1 &= c \\ y_0 b_0 + y_1 b_1 &\leq d + \varepsilon. \end{aligned}$$

Let z_0 be any integral vector such that $y_0 - z_0$ is non-negative — for instance, we could choose $z_0 = \lfloor y_0 \rfloor$. Let $c' = c - z_0 A_0$ and $d' = d - z_0 b_0$. Observe that $c'x \leq d' + \varepsilon$ is valid for P as it can be deduced from P since $y_0 - z_0$ is non-negative. Indeed, we have shifted $cx \leq d + \varepsilon$ so that it no longer depends on the inequalities in $A_0 x \geq b_0$. Let $P' := P \cap \{c'x \leq d'\}$, noting that $P \vdash P'$ as c' and d' are integral and $P \cap \{c'x \geq d'+1\} = \emptyset$ because $c'x \leq d' + \varepsilon$ is valid for P.

Let $x \in P' \cap F$, and thus $x \in F$ and $c'x \leq d'$. Expanding c' and d' we have

$$c'x = (c - z_0 A_0)x \leq d' = d - z_0 b_0.$$

Rearranging we have that $c'x \leq d'$ implies that

$$cx \leq d + z_0(A_0 x - b_0).$$

However since $x \in F$, $A_0 x = b_0$ and thus $cx \leq d$, implying that $x \in F'$. □

Lemma 4.4.5. *Facelike* SP *is polynomially equivalent to Pathlike* SP.

Proof. That Facelike SP simulates Pathlike SP follows by the fact that any Pathlike SP query is a valid query in Facelike SP. For the other direction, consider an SP refutation π of size t. We describe a recursive algorithm for generating a Pathlike SP proof from π. The next claim will enable our recursive case.

Claim. Let P be a polytope and suppose $ax \geq b$ is valid for P. Assume that $P \cap \{ax = b\}$ has a Pathlike SP refutation using s queries. Then there is a polytope contained in $P \cap \{ax \geq b+1\}$ which can be derived from P in Pathlike SP using $s+1$ queries.

Proof of Claim. Since $cx \geq d$ is valid for P it follows that $F = P \cap \{cx = d\}$ is either a face of P or empty by definition. Consider the Pathlike SP refutation $F_0, F_1, \ldots F_s = \emptyset$, where the ith polytope F_i for $i < s$ is obtained from F_{i-1} by applying a pathlike SP query and proceeding to the non-empty child. That is, $F_{i-1} \vdash F_i$ for all i. Thus, by applying Lemma 4.4.4 repeatedly, we get a sequence of polytopes $P = P_0 \vdash P_1 \vdash \cdots \vdash P_s$ such that $P_i \cap F = P_i \cap \{cx = d\} \subseteq F_i$. This means that $P_s \cap \{cx = d\} \subseteq F_s = \emptyset$, and so $(cx \leq d,\ cx \geq d+1)$ is Pathlike SP query for P_s. Therefore, $P_s \vdash P_s \cap \{cx \geq d+1\} \subseteq P \cap \{cx \geq d+1\}$, completing the proof. □

We generate a Pathlike SP refutation by the following recursive algorithm, which performs an *in-order* traversal of π. At each step of the recursion (corresponding to a node in π) we maintain the current polytope P we are visiting and a Pathlike SP proof Π — initially, P is the initial polytope and $\Pi = \emptyset$. We maintain the invariant that when we finish the recursive step at node P, the Pathlike SP refutation Π is a refutation of P. The algorithm is described next:

1. Let $(cx \leq d-1,\ cx \geq d)$ be the current query. Because the query is pathlike, either $cx \leq d$ or $cx \geq d-1$ is valid for P; suppose that it is $cx \geq d-1$.
2. Recursively refute $P \cap \{cx \leq d-1\} = P \cap \{cx = d-1\}$, obtaining a Pathlike SP refutation Π with t queries.
3. Apply the above Claim to deduce $P \cap \{cx \geq d\}$ from P in $t+1$ queries.
4. Refute $P \cap \{cx \geq d\}$ by using the SP refutation for the right child.

Correctness follows immediately from the Claim, and also since the size of the resulting proof is the same as the size of the SP refutation. □

Theorem 4.4.1 then follows by combining Lemma 4.4.3 with Lemma 4.4.5.

Application: Cutting Planes Proofs of Systems of Linear Equations

We end by giving one application of the characterization of Cutting Planes as a subsystem of Stabbing Planes. We show that it can be used to translate the short SP refutations of systems of linear equations over a prime finite field from Theorem 4.3.1 into Cutting Planes. This extends the result of Dadush and Tiwari [48] who gave quasipolynomial size CP refutations of the Tseitin formulas.

Corollary 1.3.6. *Let F be the CNF encoding of an unsatisfiable system of m linear equations over a prime finite field. Then there is a Cutting Planes refutation of F of size $|F|^{O(\log m)}$.*

Proof. Observe that the SP refutation from Theorem 4.3.1 is facelike. Indeed, to perform step (2) we query $(\sum_{i \in I} \alpha_i f_i \leq t-1,\ \sum_{i \in I} \alpha_i f_i \geq t)$ for $t = 1, \ldots, q^2md$. For $t = 1$, the halfspace $\sum_{i \in I} \alpha_i f_i \geq 0$ is valid for the current polytope because the polytope belongs to the $[0,1]^n$ cube. For each subsequent query, $\sum_{i \in I} \alpha_i f_i \geq t-1$ is valid because the previous query deduced $\sum_{i \in I} \alpha_i f_i \geq t-1$. Similar arguments show that the remaining queries are facelike. Thus, Lemma 4.4.5 completes the proof. □

4.4.2 Translating Stabbing Planes into Cutting Planes

In this section we give a more general application of the characterization of Cutting Planes by Facelike Stabbing Planes. We prove Theorem 1.3.5 restated below for convenience.

Theorem 1.3.5. *Let F be any unsatisfiable CNF formula on n variables and suppose that there is an SP^* refutation of F in size s and maximum coefficient size c. Then there is a Cutting Planes refutation of F of size $s(cn)^{\log s}$.*

To prove this theorem, we will show that *any* SP^* proof can be converted into a Facelike SP proof with only a quasipolynomial loss. If P is a polytope let $d(P)$ denote the *diameter* of P, which is the maximum Euclidean distance between any two points in P. Theorem 1.3.5 follows immediately from the following theorem.

Theorem 4.4.6. *Let P be a polytope and suppose there is an SP refutation of P with size s and maximum coefficient size c. Then there is a Facelike SP refutation of P in size*

$$s(c \cdot d(P)\sqrt{n})^{\log s}.$$

Proof. The theorem is by induction on s. Clearly, if $s = 1$ then the tree is a single leaf and the theorem is vacuously true.

We proceed to the induction step. Let P be the initial polytope and π be the SP proof. Consider the first query $(ax \leq b,\ ax \geq b+1)$ made by the proof, and let π_L be the SP proof rooted at the left child (corresponding to $ax \leq b$) and let π_R be the SP proof rooted at the right child. Let P_L denote the polytope at the left child and P_R denote the polytope at the right child. By induction, let π'_L and π'_R be the Facelike SP refutations for P_L and P_R guaranteed by the statement of the theorem.

Suppose w.l.o.g. that $|\pi_L| \leq |\pi|/2$. Let b_0 be the largest integer such that $ax \geq b_0$ is satisfied for any point in P. The plan is to replace the first query $(ax \leq b,\ ax \geq b+1)$ with a sequence of queries $q_0, q_1, \ldots, q_{t-1}$ such that

- For each i, $q_i = (ax \leq b_0 + i,\ ax \geq b_0 + i + 1)$.
- The query q_0 is the root of the tree and q_i is attached to the right child of q_{i-1} for $i \geq 1$.
- $q_{t-1} = (ax \leq b,\ ax \geq b+1)$.

After doing this replacement, instead of having two child polytopes P_L, P_R below the top query, we have $t+1$ polytopes $P_0, P_1, \ldots, P_{t+1}$ where $P_i = P \cap \{ax = b_0 + i\}$ and $P_{t+1} = P_R$. To finish the construction, for each $i \leq t$ use the proof π'_L to refute P_i and the proof π'_R to refute P_{t+1}.

We need to prove three statements: this new proof is a valid refutation of P, the new proof is facelike, and that the size bound is satisfied.

First, it is easy to see that this is a valid proof, since for each $i \leq t$ the polytope $P_i \subseteq P_L$ and $P_{t+1} \subseteq P_R$ — thus, the refutations π'_L and π'_R can be used to refute the respective polytopes.

Second, to see that the proof is facelike, first observe that all the queries in the subtrees π'_L, π'_R are facelike queries by the inductive hypothesis. So, we only need to verify that the new queries at the top of the proof are facelike queries, which can easily be shown by a quick induction. First, observe that the query q_0 is a facelike query, since b_0 was chosen so that $ax \geq b_0$ is valid for the polytope P. By induction, the query $q_i = (ax \leq b_0 + i,\ ax \geq b_0 + i + 1)$ is a facelike query since the polytope P_i associated with that query is $P \cap \{ax \geq b_0 + i\}$ by definition. Thus $ax \geq b_0 + i$ is valid for the polytope at the query.

Finally, we need to prove the size upper bound. Let s be the size of the original proof, s_L be the size of π_L and s_R be the size of π_R. Observe that the size of the new proof is given by the recurrence relation

$$f(s) = t \cdot f(s_L) + f(s_R).$$

where $f(1) = 1$. Since the queries $q_0, q_1, \ldots, q_{t-1}$ cover the polytope P_L with slabs of width $1/\|a\|_2$, it follows that

$$t \leq d(P_L)\|a\|_2 \leq d(P)\sqrt{n}\|a\|_\infty = d(P)c\sqrt{n}$$

where we have used that the maximum coefficient size in the proof is c. Thus, by induction, the previous inequality, and the assumption that $s_L \leq s/2$, we can conclude that the size of the proof is

$$\begin{aligned} f(s) &\leq s_L(c \cdot d(P)\sqrt{n})(c \cdot d(P_L)\sqrt{n})^{\log s_L} + s_R(c \cdot d(P_R)\sqrt{n}+)^{\log s_R} \\ &\leq s_L(c \cdot d(P)\sqrt{n})(c \cdot d(P)\sqrt{n})^{\log(s/2)} + s_R(c \cdot d(P)\sqrt{n})^{\log s} \\ &\leq s_L(c \cdot d(P)\sqrt{n})^{\log s} + s_R(c \cdot d(P)\sqrt{n})^{\log s} \\ &\leq s(c \cdot d(P)\sqrt{n})^{\log s}. \end{aligned}$$

□

Theorem 1.3.5 follows immediately, since for any CNF formula F the encoding of F as a system of linear inequalities is contained in the n-dimensional cube $[0,1]^n$, which has diameter $\sqrt{n}$. We may also immediately conclude Theorem 1.3.7 by applying the known lower bounds on the size of Cutting Planes proofs [68, 72, 86, 127]. Indeed, lower bounds on random $\Theta(\log n)$-CNF formulas follow from Chapter 3.

Application: Non-automatability of SP^*

As an application of Theorem 1.3.5 we can show that SP^* — and therefore branch-and-cut — proofs cannot be found efficiently unless $\mathsf{P} = \mathsf{NP}$. That is, they are not *automatable* as defined in Section 1.1.

Corollary 4.4.7. *The SP^* proof system is not automatable unless $\mathsf{P} = \mathsf{NP}$.*

This follows almost immediately from the recent proof of Göös et al. [78] that Cutting Planes is not automatable. Indeed, for unsatisfiable formulas F, the requisite lower bound on $\mathsf{Ref}(F) \circ \mathsf{IND}$ follows immediately from their lower bound together with Theorem 1.3.5. On the other hand, when F is satisfiable, their upper bound on $\mathsf{Ref}(F) \circ \mathsf{IND}$ in Cutting Planes can be implemented in SP^* by observing that it does not require large coefficients.

4.4.3 Towards a Topology Preserving Simulation of Cutting Planes

An artifact of both the simulations of CP by variants of SP and of SP^* by CP is that they are far from being *depth-preserving*; they convert shallow proofs into ones that are *extremely* deep. In this section, we explore whether this explosion in depth is inherent to the simulation of CP by SP (Chapter 5 is devoted to the other direction, understanding the depth of efficiently simulating SP^* by CP). While we are unable to conclusively resolve this question — indeed, at this time the only technique for proving superlogarithmic depth lower bounds on SP works equally well for CP — we provide a number of depth-preserving simulations of subsystems of CP.

To motivate our results, we will take a detour and discuss the relationship between SP, CP and real communication protocols. Presently, almost all known lower bounds for CP (including the ones from Chapter 3) are obtained by studying the communication complexity of the CNF search problem (defined in Chapter 2). For instance, it is known that:

- A depth d CP refutation yields a d-round real communication protocol for the associated CNF search problem.

- A size s treeCP refutation yields a real communication protocol $O(\log s)$-round real communication protocol for the associated CNF search problem.
- A size s and space ℓ CP refutation yields a $O(\ell \log s)$-round real communication protocol for the CNF associated search problem.
- A size s CP proof yields a dag-like real communication protocol for the associated CNF search problem.

All of these results have been used to derive strong lower bounds on Cutting Planes by proving the corresponding lower bound against the CNF search problem [28, 56, 68, 86, 100, 127]. Furthermore, this technique applies even to the stronger *semantic* CP system, as all one needs to exploit is that the lines are linear inequalities, rather than expoiting some weakness of the deduction rules. However, this strength also illustrates a weakness of current techniques, as once the lines of a proof system become expressive enough, proof techniques which work equally well for semantic proof systems break down since every tautology has a short semantic proof. Therefore, it is of key importance to develop techniques which truly exploit the "syntax" of proof systems, and not just the expressive power of the lines.

Hence, it is somewhat remarkable that we are able to show that each of the simulation results above still hold if we replace real communication protocols with SP refutations, which are syntactic objects. That is, we show

(i) A depth d CP refutation yields a depth $2d$ SP refutation.

(ii) A size s treeCP refutation yields a size $O(s)$ and depth $O(\log s)$ SP refutation.

(iii) A size s and space ℓ CP refutation yields a size $O(2^\ell s)$ and depth $O(\ell \log s)$ SP refutation.

(iv) A size s CP refutation yields a size $O(s)$ SP refutation

Simulating CP Depth

First, we exhibit a depth-preserving simulation of CP by SP, which establishes (i). Furthermore, if the proof is *tree-like* then this simulation simultaneously preserves the size.

Theorem 4.4.8. $\mathsf{depth}_{\mathsf{CP}}(F) \geq 2 \cdot \mathsf{depth}_{\mathsf{SP}}(F)$. *Moreover, for any* treeCP *refutation of depth* d *and size* s *there is an* SP *refutation of depth* $2d$ *and size* $O(s)$.

Proof. It is sufficient to prove the "moreover" part of the statement, since, by recursive doubling, any CP refutation can be converted into a treeCP refutation where the depth remains the same.

Fix a treeCP refutation of size s and depth d, and let G be the its underlying tree. We will construct an SP refutation of the same system of linear inequalities by proceeding from the root of G to the leaves. In the process, we will keep track of a subtree T of G, which we have left to simulate, and an associated "current" node v of the SP refutation that we are constructing. Along the way, the following invariant will be maintained: at every recursive step (T, v) with $T \neq G$, if the root of T is labelled with the inequality $ax \geq b$, then the edge leading to v in the SP refutation is labelled with $ax \leq b - 1$.

Initially, $T = G$ and the SP refutation contains only a single root node v. Consider a recursive step (T, v). We break into cases based on which rule was used to derive the root of T.

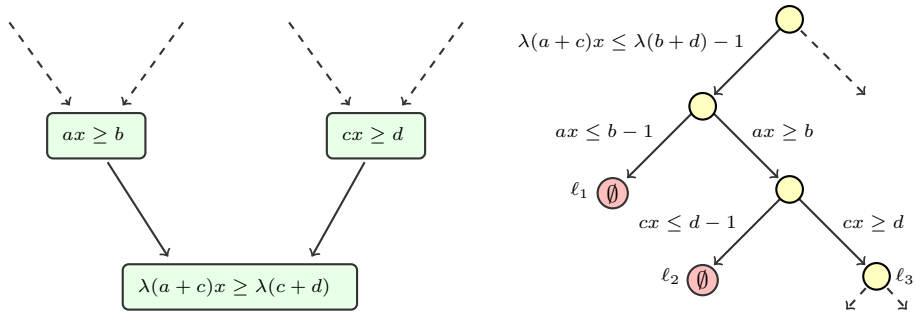


Figure 4.4: A treeR(CP) refutation invoking the conic combination rule (left) and the corresponding partial SP refutation (right).

- *Conic Combination.* Suppose that the root of T is labelled with an inequality $\lambda(a+c)x \geq \lambda(b+d)$ which was derived as a conic combination of $ax \geq b$ and $cx \geq d$. At the current node v in the SP refutation, query $(ax \leq b-1,\ ax \geq b)$. On the branch labelled with $ax \geq b$, query $(cx \leq d-1,\ cx \geq d)$. This sequence of queries results in three leaf nodes; see Figure 4.4. Let the leaf of the branch labelled with $ax \leq b-1$ be ℓ_1 and let T_1 be the subtree rooted at the child of the root of T labelled with $ax \geq b$; recurse on (T_1, ℓ_1). Similarly, for the leaf ℓ_2 of the branch labelled with $cx \leq d-1$, let T_2 be the sub-tree rooted at the child of the root of T labelled with $cx \geq d$, and recurse on (T_2, ℓ_2).

  For the final leaf, obtained by traversing the edges labelled with $ax \geq b$ and $cx \geq d$, we can derive $0 \geq 1$. To see this, first observe that if $T = G$ (i.e. the base case) then the root node of T is labelled with $0 \geq 1$ and $ax \geq b$ and $cx \geq d$ are the premises used to derive it by a conic combination. In this case, we can derive $0 \geq 1$ by the same conic combination in SP. Otherwise, by the invariant, the edge leading to v is labelled with the inequality $\lambda(a+c)x \leq \lambda(b+d)-1$. Therefore, a conic combination of this inequality with $ax \geq b$ and $cx \geq d$ yields $0 \geq 1$.

- *Division.* If the root of T is labelled with an inequality $ax \geq \lceil b/\delta \rceil$ obtained by division from $\delta ax \geq b$, then query $(\delta ax \leq b-1,\ \delta ax \geq b)$. At the leaf ℓ_1 corresponding to the edge $\delta ax \leq b-1$, let T_1 be the subtree of T rooted at the child of the root of T and recurse on (T_1, ℓ_1). At the leaf corresponding to the edge $\delta ax \geq b$ we derive $0 \geq 1$ by a conic combination with $ax \leq \lceil b/\delta \rceil - 1$, which we have already deduced by the invariant. To see this, observe that $b - \delta(\lceil b/\delta \rceil - 1) > 0$

- *Axiom.* If T is a single node — a leaf of the treeR(CP) refutation labelled with some initial inequality $ax \geq b$ of the system that it is refuting — then, by the invariant, we have already deduced $ax \leq b-1$ and this can be added to $ax \geq b$ to derive $0 \geq 1$.

To see that the SP refutation that we have constructed has depth at most twice that of the treeCP refutation, observe that conic combinations are the only inference rule of CP for which this construction requires depth 2 to simulate, while all other rules require depth 1.

To measure the size, note that every CP rule with a single premise is simulated in SP by a single query, where one of the outgoing edges of that query is immediately labelled with $0 \geq 1$. Each rule with two premises is simulated by two queries in the SP refutation, where one of the three outgoing edges is labelled with $0 \geq 1$. Therefore, the size of the SP refutation is $O(s)$. □

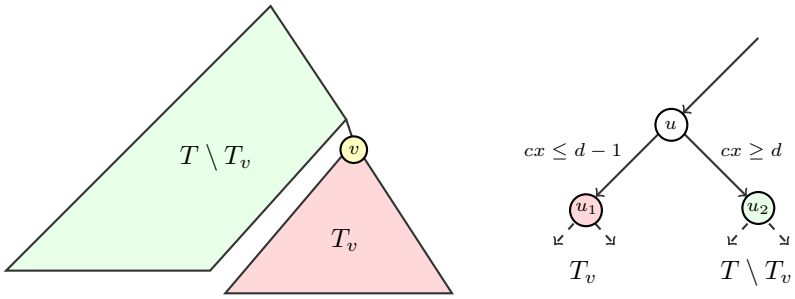


Figure 4.5: A decomposition of treeR(CP) tree T into T_v and $T \setminus T_v$ (left) and the corresponding partial SP refutation (right).

Balancing treeCP Proofs into SP

A proof system can be *balanced* if any proof of size s implies one of simultaneous size $\mathsf{poly}(s)$ depth $\mathsf{polylog}(s)$. While it is known that treeCP refutations cannot be balanced, we show next that if we permit the resulting refutation to be in SP, then we can balance. This establishes (ii).

Theorem 4.4.9. *Any size s treeCP refutation of a system of linear inequalities $Ax \geq b$ implies a size $O(s)$ and depth $O(\log s)$ SP refutation of $Ax \geq b$.*

Proof. Consider a treeCP refutation of $Ax \geq b$ and let T be its corresponding tree. As well, let $|T|$ denote the number of nodes in T. We will construct the SP refutation recursively; at each step we will keep track of a current node u in the SP proof we are constructing. The base case is when $|T| = O(1)$, in which case we can use one of the previous simulation theorems (Lemma 4.4.3 or Theorem 4.4.8) to create an SP refutation of $Ax \geq b$ satisfying these properties, and append it to u.

For the recursive step, observe that because the tree has fanin at most 2, there exists a node v in T such that the subtree T_v rooted at v satisfies $|T|/3 \leq |T_v| \leq 2|T|/3$. Let $cx \geq d$ be the line corresponding to v. At node u in the SP proof, query $(cx \leq d-1,\ cx \geq d)$. Let u_1 (resp. u_2) be the child of u obtained by following the edge labelled with $cx \leq d-1$ (resp. $cx \geq d$); see Figure 4.5. We recurse as follows:

- At u_1; Observe that the sub-proof T_v is a treeCP derivation of the inequality $cx \geq d$. Because we have deduced $cx \leq d-1$ on the path to u_1, if we also deduce $cx \geq d$ then this is sufficient to derive $0 \geq 1$. Therefore, at u_1 we recurse on the treeCP derivation T_v.
- At u_2: Observe that the sub-proof $T \setminus T_v$ is a treeCP refutation of $Ax \geq b$ where we have assumed $cx \geq d$ as an axiom. Therefore, at u_1 we recursively construct an SP refutation of the set of inequalities $\{Ax \geq b, cx \geq d\}$ using tree $T \setminus T_v$.

The size of the treeCP refutation is clearly preserved. Observe that the depth of the resulting SP refutation becomes logarithmic in s, since we are reducing the size of the proof to be simulated by a constant factor on each branch of a query. □

Balancing Low-Space CP Proofs into SP

Next,we show how to balance CP proofs into SP, provided the *space* at each step of the proof is bounded.

The space for a proof system models the amount of information that must be remembered at each state in the nondeterministic Turing machine that underlies a proof system. To capture this, we redefine a Cutting

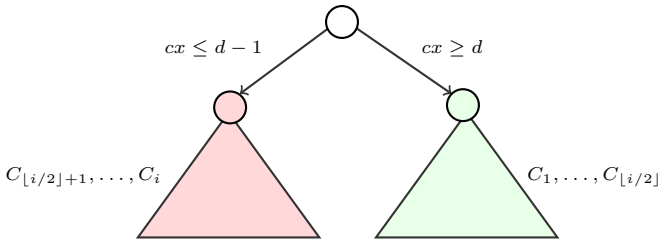


Figure 4.6: The SP tree corresponding to a configuration $C_i = \{cx \geq d\}$.

Planes refutation of a system of linear inequalities $Ax \geq b$ as a sequence of configurations $C_1, \ldots, C_s$ where a configuration C_i is a set of integer linear inequalities satisfying the following conditions: (i) $C_1 = \emptyset$, (ii) C_s contains the inequality $0 \geq 1$, (iii) each configuration C_t follows from C_{t-1} by removing any number of inequalities and including an inequality which was derived from inequalities in C_{t-1} by one of the rules of CP or an initial inequality belonging to $Ax \geq b$. The *line space* of a refutation is $\max_{i \in [s]} |C_i|$, the maximum number of inequalities in any configuration.

Theorem 4.4.10. *For any* CP *refutation of size s and line space ℓ of a system of linear inequalities $Ax \geq b$ there is an* SP *refutation of depth $O(\ell \log s)$ and size $O(s \cdot 2^\ell)$.*

This implies that strong depth lower bounds on SP proofs can lead to size-depth tradeoffs for CP.

Proof. Fix a Cutting Planes refutation $C_1, \ldots, C_\ell$ where $|C_i| \leq \ell$ for all $i \in [s]$. The theorem will follow by taking $i = s$ in the following claim.

Claim. For any $i \in [s]$ there exists an SP tree of depth $2\ell \log i$ such that every root-to-leaf path ends in a leaf labelled with $0 \geq 1$, except for one, along which we have deduced all of the inequalities in C_i.

Proof of Claim. It remains to prove the claim. If C_i contains only a single inequality $a_i x \geq b_i$ and it belongs to $Ax \geq b$, then take the tree to be the one corresponding to the SP query $(a_i x \leq b_i - 1,\ a_i x \geq b_i)$. Otherwise, the SP tree begins with a complete binary tree in which every inequality in $C_{\lfloor i/2 \rfloor}$ is queried. Exactly one path in this tree is labelled with the inequalities in $C_{\lfloor i/2 \rfloor}$, and the remaining paths contain the integer negation (i.e., $cx \leq d - 1$) of at least one inequality $cx \geq d$ in $C_{\lfloor i/2 \rfloor}$. We consider these two cases separately (seen in Figure 4.6).

In the case that a path contains a negation of a line from $C_{\lfloor i/2 \rfloor}$, we attach to its leaf the SP tree we obtain recursively by running our construction on $C_1, \ldots, C_{\lfloor i/2 \rfloor}$. The leaves of the resulting tree are all labelled with $0 \geq 1$, except for one. By construction, at the leaf not labelled with $0 \geq 1$ we have deduced all inequalities in $C_{\lfloor i/2 \rfloor}$. Since we attached this tree to a path along which we had deduced the negation of a line in $C_{\lfloor i/2 \rfloor}$, we can label this leaf with a conic combination of these inequalities equalling $0 \geq 1$. The overall depth in this case is ℓ for the initial tree and $2\ell \log(\lfloor i/2 \rfloor)$ for the tree obtained recursively. Altogether, $\ell + 2\ell \log(\lfloor i/2 \rfloor) \leq \ell(1 + 2\log(i) - 2) \leq 2\ell \log i$.

For the path labelled with the inequalities in $C_{\lfloor i/2 \rfloor}$, note that $C_{\lfloor i/2 \rfloor+1}, \ldots, C_i$ can viewed as configurations of a refutation of the original inequalities $Ax \geq b$ together with the inequalities in $C_{\lfloor i/2 \rfloor}$. At the leaf of this path we have deduced all inequalities in $C_{\lfloor i/2 \rfloor}$. Thus, we can apply the recursive construction to $C_{\lfloor i/2 \rfloor+1}, \ldots, C_i$ to refute this leaf. The overall depth is $\ell + 2\ell \log(\lfloor i/2 \rfloor) \leq \ell(1 + 2\log(i+2) - 2) \leq 2\ell \log i$. □

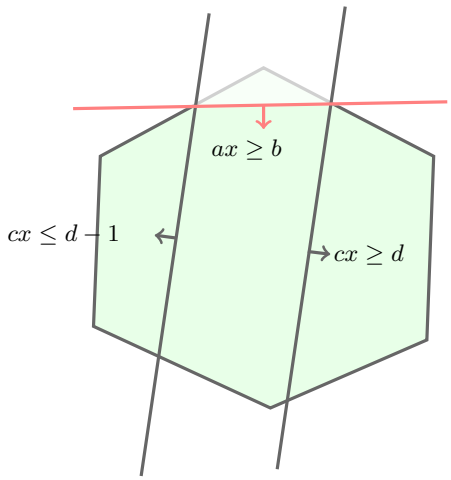


Figure 4.7: A split cut $ax \geq b$ witnessed by $(cx \leq d-1,\ cx \geq d)$ on the polytope in green.

4.4.4 Simulating non-CG Cuts

So far we have focused on the relationship between Stabbing Planes and Chvátal-Gomory cutting planes. In this section we discuss the relationship between SP and other popular types of cutting planes. First, we cover the result of Basu et al. [13] which shows that SP can simulate *split cuts*. Split cuts, which were introduced by Cook et al. [44], and form one of the most popular classes of cutting planes in practical integer linear programming. Recall that an inequality $ax \geq b$ is valid for a polytope P if for every $x^* \in P$, $ax^* \geq b$.

Split Cut. A *split cut* for a polytope P is any integer-linear inequality $ax \geq b$ for which there exists a *witnessing* pair $c \in \mathbb{Z}^n$ and $d \in \mathbb{Z}$ such that $ax \geq b$ is valid for both $P \cap \{x \in \mathbb{R}^n : cx \leq d-1\}$ and $P \cap \{x \in \mathbb{R}^n : cx \geq d\}$.

Split cuts are known to be equivalent to *mixed-integer rounding* (MIR) cuts [117] and *Gomory mixed integer* cuts [75], and generalize *lift-and-project* cuts [11]. As well, Dash [52] gave an example on which split cuts are exponentially separated from Chvátal-Gomory cuts and lift-and-project cuts. Basu et al. [13] showed that split cuts can be simulated in Stabbing Planes. For completeness, we include a proof of this.

Lemma 4.4.11 ([13]). *Let P be a polytope and let $P' = P \cap \{ax \geq b\}$ be obtained by a split cut from P. Then, there is an* SP *tree of size $O(1)$ beginning from P such that every leaf is empty, except for one whose corresponding polytope is P'.*

Proof. We simulate the deduction of P' from P in Stabbing Planes as follows:

(i) Query $(ax \leq b-1,\ ax \geq b)$

(ii) On the branch labelled with $ax \leq b-1$, query $(cx \leq d-1,\ cx \geq b)$. Observe that because $ax \geq b$ was valid for both $P \cap \{cx \geq d\}$ and $P \cap \{cx \leq d-1\}$, it follows that both $P \cap \{cx \geq d\} \cap \{ax \leq b-1\}$ and $P \cap \{cx \geq d\} \cap \{ax \leq b-1\}$ are empty.

Therefore, the only non-empty leaf is the one corresponding to P'. □

Dash [52] studied split cuts as a proof system, and showed that the Cutting Planes lower bound of Pudlák [127] could be extended to prove exponential lower bounds on the length of split cut proofs. A split cut refutation of a system of integer linear inequalities $Ax \geq b$ (representing a polytope P) is a sequence of inequalities $c_1 x \geq d_1, \ldots, c_s x \geq d_s = 0 \geq 1$ such that $c_i x \geq d_i$ is a split cut for the polytope $P \cap \{c_j x \geq d_j\}_{j<i}$. The following is immediate corollary of Lemma 4.4.4.

Corollary 4.4.12. *Stabbing Planes polynomially simulates split cut proofs.*

Proof. To simulate any split cut refutation $c_1 x \geq b_1, \ldots, c_s x \geq b_s$, we simulate each cut inductively using Lemma 4.4.4. □

Finally, we note that there exist cutting planes that cannot be efficiently simulated by SP. This is witnessed by the fact that SP cannot polynomially simulate *semantic* CP [64]. A concrete example of a type of cutting plane that SP likely cannot simulate are the *matrix cuts* of Lovász and Schrijver [110]. As we describe in Section 4.5 CP, and therefore, SP^* cannot quasipolynomially simulate the Lovasz-Schrijver proof system. However, whether this holds for SP with unbounded coefficients remains an interesting open question.

4.5 Relationship to Other Proof Systems

Having explored in depth the relationship between Cutting Planes and Stabbing Planes, we now describe how Stabbing Planes relates to other proof systems. A summary of these relationships can be seen in Figure 4.2.

Let us first note some of the separations that have already been established.

- Lower bounds for unsatisfiable systems of linear equations over prime finite fields, which are known for *Nullstellensatz* [80], the *Polynomial Calculus* [32], *Sum-of-Squares* [81, 137], AC^0*-Frege* [20, 71, 84, 124], rule out the possibility of these systems simulating CP.
- Göös et al. [77] gave an exponential separation between *Nullstellensatz* and Cutting Planes by observing that, for any unsatisfiable system of linear equations F, composing with the m-bit *index gadget* can only increase the degree of refuting F in Nullstellensatz by $O(\log m)$. On the other hand, Garg et al. [72] showed that composing any function which requires resolution refutations of *width* w when composed with the index gadget requires Cutting Planes proofs of size $n^{\Omega(w)}$. Thus, any function which requires large resolution width but small Nullstellensatz degree provides such a separation.
- Semantic CP is not polynomially verifiable, and therefore, assuming $\mathsf{P} \neq \mathsf{NP}$, no propositional proof system can simulate it. Indeed, Filmus, Hrubeš, and Lauria observed that it has $O(1)$ size refutations of unsatisfiable instances of the NP-complete subset sum problem.

We establish the remaining simulations and separations in Figure 4.2 next.

4.5.1 Equivalence Between Stabbing Planes and treelike R(CP)

The Resolution over Cutting Planes (R(CP)) proof system was introduced by Krajíček [99] as a mutual generalization of both Cutting Planes and resolution — the lines of an R(CP) proof are clauses of integer-linear inequalities, and in a single step one can take two previously derived disjunctions and either apply a Cutting Planes rule to a single inequality in the disjunctions, or apply a resolution-style "cut".

Resolution over Cutting Planes. An R(CP) *proof* of a disjunction Γ_s from a system of integer-linear inequalities $Ax \geq b$ is a sequence of disjunctions $P = \{\Gamma_i\}_{i\in[s]}$ such that each Γ_i is a disjunction which is either an inequality from $Ax \geq b$ or was derived from earlier disjunctions by one of the following deduction rules:

- *Conic Combination*. From $(ax \geq b) \vee \Gamma$ and $(cx \geq d) \vee \Gamma$ deduce $(\lambda(a+c)x \geq \lambda(b+d)) \vee \Gamma$ for any non-negative integer λ.
- *Division*. From $(ax \geq b) \vee \Gamma$ and integer δ dividing each entry of a, deduce $((a/\delta)x \geq \lceil b/\delta \rceil) \vee \Gamma$.
- *Cut*. From $(ax \geq b) \vee \Gamma$ and $(ax \leq b-1) \vee \Gamma$ derive Γ.
- *Weakening*. From Γ deduce $\Gamma \vee (ax \geq b)$
- *Axiom Introduction*. Deduce $(ax \geq b) \vee (ax \leq b-1)$ for any integer-linear inequality $ax \geq b$.
- *Elimination*. From $(0 \geq 1) \vee \Gamma$ deduce Γ.

The *size* of a proof is the number of disjunctions s in the proof and the *width* of the proof is the maximal number of inequalities in any disjunction in the proof. An R(CP) *refutation* of $Ax \geq b$ is a proof of the empty clause Λ from $Ax \geq b$. The proof system treeR(CP) is the tree-like restriction of R(CP) in which the underlying dag of the proof is required to be a tree, or equivalently, every line can be used at most once before it must be re-derived.

The main result of this sub-section is that SP is polynomially equivalent to treeR(CP).

Theorem 4.5.1. *The proof systems* SP *and* treeR(CP) *are polynomially equivalent.*

Even though SP turns out to be equivalent to a system already in the literature, this new perspective has already shown to be useful: none of aforementioned results were known for treeR(CP). Furthermore, this theorem together with Theorem 1.3.7 allows us to obtain the treeR(CP) lower bounds under mild assumptions (a bound on the magnitude of the coefficients) that were not previously known.

We will prove Theorem 4.5.1 in two parts.

Claim 4.5.2. Let $Ax \geq b$ be any system of m linear inequalities with no integer solutions. Any size s and depth d SP refutation implies a treeR(CP) refutation of size $O(s(d^2 + dm))$ and width $d+1$.

Proof. Consider an SP refutation of $Ax \geq b$ of size s and depth d. Fix any root-to-leaf path p in the refutation and let $c_1x \geq d_1, \ldots, c_tx \geq d_t$ be the sequence of linear inequalities labelling p. We will first show how to derive the clause

$$(c_1x \leq d_1 - 1) \vee \ldots \vee (c_tx \leq d_t - 1) \tag{4.1}$$

in treeR(CP). For every $i \in [t]$, using *axiom introduction*, introduce $(c_ix \leq d_i - 1) \vee (c_ix \geq d_i)$ and *weaken* it to obtain

$$(c_ix \geq d_i) \vee (c_1x \leq d_1 - 1) \vee \ldots \vee (c_tx \leq d_t - 1). \tag{4.2}$$

As well, weaken each initial inequality $a_ix \geq b_i$ in $Ax \geq b$ to

$$(a_ix \geq b_i) \vee (c_1x \leq d_1 - 1) \vee \ldots \vee (c_tx \leq d_t - 1) \tag{4.3}$$

Because p is a root-to-leaf path in the SP proof, it is labelled with a conic combination of $Ax \geq b$ and $c_i x \geq d_i$ for every $i \in [t]$ equalling $0 \geq 1$. By taking this conic combination of the first inequalities of the lines in (4.2) and (4.3) we can deduce

$$(0 \geq 1) \vee (c_1 x \leq d_1 - 1) \vee \ldots \vee (c_t x \leq d_t - 1),$$

from which we can obtain (4.1) by *elimination*.

Repeat this process to deduce (4.1) for every root-to-leaf path in the SP proof. Applying the *cut* rule appropriately to these inequalities yields the empty clause. To see this, let p and p' be two root-to-leaf paths which differ only on their leaf nodes. Then, their corresponding inequalities (4.1) are of the form

$$(c_i x \geq d_i) \vee (c_1 x \leq d_1 - 1) \vee \ldots \vee (c_{t-1} x \leq d_{t-1} - 1) \vee (c_t x \leq d_t - 1),$$
$$(c_i x \geq d_i) \vee (c_1 x \leq d_1 - 1) \vee \ldots \vee (c_{t-1} x \leq d_{t-1} - 1) \vee (c_t x \geq d_t).$$

That is, they differ in their final inequality. Applying the *cut* rule, we can deduce

$$(c_i x \geq d_i) \vee (c_1 x \leq d_1 - 1) \vee \ldots \vee (c_{t-1} x \leq d_{t-1} - 1).$$

Therefore, by repeating this process we can derive the empty clause.

Each deduction of a clause (4.1) can be done in size $O(t^2 + tm + t + m) = O(d^2 + dm)$ and has width at most $d + 1$. Thus, the size of the proof is at most $O(s(d^2 + dm))$. □

We now prove the converse.

Claim 4.5.3. Let $Ax \geq b$ be any system of m linear inequalities with no integer solutions. If there is a treeR(CP) proof of the line $(a_1 x \leq b_1 - 1) \vee \ldots \vee (a_m x \leq b_m - 1)$, where $a_i x \geq b_i$ is the ith row of $Ax \geq b$, of size s and depth d then there is an SP refutation of $Ax \geq b$ of size $O(s)$ and depth $2d$.

Proof. Fix such a treeR(CP) proof of the disjunction. For any clause $\Gamma = (c_1 x \geq d_1) \vee \ldots (c_m x \geq d_m)$, we will denote by $\neg\Gamma$ the set of inequalities $\{c_1 x \leq d_1 - 1, \ldots, c_m x \leq d_m - 1\}$. We will construct the SP refutation by structural induction, beginning at the leaves of the refutation and proceeding towards the root. At each line Γ in the proof, deduced from children Γ_1 and Γ_2, we will assume that we have constructed SP refutations $\neg\Gamma_1$ and $\neg\Gamma_2$ and use them to construct an SP refutation of $\neg\Gamma$.

First, consider a leaf of the proof which, by definition, is an *axiom introduction* of $(cx \leq d-1) \vee (cx \geq d)$ for some arbitrary integer-linear inequality $cx \geq d$. We can construct an SP refutation of $(cx \geq d) \wedge (cx \leq d - 1)$ by querying $(cx \leq d - 1,\ cx \geq d)$, and then labelling each leaves with the appropriate conic combination equalling $0 \geq 1$.

Now, let Γ be some line in the proof which was derived from earlier lines $\{\Gamma_i\}$, and suppose that we have constructed SP refutations of $\{\neg\Gamma_i\}$. To construct a refutation of $\neg\Gamma$, we break into cases based on the rule used to derive Γ.

- *Conic combination.* Let $\Gamma := (\lambda(c_1 + c_2)x \geq \lambda(d_1 + d_2)) \vee \Delta$, let $\Gamma_1 := (c_1 x \geq d_1) \vee \Delta$, and let $\Gamma_2 := (c_2 x \geq d_2) \vee \Delta$. We construct an SP refutation of $\neg\Gamma$ by first querying $(c_1 x \leq d_1 - 1,\ c_1 x \geq d_1)$. On the branch labelled with $c_1 x \geq d_1$, apply the SP refutation of $\neg\Gamma_1$. On the branch labelled with $c_1 x \geq d_1$, query $(c_2 x \leq d_2 - 1,\ c_2 x \geq d_2)$, and use the refutation of $\neg\Gamma_2$ to refute the branch labelled with $c_2 x \geq d_2$. On the remaining branch, where we have deduced $c_1 x \leq d_1 - 1$ and $c_2 x \leq d_2 - 1$, we have that $0 \geq 1$ is a conic combination with $\lambda(c_1 + c_2)x \geq \lambda(d_1 + d_2)$.

- *Division.* Let $\Gamma := ((c/\delta)x \geq \lceil d/\delta \rceil) \vee \Delta$ and let $\Gamma_1 = (\delta cx \geq d) \vee \Delta$. Query $(\delta cx \leq d-1,\ \delta cx \geq d)$. On the branch labelled with $cx \leq d-1$ we can use the refutation of $\neg\Gamma_1$. On the branch labelled with $cx \geq d$, it is enough to observe that the intersection of $cx \geq d$ and $((c/\delta)x \leq \lceil d/\delta \rceil - 1)$, provided by $\neg\Gamma$, is empty.

- *Cut.* Suppose $\Gamma := \Delta$ was derived by cutting on $\Gamma_1 := (cx \geq d) \vee \Delta$ and $\Gamma_2 := (cx \leq d-1) \vee \Delta$. Query $(cx \leq d-1,\ cx \geq d)$. On the branch labelled with $cx \leq d-1$ apply the refutation of $\neg\Gamma_1$, and on the branch labelled with $cx \geq d$ use the refutation of $\neg\Gamma_2$.

- *Weakening.* If $\Gamma := (cx \geq d) \vee \Delta)$ was derived by weakening $\Gamma_1 := \Delta$, then query $(cx \leq d-1,\ cx \geq d)$. On the branch labelled with $cx \leq d-1$ use the refutation of $\neg\Gamma_1$, and the branch labelled with $cx \geq d$ we can deduce $0 \geq 1$ by adding this inequality to $cx \leq d-1$ with is an inequality of $\neg\Gamma$.

Simulating each rule requires at most two queries, of which at most two of the children are not immediately the empty polytope. Therefore, the size of the resulting tree is at most $2s$ and the depth is at most $2d$. □

4.5.2 Stabbing Planes Simulates Tree-like DNF Resolution

Next, we show how to simulate the k-DNF resolution proof systems by variants of Stabbing Planes.

k-DNF Resolution. A $\mathsf{Res}(k)$ refutation of a CNF formula F is a sequence of k-DNF formulas $P = \{\Gamma_i\}_{i\in[s]}$ such that Γ_s is the empty clause Λ, and each Γ_i is either a clause of F or was derived from earlier DNFs by one of the following deduction rules, where a literal ℓ_i is either x_i or $\neg x_i$:

- *Cut*. From k-DNFs $A \vee (\wedge_{i\in I}\ell_i)$ and $B \vee (\vee_{i\in I}\neg\ell_i)$ deduce $A \vee B$.

- *Weakening*. From a $(k-1)$-DNF A deduce $A \vee \ell$ for any literal ℓ.

- $\wedge$*-Introduction*. From $\{A \vee \ell_i\}_{i\in I}$ deduce $A \vee (\wedge_{i\in I}\ell_i)$.

- $\wedge$*-Elimination*. From $A \vee (\wedge_{i\in I}\ell_i)$ deduce $A \vee \ell_i$ for any $i \in I$.

A refutation is *tree-like* if every deduced inequality is used at most once in the refutation (i.e., the underlying implication graph is a tree). The proof system which produces only treelike $\mathsf{Res}(k)$ refutations is denoted $\mathsf{treeRes}(k)$.

Theorem 4.5.4. *For any integer $k \geq 1$, any $\mathsf{Res}(k)$ refutation of size s implies an $\mathsf{R(CP)}$ refutation of size $O(ks)$. Similarly, any $\mathsf{treeRes}(k)$ refutation of size s implies an SP refutation of size $O(ks)$.*

The proof will follow by a straightforward application of the following claim.

Claim 4.5.5. From any disjunction $\bigvee_{i\in S}(x_i \geq 1) \vee \bigvee_{j\in T}(-x_i \geq 0)$, together with inequalities $x_i \geq 0$ and $x_i \leq 1$ for every $i \in [n]$, there is a size $O(|S|+|T|)$ $\mathsf{treeR(CP)}$ derivation of $\sum_{i\in S} x_i + \sum_{j\in T}(1-x_j) \geq 1$.

Proof. For every $v \in T \cup S$, derive $\sum_{i\in T\setminus\{v\}} x_i + \sum_{j\in S\setminus\{v\}}(1-x_j) \geq 0$ by adding together the inequalities $x_i \geq 0$ and $x_i \leq 1$. For $v \in T \cup S$ add the corresponding inequality to the disjunction in $\bigvee_{i\in S}(x_i \geq 1) \vee \bigvee_{j\in T}(-x_j \geq 0)$ containing the variable v. The result is the disjunction

$$\bigvee_{S\cup T}\Big(\sum_{i\in S} x_i + \sum_{j\in T}(1-x_j) \geq 1\Big),$$

which is the inequality $\sum_{i\in S} x_i + \sum_{j\in T}(1-x_j) \geq 1$. □

Proof of Theorem 4.5.4. We will show that R(CP) can simulate Res(k). That SP simulates treeRes(k) will follow by observing that the same proof also shows a simulation of treeRes(k) by treeR(CP), and then applying Theorem 4.5.1.

Let $\{\Gamma_i\}_{i\in[s]}$ be a Res(k) refutation of a CNF formula F, and note that the encoding of F as a system of inequalities (given by Proposition 2.2.1) includes $x_i \geq 0$ and $x_i \leq 1$ for every $i \in [n]$. We will encode each disjunction $\Gamma := \Delta_1 \vee \ldots \vee \Delta_t$ as follows: each $\Delta := (\wedge_{i\in S} x_i) \wedge (\wedge_{j\in T} \neg x_j)$ is represented by the inequality $\sum_{i\in S}(x_i - 1) + \sum_{j\in T} -x_j \geq 0$; observe that both representations are satisfied by the same set of $\{0,1\}$-assignments. Let L_Γ be the encoding of Γ obtained by replacing each Δ_i by its encoding as an inequality.

It remains to show that R(CP) can simulate the deduction rules of Res(k).

- *Cut*. Suppose that $\Gamma := A \vee B$ be deduced by cutting on $\Gamma_1 := A \vee (\wedge_{i\in S} x_i) \wedge (\wedge_{j\in T} \neg x_j)$ and $\Gamma_2 := B \vee (\vee_{i\in S} \neg x_i) \vee (\vee_{j\in T} x_j)$. As well, suppose that we have already deduced the corresponding lines $L_{\Gamma_1} := L_A \vee (\sum_{i\in S}(x_i - 1) + \sum_{j\in T} -x_j \geq 0)$ and $L_{\Gamma_2} := L_B \vee \bigvee_{i\in S}(-x_i \geq 0) \vee \bigvee_{j\in T}(x_j \geq 1)$. By Claim 4.5.5, R(CP) can reencode L_{Γ_2} as $L_B \vee (\sum_{i\in S}(1 - x_i) + \sum_{j\in T} x_j \geq 1)$, which when added to L_{Γ_1} gives $L_A \vee L_B \vee (0 \geq 1)$, which is L_Γ.

- *Weakening*. This is already a rule of R(CP).

- $\wedge$-*Introduction*. If $\Gamma := A \vee (\wedge_{i\in S} x_i) \wedge (\wedge_{j\in T} \neg x_j)$ was deduced from $\{A \vee x_i\}_{i\in S}$ and $\{A \vee \neg x_j\}_{j\in T}$ and we have already deduced $L_{\Gamma_i} := L_A \vee (x_i \geq 1)$ and $L_{\Gamma_j} := L_A \vee (-x_j \geq 0)$ for all $i \in S$ and $j \in T$. Then L_Γ can be deduced by adding together all of the L_{Γ_i} and L_{Γ_j}.

- $\wedge$-*Elimination*. If $\Gamma = A \vee x_i$ was deduced from $\Gamma_1 := A \vee (\wedge_{j\in S} x_j) \wedge (\wedge_{t\in T} x_t)$, then L_Γ can be deduced from $L_{\Gamma_1} := A \vee (\sum_{j\in S}(x_j - 1) + \sum_{t\in T} -x_t \geq 0)$ by adding the inequalities $x_j \leq 1$ for every $j \in S \setminus \{i\}$ and $x_t \geq 0$ for every $t \in T$. A similar argument holds if x_i is negated.

□

Atserias, Bonet, and Estaban [8] gave polynomial-size proofs of the *clique-coclique* formulas, for cliques of size $\Omega(\sqrt{n})$ and cocliques of size $o(\log^2 n)$. For this range of parameters, quasipolynomial size lower bounds are known [127]. This rules out the possibility of a *polynomial* simulation of R(CP) or treeRes(k) by Cutting Planes.

4.6 Lower Bounds on Unrestricted Stabbing Planes Proofs

Next, we tackle the problem of proving lower bounds on Stabbing Planes proofs with *unbounded* coefficients. First, we show that near-maximal depth lower bounds on unrestricted Stabbing Planes proofs can be obtained by a straightforward reduction to communication complexity. While we are unable to prove unrestricted size lower bounds, we explain why current techniques that would attempt to leverage the depth lower bounds fail. In doing so, we show that real communication protocols cannot be balanced by establishing the first superlogarithmic lower bound on the real communication complexity of the set disjointness function.

4.6.1 Depth Lower Bounds

In this section we prove Theorem 1.3.8, which we restate next for convenience.

Theorem 1.3.8. *There exists a family of unsatisfiable CNF formulas $\{F_n\}$ for which any* SP *refutation requires depth $\Omega(n/\log^2 n)$*

Note that every unsatisfiable CNF formula has a refutation in depth n by simply querying $(x_i \leq 0,\ x_i \geq 1)$ for all $i \in [n]$. Therefore, this lower bound is tight up to a $\log^2 n$ factor.

Our proof proceeds by a reduction to communication complexity, an approach that was pioneered in [89]. We show that from any shallow SP refutation we can extract a short *randomized* or *real* communication protocol for the associated CNF Search Problem. The lower bound follows by appealing to known lower bounds on the communication complexity of this problem.

Randomized Communication. A (bounded error) randomized communication protocol that solves a search problem $\mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y} \times \mathcal{O}$ is a distribution over deterministic communication protocols such that for every $(x, y) \in \mathcal{X} \times \mathcal{Y}$, with probability at least $2/3$, the protocol outputs o for which $(x, y, o) \in \mathcal{S}$. The randomized communication complexity of $\mathcal{S}$ is the minimum number of rounds of any randomized protocol computing $\mathcal{S}$, where the number of rounds of a randomized protocol is the maximum number of rounds of any protocol with non-zero support in the distribution.

Lemma 4.6.1. *Let $Az \geq b$ be the system of linear equations encoding an unsatisfiable CNF formula F and let (X, Y) be any partition of the variables z. Every depth d* SP *refutation of $Az \geq b$ implies a $O(d \log n + \log^2 n)$-round randomized communication protocol and a $O(d + \log n)$-round real communication protocol for solving* $\mathsf{Search}_{X,Y}(F)$.

Proof. We will first present a general procedure for solving the CNF search problem and then show how to instantiate it in both models of communication.

Fix an SP refutation of $Az \geq b$. Let Alice be given a boolean assignment to X and Bob be given a boolean assignment to Y. To solve the search problem, they will follow the root-to-leaf path through the refutation, maintaining the invariant that their joint assignment (X, Y) satisfies all of the inequalities labelling the root to leaf path. Suppose that they have arrived at a node in the refutation corresponding to a query $(cz \leq d - 1,\ cz \geq d)$. Observe that their joint assignment (X, Y) to z satisfies exactly one of these two inequalities. They will proceed down the path corresponding to the satisfied inequality, thus preserving their invariant.

Once they arrive at a leaf, they will use the conic combination of inequalities which evaluates to $0 \geq 1$ that labels it in order to search for an inequality of $Az \geq b$ (corresponding to a clause of F) which is falsified by (X, Y). Indeed, by the invariant, the only inequalities in this conic combination which could be falsified by (X, Y) are those belonging to $Az \geq b$, and a falsified inequality must exist because (X, Y) falsifies $0 \geq 1$. Let the conic combination be $\sum_{i\in[\ell]} \alpha_i c_i z \leq \sum_{i\in[\ell]} \alpha_i d$, where $\alpha_i \geq 0$. By Carathéodory's Theorem (point (ii) in Farkas' Lemma), we can assume that $\ell \leq n + 2$. To find a falsified inequality, we binary search over the conic combination: test whether $\sum_{i=1}^{\ell/2} \alpha_i c_i z \leq \sum_{i=1}^{\ell/2} \alpha_i d_i$ is falsified by (X, Y). If it is, recurse on it; otherwise, recurse on $\sum_{i=\ell/2+1}^{\ell} \alpha_i c_i z \leq \sum_{i=\ell/2+1}^{\ell} \alpha_i d_i$. Because $\ell \leq n + 2$, this process terminates in $O(\log n)$ rounds having found an inequality belonging to $Az \geq b$ which is falsified by (X, Y).

To implement this procedure in communication, it remains to show that linear inequalities can be evaluated efficiently in each of the models.

- *Real communication*: this can be done in a single round of communication. if Alice and Bob want to evaluate $c_1 x + c_2 y \geq d$, then Alice can send $d - c_1 x$ to the referee and Bob can send $c_2 y$. The referee returns whether $c_1 x \geq d - c_2 y$.

- *Randomized communication*: this can be done in $O(\log n)$ rounds of communication by combining the following two results. The first is the $O(\log b + \log 3)$ protocol of Nisan [118] for deciding a linear inequality representable in b bits. The second is a result due to Muroga [116] which states that for any linear inequality on n variables, there exists a linear inequality whose coefficients are represented in $O(n \log n)$ bits and which has the same output on points in $\{0,1\}^n$.

□

To establish Theorem 1.3.8, it remains to lower bound the communication complexity of $\mathsf{Search}(F)$. Strong lower bounds on the randomized communication complexity of the CNF search lower bound were proven by Göös and Pitassi [79]. In particular, Theorem 8.1 in [79] gives an unsatisfiable CNF formula F on $\mathsf{poly}(n)$ many clauses and partition (x, y) of the variables for which the randomized commutation complexity of $\mathsf{Search}_{x,y}(F)$ requires $\Omega(n/\log n)$ rounds. Together with Lemma 4.6.1, this establishes Theorem 1.3.8.

We remark that the formula provided by Göös and Pitassi is somewhat artificial — it is the formula obtained by *composing* the Tseitin formulas with a "versatile gadget". By Theorem 4.3.1 we know that the Tseitin formulas have $O(\log^2 n)$-depth SP refutations, and hence the hardness of these formulas of Göös and Pitassi is derived from the composition with this gadget. It remains an open problem to obtain strong lower bounds on the depth of SP refutations for more natural families of formulas.

4.6.2 Barriers to Size Lower Bounds

Next, we explore whether it is possible to leverage this depth lower bound in order to obtain size bounds.

Throughout this section, we will heavily make use of results of de Rezende, Nordström, and Vinyals [56]. They established a *lifting theorem* that translates *decision tree* lower bounds for a function $f : \{0,1\}^n \to \{0,1\}$ to lower bounds on the real communication complexity of the *composed function* $f \circ \mathsf{IND}_t^n$, which we define next. Let $\mathsf{IND}_t : [t] \times \{0,1\}^t \to \{0,1\}$ be the t-bit *index* function mapping (x, y) to y_x. The function $f \circ \mathsf{IND}_t^n$ is obtained by replacing each variable of f with a copy of IND_t^n on new variables. For any function f, composing with IND_t induces a *standard partition*, where Alice is given $x \in [t]^n$ and Bob is given $y \in \{0,1\}^{tn}$.

The *decision tree complexity* of a function f is closely related to the DPLL complexity of refuting an unsatisfiable formula. A decision tree is a binary tree in which: (i) every internal node is labelled by a variable x_i and has two outgoing edges labelled with 0 and 1, (ii) the leaves are labelled with either 0 or 1. A decision tree computes f if for every $x \in \{0,1\}^n$, the leaf obtained by following the root-to-leaf path which agrees with x is labelled with $f(x)$. The decision tree complexity of f, denoted $\mathsf{DT}(f)$, is the minimal depth of any decision tree computing f.

Theorem 4.6.2 (de Rezende et al. [56])**.** *The following statements hold:*

- *For any function $f : \{0,1\}^n \to \{0,1\}$, the real communication complexity of $f \circ \mathsf{IND}_{n^4}^n$ is at least $\mathsf{DT}(f)$.*
- *There is CNF formula F with $\mathsf{poly}(n)$ many clauses which has $\mathsf{poly}(n)$ size resolution refutation but for which any real communication protocol for $\mathsf{Search}_{x,y}(F)$ requires $\Omega(\sqrt{n^{1/4} \log n})$ rounds, for some partition of the variables.*

SP Proofs Cannot be Balanced

As an immediate corollary of Lemma 4.6.1 and Theorem 4.6.2, we show that SP proofs cannot be balanced. That is, an SP refutation of size s does not imply one of size $\mathsf{poly}(s)$ and depth $\mathsf{poly}(\log s)$. Thus, superpolynomial SP size lower bounds do not immediately follow from depth lower bounds.

Corollary 4.6.3. *There exists a CNF formula F which has $\mathsf{poly}(n)$ size SP refutations but any SP refutation requires depth $\Omega(n^{1/8}/\log n)$.*

Proof. This follows immediately by combining Theorem 4.6.2 with Lemma 4.6.1 together with the fact that SP can simulate resolution proofs. □

Real Communication Cannot be Balanced

Unlike the randomized protocols, the real communication protocols that result from Lemma 4.6.1 preserve the topology of the SP proof. That is, the *size* — the number of nodes — of the resulting real communication protocol is equivalent, up to a $\mathsf{poly}(n)$ factor, to the size of the SP refutation. Therefore, while SP proofs cannot be balanced, one might hope that the resulting real communication protocols could be, and thus size lower bounds could still be obtained from depth bounds. This is not without precedent; both deterministic and randomized communication complexity *can* be balanced. Furthermore, although it known that treeCP cannot be balanced, Impagliazzo et al. [89] show that treeCP refutations of size s can be balanced into $O(\log s)$-round randomized communication protocols for the CNF search problem.

Surprisingly, we show that real communication protocols *cannot* be balanced. To do so, we establish the first lower bound on the real communication of the *set disjointness* function, perhaps the most well-studied function in communication complexity, which we define next. Let $\mathsf{OR}_n : \{0,1\}^n \to \{0,1\}$ be the n-bit $\vee$-function and $\mathsf{AND}_2 : \{0,1\}^n \to \{0,1\}$. Then the set disjointness function, $\mathsf{DISJ}_n := \mathsf{OR}_n \circ \mathsf{AND}_2^n$, is obtained by replacing each of the n input variables of OR_n by a copy of AND_2 on new variables. As before, this function induces a *standard partition* where Alice is given one of the two input bits of each AND_2 function, and Bob is given the other.

Theorem 4.6.4. *There is a partition of the variables such that DISJ_n has a real communication protocol of size $O(n)$, but any real communication protocol requires $\Omega((n \log n)^{1/5})$ rounds of communication.*

This lower bound was subsequently improved to $\Omega(n/\log^2 n)$ in [35].

The main technique for obtaining lower bounds on the real communication complexity of a function is by a *lifting theorem*, reducing the task of proving lower bounds on certain *composed functions* to the decision tree complexity of the un-composed function. Although DISJ_n is a composed function, there is currently no lifting theorem for composition with the AND_2 function. We circumvent this by exploiting the fact that DISJ_n is *complete* for the class NP^{cc} of functions with polylogarithmic *nondeterministic communication* protocols.

Nondeterministic Communication Complexity. The *nondeterministic communication complexity* of a function $f : \{0,1\}^n \to \{0,1\}$ and a partition (X, Y), is the length of the shortest string $z \in \{0,1\}^\ell$ that can convince Alice and Bob to accept an input $(x, y) \in f^{-1}(1)$ (without communicating). That is, it is the smallest ℓ such that for every $(x, y) \in f^{-1}(1)$ there is a string $z \in \{0,1\}^\ell$ such that both Alice and Bob accept, and for every $(x, y) \in f^{-1}(0)$ and every strong $z \in \{0,1\}^\ell$, either Alice or Bob rejects.

To prove the lower bound on DISJ_n we find a function in NP^{cc} to which known lifting theorems for real communication can be applied. Then, we use NP^{cc}-completeness to transfer this lower bound to DISJ_n. The function that we will use is $\mathsf{OR}_n \circ \mathsf{IND}_t$. First, we show that this function belongs to NP^{cc}.

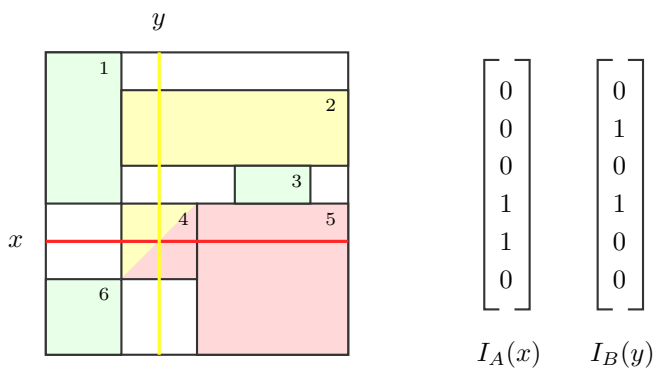


Figure 4.8: A covering of the communication matrix with monochromatic rectangles (left) and the corresponding DISJ_n instance (right).

Lemma 4.6.5. *There is a $O(\log t + \log n)$ NP^{cc} protocol computing $\mathsf{OR}_n \circ \mathsf{IND}_t^n$ for the standard partition associated with IND_t.*

Proof. Fix some input $(x, y) \in [t]^n \times \{0,1\}^{nt}$ and observe that the ith input bit to OR_n can be computed in $\log t + 1$ rounds of communication by brute-forcing the index gadget: Alice sends $x_i := x_{i,1}, \ldots, x_{i,\log t}$ to Bob who can then compute $\mathsf{IND}_t(x_i, y_i)$, where $y_i := y_{i,1}, \ldots, y_{i,t}$, and return the answer to Bob in a single bit.

Consider the following protocol NP^{cc} for $\mathsf{OR}_n \circ \mathsf{IND}_t^n$: Alice and Bob are given a $\log n$-bit string encoding the index $i \in [n]$ where $\mathsf{IND}_t(x_i, y_i) = 1$; that is, i witnesses that (x, y) is accepting input of $\mathsf{OR}_n \circ \mathsf{IND}_t^n$; see Figure 4.8. Alice and Bob verify that indeed $\mathsf{IND}_t(x_i, y_i) = 1$ by performing the above brute force protocol in $\log t$ rounds of communication. □

Lemma 4.6.6. *Let $m = n^4$, then any real communication protocol computing $\mathsf{OR}_n \circ \mathsf{IND}_t^n$ over the standard partition requires $\Omega(n \log n)$ rounds.*

Proof. Observe that the decision tree complexity of computing OR_n is n (since the *sensitivity* of OR_n is n). The proof follows by combining this with Theorem 4.6.2. □

Finally, we are ready to prove the main theorem of this section.

Proof of Theorem 4.6.4. First, we prove the lower bound. We will reduce $\mathsf{OR}_n \circ \mathsf{IND}_t^n$ for $t = n^4$ to DISJ_n. By Lemma 4.6.5 there is an cover of the 1-entries of the communication matrix of $\mathsf{OR}_n \circ \mathsf{IND}_t^n$ by at most $2nt$ monochromatic rectangles. Enumerating this rectangle covering gives us an instance of set disjointness: on input (x, y) to $\mathsf{OR}_n \circ \mathsf{IND}_t^n$, Alice and Bob construct indicator vectors $I_A(x)$ and $I_B(y)$ of the rectangles in this rectangle covering in their respective inputs lie. Then $\mathsf{OR}_n \circ \mathsf{IND}_t^n = 1$ iff $\mathsf{DISJ}(I_A(x), I_B(y)) = 1$.

This instance of DISJ_n is on $tn/2$ variables, and therefore Lemma 4.6.6 implies a lower bound of $\Omega(n \log n)$. Letting $\ell = tn$ be the total number of variables, this is a bound of the form $\ell^{1/5} \log \ell$.

For the upper bound, we give a real communication protocol for $\mathsf{DISJ}_n = \mathsf{OR}_n \circ \mathsf{AND}_2^n$ that has $O(n)$ nodes. Let x, y be the inputs given to Alice and Bob respectively. Sequentially from $i = 1, \ldots, n$, they will solve $x_i \wedge y_i$ by Alice sending x_i to the referee and Bob sending $2 - y_i$. If they discover that $x_i \wedge y_i = 0$ then they halt and output 0, otherwise they continue. □

4.7 Conclusion

We end this chapter by discussing some problems left open by this work.

Problem 4.1. An Improved Simulation. An important question left open by this work is whether the simulation of SP^* by CP can be improved. First, is it possible to remove the assumption of on the coefficients from this simulation and obtain a quasipolynomial simulation of SP? Another direction in which this simulation could be improved is by reducing the complexity: is it possible to obtain a *polynomial* simulation of SP^* by CP?

Problem 4.2. Coefficient-Preserving Simulation of Stabbing Planes. The simulation of SP^* by CP incurs incurs a significant blow-up in the coefficients due to Lemma 4.4.4, and therefore does not imply a quasipolynomial simulation of SP^* by CP^*. This is a consequence of the fact that we apply Schrijver's lemma (Lemma 4.4.4) a number of times that is proportional to the depth of the SP^* proof, with each application potentially blowing up the size of the coefficients by a factor of 2.

Problem 4.3. Relationships Between CP, SP, and R(CP). Another question is how SP and CP relate to the $\mathsf{R}(\mathsf{CP})$ proof system, which corresponds to dag-like SP. As discussed in Section 4.5, CP cannot *polynomially* simulate $\mathsf{R}(\mathsf{CP})$, since there are upper bounds on the clique-coclique formulas for certain ranges of parameters in $\mathsf{Res}(k)$ (and therefore $\mathsf{R}(\mathsf{CP})$) [8]; however, a quasipolynomial simulation has not been ruled out. A potential approach for resolving this question is to use the added expressibility of $\mathsf{R}(\mathsf{CP})$ over $\mathsf{Res}(k)$ to extend the upper bound on clique-coclique to the range of parameters for which superpolynomial CP lower bounds are known.

Chapter 5

Depth Lower Bounds and Supercritical Tradeoffs

5.1 Introduction

Depth captures the degree to which proofs — and therefore algorithms which employ the reasoning used in these proofs — can be parallelized. For proof systems such as Cutting Planes, depth is closely related to *rank measures* of polytopes, which have been extensively studied in integer programming theory.

The results from Chapter 4 suggest an interesting interplay between the depth and size of Cutting Planes proofs. In particular, we note that there are *trivial* depth n and exponential size refutation of any unsatisfiable CNF formula in Cutting Planes. However, our translation (Theorem 1.3.3) converts *shallow* Facelike SP proofs into *very* deep Cutting Planes proofs: the Stabbing Planes refutations of the Tseitin formulas have depth $O(\log^2 n)$ and quasipolynomial size, while the resulting Cutting Planes refutations have quasipolynomial size *and* depth. This is also true for the Cutting Planes refutations given by Dadush and Tiwari [48]. This is quite unusual since simulations between proof systems typically preserve the structure of the proofs. For contrast: another simulation from the literature which emphatically does *not* preserve the structure of proofs is the simulation of *bounded-size* resolution by *bounded-width* resolution by Ben-Sasson and Wigderson [22]. In this setting, it is known that this simulation is tight [29], and moreover that there exist formulas which are refutable in resolution width w but require maximal size $n^{\Omega(w)}$ [9]. Furthermore, under the additional assumption that the proofs are *tree-like*, Razborov [132] proved a *supercritical* size/width tradeoff. Roughly speaking, a supercritical tradeoff shows that if one parameter is restricted, then another must be pushed past the "critical" (i.e., worst-case) regime.

These small but deep Cutting Planes proofs bring up the possibility that the Tseitin formulas yield a supercritical *size/depth* tradeoff, which we formalize as the following conjecture.

Conjecture 1.3.9. *There exists a family of unsatisfiable formulas $\{F_n\}$ such that F_n has quasipolynomial size* CP *proofs, but any quasipolynomial-size proof requires superlinear depth. Furthermore, $\{F_n\}$ can be taken to be the Tseitin formulas on some family of graphs.*

A supercritical tradeoff for CP, roughly speaking, states that small size CP proofs must sometimes necessarily be very deep — that is, beyond the trivial depth upper bound of $O(n)$ [24, 132]. Establishing supercritical tradeoffs is a major challenge, both because hard examples witnessing such a tradeoff are rare,

and because current methods seem to fail beyond the critical (i.e., worst-case) regime. In fact, to date the only supercritical tradeoffs between size and depth for known proof systems are due to Razborov, under the additional assumption that the proofs have *bounded width*. Namely, Razborov exhibited a supercritical size-depth tradeoff for bounded width tree-like resolution [132], and then extended this result to CP proofs in which each inequality has a bounded number of distinct variables [133]. While a supercritical tradeoff for Tseitin appears to be out of reach of current techniques, "critical" lower bounds are known for Cutting Planes. Buresh Oppenheim et al. [31] gave *linear* lower bounds on the depth of Cutting Planes proofs of the Tseitin formulas.

While Cutting Planes can somewhat efficiently reason about systems of linear equations over prime finite fields, the quasipolynomial depth of these proofs prohibits any possibility of implementation. This motivates up a second important question — by how much do we need to increase the power of Cutting Planes before proofs of Tseitin, and systems of linear equations over prime finite fields, can be parallelized. A natural candidate proof system is semantic Cutting Planes, where the CG cut rule is replaced by allowing for any sound deductions from a constant number of previously derived linear inequalities. Semantic Cutting Planes is known to be extremely powerful, Filmus Hrubeš and Lauria [64] showed that there are instances that can be refuted in semantic CP but which require exponential size Cutting Planes refutations. Furthermore, its proofs are not even polynomially verifiable unless $\mathsf{P} = \mathsf{NP}$.

Contributions

In this chapter we make progress towards resolving both of these questions. Our first contribution is a linear lower bound on the depth of semantic CP refutations of the Tseitin formulas.

Theorem 5.1.1. *For all sufficiently large n, there exists a graph G on n vertices and labelling $\ell : V \to \{0,1\}$ such that any semantic Cutting Planes refutation of $\mathsf{Tseitin}(G,\ell)$ requires depth $\Omega(n)$.*

Theorem 5.1.1 is established via a new technique for proving lower bounds on the depth of semantic Cutting Planes proofs. Our technique is inspired by the result of Buresh-Oppenheim et al. [31], who proved lower bounds on the depth of Cutting Planes refutations of Tseitin by studying the *Chátal rank* of the associated polytope P. Letting $P^{(d)}$ be the polytope composed of all inequalities which can be derived in depth d in Cutting Planes. The Chátal rank of P is the minimum d such that $P^{(d)} = \emptyset$. Thus, in order to establish a depth lower bound of depth d, one would like to show the existence of a point $p \in P^{(d)}$. To do so, they give a sufficient criterion for a point p to be in $P^{(i)}$ in terms of the points in $P^{(i-1)}$. This criterion relies on a careful analysis of the specific rules of Cutting Planes, and is no longer sufficient for semantic CP. Instead, we develop an analogous criterion for semantic CP by using novel *geometric* argument (Lemma 5.3.5) which we believe will be of independent interest. Indeed, part of our motivation behind this depth bound is as a step towards proving a supercritical tradeoff in CP for Tseitin formulas.

As our second contribution, we establish supercritical size/depth tradeoffs for resolution (Res), k-DNF resolution ($\mathsf{Res}(k)$), and semantic CP (sCP).

Theorem 1.3.11. *For any constant $\varepsilon > 0$, positive integers k, n sufficiently large, $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$, and any arbitrary real parameter $1 \le c < n^{\frac{1-\varepsilon}{2+\varepsilon}}$, there is a CNF formula F on n variables and $n^{O(c)}$ clauses such that*

- *There is a $\mathcal{P}$-refutation of F of size $n^{O(c)}$.*

- *If* Π *is a* $\mathcal{P}$*-refutation of* F *with* $\mathsf{size}(\Pi) = 2^{o\left(n^{\frac{1-\varepsilon}{2+\varepsilon}}/c\right)}$ *then*

$$\mathsf{depth}(\Pi)\log^2 \mathsf{size}(\Pi) = \Omega\left(\frac{n^{c/(2+\varepsilon)}}{c\log n}\right).$$

Varying the "compression parameter" c between $O(1)$ and n^δ, for some small constant δ, allows us to obtain an, interesting family of tradeoff results. In one extreme, when $c = O(1)$ we obtain a formula F which has refutations of size $\mathsf{poly}(n)$, however any proof of size $\ll 2^{n^{1-\varepsilon}}$ must have *polynomial* depth. In the other extreme, setting $c = n^\delta$ implies an *exponential* blowup in the depth.

The main technical lemma behind these tradeoff is Razborov's supercritical width/depth tradeoff for resolution [132]. We give a simplified proof of this tradeoff in Subsection 5.4.2 which uses the "top-down" language of lifting theorems. Our tradeoffs follow by combining this with known lifting theorems [72, 139], which preserve depth.

5.1.1 Related Work

Supercritical Tradeoffs. Besides the work of Razborov [132], a number of supercritical tradeoffs have been observed in proof complexity. Perhaps most relevant for our work, Razborov [133] proved a supercritical tradeoff for Cutting Planes proofs under the assumption that each inequality has a bounded number of distinct variables (mimicking the bound on the width of each clause in the supercritical tradeoff of [132]).

A number of supercritical tradeoffs are also known between proof width and proof *space*, which is known to be closely related to depth [62, 134]. Beame et al. [14] and Beck et al. [19] exhibited formulas which admit polynomial size refutations in Resolution and the Polynomial Calculus respectively, and such that any refutation of sub-linear space necessitates a superpolynomial blow-up in size. Recently, Berkholz and Nordström [24] gave a supercritical trade-off between width and space for Resolution.

Depth in Cutting Planes and Stabbing Planes. It is widely known (and easy to prove) that any unsatisfiable family of CNF formulas can be refuted by exponential size and *linear* depth Cutting Planes. It is also known that neither Cutting Planes nor Stabbing Planes can be *balanced*, in the sense that a depth-d proof can always be transformed into a size $2^{O(d)}$ proof [15, 31]. This differentiates both of these proof systems from more powerful proof systems like Frege, for which it is well-known how to balance arbitrary proofs [43]. Furthermore, even though both the Tseitin principles and systems of linear equations in prime finite fields can be proved in both quasipolynomial-size *and* $O(\log^2 n)$ depth in Facelike SP, the simulation of Facelike SP by CP *cannot* preserve both size and depth, as the Tseitin principles are known to require depth $\Theta(n)$ to refute in CP [31].

We first recall the known depth lower bound techniques for Cutting Planes, semantic Cutting Planes, and Stabbing Planes proofs. In all of these proof systems, arguably the primary method for proving depth lower bounds is by reducing to *real communication complexity* [15, 89]; however, communication complexity is always trivially upper bounded by n, and it is far from clear how to use the assumption on the size of the proof to boost this to superlinear.

A second class of methods have been developed for *syntactic* Cutting Planes, which lower bound *rank measures* of a polytope, such as the Chvátal rank. In this setting, lower bounds are typically proven using so-called *protection lemmas* [31], which seems much more amenable to applying a small-size assumption on the proof. We also remark that for many formulas (such as the Tseitin formulas!) it is known how to

achieve $\Omega(n)$-depth lower bounds in Cutting Planes via protection lemmas, while proving even $\omega(\log n)$ lower bounds via communication complexity is impossible, due to a known folklore upper bound.

The first lower bound on the Chvátal rank was established by Chvátal et al. [36], who proved a linear bound for a number of polytopes in $[0,1]^n$. Much later, Pokutta and Schulz [126] characterized the polytopes $P \subseteq [0,1]$ with $P \cap \mathbb{Z}^n = \emptyset$ which have Chvátal rank exactly n. However, unlike most other cutting planes procedures, the Chvátal rank of polytopes $P \cap [0,1]^n$ with $P \cap \mathbb{Z}^n = \emptyset$ is not upper bounded by n. Eisenbrand and Schulz [60] showed that the Chvátal rank of any polytope $P \subseteq [0,1]^n$ is at most $O(n^2 \log n)$ and gave examples where it is $\Omega(n)$; a nearly-matching quadratic lower bound was later established by Rothvoß and Sanita [135]. For CNF formulas, the Chvátal rank is (trivially) at most n. Buresh-Oppenheim et al. [31] gave the first lower bounds on the Chvátal rank a number of CNF formulas, including an $\Omega(n)$ lower bound for the Tseitin formulas.

The rank of a number of generalizations of Cutting Planes has been studied as well. However, none of these appear to capture the strength of semantic Cutting Planes. Indeed, semantic Cutting Planes is able to refute Knapsack in a single cut, and therefore is known not to be polynomially verifiable unless $\mathsf{P} = \mathsf{NP}$ [64]. Lower bounds on the rank when using split cuts and mixed integer cuts, instead of CG cuts, was established in [47]. Pokutta and Schulz [125] obtained $\Omega(n/\log n)$ rank lower bounds on the complete tautology (which includes every clause of width n) for the broad class of *admissible cutting planes*, which includes syntactic Cutting Planes, split cuts, and many of the lift-and-project operators. Bodur et al. [26] studied the relationship between rank and integrality gaps for another broad generalization of Cutting Planes known as *aggregate cuts*.

5.2 Games for Depth and Expansion

We begin with some definitions that will be used throughout this chapter. To prove our depth lower bounds, it will be convenient to work with the following characterization of resolution depth by the *Prover-Adversary* games of Pudlák [128].

Prover–Adversary Game. The *Prover–Adversary* game associated with an n-variate formula F is played between two competing players, Prover and Adversary. The game proceeds in rounds, where in each round the state of the game is recorded by a partial assignment $\rho \in \{0,1,*\}^n$ to the variables of F. Initially the state is the empty assignment $\rho = *^n$. Then, in each round, the Prover performs the following:

- *Query.* The Prover chooses an $i \in [n]$ with $\rho_i = *$, and the Adversary chooses $b \in \{0,1\}$. The state is updated by $\rho_i \leftarrow b$ and play continues.
- *Forget.* The prover chooses a (possibly empty) set $S \subseteq [n]$ with $\rho_i \neq *$ for all $i \in S$. The state is updated by $\rho_i \leftarrow *$ for all $i \in S$.

The game ends when the state ρ falsifies an axiom of F.

It is known [128] that $\mathsf{depth}_{\mathsf{Res}}(F)$ is exactly the smallest d for which there is a Prover strategy that ends the game in d rounds, regardless of the strategy for the Adversary.

As is the case for most lower bounds in proof complexity, our depth lower bounds will, in a sense, be reductions to strong *expansion* properties of graphs which underly our formulas. Let $G = (U \cup V, E)$ and a subset $S \subseteq U \cup V$, define the *neighbourhood* of S in G as $\Gamma(S) := \{v \in U \cup V : \exists u \in S, (u,v) \in E\}$. The following definition records the expansion properties of bipartite graphs that we will use.

Boundary Expansion. For a bipartite graph $G = (U \cup V, E)$ the *boundary* of a set $W \subseteq U$ is

$$\delta(W) := \{v \in V : |\Gamma(v) \cap W| = 1\}.$$

The *boundary expansion* of a set $W \subseteq U$ is $|\delta(W)|/|W|$. The graph G is a (r, s)-*boundary expander* if the boundary expansion of every set $W \subseteq U$ with $|W| \leq r$ has boundary expansion at least s.

That is, G is a boundary expander if for any small enough subset of left-vertices, the number of *unique neighbours* is large.

5.3 Depth Lifting for Semantic Cutting Planes

In this section we develop a new method for proving depth lower bounds which we believe should be more useful for resolving the Conjecture 1.3.9. Our method works not only for CP but also for semantic CP. Using our technique, we establish the first linear lower bounds on the depth of Semantic CP refutations of the Tseitin formulas.

Lower bounds on the depth of *syntactic* CP refutations of Tseitin formulas were established by Buresh-Openheim et al. [31] using a rank-based argument. Our proof is inspired by their work, and so we describe it next. Briefly, their proof proceeds by considering a sequence of polytopes $P^{(0)} \supseteq \ldots \supseteq P^{(d)}$ where $P^{(i)}$ is the polytope defined by all inequalities that can be derived in depth i from the axioms in F; in the language of integer programming, $P^{(i)}$ is the polytope obtained by taking the *Chvátal closure* i times starting with P. The goal is to show that $P^{(d)}$ is not empty. To do so, they show that a point $p \in P^{(i)}$ is also in $P^{(i+1)}$ if for every coordinate j such that $0 < p_j < 1$, there exists points $p^{(j,0)}, p^{(j,1)} \in P^{(i)}$ such that $p^{(j,b)}_k = b$ if $k = j$ and $p^{(j,b)}_k = p_k$ otherwise. The proof of this fact is syntactic: it relies on the careful analysis of the precise rules of CP.

When working with *semantic* CP, we can no longer analyze a finite set syntactic rules. Furthermore, it is not difficult to see that the aforementioned criterion for membership in $P^{(i+1)}$ is no longer sufficient for semantic CP. We develop an analogous criterion for semantic CP given later in this section. As well, we note that the definition of $P^{(i)}$ is not well-suited to studying the depth of bounded-size CP proofs like those in Conjecture 1.3.9 — there does not appear to be a useful way to limit $P^{(i)}$ to be a polytope derived by a bounded number of halfspaces. Therefore we develop our criterion in the language of lifting, which is more amenable to supercritical tradeoffs [24, 132].

Through this section we will work with the following *top-down* definition of Semantic CP.

Semantic Cutting Planes. Let F be an n-variate unsatisfiable CNF formula. An sCP refutation of F is a directed acyclic graph of fan-out ≤ 2 where each node v is labelled with a halfspace $H_v \subseteq \mathbb{R}^n$ (understood as a set of points satisfying a linear inequality) satisfying the following:

1. *Root.* There is a unique source node r labelled with the halfspace $H_v = \mathbb{R}^n$ (corresponding to the trivially true inequality $1 \geq 0$).

2. *Internal-Nodes.* For each non-leaf node u with children v, w, we have

$$H_u \cap \{0,1\}^n \subseteq H_v \cup H_w.$$

3. *Leaves.* Each sink node u is labeled with a unique clause $C \in F$ such that $H_v \cap \{0,1\}^n \subseteq C^{-1}(0)$.

The above definition is obtained by taking a (standard) sCP proof and *reversing all inequalities*: now, a line is associated with the set of assignments *falsified* at that line, instead of the assignments *satisfying* the line. Another way to view this top down definition is that an sCP refutation is a Stabbing Planes refutation where at every node we *remember* a halfspace, rather than a polytope.

To establish the lower bound on the depth of an sCP proof we will need to find a long path in the proof. To find this path we will be take a root-to-leaf walk down the proof while constructing a partial assignment $\rho \in \{0,1,*\}^n$ to the variables. For a partial restriction ρ, denote by $\mathsf{free}(\rho) := \rho^{-1}(*)$ and $\mathsf{fix}(\rho) := [n] \setminus \mathsf{free}(\rho)$. Let the *restriction* of H by ρ be the halfspace

$$H \restriction \rho := \{x \in \mathbb{R}^{\mathsf{free}(\rho)} : \exists \alpha \in H,\ \alpha_{\mathsf{fix}(\rho)} = \rho_{\mathsf{fix}(\rho)},\ \alpha_{\mathsf{free}(\rho)} = x\}.$$

It is important to note that $H \restriction \rho$ is itself a halfspace on the *free* coordinates of ρ.

One of our key invariants needed in the proof is the following.

Good Halfspace. A halfspace $H \subseteq \mathbb{R}^n$ is *good* if it contains the all-$\frac{1}{2}$ vector. That is, $(\frac{1}{2})^n = (\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}) \in H$.

To prove our depth lower bound, we will need two technical lemmas. The first lemma shows that if a good halfspace H has its boolean points covered by halfspaces H_1 and H_2, then one of the two covering halfspaces must also be good modulo restricting a small set of coordinates.

Lemma 5.3.1. *Let $H \subseteq \mathbb{R}^n$ be any good halfspace, and suppose $H \cap \{0,1\}^n \subseteq H_1 \cup H_2$ for halfspaces H_1, H_2. Then there is a restriction ρ and an $i = 1, 2$ such that $|\mathsf{fix}(\rho)| \leq 2$ and $H_i \restriction \rho$ is good.*

The second lemma shows that good halfspaces are *robust*, in the sense that we can restrict a good halfspace to another good halfspace while also satisfying any mod-2 equation.

Lemma 5.3.2. *Let $n \geq 2$ and $H \subseteq \mathbb{R}^n$ be a good halfspace. For any $I \subseteq [n]$ with $|I| \geq 2$ and $b \in \{0,1\}$, there is a partial restriction $\rho \in \{0,1,*\}^n$ with $\mathsf{fix}(\rho) = I$ such that*

- $\bigoplus_{i \in I} \rho(x_i) = b$ *and*
- $H \restriction \rho \subseteq \mathbb{R}^{\mathsf{free}(\rho)}$ *is good.*

With these two lemmas we can already get an idea of how to construct a long path in the proof. Suppose we start at the root of the proof; the halfspace is $1 \geq 0$ (which is clearly good) and the restriction we maintain is $\rho = *^n$. We can use the first lemma to move from the current good halfspace to a good child halfspace while increasing the number of fixed coordinates by at most 2. However, we have no control over the two coordinates which are fixed by this move, and so we may fall in danger of falsifying an initial constraint. Roughly speaking, we will use the second lemma to satisfy constraints that are in danger of being falsified.

We delay the proofs of these technical lemmas to the end of the section, and first see how to prove the depth lower bounds.

5.3.1 Lifting Decision Tree Depth to Semantic CP Depth

As a warm up, we show how to lift lower bounds on resolution depth to semantic CP depth by composing with a constant-width XOR gadget. If F is a CNF formula then we can create a new formula by replacing by

replacing each variable z_i with an XOR of 4 new variables $x_{i,1}, \ldots, x_{i,4}$:

$$z_i := \mathsf{XOR}_4(x_{i,1}, \ldots, x_{i,4}) = x_{i,1} \oplus \cdots \oplus x_{i,4}.$$

We call z_i the *unlifted* variable associated with the output of the XOR_4 *gadget* applied to the i-th *block* of variables. Formally, let $\mathsf{XOR}_4^n : \{0,1\}^{4n} \to \{0,1\}^n$ be the application of XOR_4 to each 4-bit block of a $4n$-bit string. Let $F \circ \mathsf{XOR}_4^n$ denote the *lifted* formula obtained by performing this substitution on F and transforming the result into a CNF formula in the obvious way.

The main result of this section is the following.

sCP Depth Lifting Theorem. *For any unsatisfiable CNF formula F,*

$$\mathsf{depth}_{\mathsf{sCP}}(F \circ \mathsf{XOR}_4^n) \geq \frac{1}{2}\mathsf{depth}_{\mathsf{Res}}(F).$$

Our proof will rely on the Prover–Adversary game characterization of resolution depth. The proof of the sCP Depth Lifting Theorem will follow by using an optimal Adversary strategy for F to construct a long path in the sCP proof of $F \circ \mathsf{XOR}_4^n$. For this, we crucially we need to understand how halfspaces H transform under XOR_4^n:

$$\mathsf{XOR}_4^n(H) := \{z \in \{0,1\}^n : \exists x \in H \cap \{0,1\}^{4n}, \mathsf{XOR}_4^n(x) = z\}.$$

As we have already stated, we will maintain a partial assignment $\rho \in \{0,1,*\}^{4n}$ on the $4n$ *lifted* variables. However, in order to use the Adversary, we will need to convert ρ to a partial assignment on the n *unlifted* variables. To perform this conversion, for any $\rho \in \{0,1,*\}^{4n}$ define $\mathsf{XOR}_4^n(\rho) \in \{0,1,*\}^n$ as follows: for each block $i \in [n]$, define

$$\mathsf{XOR}_4^n(\rho)_i = \begin{cases} \mathsf{XOR}_4(\rho(x_{i,1}), \ldots, \rho(x_{i,4})) & \text{if } (i,j) \in \mathsf{fix}(\rho) \text{ for } j \in [4], \\ * & \text{otherwise.} \end{cases}$$

We are now ready to prove the sCP Depth Lifting Theorem.

Proof of the sCP *Depth Lifting Theorem.* Fix any semantic CP refutation of $F \circ \mathsf{XOR}_4^n$, and suppose that there is a strategy for the Adversary in the Prover-Adversary game of F certifying that F requires depth d. Throughout the walk, we maintain a partial restriction $\rho \in \{0,1,*\}^{4n}$ to the lifted variables satisfying the following three invariants with respect to the current visited halfspace H.

- *Block Closed.* In every block either all variables in the block are fixed or all variables in the block are free.
- *Good Halfspace.* $H \restriction \rho$ is good.
- *Strategy Consistent.* The unlifted assignment $\mathsf{XOR}_4^n(\rho)$ does not falsify any clause in F.

Initially, we set $\rho = *^{4n}$ and the initial halfspace is $1 \geq 0$, so the pair (H, ρ) trivially satisfy the invariants. Suppose we have reached the halfspace H in our walk and ρ is a restriction satisfying the invariants. We claim that H cannot be a leaf. To see this, suppose that H is a leaf, then by definition $H \cap \{0,1\}^{4n} \subseteq C^{-1}(0)$ for some clause $C \in F \circ \mathsf{XOR}_4^n$. By the definition of the lifted formula, this implies that $\mathsf{XOR}_4^n(H) \subseteq D^{-1}(0)$ for some clause $D \in F$. Since (H, ρ) satisfy the invariants, the lifted assignment $\mathsf{XOR}_4^n(\rho)$ does not falsify D, and so by the *block-closed* property it follows that there must be a variable $z_i \in D$ such that all lifted variables

in the block i are free under ρ. But then applying Lemma 5.3.2 to the block of variables $\{x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}\}$, we can extend ρ to a partial assignment ρ' such that $z_i = \mathsf{XOR}_4(\rho(x_{i,1}), \rho(x_{i,2}), \rho(x_{i,3}), \rho(x_{i,4}))$ satisfies D. But $H \restriction \rho'$ is a projection of $H \restriction \rho$ and so this contradicts that $\mathsf{XOR}_4^n(H)$ violates D.

It remains to show how to take a step down the proof. Suppose that we have taken $t < d/2$ steps down the Semantic CP proof, the current node is labelled with a halfspace H, and the partial assignment ρ satisfies the invariants. If H has only a single child H_1, then $H \cap \{0,1\}^{4n} \subseteq H_1 \cap \{0,1\}^{4n}$ and ρ will still satisfy the invariants for H_1. Otherwise, if H has two children H_1 and H_2 then applying Lemma 5.3.1 to the halfspaces $H \restriction \rho, H_1 \restriction \rho, H_2 \restriction \rho$ we can find an $i \in \{1,2\}$ and a restriction τ such that $H_i \restriction (\rho\tau)$ is good and τ restricts at most 2 extra coordinates. Let $i_1, i_2 \in [n]$ be the two blocks of variables in which τ restricts variables, and note that it could be that $i_1 = i_2$.

Finally, we must restore our invariants. We do this in the following three step process.

- Query the Adversary strategy at the state $\mathsf{XOR}_4^n(\rho)$ on variables z_{i_1}, z_{i_2} and let $b_1, b_2 \in \{0,1\}$ be the responses.
- For $i = i_1, i_2$ let I_i be the set of variables free in the block i, and note that $|I_i| \geq 2$. Apply Lemma 5.3.2 to $H \restriction (\rho\tau)$ and I_i to get new restrictions ρ_{i_1}, ρ_{i_2} so that blocks i_1 and i_2 both take values consistent with the Adversary responses b_1, b_2.
- Update $\rho \leftarrow \rho\tau\rho_{i_1}\rho_{i_2}$.

By Lemma 5.3.2 the new restriction ρ satisfies the block-closed and the good halfspace invariants. At each step we fix at most two blocks of variables, and thus the final invariant is satisfied as long as $t < d/2$. This completes the proof. □

5.3.2 Semantic CP Depth Lower Bounds for Unlifted Formulas

Next we show how to prove depth lower bounds directly on *unlifted* families of $\mathbb{F}_2$-linear equations. The strength of these lower bounds will depend directly on the *expansion* of the underlying constraint-variable graph of F. That is, expansion will essentially play the role that lifting did in the previous section.

Throughout this section, let F denote a set of $\mathbb{F}_2$-linear equations. In a Semantic CP proof, we must encode F as a CNF formula, but while proving the lower bound we will instead work with the underlying system of equations. For a set F of $\mathbb{F}_2$-linear equations let $G_F := (F \cup V, E)$ be the bipartite *constraint-variable* graph defined as follows. Each vertex in F corresponds to an equation in F and each vertex in V correspond to variables x_i. There is an edge $(C_i, x_j) \in E$ if x_j occurs in the equation C_i. For a subset of vertices $X \subseteq F \cup V$ define the *neighbourhood* of X in G_F as $\Gamma(X) := \{v \in F \cup V : \exists u \in X, (u,v) \in E\}$. Recall that we say that a system of linear equations F is an (r,s)-boundary expander if its constraint graph G_F is. The main result of this section is the following theorem, analogous to the sCP Depth Lifting Theorem.

Theorem 5.3.3. *For any system of $\mathbb{F}_2$-linear equations F that is an $(r, s+3)$-boundary expander,*

$$\mathsf{depth}_{\mathsf{sCP}}(F) \geq rs/2.$$

The proof of this theorem follows the proof of sCP Depth Lifting Theorem with some small changes. As before, we will maintain a partial assignment $\rho \in \{0,1,*\}^n$ that will guide us on a root-to-leaf walk through a given Semantic CP proof; we also require that each halfspace H that we visit is *good* relative to

our restriction ρ. Now our invariants are (somewhat) simpler: we will only require that $F \restriction \rho$ is a sufficiently good boundary expander.

We first prove an auxiliary lemma that will play the role of Lemma 5.3.2 in the proof of Theorem 5.3.3. We note that it follows immediately from Lemma 5.3.2 and boundary expansion.

Lemma 5.3.4. *Suppose F is a system of $\mathbb{F}_2$-linear equations that is an (r, s)-boundary expander for $s > 1$, and suppose $F' \subseteq F$ with $|F'| \leq r$. Let H be a good halfspace. Then there exists a $\rho \in \{0, 1, *\}^n$ with $\mathsf{fix}(\rho) = \Gamma(F')$ such that*

- *F' is satisfied by ρ, and*
- *$H \restriction \rho$ is good.*

Proof. We first use expansion to find, for each constraint $C_i \in F'$, a pair of variables $y_{i,1}, y_{i,2}$ that are in C_i's boundary. To do this, first observe that $|\delta(F')| \geq s|F'| > |F'|$ by the definition of boundary expansion. The pigeonhole principle then immediately implies that there are variables $y_{i,1}, y_{i,2} \in \delta(F')$ and a constraint $C_i \in F'$ such that $y_{i,1}, y_{i,2} \in C_i$. Since $y_{i,1}, y_{i,2}$ do not occur in $F' \setminus \{C_i\}$, it follows that $F' \setminus \{C_i\}$ is still an (r, s)-boundary expander. So, we update $F' = F' \setminus \{C_i\}$ and repeat the above process.

When the process terminates, we have for each constraint $C_i \in F'$ a pair of variables $y_{i,1}, y_{i,2}$ that occur *only* in C_i. Write the halfspace $H = \sum_i w_i x_i \geq c$, and let $I = \Gamma(F') \setminus \bigcup_{i \in I} \{y_{i,1}, y_{i,2}\}$ be the set of variables occurring in F' that were not collected by the above process. We define a partial restriction ρ with $\mathsf{fix}(\rho) = I$ that depends on $|I|$ as follows.

- If $|I| = 0$ then $\rho = *^n$.
- If $I = \{x_i\}$ then define $\rho(x_i) = 1$ if $w_i \geq 0$ and $\rho(x_i) = 0$ otherwise, and for all other variables set $\rho(x) = *$.
- If $|I| > 2$ then apply Lemma 5.3.2 to generate a partial restriction ρ with $\mathsf{fix}(\rho) = I$ that sets the XOR of I arbitrarily.

Observe that $H \restriction \rho$ is good. The only non-trivial case is when $|I| = 1$, but, in this case we observe that $(H \restriction \rho)((1/2)^{n-1}) = 1$ because

$$w_i \rho(x_i) + \sum_{j \neq i} w_i/2 \geq \sum_i w_i/2 \geq c,$$

where we have used that H is good and the definition of ρ.

Next we extend ρ as follows: for each $i = 1, 2, \ldots, |F'|$ apply Lemma 5.3.2 to $I_i = \{y_{i,1}, y_{i,2}\}$ to generate a partial restriction ρ_i with $\mathsf{fix}(\rho_i) = I_i$ so that the constraint $C_i \restriction \rho\rho_1 \cdots \rho_{i-1}$ is satisfied by ρ_i. Observe that this is always possible since I_i is in the boundary of C_i. Finally, we update $\rho \leftarrow \rho\rho_1 \cdots \rho_{|F'|}$. It follows by Lemma 5.3.2 that F' is satisfied by ρ and $H \restriction \rho$ is good. □

We are now ready to prove Theorem 5.3.3.

Proof of Theorem 5.3.3. We are now ready to prove Theorem 5.3.3. Fix any Semantic CP refutation of F and let n be the number of variables. We take a root-to-leaf walk through the refutation while maintaining a partial assignment $\rho \in \{0, 1, *\}^n$ and an integer valued parameter $k \geq 0$. Throughout the walk we maintain the following invariants with respect to the current halfspace H:

– *Good Expansion.* $F \restriction \rho$ is a (k,t)-boundary expander with $t > 3$.

– *Good Halfspace.* $H \restriction \rho$ is good.

– *Consistency.* The partial assignment ρ does not falsify any clause of F.

Initially, we set $k = r$, $\rho = *^n$, and $t = s + 3$, so the invariants are clearly satisfied since F is an $(r, s+3)$-expander. So, suppose that we have reached a halfspace H in our walk, and let k, ρ be parameters satisfying the invariants. We first observe that if $k > 0$ then H cannot be a sink node of the proof. To see this, it is enough to show that H contains a satisfying assignment for each equation $C \in F$. Because $H \restriction \rho$ is non-empty (since it is good) there exists a satisfying assignment in H for every equation satisfied by ρ, so, assume that C is not satisfied by ρ. In this case, since $F \restriction \rho$ is a (k,t)-expander for $k > 0$ we can apply Lemma 5.3.4 to $\{C\}$ and $H \restriction \rho$ and obtain a partial restriction τ with $\mathsf{fix}(\tau) = \Gamma(C)$ such that τ satisfies C. It follows that H is not a leaf.

Next, we show how to take a step down the proof while maintaining the invariants. If H has only a single child H_1, then $H \subseteq H_1$ and we can move to H_1 without changing ρ or k. Otherwise, let the children of H be H_1 and H_2. Applying Lemma 5.3.1 to $H \restriction \rho, H_1 \restriction \rho, H_2 \restriction \rho$ we get a partial restriction τ and an $i \in \{1,2\}$ such that $H_i \restriction \rho\tau$ is good and $|\mathsf{fix}(\tau)| \leq 2$. Due to this latter fact, since $F \restriction \rho$ is a (k,t)-expander it follows that $F \restriction \rho\tau$ is a $(k, t-2)$-expander in the worst case. Observe that since $t > 3$ it follows that $F \restriction \rho\tau$ still satisfies the consistency invariant. It remains to restore the expansion invariant.

To restore the expansion invariant, let W be the largest subset of equations such that $|W| \leq k$ and W has boundary expansion at most 3 in $F \restriction \rho\tau$, and note that W has boundary expansion at least $t-2 > 1$. Applying Lemma 5.3.4, we can find a restriction ρ' such that $W \restriction \rho\tau\rho'$ is satisfied, and $H \restriction \rho\tau\rho'$ is a good halfspace. Since W is the largest subset with expansion at most 3, it follows that $F \restriction \rho\tau\rho'$ is now a $(k-|W|, t')$-boundary expander with $t' > 3$. Suppose otherwise, then there exists a subset of equations W' which has boundary expansion at most 3 in $F \restriction \rho\tau\rho'$. Then $W \cup W'$ would have had boundary expansion at most 3 in $F \restriction \rho\tau$, contradicting the maximality of W. Now update $\rho \leftarrow \rho\tau\rho'$ and $k \leftarrow k - |W|$. Finally, we halt the walk if $k = 0$.

We now argue that this path must have had depth at least $rs/2$ upon halting. Assume that we have taken t steps down the proof. For each step $i \leq t$ let W_i be the set of equations which lost boundary expansion during the ith cleanup step. Note that $W_i \cap W_j = \emptyset$ for every $i \neq j$. Let $W^* = \cup_{i=1}^{t} W_i$, note that $|W^*| = r$ because at the ith step we decrease k by $|W_i|$. Furthermore, at the end of the walk, W^* has no neighbours and therefore no boundary in $F \restriction \rho$. Before the start of the ith cleanup step, W_i has at most $3|W_i|$ boundary variables. Therefore, at most $3|W^*| = 3r$ boundary variables were removed during the cleanup step. Since F started as an $(r, s+3)$-boundary expander, it follows that W^* had at least $r(s+3)$ boundary variables at the start of the walk. But, since *all* variables have been removed from the boundary by the end, this means that rs variables must have been removed from the boundary during the move step. Thus, as each move step sets at most 2 variables, it follows that $t \geq rs/2$ before the process halted.

□

5.3.3 Proofs of the Technical Lemmas

In this section we prove our two key technical lemmas: Lemma 5.3.1 and Lemma 5.3.2. We begin by proving the latter, which is restated next, as it is simpler.

Lemma 5.3.2. *Let $n \geq 2$ and $H \subseteq \mathbb{R}^n$ be a good halfspace. For any $I \subseteq [n]$ with $|I| \geq 2$ and $b \in \{0,1\}$, there is a partial restriction $\rho \in \{0,1,*\}^n$ with $\mathsf{fix}(\rho) = I$ such that*

- $\bigoplus_{i \in I} \rho(x_i) = b$ *and*
- $H \restriction \rho \subseteq \mathbb{R}^{\mathsf{free}(\rho)}$ *is good.*

Proof. Let H be represented by $\sum_{i \in [n]} w_i x_i \geq c$ and suppose without loss of generality that $c \geq 0$ and that $I = \{1, \ldots, k\}$. Let the weights of I in H be ordered $|w_1| \geq |w_2| \geq \ldots |w_k|$. Define ρ by setting $\rho(x_i) = *$ for $i \notin I$, for $i \leq k-1$ set $\rho(x_i) = 1$ if $w_i \geq 0$ and $\rho(x_i) = 0$ otherwise, and set $\rho(x_k)$ so that $\bigoplus_{i \in I} \rho(x_i) = b$. Clearly the parity constraint is satisfied, we show that $H \restriction \rho$ is good. This follows by an easy calculation:

$$\begin{aligned} &w_{k-1}\rho(x_{k-1}) + w_k \rho(x_k) + \sum_{i \leq k-2} w_i \rho(x_i) + \sum_{i \geq k+1} w_i/2 \\ &\geq w_{k-1}/2 + w_k/2 + \sum_{i \leq k-2} w_i \rho(x_i) + \sum_{i \geq k+1} w_i/2 \\ &\geq \sum_{i \in [n]} w_i/2 \geq c \end{aligned}$$

where the first inequality follows by averaging since $|w_{k-1}| \geq |w_k|$, and the final inequality follows since H is good. Therefore, $(H \restriction \rho)((1/2)^{[n] \setminus I}) = 1$, and $H \restriction \rho$ is good. □

In the remainder of the section we prove Lemma 5.3.1. It will be convenient to work over $\{-1,1\}^n$ rather than $\{0,1\}^n$, so, we restate it over this set and note that we can move between these basis by using the bijection $v \mapsto (1-v)/2$. We restate Lemma 5.3.1 next for $\{-1,1\}^n$.

Lemma 5.3.5. *Let $H \in \mathbb{R}^n$ be a halfspace such that $0^n \in H$ and suppose that $H \cap \{-1,1\}^n \subseteq H_1 \cup H_2$. Then one of H_1 or H_2 contains a point $y \in \{-1,0,1\}^n$ such that y has at most two coordinates in $\{-1,1\}$.*

The key ingredient in our proof of Lemma 5.3.5 is the following simple topological lemma, which will allow us to find a well-behaved point lying on a 2-face of the $\{-1,1\}^n$ cube

2-Face. A 2-face of the n-cube with vertices $\{-1,1\}^n$ are the 2-dimensional 2-by-2 squares spanned by four vertices of the cube that agree on all but two coordinates; That is, a two face is a set $A \subseteq [-1,1]^n$ such that there exists $\rho \in \{-1,1,*\}^n$ with $|\mathsf{free}(\rho)| = 2$ and $A = [-1,1]^n \restriction \rho$.

Lemma 5.3.6. *Let $w_1, w_2 \in \mathbb{R}^n$ be any pair of non-zero vectors, then we can find a vector $v \in \mathbb{R}^n$ orthogonal to w_1, w_2, such that v lies on a 2-face.*

Proof. We will construct the vector v iteratively by rounding one coordinate at a time to a $\{-1,1\}$-value until v contains exactly $n-2$ coordinates fixed to $\{-1,1\}$. At each step, we will maintain that $v \in [-1,1]^n$ and that v is orthogonal to w_1 and w_2. Therefore when the process halts v will lie on a 2-face.

Initially, set $v = 0^n$ and observe that the invariants are satisfied. Suppose that we have constructed a vector v that is orthogonal to w_1 and w_2, all of its coordinates belong to $[-1,1]$, and exactly $i < n-2$ of its coordinates belong to $\{-1,1\}$; suppose w.l.o.g. that they are the first i coordinates. We will show how to "booleanize" an additional coordinate of v. Let u be any non-zero vector that is orthogonal to $\{w_1, w_2, e_1, \ldots, e_i\}$, where e_j is the jth standard basis vector. Begin moving from v in the direction of

u and let $\alpha > 0$ be the smallest value such that one of the coordinates $j > i$ of $v + \alpha u$ is in $\{-1, 1\}$. We verify that the following properties hold:

1. The first i coordinates of $v + \alpha u$ are in $\{-1, 1\}$. This follows because we moved in a direction that is orthogonal to $e_1, \ldots, e_i$.

2. $v + \alpha u$ is orthogonal to w_1 and w_2. Let w be either of the vectors w_1 or w_2 and observe that $v_{i+1}w = v_i w + \alpha(uw) = 0$, where the final equality follows because w is orthogonal to v_i by induction and to u by assumption.

Finally, set v to be $v + \alpha u$. □

We are now ready to prove Lemma 5.3.5.

Proof of Lemma 5.3.5. Let the children H_1 and H_2 of H be given by the halfspaces $w_1 x \geq b_1$ and $w_2 x \geq b_2$ respectively. By Lemma 5.3.6 we can find a vector v which is orthogonal to w_1 and w_2, and which lies on some 2-face F of the $[-1, 1]^n$ cube corresponding to some restriction $\rho \in \{0, 1, *\}^n$. Then, v lies in (at least) one of the four 1-by-1 quadrants of the 2-face, $[0, 1]^2$, $[0, 1] \times [-1, 0]$, $[-1, 0] \times [0, 1]$, or $[-1, 0]^2$; suppose that v lies in the $[-1, 0] \times [0, 1]$ quadrant of F (see Figure 5.1), the other cases will follow by symmetry.

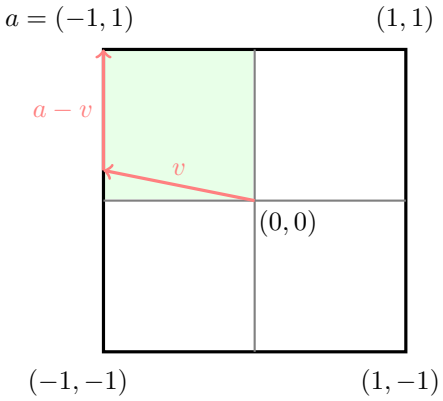


Figure 5.1: A 2-face of the n-cube together with a depiction of the booleanizing process.

Let $a \in \mathbb{R}^n$ be the vector corresponding to the $(-1, 1)$ corner of F, i.e., a is ρ extended by setting the two free bits to -1 and 1. By symmetry and the fact that H is good (and therefore $0^n \in H$), we can assume that a is contained in H — otherwise, simply exchange a and v for $-a$ and $-v$. Since $H \cap \{-1, 1\}^n \subseteq H_1 \cup H_2$ and $a \in \{-1, 1\}^n$, it follows that a is in one of H_1 or H_2. Assume that $a \in H_1$; that is, $w_1 a \geq b_1$. Our goal is to construct a vector $y \in H_1$ that satisfies the statement of the lemma. Consider the following two cases:

(i) If $w_1(a - v) \leq 0$, then it follows that $y := 0^n \in H_1$. Indeed, $w_1 y = w_1 v \geq w_1 a \geq b_1$, where first equality follows because w_1 and p are orthogonal by assumption, and the final inequality follows because $a \in H_1$.

(ii) Otherwise, we have that $w_1(a - v) > 0$. We construct a point that satisfies the statement of the lemma as follows. First, note that since $a, v \in F$, it follows that the vector $a - v$ has at most two non-zero coordinates. Beginning at the origin 0^n, move in the direction $a - v$ until a free coordinate coordinate becomes fixed to -1 or 1; that is, let $\alpha > 0$ be the minimum value such that $\alpha(a - v)$ has at most one

coordinate which is not $\{-1, 1\}$-valued. Since both a and v belong to the same 1×1 quadrant of the 2-face, $\|a - v\|_\infty \leq 1$ and so $\alpha \geq 1$. We can then verify that $\alpha(a - v) \in H_1$, since

$$w_1\alpha(a - v) = \alpha(w_1 a) - 0 \geq w_1 a \geq b_1,$$

where we have used the fact that v is orthogonal to w_1 and $\alpha \geq 1$. Finally, since $\alpha(a - v) \in H_1$ we can round the final non-zero coordinate to -1 or 1; since H_1 is a halfspace one of the two vectors will remain in H_1.

□

5.3.4 Applications

We now use the theorems from the previous sections to obtain several concrete lower bounds. First, we give strong depth lower bounds for sCP proofs of Tseitin formulas on expander graphs.

Theorem 5.1.1. *For all sufficiently large n, there exists a graph G on n vertices and labelling $\ell : V \to \{0, 1\}$ such that any semantic Cutting Planes refutation of $\mathsf{Tseitin}(G, \ell)$ requires depth $\Omega(n)$.*

Proof. A graph $G = (V, E)$ is a *γ-vertex expander* if

$$\min\{|\Gamma(W)| : W \subseteq V, |W| \leq |V|/2\} \geq \gamma|W|,$$

where $\Gamma(W)$ is the neighbourhood of W. We claim that if G is a γ-vertex expander then any Tseitin formula over G is a $(n/2, \gamma)$-boundary expander. Fix any subset W of the equations with $|W| \leq n/2$. By the definition of vertex expansion we have that $|\Gamma(W)| \geq \gamma|W|$, and since each variable is contained in exactly two constraints, it follows that the boundary of W in $\mathsf{Tseitin}(G, \ell)$ has size at least $|\delta(W)| \geq \gamma|W|$. The result then follows from Theorem 5.3.3 and the existence of strong vertex expanders G (e.g. d-regular Ramanujan graphs are at least $d/4$-vertex expanders, and exist for all d and n [113]). □

Next, we give lower bounds on the depth of Semantic CP refutations of random k-XOR and random k-CNF formulas for constant k. Recall from Chapter 3 that $\mathcal{F}(m, n, k)$ denotes the distribution on k-CNF formulas where m clauses oare chosen uniformly at random over n variables.

Definition 5.3.7. Let $\mathsf{XOR}(m, n, k)$ be the distribution on random k-XOR formulas obtained by sampling m equations from the set of all $\mod 2$ linear equations with exactly k variables.

Theorem 5.3.8. *The following holds for Semantic* CP :

1. *For any $k \geq 6$ there exists $m = O(n)$ such that $F \sim \mathsf{XOR}(m, n, k)$ requires refutations of depth at least $\Omega(n)$ with high probability.*
2. *For any $k \geq 6$ there exists $m = O(n)$ such that $F \sim \mathcal{F}(m, n, k)$ requires refutations of depth at least $\Omega(n)$ with high probability.*

Proof. We first prove (1) and obtain (2) via a reduction. Fix $m = O(n)$ so that F is unsatisfiable with high probability. For any constant k, δ and $m = O(n)$, $F \sim \mathsf{XOR}(m, n, k)$ is an $(\alpha n, k - 2 - 2\delta)$-boundary expander for some $\alpha > 0$ (see e.g. [31, 40]). Thus, setting $k \geq 6$ and ε to be some small constant, the

boundary expansion of G_F is at least 3. By Theorem 5.3.3, F requires depth $\Omega(n)$ to refute in Semantic CP with high probability.

The proof of (2) is via a reduction from $\mathcal{F}(m,n,k)$ to $\mathsf{XOR}(m,n,k)$. Every k-clause occurs in the clausal encoding of exactly one k-XOR constraint. It follows that from any k-CNF formula F we can generate a k-XOR formula whose clausal expansion F' contains F as follows: for each clause $C \in F$, if C contains an even (odd) number of positive literals then add to F' every clause on the variables of C which contains an even (odd) number of positive literals. The resulting F' is the clausal encoding of a set of $|F|$ k-XOR constraints. As there is a unique k-XOR consistent with the clauses of F, we can define the distribution $\mathsf{XOR}(m,n,k)$ equivalently as follows:

1. Sample $F \sim \mathcal{F}(m,n,k)$,
2. Return the k-XOR F' generated from F according to the aforementioned process.

It follows that the complexity of refuting $F \sim \mathcal{F}(m,n,k)$ is at least that of refuting $F' \sim \mathsf{XOR}(m,n,k)$ and (2) follows from (1) with the same parameters. □

Finally, we use Theorem 5.3.3 to extend the integrality gaps from [31] to sCP by essentially the same argument. For a linear program with constraints given by a system of linear inequalities $Ax \leq b$, the *r-round* sCP *relaxation* adds all inequalities that can be derived from $Ax \leq b$ by a depth-r sCP proof. We show that the r-round Semantic sCP linear program relaxation cannot well-approximate the number of satisfying assignments to a random k-SAT or k-XOR instance.

First we define our LP relaxations. Suppose that F is a k-CNF formula with m clauses $C_1,\ldots,C_m$ and n variables $x_1,x_2,\ldots,x_n$. If $C_i = \bigvee_{i\in P} x_i \vee \bigvee_{i\in N} \overline{x}_i$ then let $E(C_i) = \sum_{i\in P} x_i + \sum_{i\in N} 1 - x_i$. We consider the following LP relaxation of F:

$$\begin{aligned} & \max \sum_{i=1}^{m} y_i \\ \text{subject to} \quad & E(C_i) \geq y_i \quad \forall i \in [m] \\ & 0 \leq x_j \leq 1 \quad \forall j \in [n] \\ & 0 \leq y_i \leq 1 \quad \forall i \in [m] \end{aligned}$$

If F is a k-XOR formula with m constraints and n variables then we consider the above LP relaxation obtained by writing F as a k-CNF. Finally, recall that the *integrality gap* is the ratio between the optimal integral solution to a linear program and the optimal solution produced by the LP.

Theorem 5.3.9. *For any $\varepsilon > 0$ and $k \geq 6$,*

1. *There is $\kappa > 0$ and $m = O(n)$ such that for $F \sim \mathsf{XOR}(m,n,k)$ the integrality gap of the κn-round* sCP *relaxation of F is at least $(2-\varepsilon)$ with high probability.*
2. *There is $\kappa > 0$ and $m = O(n)$ such that for $F \sim \mathcal{F}(m,n,k)$ the integrality gap of the κn-round* sCP *relaxation of F is at least $2^k/(2^k-1) - \varepsilon$ with high probability.*

Proof. Let $F \sim \mathsf{XOR}(m,n,k)$ and let Y_i be the event that the ith constraint is falsified by a uniformly random assignment. Let $\delta := \varepsilon/(2-\varepsilon)$, then by a multiplicative Chernoff Bound, the probability that a uniformly random assignment satisfies at least a $1/(2-\varepsilon)$-fraction of F is $\Pr[\sum_{i\in[m]} Y_i \geq (1+\delta)\frac{m}{2}] \leq 2^{-\delta m/6}$. By

a union bound, the probability that there exists an assignment satisfying at least a $1/(2-\varepsilon)$ fraction of F is $2^{n-\delta m/6}$ which is exponentially small when $m \geq 7n(2-\varepsilon)/\varepsilon$.

On the other hand, consider the partial restriction to the LP relaxation of F that sets $y_i = 1$ for all $i \in [m]$. Setting $m \geq 7n(2-\varepsilon)/\varepsilon$ large enough, by Theorem 5.3.8 there some $\kappa > 0$ such that with high probability F requires depth κn. Hence, the κn round Semantic CP LP relaxation is non-empty, and there is a satisfying assignment $\alpha \in \mathbb{R}^n$. Thus $\alpha \cup \{y_i = 1\}$ satisfies all constraints of $\max(F)$.

The second result follows by an analogous argument. □

5.4 Supercritical Size/Depth Tradeoffs in Proof Complexity

In this section we observe supercritical size/depth tradeoffs for a number of prominent proof systems — resolution, k-DNF resolution (defined in Section 4.5), and semantic Cutting Planes. Our proof will follow the general approach of Razborov [132] who established a supercritical width/size tradeoff for tree-like resolution. The key machinery in his proof is a technique — which has been termed *hardness condensation* — that compresses the number of variables of the formula F in such a way that the depth of any *bounded width* tree-like resolution refutation of the compressed formula remains proportional to the tree-like resolution depth of refuting F.

The compression is done by composing the formula F with an XOR gadget. However, unlike the sCP Depth Lifting Theorem, the XOR gadgets will be defined on overlapping sets of variables. This will allow us to reduce the total number of variables of the composed function.

XOR Substitution. Let $G = ([N] \cup [n], E)$ be a bipartite graph and $\{y_1, \ldots, y_N\}$, $\{x_1, \ldots, x_n\}$ be sets of propositional variables. For a clause C in the variables $\{y_1, \ldots, y_N\}$ we will denote by $C \circ \mathsf{XOR}_G$ the CNF obtained from C by the $\mathbb{F}_2$-linear substitution

$$y_i \mapsto \bigoplus_{j:(i,j)\in E} x_j,$$

and then rewriting the formula in CNF (see Figure 5.2). For a CNF F, let $F \circ \mathsf{XOR}_G$ be the CNF formula that results from this substitution. If the clauses of F have width at most k and G has left-degree at most ℓ, then $F \circ \mathsf{XOR}_G$ is a CNF formula on n variables and at most $m \cdot 2^{k\ell-1}$ clauses of width at most ℓk.

Similarly, we write $F \circ \mathsf{XOR}_2^N$ to represent the CNF formula obtained by the substitution $y_i \mapsto u_i \oplus v_i$ for each i, where u_i and v_i are new variables.

If G disperses the variables of $[n]$ among the gadgets in such a way that learning the value of y_i does not tell us too much information about any y_j, then provided we cannot remember the value of too many variables x_j at once, Razborov showed that the depth of refuting $F \circ \mathsf{XOR}_G$ in tree-like resolution is proportional to the depth of refuting F [132].

We are now ready to state Razborov's hardness condensation theorem. [132]. We note that Razborov originally proved his theorem for the case of tree-like Resolution, but (as we will see) it also holds for general Resolution.

Depth Condensation Theorem. *Let F be an unsatisfiable CNF formula on N variables and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. If Π is a tree-like Resolution refutation of $F \circ \mathsf{XOR}_G$ with $\mathsf{width}(\Pi) \leq$*

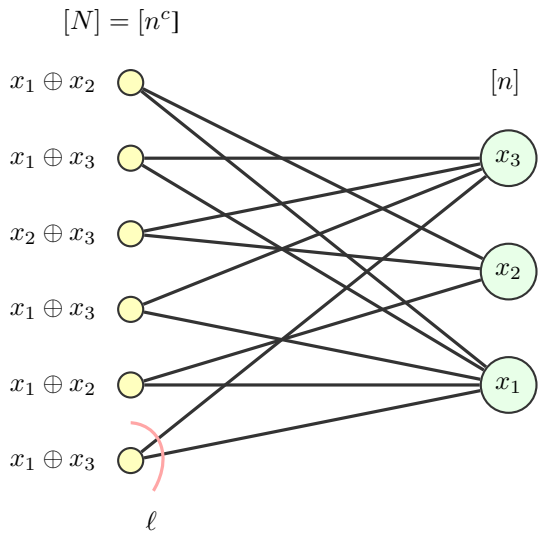


Figure 5.2: The bipartite graph G, together with the XOR constraints that each left-vertex (y-variable) is replaced with.

$r/4$, then

$$\mathsf{depth}(\Pi)\mathsf{width}(\Pi) \geq \frac{\mathsf{depth}_{\mathsf{Res}}(F)}{2}.$$

In Subsection 5.4.2, we give a simplified proof of Depth Condensation Theorem. By combining this theorem with known reductions from Resolution size to Resolution width [72, 139], we are able to establish the following depth-to-size lifting theorems. When instantiated, these will allow us to prove our tradeoffs.

Theorem 5.4.1. *Let F be any CNF formula on N variables and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. If Π is a resolution refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$ such that $\log(\mathsf{size}(\Pi) + 1) \leq r/12$, then*

$$\mathsf{depth}(\Pi) \log(\mathsf{size}(\Pi) + 1) \geq \frac{\mathsf{depth}_{\mathsf{Res}}(F)}{6}.$$

Theorem 5.4.2. *Let $k \geq 1$ be any constant, let F be any CNF formula on N variables, and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. There is a constant $\delta := \delta(k) > 0$ such that if Π is a $\mathsf{Res}(k)$ refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$ with $\log(\mathsf{size}(\Pi) + 1) \leq \delta \cdot r$, then*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega\big(\mathsf{depth}_{\mathsf{Res}}(F)\big)$$

To obtain the depth-to-size lifting theorem for semantic Cutting Planes, we will instead use the t-bit index function as our outer gadget. Recall that $\mathsf{IND}_t : [t] \times \{0,1\}^t \to \{0,1\}$ maps (x, y) to y_x. For a CNF formula on variables $z_1, \ldots, z_n$, let $F \circ \mathsf{IND}_t^n$ be the CNF formula obtained from F by the substitution $z_i \mapsto \mathsf{IND}_t(x_i, y_i)$ on new sets of variables x_i, y_i. Note that if F is an unsatisfiable k-CNF formula with m clauses, then $F \circ \mathsf{IND}_t^n$ is an unsatisfiable CNF formula on $O(nt)$ variables and $O(tm^k + n)$ clauses.

Theorem 5.4.3. *Let $\varepsilon > 0$ be any constant, let F be any CNF formula on N variables, and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. There is a constant $\delta > 0$ such that if Π is a semantic CP*

refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{IND}^n_{n^{1+\varepsilon}}$ *with* $\log(\mathsf{size}(\Pi)) \leq \delta \cdot r \log n$*, then*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega(\mathsf{depth}_{\mathsf{Res}}(F) \log^2 n).$$

We delay the proofs of these theorems until Subsection 5.4.3. Instead, we will first instantiate them to obtain our tradeoffs.

5.4.1 Parameterizing the Tradeoffs

From the previous theorems we can obtain a family of supercritical tradeoffs. To do so, we will need a formula which has small resolution refutations, but requires large depth. The canonical example of such formulas are the *pebbling formulas* of [1] on some hard-to-pebble graph H. It is known that for N-vertex graph H, Peb_H has resolution refutations of size $O(N)$ and width $O(1)$. However, resolution depth required to refute these formulas is equal to the reversible pebbling number of the graph [34]. Furthermore, there exist $O(1)$-degree graphs with reversible pebbling number $\Omega(N/\log N)$ [123].

By combining the lower and upper bounds for the pebbling formulas with the previous lemmas we can obtain a family of supercritical tradeoffs by varying the underlying expander graph $G = ([N] \cup [n], E)$. The following lemma provides us with a sufficient family of expander graphs.

Lemma 5.4.4 (Razborov [132])**.** *Let* n *be any sufficiently large positive integer, and let* N, r, ℓ *be positive integers depending on* n *such that* $\ell \geq 4$*. If*

$$rN^{4/\ell} = o(n/\ell)$$

then an $(r, 2)$ *boundary expander* $G = ([N] \cup [n], E)$ *exists with left-degree* ℓ*.*

Our tradeoffs will be in terms of the number of variables n that we are"compressing" $[N]$ into. It will be convenient to set $N = n^c$, for some real parameter $c \geq 1$, which we will call our *compression parameter*. As well, let $\varepsilon > 0$ be some arbitrarily small real parameter. We will set

$$\begin{aligned} N &:= n^c, \\ r &:= n^{1-\varepsilon}/c, \\ \ell &:= 8c/\varepsilon. \end{aligned}$$

We can verify that

$$rN^{4/\ell} = \frac{1}{c} n^{1-\varepsilon} n^{4c/\ell} \leq \frac{1}{c} n^{1-\varepsilon} n^{\varepsilon/2} = \frac{n^{1-\varepsilon/2}}{c} = o(n/\ell).$$

Now, choosing different ranges of c allow us to obtain the following interesting tradeoff results.

For convenience, we record the following proposition stating how the parameters of F transform under composition.

Proposition 5.4.5. *Let* H *be any graph on* N *vertices with indegree* $O(1)$*. Let* $G = ([N] \cup [n], E)$ *be a bipartite graph with left-degree at most* ℓ*. Then,*

- Peb_H *has* N *variables,* $N + 1$ *clauses, and width* $O(1)$*.*

- $\mathsf{Peb}_H \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$ *has* $2n$ *variables,* $n2^{O(\ell)}$ *clauses, and width* $O(\ell)$.
- $\mathsf{Peb}_H \circ \mathsf{XOR}_G \circ \mathsf{IND}_{n^{1+\varepsilon}}^n$ *has* $O(n^{2+\varepsilon})$ *variables,* $n^{O(\ell)}$ *clauses, and width* $O(\ell)$

Proof. We obtain $\mathsf{Peb}_H \circ \mathsf{XOR}_G$ by replacing each variable y_i with an XOR of at most ℓ variables. After expanding, this yields a CNF formula with $n2^{(\ell-1)\mathsf{width}(\mathsf{Peb}_H)} + n = n2^{O(\ell)}$ clauses and width $O(\ell)$. Composing this with XOR_2 has the same effect. To handle composition with the index gadget, we use the encoding of [72] which, for any k-CNF formula F on n variables and m clauses, encodes $F \circ \mathsf{IND}_{n^{1+\varepsilon}}^n$ as a $2k$-CNF formula on $O(n^{2+\varepsilon})$ variables and $O(m \cdot n^{k(1+\varepsilon)})$ clauses. For our choice of parameters, including $\varepsilon > 0$ an arbitrarily small constant, this will be $n^{O(\ell)}$. □

Now we can test different parameter regimes. In each of our regimes our tradeoffs are basically as follows: we have a trivial proof of size 2^n and depth n. However, if we demand that the proof has size $\ll 2^{n^{1-\varepsilon}}$, then the depth of the proof will explode to roughly n^c (which is *supercritical* in that it lies above the worst-case upper bound of n). Increasing c obviously increases the final depth lower bound, but since we must choose $\ell = O(c)$ it also increases the number of clauses proportionally. We first state a general tradeoff parameterized by c (a formal version of Theorem 1.3.11), and then instantiate c.

Theorem 5.4.6. *For all constants* $\varepsilon > 0$*, positive integers* k*,* n *sufficiently large,* $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$*, and arbitrary real parameter* $c \geq 1$*, there is an unsatisfiable CNF formula* $F_{\mathcal{P}}$ *on* n *variables such that*

- *If* $\mathcal{P} = \mathsf{Res}$*, then* $F_{\mathcal{P}}$ *has a resolution refutation of size* $n^c \cdot 2^{O(c)}$*. However, any resolution refutation* Π *of* $F_{\mathcal{P}}$ *with* $\mathsf{size}(\Pi) = 2^{o(n^{1-\varepsilon}/c)}$ *satisfies*

$$\mathsf{depth}(\Pi) \log \mathsf{size}(\Pi) = \Omega\left(\frac{n^c}{c \log n}\right).$$

- *If* $\mathcal{P} = \mathsf{Res}(k)$*, then* $F_{\mathcal{P}}$ *has a* $\mathsf{Res}(k)$ *refutation of size* $n^c \cdot 2^{O(c)}$*. However, any* $\mathsf{Res}(k)$ *refutation* Π *of* $F_{\mathcal{P}}$ *with* $\mathsf{size}(\Pi) = 2^{o(n^{1-\varepsilon}/c)}$ *satisfies*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega\left(\frac{n^c}{c \log n}\right)$$

- *If* $\mathcal{P} = \mathsf{sCP}$*, then* $F_{\mathcal{P}}$ *has a* sCP *refutation of size* $n^{O(c)}$*. However, any* sCP *refutation* Π *of* $F_{\mathcal{P}}$ *with* $\mathsf{size}(\Pi) = \exp\left(o\left(n^{\left(\frac{1-\varepsilon}{2+\varepsilon}\right)}/c\right)\right)$ *satisfies*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega\left(\frac{n^{c/(2+\varepsilon)} \log n}{c}\right).$$

Proof. Let the parameters N, r, ℓ be set as above, and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander whose existence is guaranteed by Lemma 5.4.4. Let H be any directed graph on N vertices with indegree $O(1)$ whose reversible pebbling number is $\Omega(N/\log N)$. Then Peb_H requires resolution refutations of depth $\Omega(N/\log N)$, but has resolution refutations of size $O(N)$. For Res and $\mathsf{Res}(k)$, let F be $\mathsf{Peb}_H \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$. The lower bounds on F follow from Theorem 5.4.1 and Theorem 5.4.2. For the upper bound, we simulate the upper bound for Peb_H: every time the resolution proof for Peb_H would query a variable y_i, we query all of the variables that y_i was replaced with in F; that is, we query all u_j, v_j such that (i, j) is an edge of G, in order to evaluate

$$\bigoplus_{j:(i,j)\in E} (u_j \oplus v_j),$$

which gives a value for y_j. Because the left-degree of G is at most ℓ, we query at most $2\ell = O(c)$ variables. This can be done in a subproof (a decision tree) of size $2^{O(c)}$. Altogether, this is a refutation of size $N \cdot 2^{O(c)} = n^c \cdot 2^{O(c)}$ of F.

For sCP, let F be $\mathsf{Peb}_H \circ \mathsf{XOR}_G \circ \mathsf{IND}^n_{n^{1+\varepsilon}}$. By Proposition 5.4.5, F has $O(n^{2+\varepsilon})$ variables. The lower bound follows from Theorem 5.4.3 together with the lower bound on Peb_H. For the upper bound, note that $\mathsf{IND}_{n^{1+\varepsilon}}$ can be evaluated in resolution by querying $(1+\varepsilon)\log n + 1$ variables (the $(1+\varepsilon)\log n$ "pointer variables" x together with the single bit y_x). Thus, by following the same strategy as before, we can simulate the resolution refutation of Peb_H by every time the Res refutation queries a variable y_i, evaluating the index gadgets of all $j \in [n]$ such that $(i,j) \in E$. As G has left-degree at most ℓ, evaluating y_i can be done by querying at most $\ell \cdot ((1+\varepsilon)\log n + 1)$ variables. This results in a Res (and therefore sCP) refutation of size $N \cdot 2^{\ell \cdot ((1+\varepsilon)\log n + 1)} = n^{O(c)}$. □

In what follows we will explore different ranges of the compression parameter c. The next corollaries are somewhat lossy, as they are stated in order to hold simultaneously for resolution, $\mathsf{Res}(k)$, and sCP. The first interesting regime is when $c = O(1)$. In this case, F is a polynomial size and constant width formula which has a trivial depth n and size 2^n proof, however any refutation of size $\ll 2^{n^{1-\varepsilon}}$, for some $\varepsilon > 0$, must have *polynomial* depth.

Corollary 5.4.7. *Let $c = O(1)$ be any constant, let $\varepsilon > 0$ be an arbitrarily small constant, and let $\Delta_{\mathsf{sCP}} = 1+\varepsilon$ and $\Delta_{\mathsf{Res}} = \Delta_{\mathsf{Res}(k)} = 0$. For any $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$ there is a CNF formula $F_{\mathcal{P}}$ on n variables, such that*

- *There is a $\mathcal{P}$-refutation of $F_{\mathcal{P}}$ of size $\mathsf{poly}(n)$.*
- *Any $\mathcal{P}$-refutation Π of F_P with $\mathsf{size}(\Pi) = \exp(o(n^{\frac{1-\varepsilon}{1+\Delta_{\mathcal{P}}}}))$ has $\mathsf{depth}(\Pi) = \tilde{\Omega}(n^{c/(1+\Delta_{\mathcal{P}})})$.*

The second interesting regime is when $c = \log^{O(1)} n$. In this case, we are compressing the number of variables quasipolynomially, and we obtain *quasipolynomial* depth lower bounds for small proofs.

Corollary 5.4.8. *Let $c = \log^{O(1)} n$ and let $\varepsilon > 0$ be an arbitrarily small constant, and let $\Delta_{\mathsf{sCP}} = 1+\varepsilon$ and $\Delta_{\mathsf{Res}} = \Delta_{\mathsf{Res}(k)} = 0$. For any $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$ there is a CNF formula $F_{\mathcal{P}}$ on n variables, such that*

- *There is a $\mathcal{P}$-refutation of $F_{\mathcal{P}}$ of size $2^{\log^{O(1)} n}$.*
- *Any $\mathcal{P}$-refutation Π of $F_{\mathcal{P}}$ has with $\mathsf{size}(\Pi) = \exp(\tilde{o}(n^{\frac{1-\varepsilon}{1+\Delta_{\mathcal{P}}}}))$ has $\mathsf{depth}(\Pi) = \tilde{\Omega}(n^{\log^{O(1)} n})$.*

Finally, we would like to test how large we can set c. The best possible compression afforded by Theorem 5.4.6 is $c = n^\delta$ for some small constant $\delta > 0$. Surprisingly, this implies an *exponential* blowup in the depth.

Corollary 5.4.9. *Let $\varepsilon > 0$ be an arbitrarily small constant, and let $\Delta_{\mathsf{sCP}} = 1+\varepsilon$ and $\Delta_{\mathsf{Res}} = \Delta_{\mathsf{Res}(k)} = 0$. For $\mathcal{P} \in \{\mathsf{Res}, \mathsf{Res}(k), \mathsf{CP}\}$ and any $0 < \delta < (1-\varepsilon)/(1+\Delta)$. There is a CNF formula $F_{\mathcal{P}}$ on n variables such that*

- *$F_{\mathcal{P}}$ has a $\mathcal{P}$-refutation of size $2^{O(n^\delta/\log n)}$.*
- *Any $\mathcal{P}$-refutation Π of $F_{\mathcal{P}}$ with $\mathsf{size}(\Pi) = \exp(o(n^{\frac{1-\varepsilon}{1+\Delta_{\mathcal{P}}} - \delta}))$ has $\mathsf{depth}(\Pi) = \exp(\Omega(n^\delta))$.*

5.4.2 Proof of the Depth Condensation Theorem

In this section we prove the Depth Condensation Theorem. To do so, it will be convenient to work with the following variant of the classic *Prover-Adversary* games of Pudlák [128]. Our variant characterizes the depth of bounded-width Resolution proofs.

Width–Bounded Prover–Adversary Game. Let $w > 0$ be an integer. A Prover–Adversary game is a *w-bounded Game* if at every step in game, the Prover's memory ρ remembers assignments to at most w variables; i.e., $|[n] \setminus \rho^{-1}(*)| \leq w$.

We will say that a game is *non-bounded* if it is not necessarily w-bounded. By a similar argument to [128], we can show the following.

Lemma 5.4.10. *For any unsatisfiable CNF formula F, there is a depth d, width w resolution refutation of F if and only if there the Prover has a strategy that ends the $(w+1)$-bounded game in d rounds, regardless of the strategy for the Adversary.*

Proof. Let Π be a resolution proof of width w and depth d. We extract a strategy for the Prover as follows: the Prover will take a root-to-leaf walk down the proof. At each step, corresponding to some clause C in the resolution proof, she will maintain that she is remembering exactly the unique falsifying assignment $\neg C$ to the clause C. If C was obtained by resolving $C_1 \vee x$ and $C_2 \vee \neg x$ then she will query the variable x. If the delayer responds with $x = 0$ then she will move to $C_1 \vee x$ and forget all assignments except for $\neg C_1 \wedge \neg x$. Proceeding in this way, we arrive at a leaf C of Π in at most d steps. By our invariant, the Prover remembers at most $w + 1$ variables at any point.

As the converse direction will not be used in our proofs, we only provide a sketch of the argument. We can view the Prover's strategy for a $(w+1)$-bounded game as a dag, where every node is labelled with the memory of the Prover at that step in the strategy, along with the variable that the Prover queries, and there are two outgoing edges labelled 0 and 1 respectively, corresponding to possible answers of the Delayer. For each node labelled with some memory ρ and variable x, we will relabel it with the clause formed by the negation of the literals fixed by ρ, which will be obtained from its children by resolving on x. □

Let us set up some notation. Let $G = ([N] \cup [n], E)$ be a bipartite graph; we will think of the left-vertices as the variables of F and the right-vertices as the variables of $F \circ \mathsf{XOR}_G$. For a set $S \subseteq [n]$ of right-vertices of G, denote by

$$\mathsf{Fixed}(S) := \{i \in [N] : \forall (i,j) \in E, j \in S\}$$

the set of left-vertices which become isolated after removing the set S of right vertices. Similarly, for a partial assignment $\rho \in \{0,1,*\}^n$ let $\mathsf{Fixed}(\rho) := \mathsf{Fixed}(S)$, where $S = [n] \setminus \rho^{-1}(*)$, be the set of variables of F which are determined by ρ. Finally, let $G \restriction \rho$ denote the graph obtained by removing all left-vertices that have been set by ρ (i.e., those in $[n] \setminus \rho^{-1}(*)$), and removing all isolated vertices.

Proof Overview. First, we give a high-level sketch of the proof. Let F be a CNF formula which requires depth d to refute in resolution. This gives us a strategy for the Adversary (for the non-bounded game) which ensures that it scores at least d points. We will use this strategy to construct a strategy for the Adversary in the w-bounded–game for $F \circ \mathsf{XOR}_G$. Ideally, we would like to proceed as follows: if the Prover (in the w-bounded game for $F \circ \mathsf{XOR}_G$) queries a variable, we would like to set it according to the Delayer strategy of the non-bounded game for F if it would determine the value of some variable y_i of F, that is, i would

be added to $\mathsf{Fixed}(\rho)$, and set it arbitrarily otherwise. However, in this setting the XOR gadgets may share variables and so variables may be correlated. To circumvent this, we will exploit *expansion*. Indeed, if G is a good enough boundary expander, then we can always set the constraints to whatever value we like. That is, for any subset $I \subseteq [N]$ of XOR-constraints, we can always find a *strong system of distinct representative variables*.

Strong SDR. If $G = ([N] \cup [n], E)$ is a bipartite graph and $I = \{I_1, \ldots, I_t\} \subseteq [N]$ then a *system of distinct representatives* (SDR) for I is a set $J = \{J_1, \ldots, J_t\} \subseteq [n]$ such that I and J form a matching where $(I_i, J_i) \in E$ for all $i \in [t]$. The SDR of I is *strong* if, furthermore, I_i is not adjacent to J_j for all $j > i$.

The following lemma can be viewed as a strengthening of the claim that expanders have matchings on small sets.

Lemma 5.4.11. *If $G = ([N] \cup [n], E)$ is an $(r/2, 1/2)$-boundary expander, then any $I \subseteq [N]$ with $|I| \leq r/2$ has a strong SDR.*

Proof. For $i = 1 \ldots t$ perform the following. Because I has boundary at least $|I|/2$ within G, by the pigeonhole principle there exists $\ell \in I$ and a column $j \in [n]$ such that $(\ell, j) \in E$ and $(\ell', j) \notin E$ for every $\ell' \in I$ with $\ell' \neq \ell$; fix $I_i := \ell$ and $J_i := j$. Set G to be the graph obtained by removing vertices I_i and J_i and any edge incident to either of them, and update $I \leftarrow I \setminus I_i$. Because J_i was not adjacent to any vertex besides I_i, removing J_i does not decrease the expansion of I in G and we can recurse. □

If a partial restriction $\rho \in \{0, 1, *\}^n$ (thought of as the Prover's memory) sets some variables, this may decrease the boundary expansion of the current graph $G \restriction \rho$. Therefore, at each step of the simulation, the Delayer will track a *closure* of ρ which will set some additional variables, but will ensure that the residual graph is a good boundary expander. The following operator will allow us to restore the expansion of G after removing a subset of the vertices.

Closure Operator. For a $J \subseteq [n]$, denote by $G \setminus J$ the graph obtained by taking the subgraph induced by the vertex set $[N] \cup ([n] \setminus J)$ and removing any isolated vertices (i.e. y_i for which $i \in \mathsf{Fixed}(J)$) from $[N]$.

The following lemma states that for any small J there is a *closure* $\mathrm{Cl}(J) \supseteq J$ such that $G \setminus \mathrm{Cl}(J)$ is still expanding; a proof can be found in [132] (Lemma 2.3), building on ideas in [3, 139].

Lemma 5.4.12. *Let $G = ([m] \cup [n], E)$ be an $(r, 2)$-boundary expander. For every $J \subseteq [n]$ with $|J| \leq r/4$ there exists $\mathrm{Cl}(J) \supseteq J$ such that $|\mathsf{Fixed}(\mathrm{Cl}(J))| \leq 2|J|$ and $G \setminus \mathrm{Cl}(J)$ is an $(r/2, 3/2)$-boundary expander.*

We are now ready to prove the main theorem of this section.

Proof of the Depth Condensation Theorem. Fix an optimal strategy D for the Delayer (in the unbounded game) which scores at least d points on F. We will construct a Delayer strategy for the w-bounded game which scores at least $d/2w$ points. We will denote the Prover's memory at each step in the bounded game by $\rho \in \{0, 1, *\}^n$. For convenience, we will also track an assignment to the closure $\mathrm{Cl}(\rho)$ of ρ, which we denote by $\rho^{\mathrm{Cl}} \in \{0, 1, *\}^n$. In each round ρ in the game, we will maintain the following invariants:

- *Closed*: $\rho^{\mathrm{Cl}}_i \in \{0, 1\}$ iff $i \in \mathrm{Cl}(\rho)$.
- *Expanding.* $G \setminus \mathrm{Cl}(\rho)$ is an $(r/2, 3/2)$-boundary expander.
- *Satisfying Closure.* ρ^{Cl} does not falsify any of the constraint of $F \circ \mathsf{XOR}_G$.

Initially $\rho = \rho^{\mathrm{Cl}} = *^n$ and the invariants are satisfied.

At each round we will query the Delayer strategy D at most w times. Suppose that, that we (as the Delayer for the bounded game) have scored at most $d/2w - 1$ points and the invariants are satisfied, then we can claim we can continue for another round and restore the invariants. First, we handle ρ.

- *Querying.* Let x_i be the variable that the Prover queries at this step. We will set x_i to a value $\alpha \in \{0, 1\}$, denoting the resulting restriction $\rho^{i \leftarrow \alpha}$ according to one of these two cases:

  1. If $i \in \mathrm{Cl}(\rho)$ then the Delayer sets $\alpha \leftarrow \rho_i^{\mathrm{Cl}}$.
  2. If $i \notin \mathrm{Cl}(\rho)$ then choose α arbitrarily.

  The Delayer sets $\rho_i^{\mathrm{Cl}} \leftarrow \alpha$.

  Note that by the expanding assumption, each y_j with $j \notin \mathsf{Fixed}(\mathrm{Cl}(\rho))$ had at least two free variables before x_i was set, and therefore setting x_i does not change the set $\mathsf{Fixed}(\mathrm{Cl}(\rho))$. Because $G \setminus \mathrm{Cl}(\rho)$ was an $(r/2, 3/2)$-boundary before setting x_i, after setting it, $G \setminus (\mathrm{Cl}(\rho) \cup \{i\})$ is at least an $(r/2, 1/2)$-boundary expander.

- *Forgetting.* If the Prover chooses to forget the value of a variable x_i, then set $\rho_i^{i \leftarrow \alpha} = *$.

It remains to restore the invariants by updating ρ^{Cl} to fix exactly the variables in $\mathrm{Cl}(\rho_i^{i \leftarrow \alpha})$. Denote by

$$I = \{I_1, \ldots, I_t\} := \mathsf{Fixed}(\mathrm{Cl}(\rho^{i \leftarrow \alpha})) \setminus \mathsf{Fixed}(\mathrm{Cl}(\rho))$$

the set of indices of left-vertices y_{I_j} (XOR-constraints) whose neighbours (variables) are all contained within $\mathrm{Cl}(\rho^{i \leftarrow \alpha})$, but are not contained within $\mathrm{Cl}(\rho)$. Because $G \setminus \mathrm{Cl}(\rho \cup \{i\})$ is an $(r/2, 1/2)$-boundary expander, by Lemma 5.4.12 we can find a strong SDR $J = \{J_1, \ldots, J_t\}$ for I.

We are now ready to restore the invariants. For every $j \in \mathrm{Cl}(\rho^{i \leftarrow \alpha}) \setminus J$, set ρ_j^{Cl} arbitrarily. It remains to set the strong SDR J, which we will do according to the Delayer strategy D. For $\ell = 1, \ldots, t$ perform the following: Query the Delayer strategy D for the response $\beta \in \{0, 1\}$ when the current state is $\mathsf{XOR}_G(\rho^{\mathrm{Cl}})$ and the Prover is querying the variable y_{I_ℓ} (of F). Observe that because we have already fixed all of the other variables in the neighbourhood of y_{I_ℓ}, the only free variable in the constraint corresponding to y_{I_ℓ} is x_{J_ℓ}. Fix $\rho_{J_\ell}^{\mathrm{Cl}}$ so that

$$\bigoplus_{j:(I_\ell,k)\in E} x_j = \beta.$$

Note that $|\mathsf{Fixed}(\rho^{i \leftarrow \alpha})| \leq w$ by definition of a w-bounded game. Thus, by Lemma 5.4.12, $|\mathsf{Fixed}(\mathrm{Cl}(\rho^{i \leftarrow \alpha}))| \leq 2w$, and we can conclude that we query the Delayer strategy at most $2w$ times during this round. Because we have scored less than $d/2w - 1$ points, the Delayer has answered at most $d - 2w$ queries so far, and therefore the Delayer can still provide answers to these at most $2w$ queries which do not falsify any constraints of F, and therefore of $F \circ \mathsf{XOR}_G$. Set $\rho \leftarrow \rho^{i \leftarrow \alpha}$; we have restored our invariants. □

5.4.3 Proofs of the Tradeoffs

5.4.4 Proof of the Resolution Tradeoff

We begin with Theorem 5.4.1, which we restate next for convenience.

Theorem 5.4.1. *Let F be any CNF formula on N variables and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. If Π is a resolution refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$ such that $\log(\mathsf{size}(\Pi) + 1) \leq r/12$, then*

$$\mathsf{depth}(\Pi) \log(\mathsf{size}(\Pi) + 1) \geq \frac{\mathsf{depth}_{\mathsf{Res}}(F)}{6}.$$

We require the following simple size-width lifting theorem for resolution.

Lemma 5.4.13. *Let F be any unsatisfiable CNF formula. For any resolution refutation Π^* of $F \circ \mathsf{XOR}_2^n$ there is a resolution refutation Π of F such that*

$$\mathsf{width}(\Pi) \leq 3 \log(\mathsf{size}(\Pi^*) + 1),$$
$$\mathsf{depth}(\Pi) \leq \mathsf{depth}(\Pi^*).$$

Proof. Let $x_1, \ldots, x_n$ be the variables of F and let $u_1, v_1, \ldots, u_n, v_n$ be the variables of $F \circ \mathsf{XOR}_2^n$. Let $\mathcal{D}$ be the collection of partial restrictions $\rho \in \{0, 1, *\}$ that, for every $i \in [n]$, set exactly one of u_i or v_i to a value in $\{0, 1\}$ and leave the other unset. Denote by $\rho \sim \mathcal{D}$ sampling a restriction ρ uniformly at random from $\mathcal{D}$. It is easy to see that for any resolution refutation Π^* of $F \circ \mathsf{XOR}_2^n$ and any $\rho^* \in \mathcal{D}$, $\Pi^* \restriction \rho$ is a resolution refutation of F, and furthermore by closure under restrictions it follows that $\mathsf{depth}(\Pi^* \restriction \rho) \leq \mathsf{depth}(\Pi^*)$.

Let t be a positive integer to be set later. For any clause C of $\mathsf{width}(C) \geq t$ in Π^*, it follows that for $\rho \sim \mathcal{D}$, the probability $C \restriction \rho$ is not satisfied is at most $(3/4)^t$. By a union bound, it follows that the probability that $\Pi^* \restriction \rho$ has a clause of width $\geq t$ is at most $\mathsf{size}(\Pi^*)(3/4)^t$, which is strictly less than 1 as long as $\mathsf{size}(\Pi^*) \leq (4/3)^t$. Choosing $t = \log_{4/3}(\mathsf{size}(\Pi^*) + 1) \leq 3 \log(\mathsf{size}(\Pi^*) + 1)$ completes the proof. □

By combining this lemma with the Depth Condensation Theorem, we can prove Theorem 5.4.1.

Proof of Theorem 5.4.1. Let Π^* be a resolution refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$. By Lemma 5.4.13, there is a resolution refutation Π of $F \circ \mathsf{XOR}_G$ with $\mathsf{depth}(\Pi) \leq \mathsf{depth}(\Pi^*)$ and

$$\mathsf{width}(\Pi) \leq 3 \log(\mathsf{size}(\Pi^*) + 1) \leq 3r/14 = r/4.$$

By the Depth Condensation Theorem, it follows that

$$\mathsf{depth}(\Pi) \log(\mathsf{size}(\Pi) + 1) \geq \mathsf{depth}_{\mathsf{Res}}(F)/6.$$

□

5.4.5 Proof of the k-DNF Resolution Tradeoff

Next, we establish Theorem 5.4.2, which we restate next.

Theorem 5.4.2. *Let $k \geq 1$ be any constant, let F be any CNF formula on N variables, and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. There is a constant $\delta := \delta(k) > 0$ such that if Π is a $\mathsf{Res}(k)$ refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{XOR}_2^n$ with $\log(\mathsf{size}(\Pi) + 1) \leq \delta \cdot r$, then*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega\big(\mathsf{depth}_{\mathsf{Res}}(F)\big)$$

To do so, we will prove a generic lifting theorem for $\mathsf{Res}(k)$. For this, it will not be necessary to recall the specific rules of $\mathsf{Res}(k)$, only that every line in a $\mathsf{Res}(k)$ proof is a k-DNF formula.

Res(k) Lifting Theorem. *Let $k \geq 1$ be an integer and F be any CNF formula. For any $\mathsf{Res}(k)$ refutation Π^* of $F \circ \mathsf{XOR}_2^n$ there is a resolution refutation Π of F such that*

$$\mathsf{width}(\Pi) \leq k\Big(\log \mathsf{size}(\Pi^*) - \log(4k)\Big)\left(\frac{4^{k+1}k}{\log e}\right)^k,$$
$$\mathsf{depth}(\Pi) \leq k \cdot \mathsf{depth}(\Pi^*)\Big(\log \mathsf{size}(\Pi^*) - \log(4k)\Big)\left(\frac{4^{k+1}k}{\log e}\right)^k.$$

This theorem follows in a straightforward way from the *switching lemma* of Segerlind Buss and Impagliazzo [139], which shows that low-width DNFs can be converted into short decision trees under a random restriction.

Definition 5.4.14. A *decision tree* is a rooted binary tree in which every non-leaf node is labelled with a variable, the edges leaving a a node are labelled with 0 and 1, and the leaves are labelled either 0 or 1. Every root-to-leaf path π in a decision tree T can be viewed as a partial assignment $\pi \in \{0,1,*\}^n$, where, if the π takes the edge labelled $\alpha \in \{0,1\}$ at node x_i then $\pi_i = \alpha$. We say that T computes a function $f : \{0,1\}^n \to \{0,1\}$ if for every $x \in \{0,1\}^n$, the leaf of the unique root-to-leaf path π in T which agrees with x is labelled with $f(x)$. The *decision tree complexity* of computing f, $\mathsf{DT}(f)$ is the minimum depth of any decision tree computing f.

For a DNF D over variables $\{x_1, \dots x_n\}$, let $C(D)$ denote the *covering number* of D — the minimum size of a set $S \subseteq \{x_1, \dots x_n\}$ such that for every term T of D, S contains at least one variable in T. The switching lemma is argued by showing that if the size of $C(D)$ is large, then many terms of D are independent and thus D is set to 1 with high probability, and if $C(D)$ is small then we can build a small decision tree computing D.

Lemma 5.4.15 ([139]). *Let $s_0, \dots, s_{k-1}$ and $p_1, \dots, p_k$ be positive numbers and let $\mathcal{D}$ set of partial assignments such that for every $i \leq k$ and every i-DNF D', if $C(D') > s_{i-1}$ then $\Pr_{\rho\sim\mathcal{D}}[D' \restriction \rho \neq 1] \leq p_i$. Then, for every k-DNF D,*

$$\Pr_{\rho\sim\mathcal{D}}\left[\mathsf{DT}(D \restriction \rho) > \sum_{i=0}^{k-1} s_i\right] \leq \sum_{i=1}^{k} p_i \cdot 2^{\left(\sum_{j=i}^{k-1} s_j\right)}.$$

In the same paper, they showed that $\mathsf{Res}(k)$ refutations in which every line can be represented by a short decision tree can be transformed into a low-width resolution refutation.

Lemma 5.4.16 ([139]). *Let F be any unsatisfiable CNF formula. If Π is a $\mathsf{Res}(k)$ refutation of F such that for every line $D \in \Pi$, $\mathsf{DT}(D) \leq t$ then there is a resolution refutation Π^* of F with*

$$\mathsf{width}(\Pi^*) \leq kt,$$
$$\mathsf{depth}(\Pi^*) \leq kt \cdot \mathsf{depth}(\Pi).$$

To prove the $\mathsf{Res}(k)$ Lifting Theorem, our strategy will be to show that for any small $\mathsf{Res}(k)$ refutation of $F \circ \mathsf{XOR}_2$ there is a restriction such that under this restriction every line in the proof can be computed by a short decision tree.

Proof of the Res(k) *Lifting Theorem.* Let $F \circ \mathsf{XOR}_2^n$ be defined over variables $u_1, v_1, \ldots, u_n, v_n$. Let $\mathcal{D}$ be the set of restrictions $\rho \in \{0,1,*\}^{2n}$ such that for every $i \in [n]$, ρ sets exactly one of u_i, v_i to some value in $\{0,1\}$ and leaves the other unset. Note that for any $\rho \in \mathcal{D}$, $F \circ \mathsf{XOR}_2^n \restriction \rho = F$. Fix a $\mathsf{Res}(k)$ refutation Π of $F \circ \mathsf{XOR}_2^n$ satisfying the assumption of the lemma. It remains to argue that there exists a restriction $\rho \in \mathcal{D}$ such that every line in $\Pi \restriction \rho$ can be computed by a short decision tree.

Fix any k-DNF D. By the pigeonhole principle, there is a set of at least $C(D)/k$ variable-disjoint terms $T_1, \ldots, T_{C(D)/k} \in D$. Denote by $\rho \sim \mathcal{D}$ sampling a restriction ρ from $\mathcal{D}$ uniformly at random, and observe that the probability that ρ satisfies any term T is at most $(1/4)^k$. Therefore,

$$\Pr_{\rho \sim \mathcal{D}}[D \restriction \rho \neq 1] \leq (1-(1/4)^k)^{C(D)/k} \leq \exp(-(1/4)^k C(D)/k).$$

Denote $w := (\log \mathsf{size}(\Pi) - \log(2k))(4^{k+1}k/\log e)^k$ and let $s_i := (w/2)(\log e/4^{i+1}i)^k$ and $p_i := 2^{-4s_i}$. Observe that $s_{i-1}/4 \leq s_i$. Therefore,

$$\sum_{j=i}^{k-1} s_j \leq \sum_{j=i}^{k-1} s_i/4^{j-i} \leq 2s_i,$$

and in particular, $\sum_{i=0}^{k-1} s_i \leq 2s_0 \leq (w/2)\log e \leq w$. For any i-DNF D with $C(D) \geq s_{i-1}$, we have

$$\Pr_{\rho \sim \mathcal{D}}[D \restriction \rho \neq 1] \leq \exp(-(1/4)^i C(D)/i) \leq \exp(-s_{i-1}p_i/i).$$

Therefore, for any k-DNF, it follows from the switching lemma (Lemma 5.4.15) that

$$\Pr_{\rho \sim \mathcal{D}}[\mathsf{DT}(D) > w] \leq \Pr_{\rho \sim \mathcal{D}}\left[\mathsf{DT}(D) > \sum_{i=0}^{k-1} s_i\right] \leq \sum_{i=1}^{k} p_i \cdot 2^{\left(\sum_{j=i}^{k-1} s_j\right)} \leq \sum_{i=1}^{k} k 2^{2s_i}(2^{-4s_i}) \leq k2^{-2s_k}.$$

Finally, we can conclude the lemma by taking a union bound over all DNFs in Π,

$$\Pr_{\rho \sim \mathcal{D}}[\exists D \in \Pi : \mathsf{DT}(D \restriction \rho) > w] \leq \mathsf{size}(\Pi)k2^{-2s_k} = \mathsf{size}(\Pi) \cdot k2^{-w\left(\frac{\log e}{4^{k+1}k}\right)^k} = 1/2,$$

where the final equality follows by our setting of w. Thus, there exists some restriction $\rho \in \mathcal{D}$ such that every $D \in \Pi \restriction \rho$ has $\mathsf{DT}(D) \leq w$. Applying Lemma 5.4.16 we can conclude that there is a resolution refutation of width at most kw and depth dkw. □

With this lifting theorem in hand, we are ready to prove Theorem 5.4.2.

Proof of Theorem 5.4.2. Set $\delta > 0$ such that $\delta = (4^{k+1}k/\log e)^{-k}/4k + \log(4k)/r$. Let Π^* be any $\mathsf{Res}(k)$ refutation of $F \circ \mathsf{XOR}_2^n$ with $\log \mathsf{size}(\Pi^*) \leq \delta \cdot r$, and denote by $t := (\log \mathsf{size}(\Pi^*) - \log(4k))(4^{k+1}k/\log e)^k$. By the Res($k$) Lifting Theorem, there exists a resolution refutation Π with $\mathsf{depth}(\Pi) \leq kt \cdot \mathsf{depth}(\Pi^*)$ and

$$\mathsf{width}(\Pi) \leq kt \leq k(\delta r - \log(4k))\left(\frac{4^{k+1}k}{\log e}\right)^k = r/4.$$

Applying the Depth Condensation Theorem, we have that

$$\frac{\mathsf{depth}_{\mathsf{Res}}(F)}{2} \leq \mathsf{depth}(\Pi)\mathsf{size}(\Pi) \leq (kt)^2\mathsf{depth}(\Pi^*) = O\Big(\log^2 \mathsf{size}(\Pi^*)\mathsf{depth}(\Pi)\Big),$$

which completes the proof.

□

5.4.6 Proof of the Semantic Cutting Planes Tradeoff

Finally, we establish Theorem 5.4.3.

Theorem 5.4.3. *Let $\varepsilon > 0$ be any constant, let F be any CNF formula on N variables, and let $G = ([N] \cup [n], E)$ be an $(r, 2)$-boundary expander. There is a constant $\delta > 0$ such that if Π is a semantic CP refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{IND}^n_{n^{1+\varepsilon}}$ with $\log(\mathsf{size}(\Pi)) \leq \delta \cdot r \log n$, then*

$$\mathsf{depth}(\Pi) \log^2(\mathsf{size}(\Pi)) = \Omega(\mathsf{depth}_{\mathsf{Res}}(F) \log^2 n).$$

This theorem follows almost immediately by applying the dag-like lifting theorem of Garg et al. [72], with the improved parameters from [111], and observing that their proof also preserves depth. We state this theorem next, specialized to semantic Cutting Planes.

Theorem 5.4.17 ([72, 111])**.** *Let $\varepsilon > 0$ be any constant and let F be an unsatisfiable CNF formula on n variables. For any semantic CP Π of $F \circ \mathsf{IND}^n_{n^{1+\varepsilon}}$ there is a resolution refutation Π^* of F satisfying*

$$\mathsf{width}(\Pi^*) = O\left(\frac{\log \mathsf{size}(\Pi)}{\log n}\right),$$
$$\mathsf{depth}(\Pi^*) = O\left(\frac{\mathsf{depth}(\Pi) \log \mathsf{size}(\Pi)}{\log n}\right).$$

By combining this lifting theorem with the Depth Condensation Theorem, we can prove Theorem 5.4.3.

Proof of Theorem 5.4.3. Let Π be a semantic CP refutation of $F \circ \mathsf{XOR}_G \circ \mathsf{IND}^n_{n^{1+\varepsilon}}$. By Theorem 5.4.17 there is a semantic Cutting Planes refutation Π^* of $F \circ \mathsf{XOR}_G$ with $\mathsf{depth}(\Pi^*) = O(\log \mathsf{size}(\Pi)/\log n)$ and $\mathsf{width}(\Pi^*) = \alpha \cdot \log \mathsf{size}(\Pi)/\log n$ for some constant $\alpha > 0$. Setting $\delta > 0$ so that $\alpha\delta = 1/4$,

$$\mathsf{width}(\Pi^*) = \frac{\alpha \cdot \log \mathsf{size}(\Pi)}{\log n} \leq \alpha\delta \cdot r = r/4.$$

By the Depth Condensation Theorem, it follows that

$$\mathsf{depth}(\Pi) \log^2 \mathsf{size}(\Pi) = \Omega\left(\mathsf{depth}_{\mathsf{Res}}(F) \log^2 n\right).$$

□

5.5 Conclusion

We end this chapter by discussing several questions left open by this work.

Problem 5.1. Supercritical Size/Depth Tradeoffs for Tseitin. The main open problem is to resolve Conjecture 1.3.9 for the Tseitin formulas. Our supercritical tradeoff for Cutting Planes relies on the lifting theorem of Garg et al. [72] which lifts resolution width lower bounds to RCC_1 (see Chapter 3) size lower bounds by composing with the *index gadget*. This poses two major issues. First, it is not clear that the Tseitin formulas

can be viewed as a function composed with any gadget beyond XOR. Second, the RCC_1 proof system has $\mathsf{poly}(n)$ and depth $O(\log n)$ refutations of the Tseitin formulas. Instead, we hope that our techniques from the proof of the sCP Depth Lifting Theorem can be useful in this context.

Problem 5.3. Supercritical Size/Depth Tradeoffs for Clause-Size. Our supercritical tradeoffs are only supercritical in terms of the number of variables, and not the number of clauses. This leaves open the possibility of establishing a tradeoff which is supercritical both in the number of clauses as well. For resolution *clause space* rather than depth, Beame Beck and Impagliazzo [14] established a tradeoff which is supercritical in terms of the number of clauses. Can the techniques used to establish this tradeoff be leveraged in order to obtain a size/depth tradeoff which is supercritical also in the number of clauses?

Chapter 6

Conclusion

In this thesis we developed and studied proof systems which formalize algorithms for integer programming. By developing deeper connections between proof complexity and monotone circuit complexity, we established strong lower bounds on the size of proofs of random CNF formulas in these systems. As well, we proved lower bounds on the depth of proofs which go far beyond worst-case when the size of the proof is limited. In what remains, we would like to record several interesting open problems related to this work.

Problem 6.1. Combinatorial Lower Bounds for (Treelike) Cutting Planes. Currently, all lower bounds on Cutting Planes are proved for the much stronger RCC_1 proof system. This includes Pudlák's lower bound [127], as well as the lifting theorem of Garg et al. [72]. The only properties of the proofs that are exploited in these lower bounds is that they are sound, and that every line can be computed by a small (real) communication protocol. This obscures our understanding of Cutting Planes. Moreover, this is also the case for the *treelike* Cutting Planes proof system, where the only lower bounds are via reduction to communication complexity [89]. Therefore, an important open problem is to develop a lower bound technique which works only for (treelike) Cutting Planes (or even semantic Cutting Planes), and not for RCC_1. This requires developing a deeper understanding how halfspaces transform throughout a Cutting Planes proof. We hope that our geometric lemma (Lemma 5.3.1) is a useful step towards this.

Problem 6.2. Treelike Proofs of the Tseitin Formulas. Dadush and Tiwari's upper bound on the Tseitin formulas [48] went against the long-standing conjecture that these formulas should be hard for Cutting Planes. Their proof is highly daglike, and therefore a natural question is whether this conjecture may still be true for *treelike* Cutting Planes. This dovetails with the previous question; the Tseitin formulas are known to have short communication protocols, and therefore they provide a natural (potentially) hard family of formulas for developing a combinatorial lower bound technique for treelike Cutting Planes.

Problem 6.3. Supercritical Size/Depth Tradeoffs for Monotone Circuits. The lifting theorem of Garg et al. [72] works equally well to prove monotone circuit lower bounds as it does Cutting Planes lower bounds. Therefore, one might hope to leverage the same techniques in order to prove the same tradeoffs for monotone circuits. While there are conversions of CNF formulas into monotone functions F (such as the one presented in Chapter 3) which preserve the complexity of refuting F, they result in functions on a number of variables which is proportional to the number of clauses of F. Unfortunately, our tradeoffs are only supercritical in

terms of the number of variables and *not* the number of clauses. Indeed, the formula F in Theorem 1.3.11 has $n^{O(c)}$ clauses, and therefore the tradeoff that one obtains is not longer supercritical.

Problem 6.4. Lower Bounds for Stabbing Planes and R(CP). One of the major questions left open by this work is to establish lower bounds on Stabbing Planes proofs with unbounded coefficients, as well as its dag-like counterpart, $\mathsf{R}(\mathsf{CP})$. One potential approach is to establish a (quasipolynomial) simulation of SP by CP which does not depend on the size of the coefficients. Garg et al. [72] suggested an approach for proving lower bounds on $\mathsf{R}(\mathsf{CP})$ proofs by establish lower bounds on a model of communication, known as *intersections of triangles dags*. Indeed, it is known that small $\mathsf{R}(\mathsf{CP})$ proofs convert to small protocols for this model of communication. Similarly, SP proofs are captured by intersections of triangles *trees*.

Bibliography

[1] Karen Aardal, Robert E. Bixby, Cor A. J. Hurkens, Arjen K. Lenstra, and Job W. Smeltink. Market split and basis reduction: Towards a solution of the cornuéjols-dawande instances. *INFORMS J. Comput.*, 12(3):192–202, 2000.

[2] Karen Aardal and Arjen K. Lenstra. Hard equality constrained integer knapsacks. *Math. Oper. Res.*, 29(3):724–738, 2004.

[3] Michael Alekhnovich. Lower bounds for k-DNF resolution on random 3-CNFs. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 251–256. ACM, 2005.

[4] Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 190–199. IEEE Computer Society, 2001.

[5] Michael Alekhnovich and Alexander A. Razborov. Satisfiability, branch-width and tseitin tautologies. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pages 593–603. IEEE Computer Society, 2002.

[6] Noga Alon and Joel H Spencer. *The probabilistic method*. John Wiley & Sons, 1996.

[7] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study*. Princeton university press, 2006.

[8] Albert Atserias, Maria Luisa Bonet, and Juan Luis Esteban. Lower bounds for the weak pigeonhole principle and random formulas beyond resolution. *Inf. Comput.*, 176(2):136–152, 2002.

[9] Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. *ACM Trans. Comput. Log.*, 17(3):19:1–19:30, 2016.

[10] Egon Balas. An additive algorithm for solving linear programs with zero-one variables. *Operations Research*, 13(4):517–546, 1965.

[11] Egon Balas, Sebastián Ceria, and Gérard Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0–1 programs. *Mathematical programming*, 58(1):295–324, 1993.

[12] Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 307–326, 2012.

[13] Amitabh Basu, Michele Conforti, Marco Di Summa, and Hongyi Jiang. Complexity of branch-and-bound and cutting planes in mixed-integer optimization - II. *CoRR*, abs/2011.05474, 2020.

[14] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space trade-offs in resolution: Superpolynomial lower bounds for superlinear space. *SIAM J. Comput.*, 45(4):1612–1645, 2016.

[15] Paul Beame, Noah Fleming, Russell Impagliazzo, Antonina Kolokolova, Denis Pankratov, Toniann Pitassi, and Robert Robere. Stabbing planes. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 10:1–10:20, 2018.

[16] Paul Beame, Richard M. Karp, Toniann Pitassi, and Michael E. Saks. On the complexity of unsatisfiability proofs for random k-CNF formulas. In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, pages 561–571. ACM, 1998.

[17] Paul Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14-16 October, 1996*, pages 274–282. IEEE Computer Society, 1996.

[18] Paul Beame and Toniann Pitassi. Propositional proof complexity: Past, present, and future. In *Current Trends in Theoretical Computer Science, Entering the 21th Century*, pages 42–70. 2001.

[19] Chris Beck, Jakob Nordström, and Bangsheng Tang. Some trade-off results for polynomial calculus: extended abstract. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 813–822. ACM, 2013.

[20] Eli Ben-Sasson. Hard examples for the bounded depth frege proof system. *Comput. Complex.*, 11(3-4):109–136, 2002.

[21] Eli Ben-Sasson and Russell Impagliazzo. Random CNFs are hard for the polynomial calculus. *Comput. Complex.*, 19(4):501–519, 2010.

[22] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001.

[23] Christer Berg and Staffan Ulfberg. Symmetric approximation arguments for monotone lower bounds without sunflowers. *Computational Complexity*, 8(1):1–20, 1999.

[24] Christoph Berkholz and Jakob Nordström. Supercritical space-width trade-offs for resolution. *SIAM J. Comput.*, 49(1):98–118, 2020.

[25] Alexander Bockmayr, Friedrich Eisenbrand, Mark Hartmann, and Andreas S Schulz. On the Chvátal rank of polytopes in the 0/1 cube. *Discrete Applied Mathematics*, 98(1-2):21–27, 1999.

[26] Merve Bodur, Alberto Del Pia, Santanu S. Dey, Marco Molinaro, and Sebastian Pokutta. Aggregation-based cutting-planes for packing and covering integer programs. *Math. Program.*, 171(1-2):331–359, 2018.

[27] Maria Luisa Bonet, Samuel R Buss, and Toniann Pitassi. Are there hard examples for frege systems? In *Feasible Mathematics II*, pages 30–56. Springer, 1995.

[28] Maria Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM J. Comput.*, 30(5):1462–1484, 2000.

[29] Maria Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Comput. Complex.*, 10(4):261–276, 2001.

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

[31] Joshua Buresh-Oppenheim, Nicola Galesi, Shlomo Hoory, Avner Magen, and Toniann Pitassi. Rank bounds and integrality gaps for cutting planes procedures. *Theory of Computing*, 2(4):65–90, 2006.

[32] Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *J. Comput. Syst. Sci.*, 62(2):267–289, 2001.

[33] Igor Carboni Oliveira. *Unconditional lower bounds in complexity theory*. PhD thesis, Columbia University, 2015.

[34] Siu Man Chan. Just a pebble game. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013, K.lo Alto, California, USA, 5-7 June, 2013*, pages 133–143. IEEE Computer Society, 2013.

[35] Arkadev Chattopadhyay, Yuval Filmus, Sajin Koroth, Or Meir, and Toniann Pitassi. Query-to-communication lifting for BPP using inner product. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 35:1–35:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[36] V. Chvátal, W. Cook, and M. Hartmann. On cutting-plane proofs in combinatorial optimization. *Linear Algebra and its Applications*, 114-115:455–499, 1989. Special Issue Dedicated to Alan J. Hoffman.

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

[38] Vašek Chvátal. *Cutting-plane proofs and the stability number of a graph*. Inst. für Ökonometrie und Operations Research, Rhein. Friedrich-Wilhelms-Univ., 1984.

[39] Vasek Chvátal and Bruce A. Reed. Mick gets some (the odds are on his side). In *33rd Annual Symposium on Foundations of Computer Science, Pittsburgh, Pennsylvania, USA, 24-27 October 1992*, pages 620–627. IEEE Computer Society, 1992.

[40] Vasek Chvátal and Endre Szemerédi. Many hard examples for resolution. *J. ACM*, 35(4):759–768, 1988.

[41] Matthew Clegg, Jeff Edmonds, and Russell Impagliazzo. Using the groebner basis algorithm to find proofs of unsatisfiability. In Gary L. Miller, editor, *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 174–183. ACM, 1996.

[42] Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli, et al. *Integer programming*, volume 271. Springer, 2014.

[43] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *J. Symb. Log.*, 44(1):36–50, 1979.

[44] William Cook, Ravindran Kannan, and Alexander Schrijver. Chvátal closures for mixed integer programming problems. *Mathematical Programming*, 47(1):155–174, 1990.

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

[46] William J. Cook and Mark Hartmann. On the complexity of branch and cut methods for the traveling salesman problem. In William J. Cook and Paul D. Seymour, editors, *Polyhedral Combinatorics, Proceedings of a DIMACS Workshop, Morristown, New Jersey, USA, June 12-16, 1989*, volume 1 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 75–82. DIMACS/AMS, 1990.

[47] Gérard Cornuéjols and Yanjun Li. On the rank of mixed 0, 1 polyhedra. *Math. Program.*, 91(2):391–397, 2002.

[48] Daniel Dadush and Samarth Tiwari. On the complexity of branching proofs. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 34:1–34:35. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[49] Stefan S. Dantchev, Nicola Galesi, Abdul Ghani, and Barnaby Martin. Depth lower bounds in stabbing planes for combinatorial principles. *CoRR*, abs/2102.07622, 2021.

[50] Adnan Darwiche. Recursive conditioning. *Artif. Intell.*, 126(1-2):5–41, 2001.

[51] Sanjeeb Dash. Exponential lower bounds on the lengths of some classes of branch-and-cut proofs. *Math. Oper. Res.*, 30(3):678–700, 2005.

[52] Sanjeeb Dash. On the complexity of cutting plane proofs using split cuts. *Electron. Colloquium Comput. Complex.*, 15(084), 2008.

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

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

[55] Wenceslas Fernandez de la Vega. On random 2-sat. Unpublished Manuscript, 1992.

[56] Susanna F. de Rezende, Jakob Nordström, and Marc Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). *Electron. Colloquium Comput. Complex.*, 28:6, 2021.

[57] Rina Dechter. Bucket elimination: a unifying framework for processing hard and soft constraints. *Constraints An Int. J.*, 2(1):51–55, 1997.

[58] Santanu S. Dey, Yatharth Dubey, and Marco Molinaro. Lower bounds on the size of general branch-and-bound trees. *CoRR*, abs/2103.09807, 2021.

[59] Jian Ding, Allan Sly, and Nike Sun. Proof of the satisfiability conjecture for large k, 2016.

[60] Friedrich Eisenbrand and Andreas S. Schulz. Bounds on the chvátal rank of polytopes in the 0/1-cube. In Gérard Cornuéjols, Rainer E. Burkard, and Gerhard J. Woeginger, editors, *Integer Programming and Combinatorial Optimization, 7th International IPCO Conference, Graz, Austria, June 9-11, 1999, Proceedings*, volume 1610 of *Lecture Notes in Computer Science*, pages 137–150. Springer, 1999.

[61] Friedrich Eisenbrand and Andreas S Schulz. Bounds on the Chvátal rank of polytopes in the 0/1-cube. *Combinatorica*, 23(2):245–261, 2003.

[62] Juan Luis Esteban and Jacobo Torán. A combinatorial characterization of treelike resolution space. *Inf. Process. Lett.*, 87(6):295–300, 2003.

[63] Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 534–543, 2002.

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

[65] Matteo Fischetti and Andrea Lodi. Local branching. *Math. Program.*, 98(1-3):23–47, 2003.

[66] Noah Fleming, Mika Göös, Russell Impagliazzo, Toniann Pitassi, Robert Robere, Li-Yang Tan, and Avi Wigderson. On the power and limitations of branch and cut. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 6:1–6:30. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[67] 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.

[68] Noah Fleming, Denis Pankratov, Toniann Pitassi, and Robert Robere. Random $\Theta(\log n)$-CNFs are hard for cutting planes. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 109–120, 2017.

[69] Noah Fleming and Toniann Pitassi. Reflections on proof complexity and counting principles. *Electron. Colloquium Comput. Complex.*, 28:61, 2021.

[70] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.

[71] Nicola Galesi, Dmitry Itsykson, Artur Riazanov, and Anastasia Sofronova. Bounded-depth frege complexity of tseitin formulas for all graphs. In Peter Rossmanith, Pinar Heggernes, and Joost-Pieter Katoen, editors, *44th International Symposium on Mathematical Foundations of Computer Science, MFCS 2019, August 26-30, 2019, Aachen, Germany*, volume 138 of *LIPIcs*, pages 49:1–49:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[72] Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 902–911. ACM, 2018.

[73] Michel X. Goemans and David P. Williamson. .879approximationn algorithms for max cut and max 2sat. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 422–431, New York, NY, USA, 1994. ACM.

[74] Andreas Goerdt. A threshold for unsatisfiability. *J. Comput. Syst. Sci.*, 53(3):469–486, 1996.

[75] Ralph Gomory. An algorithm for the mixed integer problem. Technical report, RAND CORP SANTA MONICA CA, 1960.

[76] Ralph E Gomory. An algorithm for integer solutions to linear programs. *Recent advances in mathematical programming*, 64(260-302):14, 1963.

[77] Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 38:1–38:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[78] Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is NP-hard. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 68–77. ACM, 2020.

[79] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM J. Comput.*, 47(5):1778–1806, 2018.

[80] Dima Grigoriev. Tseitin's tautologies and lower bounds for nullstellensatz proofs. In *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8-11, 1998, Palo Alto, California, USA*, pages 648–652. IEEE Computer Society, 1998.

[81] Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theor. Comput. Sci.*, 259(1-2):613–622, 2001.

[82] M. Grötschel, L. Lovász, and A. Schrijver. Geometric methods in combinatorial optimization. In William R. Pulleyblank, editor, *Progress in Combinatorial Optimization*, pages 167–183. Academic Press, 1984.

[83] Armin Haken and Stephen A. Cook. An exponential lower bound for the size of monotone real circuits. *J. Comput. Syst. Sci.*, 58(2):326–335, 1999.

[84] Johan Håstad. On small-depth frege proofs for tseitin for grids. *Electron. Colloquium Comput. Complex.*, 24:142, 2017.

[85] Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, volume 79 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

[86] Pavel Hrubeš and Pavel Pudlák. Random formulas, monotone circuits, and interpolation. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 121–131, 2017.

[87] Pavel Hrubeš and Pavel Pudlák. A note on monotone real circuits. *Inf. Process. Lett.*, 131:15–19, 2018.

[88] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[89] Russell Impagliazzo, Toniann Pitassi, and Alasdair Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94), Paris, France, July 4-7, 1994*, pages 220–228. IEEE Computer Society, 1994.

[90] Robert G Jeroslow. Trivial integer programs unsolvable by branch-and-bound. *Mathematical Programming*, 6(1):105–109, 1974.

[91] Hendrik W. Lenstra Jr. Integer programming with a fixed number of variables. *Math. Oper. Res.*, 8(4):538–548, 1983.

[92] Roberto J. Bayardo Jr. and Robert Schrag. Using CSP look-back techniques to solve real-world SAT instances. In Benjamin Kuipers and Bonnie L. Webber, editors, *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Innovative Applications of Artificial Intelligence Conference, AAAI 97, IAAI 97, July 27-31, 1997, Providence, Rhode Island, USA*, pages 203–208. AAAI Press / The MIT Press, 1997.

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

[94] Miroslav Karamanov and Gérard Cornuéjols. Branching on general disjunctions. *Math. Program.*, 128(1-2):403–436, 2011.

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

[96] Scott Kirkpatrick and Bart Selman. Critical behavior in the satisfiability of random boolean expressions. *Science*, 264(5163):1297–1301, 1994.

[97] Arist Kojevnikov. Improved lower bounds for tree-like resolution over linear inequalities. In João Marques-Silva and Karem A. Sakallah, editors, *Theory and Applications of Satisfiability Testing - SAT 2007, 10th International Conference, Lisbon, Portugal, May 28-31, 2007, Proceedings*, volume 4501 of *Lecture Notes in Computer Science*, pages 70–79. Springer, 2007.

[98] Jan Krajícek. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symb. Log.*, 62(2):457–486, 1997.

[99] Jan Krajícek. Discretely ordered modules as a first-order extension of the cutting planes proof system. *J. Symb. Log.*, 63(4):1582–1596, 1998.

[100] Jan Krajícek. Interpolation by a game. *Math. Log. Q.*, 44:450–458, 1998.

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

[102] Bala Krishnamoorthy and Gábor Pataki. Column basis reduction and decomposable knapsack problems. *Discret. Optim.*, 6(3):242–270, 2009.

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

[104] Ailsa H. Land and Alison G. Doig. An automatic method for solving discrete programming problems. In Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey, editors, *50 Years of Integer Programming 1958-2008 - From the Early Years to the State-of-the-Art*, pages 105–132. Springer, 2010.

[105] Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[106] Neha Lodha, Sebastian Ordyniak, and Stefan Szeider. A SAT approach to branchwidth. *ACM Trans. Comput. Log.*, 20(3):15:1–15:24, 2019.

[107] László Lovász. On determinants, matchings, and random algorithms. In Lothar Budach, editor, *Fundamentals of Computation Theory, FCT 1979, Proceedings of the Conference on Algebraic, Arthmetic, and Categorial Methods in Computation Theory, Berlin/Wendisch-Rietz, Germany, September 17-21, 1979*, pages 565–574. Akademie-Verlag, Berlin, 1979.

[108] László Lovász, Moni Naor, Ilan Newman, and Avi Wigderson. Search problems in the decision tree model. *SIAM J. Discret. Math.*, 8(1):119–132, 1995.

[109] László Lovász and Herbert E. Scarf. The generalized basis reduction algorithm. *Math. Oper. Res.*, 17(3):751–764, 1992.

[110] László Lovász and Alexander Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM J. Optim.*, 1(2):166–190, 1991.

[111] Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with sunflowers. unpublished.

[112] Ashutosh Mahajan and Theodore K Ralphs. Experiments with branching using general disjunctions. In *Operations Research and Cyber-Infrastructure*, pages 101–118. Springer, 2009.

[113] Adam W. Marcus, Daniel A. Spielman, and Nikhil Srivastava. Interlacing families IV: bipartite ramanujan graphs of all sizes. *SIAM J. Comput.*, 47(6):2488–2509, 2018.

[114] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[115] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference, DAC 2001, Las Vegas, NV, USA, June 18-22, 2001*, pages 530–535. ACM, 2001.

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

[117] George L Nemhauser and Laurence A Wolsey. A recursive procedure to generate all cuts for 0–1 mixed integer programs. *Mathematical Programming*, 46(1):379–390, 1990.

[118] Noam Nisan. The communication complexity of threshold gates. *Combinatorics, Paul Erdos is Eighty*, 1:301–315, 1993.

[119] Robert Robere Noah Fleming, Toniann Pitassi. Extremely deep proofs. 2021. unpublished.

[120] Jonathan H. Owen and Sanjay Mehrotra. Experimental results on using general disjunctions in branch-and-bound for general-integer linear programs. *Comput. Optim. Appl.*, 20(2):159–170, 2001.

[121] Manfred Padberg and Giovanni Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33(1):60–100, 1991.

[122] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

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

[124] Toniann Pitassi, Benjamin Rossman, Rocco A. Servedio, and Li-Yang Tan. Poly-logarithmic frege depth lower bounds via an expander switching lemma. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 644–657. ACM, 2016.

[125] Sebastian Pokutta and Andreas S. Schulz. On the rank of cutting-plane proof systems. In Friedrich Eisenbrand and F. Bruce Shepherd, editors, *Integer Programming and Combinatorial Optimization, 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9-11, 2010. Proceedings*, volume 6080 of *Lecture Notes in Computer Science*, pages 450–463. Springer, 2010.

[126] Sebastian Pokutta and Andreas S. Schulz. Integer-empty polytopes in the 0/1-cube with maximal gomory-chvátal rank. *Oper. Res. Lett.*, 39(6):457–460, 2011.

[127] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *J. Symb. Log.*, 62(3):981–998, 1997.

[128] Pavel Pudlák. Proofs as games. *Am. Math. Mon.*, 107(6):541–550, 2000.

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

[130] Alexander Razborov. Unprovability of lower bounds on circuit size in certain fragments of bounded arithmetic. *Izvestiya Mathematics*, 59(1):205–227, 1995.

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

[132] Alexander A. Razborov. A new kind of tradeoffs in propositional proof complexity. *J. ACM*, 63(2):16:1–16:14, 2016.

[133] Alexander A. Razborov. On the width of semialgebraic proofs and algorithms. *Math. Oper. Res.*, 42(4):1106–1134, 2017.

[134] Alexander A. Razborov. On space and depth in resolution. *Comput. Complex.*, 27(3):511–559, 2018.

[135] Thomas Rothvoß and Laura Sanita. 0/1 polytopes with quadratic chvátal rank. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 349–361. Springer, 2013.

[136] Rahul Santhanam. Pseudorandomness and the minimum circuit size problem. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference, ITCS 2020, January 12-14, 2020, Seattle, Washington, USA*, volume 151 of *LIPIcs*, pages 68:1–68:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[137] Grant Schoenebeck. Linear level lasserre lower bounds for certain k-CSPs. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 593–602. IEEE Computer Society, 2008.

[138] A. Schrijver. On cutting planes. In Peter L. Hammer, editor, *Combinatorics 79*, volume 9 of *Annals of Discrete Mathematics*, pages 291–296. Elsevier, 1980.

[139] Nathan Segerlind, Samuel R. Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for k-DNF resolution. *SIAM J. Comput.*, 33(5):1171–1200, 2004.

[140] Bart Selman, David G. Mitchell, and Hector J. Levesque. Generating hard satisfiability problems. *Artif. Intell.*, 81(1-2):17–29, 1996.

[141] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discret. Math.*, 3(3):411–430, 1990.

[142] João P. Marques Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Trans. Computers*, 48(5):506–521, 1999.

[143] Dmitry Sokolov. Dag-like communication and its applications. In Pascal Weil, editor, *Computer Science - Theory and Applications - 12th International Computer Science Symposium in Russia, CSR 2017, Kazan, Russia, June 8-12, 2017, Proceedings*, volume 10304 of *Lecture Notes in Computer Science*, pages 294–307. Springer, 2017.

[144] Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987.

[145] Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012.

[146] Laurence A Wolsey and George L Nemhauser. *Integer and combinatorial optimization*, volume 55. John Wiley & Sons, 1999.