PROOF COMPLEXITY & AUTOMATIZABILITY

by

Ian Mertz

A project submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

Abstract

Proof Complexity & Automatizability

Ian Mertz
Master of Science
Graduate Department of Computer Science
University of Toronto
2018

Just putting the abstract here as a reminder to myself. Also remember that you only get 150 words for MSc.

Acknowledgements

The authors are grateful to Noah Fleming, Pravesh Kothari, Denis Pankratov, Robert Robere and Avi Wigderson for helpful conversations. We also thank Yijia Chen for providing more details on the construction in [14], and Pasin Manurangsi for giving feedback on the parameters in [12].

Contents

Chapter 1

Introduction

Proof complexity aims to study proofs. This seems simple and self-evident, but it is worth thinking about what we computer scientists care about when it comes to studying proofs, or rather studying proof *systems*. Simple systems like Resolution have given us insight into the power and limitations of SAT solvers, while saying anything worthwhile about stronger systems like Extended Frege seems to still pose an indomidable challenge. As a complexity theorist the first natural way to classify proof systems is the following: given a tautology, is it easy or hard for the proof system to prove it? Can the system refute a random CNF with a polynomial number of lines, or can we prove that even the Pigeonhole Principle takes exponential sized proofs? This is a fascinating and central problem, but it seems to suggest an equally natural yet perhaps more practical and certainly more overlooked challenge: even if the system has a short proof, can we find it?

The notion of *automatizability* [11] captures the algorithmic side of proof complexity, asking whether or not there exists an algorithm that can return some $\mathcal{Q}$-proof of any tautology τ efficiently with regards to the length of the *shortest* $\mathcal{Q}$-proof of τ. As proof complexity has found connections to many areas such as learning theory, SAT solvers, and approximation, the algorithmic question of automatizability has provided a lens and a hammer for deciding the tractability of many core problems. I stole this next part from the paper. Most notably, the Sum-of-Squares algorithm has emerged as a very powerful algorithmic tool, as short SoS proofs capture many of our best known approximation techniques. This idea has led to a flurry of exciting papers that give new and improved algorithms for a variety of learning problems [6, 7, 23, 27, 30], but the runtime of these algorithms is dependent on the *degree* of the proofs rather than the size. If small size SoS proofs could also be found efficiently, this would give rise to an even richer family of algorithms.

At first glance it would seem like the efficiency of finding $\mathcal{Q}$-proofs would have some sort of tradeoff with the efficiency of $\mathcal{Q}$ itself. For example, if we learned some day that Extended Frege could refute random CNFs in polynomial time, modulo $\mathsf{P} \neq \mathsf{NP}$ it would be impossible to automatize Extended Frege in polynomial time, and under something stronger like the Exponential Time Hypothesis even exponential lower bounds could follow. By contrast the efficiency of all known SAT solvers is implicitly based on the automatizability of Resolution and tree-like Resolution, and so it would seem beneficial to prove some sort of upper bound on automatizability therein.

The first question, of lower bounds against strong systems, quickly had some progress. The first observation was that any propositional proof system which for any τ has $\mathsf{poly}(|\tau|)$ sized proofs is not polynomially automatizable assuming $\mathsf{NP} \not\subseteq \mathsf{P/poly}$. This is in line with our earlier intuition about a hypothetical world where EF could easily refute random CNFs, and uses a very well-believed assumption about NP. Next, using various cryptographic assumptions a line of work [10, 11, 26] showed that many Frege systems are not polynomially automatizable. The argument was once again similar to our intuition: if a system has short proofs for hard problems, we shouldn't expect to get those proofs too easily.

All these arguments used a similar technique for "solving hard problems" using proofs: first, consider some decision problem that is hard, perhaps an NP-complete problem or inverting some function believed to be one-way. Our algorithm first transforms an instance of the problem into a tautology, such that the "yes" instances have some structure that the proof system can exploit to find a short proof, and such that the "no" instances have strong lower bounds in the system. By automatizing the proof system the algorithm can obtain a proof that's not too large compared to the shortest proof, and then using the gap in the shortest proofs of the yes and no instances can decide the original problem. Note that we only use the length of the proof the automatizing algorithm outputs and never look at the proof itself, and so it may not be too surprising that [1] showed that approximating the length of the shortest proof to within a factor of $2^{\log^{1-o(1)} n}$ for *any* propositional proof system is NP-hard.

Shockingly this technique also worked for weaker systems like Resolution. In a breakthrough result [2] showed that Resolution and tree-like Resolution are not polynomially automatizable under a very plausible hypothesis from parameterized complexity, $\mathsf{FPT} \neq \mathsf{W[P]}$. The key idea was simple, the same transformation of instances to tautologies with a gap in the proof lengths as before, and for all the technical considerations involved the execution was equally simple and beautiful. The same transformation proved to go beyond Resolution; a few years later [19] proved the same automatizability lower bound for Nullstellensatz and Polynomial Calculus, using a specialized but morally equivalent lower bound technique. With a number of systems hanging between the frontier these two works carved out and the powerful Frege systems for which lower bounds were known, it seems natural to see how far this tautology can be pushed, maybe even to the more elusive systems like Sum-of-Squares and Cutting Planes.

Another question would be if changing the assumption or other fine details of the argument would give a stronger automatizability lower bound. Until now no lower bounds beyond superpolynomial are known for the automatizability of any proof system, and in using a parameterized complexity assumption there is an inherent and insurmountable barrier to going beyond superpolynomial lower bounds. There is also an upper bound hanging close by: Resolution, tree-like Resolution, Nullstellensatz, Polynomial Calculus, and many other systems are width (or degree) automatizable in the sense that if there exists a width (degree) d proof, it can be found in time $n^{O(d)}$, which immediately implies an $n^{O(\log S)}$ automatizability upper bound for both tree-like Resolution and Nullstellensatz (where $S := S(\tau)$ is the size of the shortest proof of τ). Thus if this technique holds for tree-like Resolution and Nullstellensatz, it cannot prove a lower bound beyond $n^{\Omega(\log S)}$, whereas for Resolution and many other systems there's no reason to rule out even exponential automatizability lower bounds.

In this paper we attempt to capture the core ideas of [2, 19] by simplifying the argument, removing many of the technical details necessary for their arguments to hold. We do so by changing our assumption to the Exponential Time Hypothesis (ETH) as well as a more recent variant called the Gap Exponential Time Hypothesis (GapETH), which while both weaker assumptions than $\mathsf{FPT} \neq \mathsf{W[P]}$ are still very widely believed and (in our estimation) more well-known in complexity theory. In doing so we also manage to break the superpolynomial wall and get close to the upper bound stated above.

Theorem 1.1 (Main Theorem)**.** *Let* $\mathcal{Q} \in \{\mathsf{Res}, \mathsf{TreeRes}, \mathsf{Nullsatz}, \mathsf{PC}, \mathsf{PCR}\}$*. Assuming* GapETH *holds,* $\mathcal{Q}$ *is not* n^f*-automatizable for any* $f = \tilde{o}(\log \log S)$*. Furthermore, assuming* ETH *holds* $\mathcal{Q}$ *is not* n^f*-automatizable for any* $f = O(\log^{1/5} \log S)$*.*

We also get a comparable statement for a new system, typically referred to as k-Resolution (we use Res(r) instead of Res(k) due to the use of k as a different parameter in the reduction). In light of the power of Res(r) to

System	Assumption	Result	Ref
Any PPS	NP-hard	$2^{\log^{1-o(1)} n}$	[1]
Any poly PPS	NP $\not\subseteq$ P/poly	superpoly(n, S)	[11]
AC^0-Frege	Diffie-Hellman requires circuits of size 2^{n^ϵ}	superpoly(n, S)	[10]
Frege	Factoring Blum integers requires circuits of size $n^{\omega(1)}$	superpoly(n, S)	[11]
E. Frege	Discrete log is not in P/poly	superpoly(n, S)	[26]
Res, TreeRes	W[P] $\neq$ FPT	superpoly(n, S)	[2]
Nullsatz, PC	W[P] $\neq$ FPT	superpoly(n, S)	[19]
Res, TreeRes, Nullsatz, PC	GapETH	$n^{\tilde{\Omega}(\log\log S)}$	this work
	ETH	$n^{\omega(\log^{1/5}\log S)}$	
Res(r)	GapETH	$n^{\tilde{\Omega}(\log\log S/\exp(r^2))}$	this work
	ETH	$n^{\omega(\log^{1/5}\log S/\exp(r^2))}$	

perform very powerful tasks when r is large, and in particular a plausible break in the argument for $r = O(\log n)$, we get an inherent tradeoff between r and the lower bound. In particular this allows us to give lower bounds beyond superpolynomial even for superconstant r. By contrast the previous works only allowed superpolynomial lower bounds for constant r, which is in some sense uninteresting given matching Resolution lower bounds.

Theorem 1.2 (Main Theorem 2)**.** *Let $\mathcal{Q} =$ Res(r). Assuming GapETH holds, $\mathcal{Q}$ is not $n^{f/\exp(r^2)}$-automatizable for any $f = \tilde{o}(\log\log n)$ if $r = O(\sqrt{\log f})$. Furthermore, assuming ETH holds $\mathcal{Q}$ is not $n^{f/\exp(r^2)}$-automatizable for any $f = O(\log^{1/5}\log n)$ if $r = O(\sqrt{\log f})$.*

A summary of all known automatizability results discussed can be found in Table 1. In Chapter 5 we discuss how the technique may be extended to other systems and what challenges arise.

Before moving into the paper we sketch our version of the technique of [2]. Let I be an instance of some problem on n variables. Consider some n-variate monotone function f_I and let $\gamma(f_I)$ be the minimum weight of any input on which f_I evaluates to 1. Furthermore we assume that for a given I, distinguishing whether $\gamma(f_I) \leq k$ or $\gamma(f_I) \geq k^2$ requires time $n^{\Omega(k)}$ assuming ETH for all k up to some value. Our tautology will create m different inputs to the function f_I for some specially chosen m, with the different inputs correlated in such a way that the following two properties hold: 1) to find the ith bit of the jth input, we need to find the ith bit of *all* inputs; 2) no matter how the inputs are chosen, at least one of them evaluates to 1 on f_I by having a 1 in the $\gamma(f_I)$ spots of the minimum weight input. Here we see the trapdoor: in order to show that some input satisfies f_I, it is necessary and sufficient to consider $\gamma(f_I)$ input bits. With the right choice of m and the right correlation, we end up with a tautology with an $n^{\Theta(1)}$ sized proof whenever $\gamma(f_I) \leq k$ and an $n^{\Theta(k)}$ sized proof whenever $\gamma(f_I) \geq k^2$. Thus any automatizing algorithm running in time $n^{o(k)}$ can distinguish between the two, and in too little time, contradicting ETH. Fixing k to be the maximum value possible, say $O(\log\log n)$, and noting that when $\gamma(f_I) \leq k$ we get $S = n^{O(1)}$, this implies that no automatizing algorithm can run in time $n^{o(k)} = n^{o(\log\log S)}$.

Chapter 2

Preliminaries

2.1 Proof complexity

Let $\tau = \{C_1, C_2, \ldots, C_m\}$ be an unsatisfiable CNF formula over $X = \{x_1 \ldots x_n\}$. We denote by $|\tau|$ the *size* of τ, and likewise for a proof π refuting τ let $|\pi|$ denote the size of π. For a proof system $\mathcal{Q}$ let $S_{\mathcal{Q}}(\tau)$ be the size of the shortest $\mathcal{Q}$-proof refuting τ. A proof system $\mathcal{Q}$ is said to be $f()$*-automatizable* if there exists an algorithm A such that for every unsatisfiable τ A runs in time $f(|\tau| + S_{\mathcal{Q}}(\tau))$ and outputs a valid $\mathcal{Q}$-proof refuting τ. A proof system $\mathcal{Q}'$ *p-simulates* $\mathcal{Q}$ if for every $\mathcal{Q}$-proof π refuting τ there is a corresponding $\mathcal{Q}'$-proof π' refuting τ such that $|\pi'| = |\pi|^{O(1)}$.

2.1.1 Proof Systems

A *Resolution* (Res) refutation of τ is a sequence of clauses $\pi = \{D_1, D_2, \ldots, D_S\}$ such that $D_S = \emptyset$, and each line D_i is either some initial clause $C_j \in \tau$ or is derived from two previous lines using the *resolution rule*: from $(E \vee x)$, $(F \vee \overline{x})$ we derive $(E \vee F)$, where $x \in X$, E and F are clauses, and $E \vee F$ is their disjunction with repeated literals removed. We can view a Res proof π as a directed acyclic graph with a unique line D_i at every vertex, with initial clauses $C_j \in \tau$ at the leaves, $\emptyset$ at the root, and having an edge from D_i to D_j if D_i was used to derive D_j. With this view, a $\mathsf{TreeRes}$ refutation requires that all non-leaf vertices of the underlying graph have outdegree 1 (so the underlying graph of any $\mathsf{TreeRes}$ proof is tree-like).

Given a Res or $\mathsf{TreeRes}$ refutation $\pi = \{D_1, D_2, \ldots, D_S\}$, the size of π is the number of lines in π, in this case S. The *width* of a clause D_i is the number of literals in it, and the width of π is the maximum width of a clause in the proof. We denote the width of a clause D_i or proof π by $w(D_i)$ and $w(\pi)$, respectively. Clearly Res can p-simulate $\mathsf{TreeRes}$ with respect to size and width, as every $\mathsf{TreeRes}$-proof is also a Res-proof.

An r*-Resolution* ($\mathsf{Res(r)}$) refutation is similar to a Res refutation, but each line D_i is an r-DNF instead of a clause, and the resolution rule is adapted as follows: from $(E \vee (\vee_{j \in J} x_j)), (F \vee (\wedge_{j \in J} \overline{x_j}))$ we derive $(E \vee F)$, where $J \subseteq [n]$ such that $|J| \leq r$, E and F are r-DNFs, and $E \vee F$ is their disjunction with repeated conjunctions removed (note that $\vee_{j \in J} x_j$ is a DNF with $|J|$ terms while $\wedge_{j \in J} \overline{x_j}$ is a single term). Note that $\mathsf{Res(1)} = \mathsf{Res}$. The size of a $\mathsf{Res(r)}$ proof is the number of r-disjunctions in it. (See [33] for more details.)

An *algebraic proof system* for refuting CNF $\tau = \{C_1 \ldots C_{m'}\}$ over variable set X is a proof system where each of the clauses C_i is converted into a polynomial equality or inequality P_i over X, such that any assignment of all x_j to $\{0,1\}^n$ satisfies C_i iff it satisfies P_i. For this paper the conversion is done is by sending every positive literal x_j to $(1 - x_j)$ and every negative literal $\overline{x_j}$ to x_j, and P_i is satisfied if the product of all converted literals in

C_i is 0. For example, the clause $C_i = x_1 \vee \overline{x_2} \vee x_3$ is converted to $P_i = (1 - x_1)(x_2)(1 - x_3) = 0$. In addition, we add the equations $x_j^2 - x_j = 0$ for all $j \leq n$. Let the resulting $m = m' + n$ equations corresponding to τ be denoted by $\mathcal{P} = \{P_1, \ldots, P_m\}$. Since every P_i is of the form $p_i = 0$ we drop the latter part and use P_i to refer to p_i.

The *Nullstellensatz* (Nullsatz) refutation system [8] is an algebraic proof system that uses Hilbert's Nullstellensatz as a certificate of unsatisfiablility. A Nullsatz proof (over a field $\mathbb{F}$) of τ is a set of polynomials $Q_1, \ldots, Q_m$ such that $\sum_i P_i Q_i$ is the formal polynomial "1". Note that this contradicts the fact that there exists an assignment such that $P_i = 0$ for all i. The size of a Nullsatz refutation π is the sum over all $i \in [m]$ of the number of monomials in the expansion of the term $P_i Q_i$, while the *degree* of the refutation is the maximum degree $\deg(P_i Q_i)$ over all $i \in [m]$. It is known that Nullsatz p-simulates TreeRes.

The *Polynomial Calculus* (PC) system is a dynamic version of Nullsatz [16], where the lines of a PC proof π are all polynomials $Q_1, Q_2, \ldots, Q_S$. The lines Q_i can be any of the initial polynomial equations $\mathcal{P}$ or can be derived from previous lines by the following rules: (1) from Q_i we can derive $x_j Q_i$ or $(1 - x_j) Q_i$ for any variable x_j; (2) from Q_i, Q_j we can derive $aQ_i + bQ_j$ for any $a, b \in \mathbb{R}$. As with Nullsatz the final line Q_S is the formal polynomial "1". Similarly to Nullsatz the degree of a PC proof π is the maximal degree of any line Q_i, and the size of π is the total number of monomials in the refutation, where multiple occurrences of the same monomial are counted for each occurrence. PC trivially p-simulates Nullsatz and the simulation is degree-preserving.

The PCR system is a simple modification to the PC proof system so that it can p-simulate Res proofs with respect to size. For PCR, polynomials are allowed to use additional variables $\overline{x}_1, \ldots, \overline{x}_n$ and axioms of the form $1 - \overline{x}_j - x_j = 0$ for all $j \in [n]$. Furthermore all terms $(1 - x_j)$ in the input polynomials in $\mathcal{P}$ are replaced by the variables $\overline{x_j}$. Intuitively although the variables x_j and $\overline{x_j}$ are distinct they stand for the negations of one another, which is enforced by the new axiom corresponding to x_j. It is not hard to see that PCR can now p-simulate Res with respect to size.

2.2 Miscellaneous

2.2.1 Gap hitting set

Let $\mathcal{S} = \{S_1, \ldots, S_n\}$ be a collection of non-empty sets S_j over $[n]$. A *hitting set* $H \subseteq [n]$ is a set of elements such that $H \cap S_j \neq \emptyset$ for all $j \in [n]$. Let $\gamma(\mathcal{S})$ be the size of the smallest hitting set for $\mathcal{S}$. The *gap hitting set problem* is the task of distinguishing, on input $(\mathcal{S}, k, hk)$, the following two cases: (1) $\gamma(\mathcal{S}) \leq k$; (2) $\gamma(\mathcal{S}) > hk$.

Definition 2.1. The *Exponential Time Hypothesis* (ETH) states [25] that for sufficiently large m and n, no algorithm running in time $2^{o(n)}$ can decide, for given CNF τ with m clauses and n variables, whether all m clauses of τ are satisfiable or not. The *Gap Exponential Time Hypothesis* (GapETH) states [18, 28] that for sufficiently large m and n, no algorithm running in time $2^{o(n)}$ can decide, for given CNF τ with m clauses and n variables and any constant $\epsilon \in (0, 1)$, whether all m clauses of τ are satisfiable or if at most $(1 - \epsilon)m$ of the clauses are satisfiable.

We state the following hardness results for the hitting set problem under GapETH and ETH, which can be inferred from recent work on parameterized complexity ([12] and [**?**], respectively). For an overview of how to find the optimal values in the result, see Appendix A.

Lemma 2.2 (Hardness of Hitting Set)**.** *Assuming* GapETH*, for sufficiently large n and $k = \tilde{O}(\log \log n)$ no algorithm can solve the gap hitting set problem $(\mathcal{S}, k, k^2)$ in time $n^{o(k)}$. Assuming* ETH*, for sufficiently large n and $k = O(\log^{1/5} \log n)$ no algorithm can solve the gap hitting set problem $(\mathcal{S}, k, k^2)$ in time $n^{o(k)}$.*

2.2.2 k-universal sets

Consider a set $A \subseteq \{0,1\}^m$ of m-bit strings such that $|A| = m$. We say that A is (m,k)-*universal* if for every subset $J \subseteq [m]$ of up to k positions in $[m]$, the projection $A|_J$ (restricting the strings in A to these positions) contains all possible $2^{|J|}$ binary strings of length $|J|$. Observe that we can take the dual of the set A in the following sense: if $A = \{a_1, \ldots, a_m\}$, and let $B \subseteq \{0,1\}^m$ be the set of all strings b_j for $j \in [m]$ such that the ith bit of b_j is the jth bit of a_i. Another way to think about B is taking the strings of A to be the columns of an $m \times m$ matrix and letting B be the columns of that matrix's transpose. We say A is (m,k)-*dual-universal* if B is (m,k)-universal. Equivalently A is (m,k)-dual-universal if for every ordered subset $I \subseteq A$ of up to k strings in A and for every string $s \in \{0,1\}^{|I|}$, there exists some position $j \in [m]$ such that s is the string formed by concatenating the jth bit of all strings in I in order. The existence of efficiently constructible $(m, \log m/4)$-universal sets is known (see [3, 29] for many examples of such sets coming from almost k-wise independent sample spaces). It is also known that there exist efficiently constructible sets that are both $(m, \log m/4)$-universal *and* $(m, \log m/4)$-dual-universal. (For a concrete example, [2] uses the *Paley graph* G_m on m vertices.) For the rest of the paper we will fix an arbitrary A that is efficiently computable and is both $(m, \log m/4)$-universal and $(m, \log m/4)$-dual-universal.

Chapter 3

Main ideas

To prove Theorem 1.1, it is sufficient to have a procedure that efficiently takes a hitting set instance $\mathcal{S}$ and construct from it a tautology $\tau_{\mathcal{S}}$ such that $S_{\mathcal{Q}}(\tau_{\mathcal{S}})$ is closely correlated with $\gamma(\mathcal{S})$. In this chapter we show that such a procedure proves Theorem 1.1, and then define $\tau_{\mathcal{S}}$. To understand why this construction gives the desired correlation with $\gamma(\mathcal{S})$ we prove decision tree upper and lower bounds on $S(\tau_{\mathcal{S}})$. This will also give the strategy behind the lower bounds for all other $\mathcal{Q}$, which we prove in Chapter 4.

3.1 Proof of main theorem

We first state our main lemma from which Theorem 1.1 is easily proven.

Lemma 3.1. *Let $\mathcal{Q} \in \{\mathsf{Res}, \mathsf{TreeRes}, \mathsf{Nullsatz}, \mathsf{PC}, \mathsf{PCR}\}$. For sufficiently large n, let $(\mathcal{S}, k, k^2)$ be an instance of the gap hitting set problem over $[n]$. Then there exists a tautology $\tau_{\mathcal{S}}$ which can be computed in time $n^{O(1)}$ such that $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) = n^{\Theta(\gamma(\mathcal{S})/k)}$. Namely the following two properties hold*

(1) if $\gamma(\mathcal{S}) \leq k$ then $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \leq n^{O(1)}$;

(2) if $\gamma(\mathcal{S}) > k^2$ then $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \geq n^{\Omega(k)}$.

Proof of Theorem 1.1. We prove the statement for GapETH, and defer the proof for ETH to Appendix B. Assuming that $\mathcal{Q}$ is n^f automatizable for some $f := f(n, S) = \tilde{o}(\log \log S)$, we describe an efficient algorithm for the gap hitting set problem. Given an instance $(\mathcal{S}, k, k^2)$ of the gap hitting set problem over $[n]$, with n sufficiently large and $k = \tilde{O}(\log \log n)$, we generate the CNF $\tau_{\mathcal{S}}$, and simulate the automatizing algorithm on $\tau_{\mathcal{S}}$ for $n^{O(f)}$ timesteps. If the automatizing algorithm outputs a legal Res refutation of $\tau_{\mathcal{S}}$ within the allotted time, then we output "$\gamma(\mathcal{S}) \leq k$" and otherwise output "$\gamma(\mathcal{S}) > k^2$". Because $S = n^{O(1)}$ it holds that $f = o(k)$, and so the correctness is guaranteed by Lemma 3.1. Thus we can decide the gap hitting set problem in time $n^{O(f)} = n^{o(k)}$, which by Lemma 2.2, contradicts GapETH. □

3.2 Reduction

The rest of the paper is devoted to the proof of Lemma 3.1. Hereafter, fix $k = \tilde{O}(\log \log n)$ and define $m := n^{1/k}$. Observe that $k \log m = \log n$ and $k < \frac{\log m}{4}$. In what follows we will abuse notation and x_i, y_j will denote a tuple of Boolean variables (rather than a single Boolean variable). The tuple size of x_i, y_j will be clear from context, but generally x_i will be a $O(\log m)$-tuple and y_j will be a $O(\log n)$-tuple. A vector of $\vec{x} = x_1, \ldots, x_n$, $\vec{y} = y_1, \ldots, y_m$ will denote a vector of tuples. α_i and β_j will denote a 0/1 assignment to the tuples x_i and y_j respectively, and $\vec{\alpha}, \vec{\beta}$ will each denote a 0/1 assignment to the vector of tuples $\vec{x}, \vec{y}$ respectively.

3.2.1 The basic reduction

Given a hitting set instance $\mathcal{S}$ we will define an unsatisfiable formula $\psi_{\mathcal{S}}$. The variables of $\psi_{\mathcal{S}}$ are:

- $\vec{x} = \{x_i \mid i \in [n]\}$ where x_i is a tuple of $\log m$ Boolean variables, and
- $\vec{y} = \{y_j \mid j \in [m]\}$ where y_j is a tuple of $\log n$ Boolean variables.

We will view an assignment α_i for x_i as choosing a position in $[m]$, and similarly we will view an assignment β_j for y_j as choosing a set S_{β_j} from $\mathcal{S}$.

Given an assignment $\vec{\alpha}$ to all of the $\vec{x}$-variables, we will associate with $\vec{\alpha}$ an n-by-m matrix $M_{\vec{\alpha}}$, where the ith row of $M_{\vec{\alpha}}$ will be the vector $a_{\alpha_i} \in A$ (interpreting α_i as a number in $[m]$). Similarly given an assignment $\vec{\beta}$ to all of the $\vec{y}$-variables, we will associate with $\vec{\beta}$ an n-by-m matrix $N_{\vec{\beta}}$, where column j is the characteristic vector corresponding to the set $S_{\beta_j} \in \mathcal{S}$ (interpreting β_j as a number in $[n]$). In other words, $N_{\vec{\beta}}[i,j]$ is 1 if and only if set S_{β_j} contains element i.

The CNF formula $\psi_{\mathcal{S}}$ will say that for any assignments $\vec{\alpha}, \vec{\beta}$, there is no location $[i,j]$ where both $M_{\vec{\alpha}}$ and $N_{\vec{\beta}}$ are 1. To motivate this definition, consider the jth column of $M_{\vec{\alpha}}$, and treat it as the characteristic vector of a set $H_j \subseteq [n]$. H_j is *not* a hitting set of $\mathcal{S}$ iff there exists some set $S^j \in \mathcal{S}$ such that $H_j \cap S^j = \emptyset$. Setting β_j such that $S_{\beta_j} = S^j$, we find that $H_j \cap S^j = \emptyset$ iff for all $i \in [n]$, either $M_{\vec{\alpha}}[i,j] = 0$ or $N_{\vec{\beta}}[i,j] = 0$. Thus $\psi_{\mathcal{S}}$ says that for all j, the set H_j defined by the jth column of M_α is not a hitting set of $\mathcal{S}$. For each H_j, the jth column of N_β witnesses this by encoding a set which is not hit by H_j.

Claim 3.2. $\psi_{\mathcal{S}}$ is unsatisfiable when $\gamma(\mathcal{S}) \leq \frac{\log m}{4}$.

Proof. Suppose without loss of generality that $H = \{1, 2, ..., \gamma(\mathcal{S})\}$ is a hitting set and $\gamma(\mathcal{S}) \leq \frac{\log m}{4}$. Consider any assignment $\alpha_1, \ldots, \alpha_{\gamma(\mathcal{S})}$ to $x_1 \ldots x_{\gamma(\mathcal{S})}$, which is a collection of at most $\gamma(\mathcal{S})$ vectors from A (note that there may be repetitions). Since A is $(m, \log m/4)$-dual-universal, and $\gamma(\mathcal{S}) \leq \log m/4$, there exists a j such that $M_{\vec{\alpha}}[i,j] = 1$ for all $i \in [\gamma(\mathcal{S})]$. Since $H = [\gamma(\mathcal{S})]$ is a hitting set, for any assignment $\vec{\beta}$ every column of $N_{\vec{\beta}}$ (in particular j) has a 1 somewhere in the first $\gamma(\mathcal{S})$ entries. Therefore there exists an i where both $M_{\vec{\alpha}}$ and $N_{\vec{\beta}}$ are 1 in entry $[i,j]$, which falsifies $\psi_{\mathcal{S}}$. □

Lastly we need to formalize $\psi_{\mathcal{S}}$ as a collection of clauses. To express that column j doesn't hit the set S_{β_j}, we will define a set $\mathbf{A}_j$ of *column axioms* consisting of clauses which together rule out all of the ways that the set H_j corresponding to column j of $M_{\vec{\alpha}}$ could hit the set S_{β_j}. It is important to view $M_{\vec{\alpha}}$ as a sequence of row vectors, where the i^{th} row is determined by α_i, and $N_{\vec{\beta}}$ as a sequence of column vectors where the j^{th} column is determined by β_j. We will sometimes write $M_{\vec{\alpha}}[i,j]$ as $M_{\alpha_i}[i,j]$ to stress that the entries $[i,*]$ of $M_{\vec{\alpha}}$ are determined by α_i. Similarly, we will sometimes write $N_{\vec{\beta}}[i,j]$ as $N_{\beta_j}[i,j]$.

Definition 3.3. For each $j \in [m]$, the set of clauses, $\mathbf{A}_j$, are defined as follows. For every $i \in [n]$ and for every pair of values $\alpha_i \in \{0,1\}^{\log m}$, $\beta_j \in \{0,1\}^{\log n}$ such that $M_{\alpha_i}[i,j] = 1$ and $N_{\beta_j}[i,j] = 1$, we have the clause

$$\overline{x_i^{\alpha_i} \wedge y_j^{\beta_j}}$$

where $x_i^{\alpha_i} = \wedge_{t\in[n]}(x_i)_t^{(\alpha_i)_t}$ is the conjunction of all variables in x_i, each of which occurs positively when the corresponding bit of α_i is 1 and negatively when the corresponding bit of α_i is 0 (we define $y_j^{\beta_j}$ in the same way). This axiom is falsified exactly when x_i is assigned value α_i and y_j is assigned value β_j.

The formula $\psi_{\mathcal{S}}$ is the conjunction of all clauses in $\cup_{j\in[m]}\mathbf{A}_j$. It is easy to check that the number of variables is $n\log m + m\log n$. For each $j \in [m]$, the number of clauses in $\mathbf{A}_j$ is at most $n^2 m$, since there are at most $n(nm)$ triples (i, α_i, β_j). Thus the total number of clauses is at most $n^2 m^2$.

3.2.2 A Redundant Encoding

In order to prove our result we will need a way of proving both upper and lower bounds on $S_{\mathcal{Q}}(\psi_{\mathcal{S}})$, but it turns out that the lower bounds are difficult to prove if we use $\psi_{\mathcal{S}}$ as is. Thus, we will employ a standard trick in proof complexity, which is to redundantly encode the variables in the formula. It is interesting to note that for our formulas, we are unable to prove even width lower bounds without the redundant encoding. In contrast, most proof complexity applications use this trick solely for the purpose of reducing size lower bounds to width lower bounds. To this effect we follow [2] and define a variant of $\psi_{\mathcal{S}}$ where the x and y variables are redundantly encoded using error correcting codes.

Definition 3.4. For $q, r, s \in \mathbb{N}$, a (q, r, s)-*code* is a function f from $\{0,1\}^q$ to $\{0,1\}^r$ with the property that for any $\rho \in \{0,1,*\}^q$ such that ρ fixes at most s values to $\{0,1\}$, $f|_\rho$ is surjective on $\{0,1\}^r$. Efficiently computable constructions using linear codes are known for any $r, q = 6r, s = 2r$ (see e.g. [2]). We say that f is r-surjective.

Let $f_x : \{0,1\}^{6\log m} \to [m]$ be a $(6\log m, \log m, 2\log m)$-code and let $f_x : \{0,1\}^{6\log n} \to [n]$ be a $(6\log n, \log n, 2\log n)$-code. We will have a vector $x_i \in \{0,1\}^{6\log m}$ for each $i \in [n]$ and a vector $y_j \in \{0,1\}^{6\log n}$ for each $j \in [m]$. Given an assignment $\vec{\alpha}$ to all of the $\vec{x}$-variables, we will associate with $\vec{\alpha}$ an n-by-m matrix $M_{\vec{\alpha}}$, where the ith row of $M_{\vec{\alpha}}$ will be the vector $a_{f_x(\alpha_i)} \in A$. Similarly given an assignment $\vec{\beta}$ to all of the $\vec{y}$-variables, we will associate with $\vec{\beta}$ an n-by-m matrix $N_{\vec{\beta}}$, where column j is the characteristic vector corresponding to the set $S_{f_y(\beta_j)} \in \mathcal{S}$ In other words, $N_{\vec{\beta}}[i,j]$ is 1 if and only if set $S_{f_y(\beta_j)}$ contains element i.

We now define our unsatisfiable CNF $\tau_{\mathcal{S}}$ in the same way as $\psi_{\mathcal{S}}$ using these redundant encodings. Note that it is unsatisfiable for exactly the same reason as stated before.

Definition 3.5. For each $j \in [m]$, the clauses $\mathbf{A}_j$ of $\tau_{\mathcal{S}}$ are defined as follows. For every $i \in [n]$ and for every pair of assignments (α_i, β_j) to (x_i, y_j) such that $M_{\alpha_i}[i,j] = 1$ and $N_{\beta_j}[i,j] = 1$, we have the clause $\overline{x_i^{\alpha_i} \wedge y_j^{\beta_j}}$.

The formula $\tau_{\mathcal{S}}$ is the conjunction of all clauses in $\cup_{j\in[m]}\mathbf{A}_j$. In the redundant encoding we have $n \cdot 6\log m$ x-variables and $m \cdot 6\log n$ y-variables, for a total of $O(n\log m)$ variables when $m = n^{1/k} \ll n$. For each $j \in [m]$ the number of clauses in $\mathbf{A}_j$ is at most $n^7 m^6$ since the total number of triples (i, α, β) is at most $n(n^6 m^6)$. Thus the size of $\tau_{\mathcal{S}}$ is at most $n^7 m^7$.

The following two lemmas, which will be the focus of the rest of the paper, give tight upper and lower bounds on $S_{\mathcal{Q}}(\tau_{\mathcal{S}})$ as a function of $\gamma(\mathcal{S})$. Since we can clearly construct $\tau_{\mathcal{S}}$ in time polynomial in n, proving these two lemmas is all we need to finish Lemma 3.1.

Lemma 3.6. *For sufficiently large n, let $(\mathcal{S}, k, k^2)$ be an instance of the gap hitting set problem over $[n]$ such that $\gamma(\mathcal{S}) \leq k$ and $k < \log m$. Then $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \leq n^{O(1)}$ for any $\mathcal{Q} \in \{$*Res, TreeRes, Nullsatz, PC, PCR, Res(r)*$\}$.*

Lemma 3.7. *For sufficiently large n, let $(\mathcal{S}, k, k^2)$ be an instance of the gap hitting set problem over $[n]$ such that $\gamma(\mathcal{S}) > k^2$ and $k < \log m$. Then $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \geq n^{\Omega(k)}$ for any $\mathcal{Q} \in \{$*Res, TreeRes, Nullsatz, PC, PCR*$\}$.*

3.2.3 Proof sketch

For any unsatisfiable CNF τ, the *search problem* associated with τ takes as input an assignment $\vec{\rho}$ to the underlying variables of τ, and should output some clause in τ that is falsified by $\vec{\rho}$. A decision tree for the search problem is defined in the obvious way: it is a decision tree over the variables of τ, where every leaf of the tree is labelled by some clause of τ, and for every assignment $\vec{\rho}$ to the variables, the (unique) path in the tree that is consistent with $\vec{\rho}$ is labelled with a clause that is falsified by $\vec{\rho}$. It is this search problem that we prove height upper and lower bounds on in the rest of the chapter, which will serve as a springboard into proving Lemmas 3.6 and 3.7 for all $\mathcal{Q}$ in Theorem 1.1

For the upper bound we simply need to formalize Claim 3.2 as a decision tree. Querying all rows in the smallest hitting set H will identify the j for which $M[i, j] = 1$ for all $i \in H$, and then we need only query y_j. For the lower bound we use the fact that if a decision tree queries too few rows then there is a set in $\mathcal{S}$ that is not hit by any row queried, and if it queries too few columns then there is a row vector from the universal set A which is 0 in all columns queried. Even if the decision tree is allowed to pick the order the rows and columns are queried in, as long as we pick a row/column that "misses" every column/row queried so far then we can always avoid violating an axiom. Finally by plugging in $O(\log m)$ variables for every row and $O(\log n)$ variables for every column we will get the lower bounds desired.

3.3 Decision tree bounds

3.3.1 Decision tree upper bound

Lemma 3.8 (Height upper bound). *If $\gamma(\mathcal{S}) \le k$ and $k \le \frac{\log m}{4}$, then there is a decision tree of height $O(\log n)$ solving the search problem on $\tau_{\mathcal{S}}$.*

Proof. We will first show that if $\gamma(\mathcal{S}) \le k$, then there is a height $2 \log n$ decision tree (and therefore size n^2) for the unencoded formula $\psi_{\mathcal{S}}$. Since $\gamma(\mathcal{S}) \le k$, assume without loss of generality that $H = \{1, \dots, k\}$ is a valid hitting set for $\mathcal{S}$. The decision tree for $\psi_{\mathcal{S}}$ consists of two phases. In the first phase the decision tree will branch on all of the Boolean variables in $x_1, \dots, x_k$. This will result in a full binary tree, call it T, of depth $k \log m$. In the second phase, at each leaf vertex of T we will query all of the variables of some y_j variable, where the choice of y_j will be a function of the path taken in T.

Consider some path in T leading to the leaf vertex $l_{\vec{\alpha}}$, corresponding to the assignment $\vec{\alpha} = \alpha_1, \dots \alpha_k$ for $x_1, \dots, x_k$. The assignment $\vec{\alpha}$ corresponds to an ordered set of strings $I \subseteq A$, where $|I| \le k$. Since $k \le \frac{\log m}{4}$, by the $(m, \log m/4)$-dual-universal property of A there is some $j \in [m]$ such that I restricted to position j is all 1's, and thus $M_{\vec{\alpha}}[i, j] = 1$ for all $i \in [k]$. In the second phase, at this leaf vertex $l_{\vec{\alpha}}$ of T we will then query all of the Boolean variables in y_j. Let β_j be one partial assignment to these variables and consider the path labeled by $\vec{\alpha}\beta_j$ leading to the leaf vertex $l_{\vec{\alpha}\beta_j}$. Since $\{1, \dots, k\}$ is a hitting set for $\mathcal{S}$ we are guaranteed that $N_{\vec{\beta}_j}[i, j] = 1$ for at least one $i \in [k]$, and since $M_{\vec{\alpha}}[i, j] = 1$ for all $i \in [k]$, one of the clauses in $\mathbf{A}_j$ must be violated by the partial assignment $\vec{\alpha}, \beta_j$, so we label $l_{\vec{\alpha}\beta_j}$ with any such clause. The resulting decision tree thus solves the search problem associated with $\psi_{\mathcal{S}}$ and has height $k \log m + \log n = 2 \log n$.

The decision tree for the redundant version $\tau_{\mathcal{S}}$ is essentially the same but now we query the redundant encodings of the variables instead. In the first phase we query $x_1, \dots, x_k$, resulting in a full binary tree of height $k \cdot 6 \log m$, and in the second phase we query a particular y_j (depending on the path taken in T), which is $6 \log n$ variables, and thus the height is $k \cdot 6 \log m + 6 \log n = 12 \log n$. □

Proof of Lemma 3.6. By Lemma 3.8 there is a decision tree solving the search problem for $\tau_{\mathcal{S}}$ of size at most $2^{12 \log n} = n^{O(1)}$. It is well-known that there is a simple size-preserving transformation between decision trees solving the search problem for τ and $\mathsf{TreeRes}$ refutations for τ. Therefore $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \leq n^{O(1)}$ for $\mathcal{Q} = \mathsf{TreeRes}$. The lemma follows by the fact that Res, $\mathsf{Nullsatz}$, PC, PCR, and $\mathsf{Res(r)}$ all p-simulate $\mathsf{TreeRes}$. □

3.3.2 Decision tree lower bound

We prove a stronger height lower bound as per our proof sketch using the following definitions:

Definition 3.9. For a collection of literals D, let $I_0(D)$ be the set of all $i \in [n]$ for which there are at least $\log m$ literals in D that correspond to variables from x_i. Likewise let $J_0(D)$ be the set of all $j \in [m]$ for which there are at least $\log n$ literals in D that correspond to variables from y_j.

Lemma 3.10 (Height lower bound)**.** *If $\gamma(\mathcal{S}) \geq k^2$ and $k \leq \frac{\log m}{4}$, then for any tree π solving the search problem on $\tau_{\mathcal{S}}$, there exists a path $p \in \pi$ from the root to a leaf such that $|I_0(p)| \geq k^2$ or $|J_0(p)| \geq k$.*

Proof. Assume for contradiction that $|I_0(p)| < k^2$ and $|J_0(p)| < k$ for all paths $p \in \pi$. We will inductively build a path p such that no clause has been violated as long as these two upper bounds hold, which is a contradiction as π must find a violated clause. Let z be the variable being queried in the current node. We perform as follows:

- if z is already in p, answer consistently with our previous answer
- if z is a variable in x_i:
  - if $i \notin I_0(p)$ and after adding z to p there are still less than $\log m$ variables from x_i in p, we branch arbitrarily.
  - if $i \notin I_0(p)$ but after adding z to p there are $\log m$ variables from x_i in p, we use the $(m, \log m/4)$-universal property of A to find a string $a_0 \in A$ such that $a_0|_{J_0(p)}$ is the all-zeros string, and use the surjective property of f_x to find an assignment α_i consistent with the assignment to the x_i variables in memory such that $f_x(\alpha_i) = a_0$. We store the assignment α_i for x_i from now on, and note that $I_0(p)$ now contains i.
  - if $i \in I_0(p)$ then we are maintaining an assignment α_i for x_i, and we answer according to α_i.
- if z is a variable in y_j:
  - if $j \notin J_0(p)$ and after adding z to p there are still less than $\log n$ variables from y_j in p, we branch arbitrarily.
  - if $j \notin J_0(p)$ but after adding z to p there are $\log n$ variables from y_j in p, we use the fact that $|I_0| < \gamma(\mathcal{S})$ to find a set $S_0 \in \mathcal{S}$ such that S_0 does not contain any element of I_0 and use the surjective property of f_y to find an assignment β_j consistent with the assignment to the y_j variables in memory such that $f_y(\beta_j) = S_0$. We store the assignment β_j for y_j from now on, and note that $J_0(p)$ now contains j.
  - if $j \in J_0(p)$ then we are maintaining an assignment β_j for y_j, and we answer according to β_j.

Clearly no axiom is violated, but for completeness assume for contradiction we reach a leaf labeled with the axiom $\overline{x_i^{\alpha_i} \wedge y_j^{\beta_j}}$, and thus π claims that $M_{\vec{\alpha}}[i,j] = N_{\vec{\beta}}[i,j] = 1$. First, consider the case when either $i \notin I_0$ or $j \notin J_0$. In either case there are is at least one variable in the axiom that is not in p, which means that it has not been falsified, which is a contradiction. So $i \in I_0$ and $j \in J_0$. Assume that i was added after j. Since j was in J_0 at the time we defined α_i, $M_{\alpha_i}[i, j] = 0$ by our choice of α_i, which is a contradiction. Finally assume that j was added after i. Then since i was in I_0 at the time we defined β_j, $f_y(\beta_j)$ does not contain i, and so $N_{\beta_j}[i, j] = 0$, which is also a contradiction. □

Unlike for Lemma 3.6, Lemma 3.10 is not enough to give a proof of Lemma 3.7 even for TreeRes. Proving Lemma 3.7 using the adversarial argument of Lemma 3.10 is the subject of our next chapter.

Chapter 4

Lower bounds

The decision tree upper bound given in Chapter 3 is enough to give the upper bound in Lemma 3.6 for Res, Nullsatz, PC, PCR, and Res(r), since all these systems p-simulate TreeRes. However Lemma 3.7 remains to be proven for all these proof systems. The core of the argument is analogous to the case of TreeRes, and so we leverage the same adversarial argument in each of the different settings to give the desired lower bound.

4.1 Res lower bounds

In this section we prove Lemma 3.7 for the case of $\mathcal{Q} = \mathsf{Res}$, which implies the result for TreeRes as well. We begin by proving a *wide clause lemma* for $\tau_{\mathcal{S}}$, which alone is enough to prove lower bounds for TreeRes (using the size-width relationship for TreeRes due to Ben-Sasson and Wigderson [9]); for general Res, we apply a standard application of random restrictions to reduce to width.

Definition 4.1. For a clause D, let $I_0(D)$ be the set of all $i \in [n]$ for which there are at least $\log m$ literals in D that correspond to variables from x_i. Likewise let $J_0(D)$ be the set of all $j \in [m]$ for which there are at least $\log n$ literals in D that correspond to variables from y_j.

Lemma 4.2 (Wide Clause Lemma)**.** *If $\gamma(\mathcal{S}) \geq k^2$ and f_x (f_y) is $\log m$-surjective ($\log n$-surjective, respectively), then for any* Res *refutation π refuting $\tau_{\mathcal{S}}$ there exists a clause $D \in \pi$ such that $|I_0(D)| \geq k^2$ or $|J_0(D)| \geq k$.*

Proof. We follow the *prover-delayer game* of [4, 31] in the style of [5]. The width-w game on an unsatisfiable formula τ is played between a Delayer, who is asserting that she has a satisfying assignment for τ, and a Prover, who is trying to force the Delayer into a contradiction by asking her values of the underlying variables. However, the Prover has limited memory and can only remember the values of up to w of the variables at a time.

Both players know τ and the contents of the Prover's memory, which is initially empty. At the start of each round there are at most $w - 1$ values in memory. The Prover asks the Delayer the value of some variable whose value is not currently in memory. The Delayer responds with an answer (either 0 or 1), and upon receiving the answer, the Prover adds this assignment to his memory (increasing the number of stored values by 1). He can then erase (forget) any existing values from memory, possibly decreasing the number of stored values. The Prover declares victory if at some point, the partial assignment written in his memory falsifies one of the clauses of τ. The Delayer has a winning strategy for the width-w game on τ if no matter how the Prover plays the game, he cannot win. It was shown [4, 31] that the Delayer has a winning strategy for the width-w game if and only if the Res width of τ is at least w.

For our tautology $\tau_{\mathcal{S}}$, the game proceeds as above, but now let D be the set of literals in the Prover's memory, and we demand instead of only holding w variables total in memory that $|I_0(D)| < k^2$ and $|J_0(D)| < k$. Now the Delayer has a winning strategy for this game if and only if the lemma holds. The Delayer's winning strategy is nearly identical to in the proof of Lemma 3.10, with the only adjustment being for what to do when a literal is erased from memory.

- if the Prover asks about a variable in memory, we answer consistently
- If the Prover asks about a variable in x_i:
  - If $i \notin I_0(D)$ and after adding this bit there are still less than $\log m$ variables from x_i in memory, the Delayer can answer with either 0 or 1 arbitrarily.
  - If $i \notin I_0(D)$ but after adding this bit to memory there are now $\log m$ variables from x_i in memory, the Delayer uses the $(m, \log m/4)$-universal property of A to find a string $a_0 \in A$ such that $a_0|_{J_0(D)}$ is the all-zeros string, and uses the surjective property of f_x to find an assignment α_i consistent with the assignment to the x_i variables in memory such that $f_x(\alpha_i) = a_0$. The Delayer will remember the assignment α_i for x_i from now on, and note that $I_0(D)$ now contains i.
  - Finally if $i \in I_0(D)$ then the Delayer is maintaining an assignment α_i for x_i, so she answers according to α_i.
- If the Prover asks about a variable in y_j:
  - If $j \notin J_0(D)$ and after adding this bit there are still less than $\log n$ variables from y_j in memory, the Delayer can answer with either 0 or 1 arbitrarily.
  - If $j \notin J_0(D)$ but there are now $\log n$ variables from y_j in memory, the Delayer uses the fact that $|I_0(D)| < \gamma(\mathcal{S})$ and finds a set S_0 that doesn't contain any element $i \in I_0(D)$, and uses the surjective property of f_y to find an assignment β_j consistent with the assignment to the y_j variables in memory such that $f_y(\beta_j) = S_0$. The Delayer will remember the assignment β_j for x_j, and note that $J_0(D)$ now contains j.
  - Finally if $j \in J_0(D)$ then the Delayer is already maintaining an assignment β_j for y_j, so she answers according to β_j.
- Whenever the Prover erases a variable from x_i from his memory, if $i \in I_0$ and now there are less than $\log m$ variables from x_i in memory, the Delayer forgets α_i. (note that i is no longer in I_0) Similarly, whenever the Prover erases a variable from y_j from his memory, if $j \in J_0$ and now there are less than $\log n$ variables from y_j in memory, the Delayer removes β_j from J_0. (note that j is no longer in J_0)

Assume for contradiction the game ends with the Prover winning. Consider when the game ends, and say the Prover claims the axiom $\overline{x_i^{\alpha_i} \wedge y_j^{\beta_j}}$ was falsified, and thus that $M_{\vec{\alpha}}[i,j] = N_{\vec{\beta}}[i,j] = 1$. First, consider the case when either $i \notin I_0$ or $j \notin J_0$. In either case there are is at least one variable in the axiom that is not in memory, which means that it has not been falsified, which is a contradiction. So assume that $i \in I_0$ and $j \in J_0$, and consider the last time that i was added to I_0 and the last time that j was added to J_0. Assume that i was added after j. Since j was in J_0 at the time we defined α_i, $M_{\alpha_i}[i,j] = 0$ by our choice of α_i, which is a contradiction. Finally assume that j was added after i. Then since i was in I_0 at the time we defined β_j, $f_y(\beta_j)$ does not contain i, and so $N_{\beta_j}[i,j] = 0$, which is also a contradiction. □

*Proof of Lemma 3.7 (*TreeRes, Res*).* Let π be a Res refutation of $\tau_{\mathcal{S}}$ and assume for contradiction that $|\pi| < n^{k/16}$. Let $\rho_{x_i} \in \{0,1,*\}^{x_i}$ and let $\rho_{y_j} \in \{0,1,*\}^{y_j}$ Let $\mathcal{R}$ be the set of all $\vec{\rho} = \{\rho_{x_1} \dots \rho_{x_n}, \rho_{y_1} \dots \rho_{y_m}\}$, such that for all $i \in [n]$ and $j \in [m]$, $|\rho_{x_i}^{-1}(*)| = 5\log m$ and $|\rho_{y_j}^{-1}(*)| = 5\log n$. Observe that every such restriction fixes exactly $\frac{5}{6}$ of the variables in each x_i and each y_j to $*$ and the rest of the variables to $\{0,1\}$ uniformly at random. Also note that f_x was $2\log m$ before the restriction, and since only $\log m$ variables are fixed in every row $f_x|_{\rho\sim\mathcal{R}}$ is still $\log m$ surjective (and similarly for f_y).

First, consider a clause $D \in \pi$ such that $|I_0(D)| \geq k^2$. For each $i \in I_0(D)$, the chance that a randomly chosen $\vec{\rho} \in \mathcal{R}$ doesn't set one of the x_i literals in D to 1 is less than $(1 - (\frac{1}{6} \cdot \frac{1}{2}))^{\log m}$. Thus the probability that no $i \in I_0(D)$ sets D to 1 is at most $(\frac{11}{12})^{k^2 \log m} = (\frac{11}{12})^{k \log n} < \frac{1}{n^{k/8}}$. By a union bound the probability that some clause D in π satisfying $|I_0(D)| \geq k^2$ survives a random restriction is less than $\frac{n^{k/16}}{n^{k/8}} = \frac{1}{n^{k/16}}$, using the fact that $|\pi| < n^{k/16}$.

Similarly the probability that some clause $D \in \pi$ satisfying $|J_0(D)| \geq k$ survives a random restriction is at most $\frac{1}{n^{k/16}}$. Thus with probability at least $1 - \frac{2}{n^{k/16}}$, all clauses D satisfying $|I_0(D)| \geq k^2$ or $|J_0(D)| \geq k$ are set to 1 by a random restriction, and thus there exists a restriction $\vec{\rho}$ setting all such clauses to 1. However even after restricting $\tau_{\mathcal{S}}$ by $\vec{\rho}$, the function f_x is still a $(5\log m, \log m, \log m)$-code and the function f_y is still a $(5\log n, \log n, \log n)$-code. Thus we can still apply Lemma 4.2 to $\tau_{\mathcal{S}}|_{\vec{\rho}}$, even with $\log m$ x_i variables missing in every row and $\log n$ y_j variables missing in every column. Since $\pi|_{\vec{\rho}}$ is a refutation of $\tau_{\mathcal{S}}|_{\vec{\rho}}$, by Lemma 4.2 it must have a clause D with either $I_0(D) \geq k^2$ or $J_0(D) \geq k$, which is a contradiction of the fact that $\vec{\rho}$ sets all such clauses to 1. Thus $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \geq n^{c_l k}$ for $c_l = \frac{1}{16}$. □

4.2 Nullsatz/PC/PCR lower bounds

Galesi and Lauria [19] extended the argument due to Alekhnovich and Razborov [2] to prove that Nullsatz, PC and PCR are also not polynomially automatizable. In this section we similarly extend our proof to apply to these systems, obtaining our improved bounds as well. Namely we prove Lemma 3.7 for the case of $\mathcal{Q} =$ PCR, and by extension Nullsatz and PC.

The strategy is to prove a degree version of the wide clause lemma for $\tau_{\mathcal{S}}$ in the style of Lemma 4.2, and then the same random restriction argument as before will prove the size lower bound needed for Lemma 3.7. Recalling the definitions for I_0, J_0 in Lemma 4.2, for any monomial t let $I_0(t)$ be the set of all $i \in [n]$ for which at least $\log m$ variables from x_i appear in t, and let $J_0(t)$ be the set of all $j \in [m]$ for which at least $\log n$ variables from y_j appear in t. Recall that for PCR there exist distinct variables z and $\overline{z}$, both of which we consider to be variables from their respective x_i or y_j.

Lemma 4.3. *If $\gamma(\mathcal{S}) \geq k^2$, then for any* PCR *refutation π refuting $\tau_{\mathcal{S}}$, there exists a monomial $t \in p \in \pi$ such that $|I_0(t)| \geq k^2$ or $|J_0(t)| \geq k$.*

Proof. Given a set $P = \{p_1, \dots, p_m\}$ of polynomials over $F[x_1, \dots, x_n]$, we denote by $span(P)$ the ideal generated by P – that is the set $\{\sum_i p_i f_i \mid f_i \in F[x_1, \dots, x_n]\}$. A set of polynomials $f_1, \dots, f_n$ semantically implies a polynomial g if any assignment that satisfies $f_i = 0$ for all $i \in [n]$ also satisfies $g = 0$. Note that $p \in span(P)$ if and only if P semantically implies p, which we write as $P \vdash p$.

Recall that $\mathcal{P}$ is our set of input clauses converted to polynomial form, and $\mathbf{A}_j$ is the set of clauses associated with column j. Accordingly let $\mathcal{P}_j$ denote the corresponding set of polynomials plus the equations $\{z^2 - z = 0\}$ for every variable z in τ. For a subset $J \subseteq [m]$ of columns, let $\mathcal{P}_J$ denote $\cup_{j \in J} \mathcal{P}_j$, and thus $span(\mathcal{P}_J)$ is the ideal generated by the polynomials $\mathcal{P}_J$.

We will prove our degree bound for PCR refutations of τ_S by defining a linear operator K which maps polynomials p where $|I_0(t)| < k^2$ and $|J_0(t)| < k$ for all $t \in p$ to polynomials q, and satisfies the following conditions:

1. For all initial polynomials $p \in \cup_{j\in[m]}\mathcal{P}_j$, $K(p) = 0$.
2. K is linear: $K(ap + bq) = aK(p) + bK(q)$ for all constants a, b and polynomials p, q.
3. $K(xt) = K(xK(t))$ for all x
4. $K(1) \neq 0$

The existence of such an operator implies our degree bound as follows. Given an alleged PCR refutation which contains no monomial t where $I_0(t) \geq k^2$ or $J_0(t) \geq k$, applying K to every line in the proof, we have by the properties of K in conditions 1, 2, and 3 that $K(p) = 0$ for every polynomial in the proof. On the other hand since the final line is 1, by property 4 $K(1) \neq 0$, which is a contradiction.

We fix the grlex (graded lexicographical) ordering on all polynomials over $F[x_1, \ldots, x_n]$. Given a polynomial q and $J \subseteq [m]$, let $R_J(q)$ be the minimal (with respect to $<$) polynomial p such that $q - p \in span(\mathcal{P}_J)$. For every monomial t we set $K(t) = R_{J_0(t)}(t)$, and for $p = \sum_i c_i t_i$ set $K(p) = \sum_i c_i K(t_i)$. Intuitively $K(t)$ is how "close" t is to being in the span of the axioms in all columns with many variables in t. Note that this definition is asymmetric with respect to $\vec{x}$ and $\vec{y}$.

We now show the conditions of the linear operator are fulfilled. Consider any initial polynomial, p. If p is $z^2 - z$, then since $K[z^2] = K[z]$, $K[z^2 - z] = 0$ as required. Otherwise p is of the form $x_i^{\alpha_i} y_j^{\beta_i} = 0$. Note that p is a single monomial with $4 \log n$ variables of the form y_j and no variables of the form $y_{j'}$ for $j \neq j'$. So $J_0(p) = \{j\}$ and thus $R_{J_0(p)}(p) = 0$, which fulfills condition 1. By definition K is a linear operator, which fulfills 2. Because $J_0(1) = \emptyset$, $1 \notin span(\mathcal{P}_{J_0(1)})$, and so condition 4 is satisfied.

To prove condition 3, let us first prove the intuitive direction of the equality, namely that $K(xt) \geq K(xK(t))$. We repeatedly make use of the fact that if $J \subseteq J'$ then $R_J(t) \geq R_{J'}(t)$.

$$\begin{aligned}
K(xt) &= R_{J_0(xt)}(xt) \\
&= R_{J_0(xt)}(xR_{J_0(xt)}(t)) \\
&\geq R_{J_0(xt)}(xR_{J_0(t)}(t)) && (1) \\
&= R_{J_0(xt)}(xK(t)) \\
&\geq R_{J_0(xK(t))}(xK(t)) && (2) \\
&= K(xK(t))
\end{aligned}$$

In order to get equality, it is enough to show that (1) and (2) can be made equalities. For (2) note that if we expand $xK(t)$ as a polynomial and apply the linear operator $R_{J_0(xt)}$ to each term, we get that equality holds iff for all monomials t' in $xK(t)$,

$$R_{J_0(xt)}(t') = R_{J_0(xK(t))}(t').$$

We now observe that $J_0(xt) \subseteq J_0(t)$ and $J_0(t') \subseteq J_0(xt), J_0(xK(t))$. Therefore to finish the proof of condition 3 and thus the lemma, we prove the following claim:

Claim 4.4. For all t where $|I_0(t)| < k^2$ and all $J \supseteq J_0(t)$ such that $|J| < k$, $R_{J_0(t)}(t) = R_J(t)$.

We need to show that $R_{J_0(t)}(t) \geq R_J(t)$ and $R_{J_0(t)}(t) \leq R_J(t)$. The first inequality holds trivially because $J_0(t) \subseteq J$, meaning that any $p \in span(\mathcal{P}_{J_0(t)})$ is also in $span(\mathcal{P}_J)$ as well. Now we prove the other direction,

$R_{J_0(t)}(t) \leq R_J(t)$. If we can show that $t - R_J(t) \in span(\mathcal{P}_{J_0(t)})$, then since $R_{J_0(t)}$ is the smallest polynomial p for which $t - p \in span(\mathcal{P}_{J_0(t)})$, it follows that $R_J(t) \geq R_{J_0(t)}$ as desired. This is equivalent to showing that $\mathcal{P}_{J_0(t)} \vdash t - R_J(t)$ by definition of $span$, which is the statement we will now prove.

Assume for contradiction that there exists an assignment $\vec{\alpha}, \vec{\beta}$ that satisfies all axioms in $\mathcal{P}_{J_0(t)}$ but falsifies $t - R_J(t)$. We then prove that there exists an assignment $\vec{\alpha}', \vec{\beta}'$ that satisfies all axioms in $\mathcal{P}_J$ but doesn't touch any variables in $t - R_J(t)$, This means that $\mathcal{P}_J$ doesn't imply $t - R_J(t)$, which contradicts the fact that by definition of R, $t - R_J(t) \in span(\mathcal{P}_J)$. For this we note that the set of variables in t is a superset of the variables in $t - R_J(t)$, and thus $|I_0(t - R_J(t))| < k^2$ and $|J_0(t - R_J(t))| < k$ as per the claim. For simplicity we will refer to these sets as simply I_0 and J_0.

Consider a row $i \in [n] - I_0$. By the $(m, \log m/4)$-universal property of A there exists a string $a \in A$ which is zero in all positions $j \in J$. Since there are at most $\log m$ x_i variables in $t - R_J(t)$, we can leave α_i untouched on those variables and change the rest to give us α_i', such that $f_x(\alpha_i') = a$. We do this for all such i, noting that no variables in $t - R_J(t)$ have been changed.

Now consider a row $j \in J - J_0$. Let S_0 be a set that doesn't contain any $i \in I_0$, given to us by the fact that $I_0 < k^2 < \gamma(S)$. Since there are at most $\log n$ y_j variables in $t - R_J(t)$, we can leave β_j untouched on those variables and change the rest to give us β_j', such that $f_y(\beta_j') = S_0$. We do this for all such β_j', noting again that no variables in $t - R_J(t)$ have been changed.

We now claim that $\vec{\alpha}', \vec{\beta}'$ satisfies all axioms in $\mathcal{P}_J$. Consider a row $j \in J_0$. Assume an axiom for row i and column j was violated. If $i \notin I_0$, we are guaranteed that $f_x(\alpha_i')$ is 0 in the jth entry, so it must be that $i \in I_0$. But then we haven't changed α_i or β_j, and since the original assignment satisfied all axioms in $\mathcal{P}_{J_0}$ the axiom could not have been violated by $\vec{\alpha}', \vec{\beta}'$. Now consider a row $j \in J - J_0$. Assume an axiom for row i and column j was violated. Again if $i \notin I_0$, we are guaranteed that $f_x(\alpha_i')$ is 0 in the jth entry, so it must be that $i \in I_0$. But then we changed β_j such that $f_y(\beta_j')$ is 0 in the ith row, and so the axiom could not have been violated by $\vec{\alpha}', \vec{\beta}'$. □

*Proof of Lemma 3.7 (*Nullsatz, PC, PCR*).* Assume for contradiction that there exists a PCR proof π refuting $\tau_{\mathcal{S}}$ in size less than $n^{k/16}$. We apply the same restriction from the proof of Lemma 3.7 to the positive variables $\vec{x}, \vec{y}$ in every line . Then for the negative variables we set $\overline{z}$ to be $*$ if z is set to $*$ and $1 - z$ otherwise. The same analysis proves that there exists a restriction ρ which sets every monomial t with $|I(t)| \geq k^2$ or $|J(t)| \geq k$ to 0, and the remaining proof $\pi|_\rho$ is a refutation of $\tau_{\mathcal{S}}|_\rho$. Because f_x is still a $(5\log m, \log m, \log m)$ code and f_y is still a $(5\log n, \log n, \log n)$ code, $\pi|_\rho$ still requires such a monomial by Lemma 4.3, which is a contradiction. □

4.3 Res(r) lower bounds

There is an inherent tradeoff between the lower bound on $S_{\mathcal{Q}}(\tau_{\mathcal{S}})$ for $\mathcal{Q} =$ Res(r) and the width of the terms r. In Chapter 5 we will see a generalization of Res(r) for which Lemma 3.7 does not hold, and in fact the argument holds for Res(r) where $r = \Omega(\log n)$. We prove a version of Lemma 3.7 that reflects this tradeoff and then state Theorem 1.1 in the case of Res(r), along with the two most extreme settings for r.

Lemma 4.5 (Lemma 3.7 for Res(r))**.** *For sufficiently large n, let $(\mathcal{S}, k, k^2)$ be an instance of the gap hitting set problem over $[n]$ such that $\gamma(\mathcal{S}) > k^2$ and $k^2 < \log m$. Then for $\mathcal{Q} =$ Res(r), $S_{\mathcal{Q}}(\tau_{\mathcal{S}}) \geq n^{k/\exp(r^2)}$.*

Proof. Suppose that π is a small Res(r) refutation of $\tau_{\mathcal{S}}$; thus each line of the proof is a disjunction of size-r conjunctions (where $r \ll \log n$). At a high level, we will show that there exists a random restriction $\rho \in \mathcal{R}$ (defined in the proof of Lemma 3.7) such that $\pi|_\rho$ is a small-width Res proof refuting $\tau_{\mathcal{S}}|_\rho$, which contradicts the Wide Clause Lemma (Lemma 4.2).

In order to prove the existence of such a restriction, we will apply the switching lemma proven in [33] which is specifically designed to work for Res(r). More specifically, the restriction will leave a constant fraction of the variables unset. In contrast, standard switching lemmas such as those used to prove bounded-depth circuit lower bounds leave at most $1/\log n$ variables unset. The fact that the restriction is small (sets only a constant fraction of variables) will allow us to maintain that $\tau_{\mathcal{S}}|_\rho$ is still an encoded version of $\tau_{\mathcal{S}}$, but where the encoding length is somewhat smaller. Whereas standard switching lemmas typically convert disjunctions of r-conjunctions to decision trees of height r (in order to apply it repeatedly), in our case, we only need to apply the switching lemma once, and therefore we are content with converting disjunctions of r-conjunctions to decision trees of height w, where r is much smaller than w. This setting of parameters is what makes it possible to obtain a switching lemma that sets only a constant fraction of the inputs. After applying the restriction, the proof is a sequence of sound inferences, where each line is a height-w decision tree. [33] show how to convert such a refutation into a width w' refutation, where w' is not much larger than w, and thus we can apply our Wide Clause Lemma in order to obtain a contradiction.

The switching lemma (showing the existence of the restriction ρ) is argued in stages; in stage i we show that for any i-DNF D either there exist many restrictions in $\mathcal{R}$ that set D to 1 or we can create a small height decision tree with each leaf labeled by D restricted by the path to the leaf leaf, and such that the resulting DNF at every leaf is a $(i-1)$-DNF. To do this we take the i-DNF from the previous round consider its *covering number*, where the covering number $c(D)$ is the size of the smallest set of variables which intersects every term in D. If the covering number is large, then many terms are independent and are thus set to 1 by a random restriction with high probability. If the covering number is small, then we can query all variables in the cover to turn D into a $(i-1)$-DNF. Continuing until $i = r$ gives us a small height decision tree for all $D \in \pi$ with small $c(D)$, while taking a union bound over all $D \in \pi$ with large $c(D)$ ensures that there exists a restriction $\rho \in \mathcal{R}$ that kills off all such DNFs. The resulting proof π can then be shown to have a small Res proof given these two facts, which completes the proof.

Let s be a parameter to be set later, and assume for contradiction that there exists a Res(r) proof π such that $|\pi| < n^s$. Define sequences $s_0 \dots s_r$, $p_1 \dots p_r$ as follows:

$$s_0 = (\prod_{i=1}^{r} \frac{2(6/5)^{i+1}}{i}) s \log n$$

$$s_i = (\frac{i}{2(6/5)^{i+1}}) s_{i-1}$$

$$p_i = 2^{-2s_i}$$

Observe that that $s_k = s \log n$, and that $s_i \gg \frac{s_{i+1}}{4}$.

Consider any i-DNF D such that $c(D) > s_i$. By the pigeonhole principle there exist s_i/i terms $T_1 \dots T_{s_i/i}$ in D which are mutually disjoint. Let $\rho \sim \mathcal{R}$ be defined as in Lemma **??**. Then the probability that D is not set to 1 by ρ is at most the probability that no term T_j is set to 1, and since they are disjoint this happens with probability $(1 - (\frac{5}{6})^i)^{s_i/i} = e^{-(6/5)^i s_i/i} < 2^{-2s_{i+1}} = p_{i+1}$.

Now consider an r-DNF D. We claim that

$$\Pr_{\rho \sim \mathcal{R}}[\mathsf{dt}(F|_\rho) > \sum_{i=0}^{r-1} s_i] \leq \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r-1} s_j)} p_i$$

where $\mathsf{dt}(F|_\rho)$ is the height of a minimal decision tree for $F|_\rho$. We prove this claim by induction on r. In the case when $r = 1$, either $c(D) \le s_0$, in which case the claim holds trivially, or $c(D) > s_0$, in which case it is killed off with probability $p_1 = p_1 2^{\sum_{j=1}^{r-1} s_j}$ for $r - 1 = 0$.

Inductively assume the claim holds for all $r-1$-DNFs and consider an r-DNF D. Again we consider two cases, when $c(D) \le s_{r-1}$ and when $c(D) > s_{r-1}$. In the former case, let H be the set of s_{r-1} variables needed to cover all terms of D, and note that $\mathsf{dt}(D) \le \mathsf{dt}(D/H) + s_{r-1}$ as we can query all variables in H first. Since D/H is an r-DNF, applying the induction hypothesis along with a union bound over all $2^{s_{r-1}}$ settings of H gives

$$\Pr_{\rho\sim\mathcal{R}}[\mathsf{dt}(D|_\rho) > \sum_{i=0}^{r-1} s_i] \le 2^{s_{r-1}} \Pr_{\rho\sim\mathcal{R}}[\mathsf{dt}((D/H)|_\rho) > \sum_{i=0}^{r-2} s_i] \le 2^{s_{r-1}} \sum_{i=1}^{r-1} 2^{(\sum_{j=i}^{r-2} s_j)} p_i \le \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r-1} s_j)} p_i$$

In the latter case, when $c(D) > s_{r-1}$, as shown before

$$\Pr_{\rho\sim\mathcal{R}}[\mathsf{dt}(D|_\rho) > \sum_{i=0}^{r-1} s_i] \le \Pr_{\rho\sim\mathcal{R}}[\mathsf{dt}(D|_\rho) > 0] \le p_r \ll \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r-1} s_j)} p_i$$

We now use this claim and take a union bound over all $D \in \pi$ to show that there exists a restriction ρ which makes all $D \in \pi$ have a small decision tree, which we then connect to the width of any Res proof of $\tau_{\mathcal{S}}|_\rho$ to get a contradiction.

$$\begin{aligned}
\Pr_{\rho\sim\mathcal{R}}[\exists D \in \pi \mid \mathsf{dt}(D|_\rho) > \sum_{i=0}^{r-1} s_i] &\le n^s \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r-1} s_j)} p_i \\
&\le \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r-1} s_j) + s\log n} p_i \\
&\le \sum_{i=1}^{r} 2^{(\sum_{j=i}^{r} s_j)} p_i \\
&\le \sum_{i=1}^{r} 2^{\frac{4}{3} s_i} 2^{-2 s_i} \\
&\le \sum_{i=1}^{r} 2^{-\frac{2}{3} s_i} \\
&\le r 2^{-\frac{2}{3} s_r} \\
&\le 2^{\log r - \frac{2}{3} s \log n} \\
&\ll \tfrac{1}{2}
\end{aligned}$$

Thus there exists a $\rho \in \mathcal{R}$ such that for all $D \in \pi$,

$$\mathsf{dt}(D|_\rho) \le \sum_{i=0}^{r-1} s_i \le r \cdot s_0 \le r((\prod_{i=1}^{r} \frac{2(6/5)^{i+1}}{i}) s \log n) \ll \frac{2^{r^2}}{r} s \log n$$

Set $s = \frac{k}{2^{r^2}}$, and thus $\mathsf{dt}(D|_\rho) \ll \frac{k \log n}{r}$. So $\pi|_\rho$ is a $\mathsf{Res(r)}$ proof where every line can be represented by a decision tree of height $\frac{k \log n}{r}$. It was shown (Theorem 5.1 in [33]) that these clauses can be made into a Res proof π' refuting $\tau_{\mathcal{S}}|_\rho$ such that $w(\pi') \ll r \frac{k \log n}{r} = k \log n$. But as usual we can still apply Lemma 4.2 after restricting $\tau_{\mathcal{S}}$ by ρ, and thus we get a contradiction. □

Theorem 4.6 (Theorem). *Let $\mathcal{Q} = \mathsf{Res(r)}$. Assuming GapETH holds, $\mathcal{Q}$ is not $n^{f/\exp(r^2)}$-automatizable for any $f = \tilde{o}(\log\log n)$ if $r = O(\sqrt{\log f})$. Furthermore, assuming ETH holds $\mathcal{Q}$ is not $n^{f/\exp(r^2)}$-automatizable for any $f = O(\log^{1/5} \log n)$ if $r = O(\sqrt{\log f})$.*

Corollary 4.7. *Assuming* GapETH*, for any constant c,* Res(c) *is not automatizable in time n^k for any $k = \tilde{O}(\log\log S)$. Assuming* ETH*, for any constant c,* Res(c) *is not automatizable in time n^k for any $k = O(\log^{1/5}\log S)$.*

Corollary 4.8. *Assuming* GapETH*,* Res(r) *is not automatizable for any $r = \tilde{o}(\sqrt{\log\log\log n})$. Assuming* ETH*,* Res(r) *is not automatizable for any $r = o(\sqrt{\log\log^{1/5}\log n})$.*

Chapter 5

Other proof systems

We have now proved Theorem 1.1 in full, getting lower bounds for Res, Nullsatz, PC, PCR, and Res(r). Following this program the next step would be to extend the argument for other well-known proof systems for which we believe the lower bound in Lemma 3.7 holds. In this chapter we address the two most logical choices, Sherali-Adams and Cutting Planes. We give an overview of known lower bounds techniques for both systems and how they apply to our tautology τ_S.

5.1 Sherali-Adams (SA)

5.1.1 Sherali-Adams and pseudodistributions

The *Sherali-Adams* (SA) refutation system was originally conceived [34] as a hierarchy of linear programs, where the 0th level is defined by a set of *inequalities* $\mathbb{A} = \{P_1 \geq 0, P_2 \geq 0, \ldots, P_{|\mathbb{A}|} \geq 0\}$ on n variables, and the nth level is the true feasible region $\mathbb{A} \cap \{0,1\}^n$ over the boolean hypercube. Each level $i \in [0, n]$ has $\binom{n}{i}$ elements, each of which is defined by the inequalities $\mathbb{A}$ restricted to all $\{0,1\}$ assignments to a different set of i variables. When $\mathbb{A}$ is infeasible over the Boolean hypercube, the nth level will be empty, but it's possible that there will be some level $d < n$ for which the feasible region is already empty, which we call a *degree-d Sherali-Adams refutation*.

This hierarchy can be formalized in a different way. For disjoint sets $S, T \subseteq X$, we call $Q = \prod_{s \in S} x_s \prod_{t \in T}(1 - x_t)$ a *junta*. Define $\mathbb{J} = \{Q \mid S, T \subseteq X\}$ to be the set of all juntas, and let $Q_1, Q_2, \ldots, Q_{|\mathbb{J}|}$ be an arbitrary ordering on $\mathbb{J}$. A *Sherali-Adams derivation* of a constant k from the axioms $\mathbb{A} = \{P_1 \geq 0, P_2 \geq 0, \ldots, P_{|\mathbb{A}|} \geq 0\}$ is a set of non-negative constants $C = \{c_{i,j} \in \mathbb{R}^{\geq 0} \mid i \in |\mathbb{A}|, j \in |\mathbb{J}|\}$ such that $R := \sum_{i,j} c_{i,j} P_i Q_j = k$ as a formal polynomial. The juntas Q_j naturally act as the restriction of the axiom P_i to an assignment of the variables in the hierarchy.

We say that Sherali-Adams *refutes* $\mathbb{A}$ if it can derive a negative value from $\mathbb{A}$. Note that setting $c_{i,j} = 0$ for all i, j allows us to derive 0 trivially from any set $\mathbb{A}$, and that if we can derive any negative value $-a$ with C, we can derive any other negative value $-b$ with $\frac{b}{a}C$, where kC is entrywise multiplication of C by positive constant k. Hence we usually focus on deriving $R = -1$ without loss of generality. A *degree-d Sherali-Adams refutation* is a Sherali-Adams refutation in which $c_{i,j} = 0$ whenever $\deg(P_i Q_j) > d$.

In this paper we assume that all axioms P_i are translated from clauses as per Chapter 2 (e.g. $x_1 \vee \overline{x_2} \rightarrow (1 - x_1)(x_2) = 0 \rightarrow \pm(1 - x_1)(x_2) \geq 0$), and thus have degree equal to the width of the original clause. Note that when we translate the equality $P_i = 0$ into two inequalities $\pm P_i \geq 0$, the positive version is captured by some junta Q_j, so we think of $\mathbb{A}$ as having the negative versions only. Additionally $\mathbb{A}$ contains the inequality $\pm(z^2 - z) \geq 0$ for each variable z, as well as the axiom $1 \geq 0$. Note that this last axiom was not present in Nullsatz or PC, as it would immediately give us the target line $1 = 0$. Without this axiom, SA would be strictly weaker than Nullsatz, but with it we are allowed to add juntas to the left hand side.

If there existed a satisfying assignment to $\mathbb{A}$, then plugging in the assignment would always give us $R \geq 0$, and so it would be impossible to derive $R = -1$ as a formal polynomial. Likewise if there existed a *distribution* on satisfying assignments to $\mathbb{A}$, then plugging in the *expectations* of each variable being 1 under the distribution would also always give us $R \geq 0$. More formally we consider the following multilinear map from the variable set of $\mathbb{A}$ to positive real numbers.

Definition 5.1. Let $\mathbb{A} = \{P_1 \geq 0 \dots P_m \geq 0\}$ be a given set of axioms on the variables $X = \{x_1 \dots x_n\}$. For all $S \subseteq X$, let ϕ_S be a map from $\{0,1\}^S$ to $\mathbb{R}$, and let E_ϕ be the multilinear map on $\mathbb{F}[X]$ where for any monomial t over the variables S, $E_\phi[t] = \phi_S(t)$.

Note that for a monomial t with variable set S, $E_\phi[t]$ is the value of ϕ_S on the all-ones assignment to S. However since this map is actually multilinear, we can also define $E_\phi[Q_j]$ for a junta Q_j to be ϕ_S on the unique assignment that satisfies Q_j. Assuming ϕ is a distribution over satisfying assignments to $\mathbb{A}$, E_ϕ provides a witness to the fact that there does not exist any SA refutation.

Lemma 5.2 (Distribution). *ϕ is a probability distribution on assignments $\{0,1\}^X$ that satisfy $\mathbb{A}$ iff the following properties hold:*

1. $E_\phi[1] = 1$

2. $E_\phi[P_i Q_j] \geq 0$ *for all* $P_i \in \mathbb{A}, Q_j \in \mathbb{J}$

Note that for an unsatisfiable $\mathbb{A}$, there cannot exist such an operator E. However, when we restrict our attention to distributions only defined over smaller sets of variables, it may be possible to define a *pseudodistribution* over the assignments to those variables such that no axiom is falsified. We can think of pseudodistributions as fooling low levels of the Sherali-Adams hierarchy by having a locally consistent distribution on satisfying assignments that do not actually exist.

Definition 5.3 (Sherali-Adams Pseudodistribution). We refer to ϕ as a *degree-d SA pseudodistribution* for $\mathbb{A}$ iff the following properties hold:

O1. $E_\phi[1] = 1$

O2. E_ϕ is a multilinear map

O3. For any $P_i \in \mathbb{A}$ and $Q_j \in \mathbb{J}$, if $\deg(P_i Q_j) \leq d$ then $E_\phi[P_i Q_j] \geq 0$

Our last job is to ensure that a degree-d pseudodistribution does actually "fool" degree-d Sherali-Adams.

Theorem 5.4. *There does not exist a degree-d Sherali-Adams refutation of $\mathbb{A}$ iff there exists a degree-d pseudodistribution for $\mathbb{A}$.*

Proof. Let $R = \sum_{i,j} c_{ij} P_i Q_j$. Define $x_S = \prod_{X \in S} x$ and for any polynomial P let P_S be the coefficient of x_S in the fully expanded P. Note that if C is a degree-d Sherali-Adams refutation of $\mathbb{A}$, then $R_S = 0$ if $S \neq \emptyset$ and $R_\emptyset = k$ for some negative k. We will have a linear program minimizing the value of k which we can derive, and note that if there exists a refutation of $\mathbb{A}$ then k will be undefined. Taking the dual of this linear program will give us a program that has a trivial optimum of 0 subject to the existence of a degree-d pseudodistribution, and so by duality a pseudodistribution exists iff the smallest constant that degree-d Sherali-Adams can derive is 0.

To make the duality work, let us observe the fact that there exists C such that $R_\emptyset = k$ and $R_S = 0$ for all $S \neq \emptyset$ iff there exists C such that $R_\emptyset = k$ and $R_S \leq 0$ for all $S \neq \emptyset$. The forward direction is obvious, so consider some C where $R_S < 0$ for some S. Consider $Q_j = \prod_{s \in S} x_S$, and let i be such that P_i is the axiom 1. Then setting $c_{ij}+ = R_S$ yields a C where $R_S = 0$ and all other R_S are unchanged (including $S = \emptyset$). Repeating this procedure for all S gives us the original definition. Therefore for now we assume without loss of generality that $R_S \leq 0$ instead of $R_S = 0$.

We now present the primal linear program, whose variables are the entries in C:

$$\begin{array}{rrll}
\text{minimize}_C & \displaystyle\sum_{i,j}(P_iQ_j)_\emptyset c_{ij} & & \text{(recall that } R = \sum_{i,j} c_{ij}P_iQ_j\text{)} \\
\text{subject to} & \displaystyle -\sum_{i,j}(P_iQ_j)_S c_{ij} & \geq 0 & \text{for all } S : 1 \leq |S| \leq d \\
 & c_{ij} & \geq 0 & \text{for all } i, j
\end{array}$$

Taking the dual, because we have one constraint for every set S of size between 1 and d, we let x_S be a variable rather than a product, and obtain the following duall:

$$\begin{array}{rrll}
\text{maximize}_{x_S} & 0 & & \\
\text{subject to} & \displaystyle -\sum_{S:1\leq|S|\leq d}(P_iQ_j)_S x_S & \leq (P_iQ_j)_\emptyset & \text{for all } i, j \\
 & x_S & \geq 0 & \text{for all } S : 1 \leq |S| \leq d
\end{array}$$

Note that if we group the terms in the constraints, we have the constraint that $\sum_{S:|S|\leq d}(P_iQ_j)_S x_S \geq 0$, where $x_\emptyset := 1$. This constraint is clearly satisfiable iff there exists a degree-d pseudodistribution E_ϕ where $E_\phi[S] = x_S$, as $E[1] = x_\emptyset = 1$ and $E[P_iQ_j] = \sum_S (P_iQ_j)_S x_S \geq 0$ whenever $\deg(P_iQ_j)_S \leq d$. Finally as stated above, this dual program has solution 0 if there exists any E satisfying the constraint and is undefined otherwise, while the primal has solution 0 if there is no refutation of $\mathbb{A}$ and is undefined otherwise. □

5.1.2 Lower bounds for τ_S

Our job now reduces to finding a pseudodistribution for τ_S in the same way as a wide clause lemma or linear functional from Chapter 4. We prove a much simpler but necessary step to obtaining a pseudodistribution for τ_S. Namely we prove the $(m, \log m/4)$-universal property of A itself has a degree $O(k \log m)$ pseudodistribution when the rows are described by an error-correcting code as in τ_S. The tautology τ will consist of x_i variables for every $i \in [k]$, where $k \leq \frac{\log m}{4}$ and x_i is of length $6 \log m$, and fix some $2 \log m$-surjective function f_x. We view an assignment of $x_1 \ldots x_k$ to $\alpha_1 \ldots \alpha_k$ as a $k \times m$ matrix M where row i is the adjacency vector of the vertex $f_x(\alpha_i)$. τ claims that for any $\alpha_1 \ldots \alpha_k$ there is no column $j \in [m]$ of M which is all 1s. To formalize this we have a clause P_j in τ for every column j as follows:

$$\sum_{i\in[k]} \sum_{\alpha_i:(f_x(\alpha_i))|_j=0} x_i^{\alpha_i} - 1 \geq 0$$

We fix our universal set A to be the *Paley graph*, which exhibits a stronger notion of universality: for any set S of size at most $\frac{\log m}{4}$ and any partition of S into A, B, the number of rows in A which are 1 in A and 0 in B is $\frac{m}{2^{|S|}}$.

This clearly implies the universal property, but also gives us much more structure to exploit. This structure will turn out to be necessary in order to make the marginals definition work.

Lemma 5.5. *There exists a degree $\frac{k}{2}\log m$ pseudodistribution for τ.*

Proof. For any monomial t of degree at most $\frac{k}{2}\log m$, we simply set $\tilde{E}[t] = 2^{-|t|}$. First note that this satisfies linearity of $\tilde{E}$, and $\phi_\emptyset(1) = 1$. Finally we need to verify that for all $P_j \in \tau$, $\tilde{E}[QP_j]$ is nonnegative for any Q such that $|Q| \le \frac{k}{2}\log m$. Recall that P_j is of the form

$$\sum_{i\in[k]} \sum_{\alpha_i:(f_x(\alpha_i))|_j=0} x_i^{\alpha_i} - 1$$

As a sanity check let us first verify that $\tilde{E}[P_j] \ge 0$. The number of terms in this sum is $k\frac{m^6}{2}$, using the surjectivity of f_x and the fact that half of the rows in A satisfy $f_x(\alpha_i)_j = 0$ (here we crucially use our choice of A). Each term has degree $6\log m$, and since all variables are assigned $\frac{1}{2}$ by the pseudodistribution the pseudoexpectation value for each term is $2^{-6\log m}$. Therefore $\tilde{E}[P_j] \approx \frac{km^6}{2}2^{-6\log m} - 1 = \frac{k}{2} - 1 \gg 0$.

Now consider multiplying P_j by some variable z. For every term in P_j which has a $(1-z)$ in it, the term will be set to 0 by the pseudodistribution as any assignment of z to 0 or 1 will force either z or $1-z$ to be 0. Let $i(J)$ be the number of x_i variables in J and let I_0 be the set of $i \in [k]$ such that $i(J) \le \log m$. Using the surjectivity of f_x, for any $i \in I_0$ we get that the number of terms in the expression $J(\sum_{\alpha_i:(f_x(\alpha_i))|_j=0} x_i^{\alpha_i})$ is $\frac{m^6/2}{2^{i(J)}}$, as each variable in J can only force about half the terms in the sum to 0. The degree of each surviving term $Jx_i^{\alpha'_i}$ is $|J| + 6\log m - i(J)$, because while every variable $z \in x_i$ in J kills all terms with $(1-z)$, the rest of the terms already have z in them, and when we apply $\tilde{E}$ we multilinearize. Finally we observe that $|I_0| \ge \frac{k}{2}$, and so

$$\begin{aligned}
\tilde{E}[JP_j] &= \tilde{E}[J(\sum_{i\in[k]} \sum_{\alpha_i:(f_x(\alpha_i))|_j=0} x_i^{\alpha_i} - 1)] \\
&= \sum_{i\in[k]} \sum_{\alpha_i:(f_x(\alpha_i))|_j=0} \tilde{E}[Jx_i^{\alpha_i}] - \tilde{E}[J] \\
&\ge \sum_{i\in I_0} \sum_{\alpha'_i:(f_x|_J(\alpha'_i))|_j=0} \tilde{E}[Jx_i^{\alpha'_i}] - \tilde{E}[J] \\
&\ge \frac{k}{2}\frac{m^6/2}{2^{i(J)}}2^{-(|J|+6\log m - i(J))} - 2^{-|J|} \\
&\ge \frac{k}{4}\frac{m^6}{2^{i(J)}}\frac{2^{i(J)}}{m^6}2^{-|J|} - 2^{-|J|} \\
&\ge 2^{-|J|}(\frac{k}{4} - 1) \ge 0
\end{aligned}$$

□

Another sanity check we can do is verify that there does not exist a pseudodistribution of degree $6k\log m$, aka over all the variables in the tautology (recall that τ is unsatisfiable). This follows from the fact that if $|J| = 6k\log m$, then I_0 could potentially be empty, at which point the only term would be $-\tilde{E}[J]$ and we would necessarily violate one of our conditions on $\tilde{E}$.

While this lower bound was fairly straightforward, to extend this to a pseudodistribution for τ_S may take a lot more work. First, τ was restricted to the assignments for a fixed set of k rows, but in τ_S we need to prove a pseudodistribution for any set of k rows from M_x, which has n rows in total. This was never an issue before, as we were proving the lower bounds in a query-type manner where the refutation's best option was to focus on the hitting set of size k, but in Sherali-Adams the proof may have a number of different sets of rows it focuses on, and we need to ensure the marginals for any such choice. Furthermore we haven't taken the columns into account, and while choosing the universal set to be the Paley graph may give us nice regularity properties in the rows the hitting set instance $\mathcal{S}$ certainly may be too erratic to rely on.

When the axioms of $\mathbb{A}$ are defined by products as in Nullsatz and PC, there is an equivalent and easier definition to use. The reader should be aware, however, that if the axioms are presented in alternate forms, particularly any form involving large sums of low degree monomials, such as in τ above, this definition may not hold.

Definition 5.6 (Sherali-Adams Pseudodistribution (marginals)). ϕ is a degree-d pseudodistribution for $\mathbb{A}$ iff the following properties hold:

M1. $\phi_\emptyset[1] = 1$

M2. For any $S \subseteq T \subseteq X$ where $|T| \leq d$ and any $\mu \in \{0,1\}^S$, $\phi_S(\mu) = \sum_{\eta \in \{0,1\}^T, \eta|_S = \mu} \phi_T(\eta)$

M3. For any $P_i \in \mathbb{A}$ over the variable set S and any $T \supseteq S$ such that $|T| \leq d$, every assignment in the support of ϕ_T satisfies $P_i \geq 0$.

Proof. First we show that Definition 5.3 implies Definition 5.6. Clearly M1 is equivalent to O1. To prove M2, consider sets $S \subseteq T$ where $|T| \leq d$. For $\mu \in \{0,1\}^S$, let Q_μ be the unique junta set to 1 by μ, and likewise define $\eta \in \{0,1\}^T$ and Q_η. Letting $P_i = 1$ and summing up we get

$$\sum_{\eta \in \{0,1\}^T, \eta|_S = \mu} \phi_T(\eta) = \sum_{\eta \in \{0,1\}^T, \eta|_S = \mu} E_\phi[P_i Q_\eta] = E_\phi[P_i Q_\mu] \prod_{x_j \in T - S} E_\phi[x_j + (1 - x_j)] = P_i Q_\mu = \phi_S(\mu)$$

where the last equality uses O1 and O2. M3 follows from O3 by letting Q_j is the assignment to the variables of $T - S$.

Now we show that Definition 5.6 implies Definition 5.3. Set $E_\phi[t] = \phi_S(t)$ for all monomials t with variable set S and extend it multilinearly, which is possible because of M2. Again O1 is equivalent to M1 and O2 is by definition. To prove O3 it is important that each $P_i Q_j$ is a single junta, and so we consider any assignment in the support of ϕ_T where P_i has variable set S and Q_j has variable set $T - S$. Applying this assignment to P_i satisfies $P_i \geq 0$ by M3, and applying it to Q_j satisfies $Q_j \geq 0$ because Q_j is a junta. Therefore since all assignments in the support of ϕ_T satisfy $P_i Q_j \geq 0$, it follows that $E_\phi[P_i Q_j] \geq 0$. □

In other words, to get a degree-d lower bound on SA for $\tau_{\mathcal{S}}$ it is necessary and sufficient to define, for all variable sets S of size at most d, a probability distribution on assignments to S such that no axiom is violated on any assignment with nonzero probability, with the additional condition that the distribution value on any assignment μ to S is the sum of the marginal distributions on assignments to any variable set $T \supseteq S$ consistent with μ (as long as $|T| \leq d$). Following the conventions of the other lower bounds we say $\tau_{\mathcal{S}}$ has an (a,b)-pseudodistribution if ϕ is defined for all variable sets D such that $I_0(D) \leq a$ and $J_0(D) \leq b$.

5.2 Cutting Planes (CP)

5.2.1 The CP proof system

The *Cutting Planes*(CP) refutation system [15, 20] is a dynamic semi-algebraic system where every line in the proof is of the form $a_1x_1 + \ldots + a_nx_n \geq c$ for constants $a_1, \ldots, a_n, c$. Like SA we are given the input clauses in the form of inequalities, but we require that they be of degree 1 with the constant terms all on the right hand side of the inequality. Instead of converting the clause $C = x_1 \vee \overline{x_2} \vee x_3$ to the dual inequalities $\pm(1-x_1)(x_2)(1-x_3) \geq 0$ as in the conversion from Nullsatz to SA, we instead convert C to the single inequality $x_1 + (1 - x_2) + x_3 \geq 1$, which we then homogenize to get $x_1 - x_2 + x_3 \geq 0$.

In contrast to SA, a CP proof is dynamic, and as with Res there is a tree-like variant TreeCP where the underlying graph is restricted to be a tree instead of a DAG. The final line in a CP proof is the inequality "$0 \geq 1$". Let $L_1 =$ "$a_1x_1 + \ldots + a_nx_n \geq c$" and $L_2 =$ "$b_1x_1 + \ldots + b_nx_n \geq d$" be two lines in a CP proof. Then we can derive a new line L via the following rules:

- *addition*: $L = L_1 + L_2 =$ "$(a_1 + b_1)x_1 + \ldots + (a_n + b_n)x_n \geq c + d$"
- *multiplication*: $L = kL_1 =$ "$ka_1x_1 + \ldots + ka_nx_n \geq kc$"
- *integer division*: $L = \frac{L_1}{k} =$ "$\frac{a_1}{k}x_1 + \ldots + \frac{a_n}{k}x_n \geq \lceil\frac{c}{k}\rceil$" *if and only if* $k \mid a_1, \ldots, a_n$

It is very important that we can only apply division when the left hand side evenly divides every coefficient. This ensures that all coefficients are integers, but also allows us to take the ceiling of the right hand side after dividing, which is one of CP's main tools for shrinking the feasible region of the input inequalities.

5.2.2 Lifting theorems

It is known that TreeCP p-simulates TreeRes and CP p-simulates Res. Thus the upper bound in Lemma 3.6 holds for both systems, and the lower bound in Lemma 3.7 remains to be proven. However the techniques for getting CP seem to require a lower bound on τ_S that is far too difficult, if not impossible. Thus we focus on the main technique for TreeCP lower bounds, known as *lifting theorems*, which take a weak lower bound for a tautology τ, namely query (decision tree) lower bounds, and convert τ into τ' such that the lower bounds "lift" to a stronger model such as communication complexity.

Query-to-communication lifting theorems have been proven in many recent papers, and used to resolve several open problems in game theory, proof complexity, circuit complexity as well as to understand the limitations of linear and semidefinite programming via extension complexity. The basic idea of a query-to-communication lifting theorem is as follows. Let $\mathcal{C}$ be a complexity class, such as P, NP, BPP. Let $f : \{0,1\}^n \to R$ be any Boolean function or relation with range R, and let $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, where $\mathcal{X}, \mathcal{Y} = \{0,1\}^{c(n)}$. Their composition $f \circ g^n : \mathcal{X}^n \times \mathcal{Y}^n \to R$ is defined by $(f \circ g^n)(x, y) = f(g(x_1, y_1), \ldots, g(x_n, y_n))$. We view $f \circ g^n$ as a communication complexity problem where Alice holds $x \in \mathcal{X}^n$ and Bob holds $y \in \mathcal{Y}^n$. Let $\mathcal{C}^{\mathsf{dt}}(f)$ denote the query complexity of f under the model $\mathcal{C}$, and similarly let $\mathcal{C}^{\mathsf{cc}}(f \circ g^n)$ denote the corresponding communication complexity of $f \circ g^n$ under the model $\mathcal{C}$. Namely if $\mathcal{C} = \mathsf{P}$ then $\mathsf{P}^{\mathsf{dt}}(f)$ is the deterministic query complexity of f and $\mathsf{P}^{\mathsf{cc}}(f \circ g^n)$ is the deterministic communication complexity of $f \circ g^n$. A general lifting theorem for $\mathcal{C}$ with gadget size $c(n)$ states that for any f, and a specific good gadget $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, that the optimal communication protocol for the composed function is that obtained by simulating the corresponding optimal decision tree for f: $\mathcal{C}^{\mathsf{dt}}(f) = \Theta(c(n) \times \mathcal{C}^{\mathsf{cc}}(f \circ g^n))$.

Lifting theorems have been proven for many complexity classes, including P, NP and BPP (see eg [13, 22, 24, 32, 35]). However, all of the general lifting theorems that have been proven so far require gadgets whose size depends logarithmically on n, and it is a well-known open problem to prove (or disprove) the following conjecture, asserting the existence of a constant-sized gadget lifting theorem for P. For special cases of f, constant-sized lifting theorems have been shown [21].

Conjecture 5.7. *(Constant-Gadget Lifting Conjecture for* P*)*

For any function or relation $f(z_1, \dots, z_n)$ *there exists a constant* $c > 1$ *and a gadget* $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *where* $\mathcal{X}, \mathcal{Y} = \{0,1\}^c$ *such that* $\mathsf{P}^{\mathsf{dt}}(f) = \Theta(\mathsf{P}^{\mathsf{cc}}(f \circ g^n))$.

In real communication, denoted $\mathsf{P}^{\mathsf{rcc}}$, instead of Alice and Bob sending bits to one another, in each round they send a real value to a referee who tells them both who had the larger value, where the cost of a query to the referee is just one. In [24], the lifting theorem due to Raz and McKenzie [32] was generalized to show that deterministic query complexity can actually be lifted to real communication complexity. Similarly, the recent BPP lifting theorem [22] can also be extended to the *real* communication complexity setting. Thus it is natural to make the following conjecture, asserting a constant-sized gadget lifting theorem from deterministic query complexity to real communication complexity.

Conjecture 5.8. *(Constant-Gadget Real Lifting Conjecture for* P*)*

For any function or relation $f(z_1, \dots, z_n)$ *there exists a constant* $c > 1$ *and a gadget* $g : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ *where* $\mathcal{X}, \mathcal{Y} = \{0,1\}^c$ *such that* $\mathsf{P}^{\mathsf{dt}}(f) = \Theta(\mathsf{P}^{\mathsf{rcc}}(f \circ g^n))$.

Assuming the above conjecture, the GapETH hardness of automatizability for tree-like Cutting Planes follows from the following Lemma, which is a restatement of our Main Lemma 3.1 for the case of TreeCP.

Lemma 5.9. *Let* $\mathcal{Q} = \mathsf{TreeCP}$ *and suppose Conjecture 5.8 holds. Let* n *be sufficiently large and let* $(\mathcal{S}, k, k^2)$ *be an instance of the gap hitting set problem. Then there exists a tautology* $\tau_{\mathcal{S}}^{cp}$ *which can be computed in time* $n^{O(1)}$ *such that the following two properties hold:*

(1) if $\gamma(\mathcal{S}) \leq k$ *then* $S_{\mathcal{Q}}(\tau_{\mathcal{S}}^{cp}) \leq n^{O(1)}$*;*

(2) if $\gamma(\mathcal{S}) > k^2$ *then* $S_{\mathcal{Q}}(\tau_{\mathcal{S}}^{cp}) \geq n^{\Omega(k)}$.

Proof. Let $\tau_{\mathcal{S}}$ be the tautology defined in Section 3.1, and let $n' := n \log m + m \log n$ be the number of variables in $\tau_{\mathcal{S}}$. We obtain $\tau_{\mathcal{S}}^{cp} \circ g^{n'}$ by replacing every variable $z \in \tau_{\mathcal{S}}$ by $g(x_z, y_z)$, $x_z, y_z \in \{0,1\}^c$, where g is the gadget and c is the constant given by Conjecture 5.8. We want the composed formula to also be in CNF form, so we convert it in the obvious way: convert each composed clause into an equivalent CNF formula, and then the conjunction of these CNFs will be the new CNF formula $\tau_{\mathcal{S}}^{cp}$ over the x_z, y_z variables. Since each original clause of size $t = \log m + \log n$ becomes a conjunction of clauses, each involving $2ct$ variables, the width of $\tau_{\mathcal{S}}^{cp}$ is $2ct = O(t)$, and the size increases by a factor of $2^{2ct} = n^{O(1)}$.

First, we observe the upper bound. We can solve the search problem via the same decision tree as in Section 3.3, but now whenever we want to query a variable $z \in \tau_{\mathcal{S}}$ we instead query the vectors $x_z, y_z \in \tau_{\mathcal{S}}^{cp}$. Thus we incur a factor of $2c = O(1)$ in the decision tree height, which translates to an upper bound of $n^{O(1)}$ for $\mathsf{TreeRes}$. Since TreeCP p-simulates $\mathsf{TreeRes}$ by size this gives the upper bound.

To obtain the lower bound we note that Section 3.3 gives a lower bound on the decision tree height of $\tau_{\mathcal{S}}$. Thus the decision tree complexity of $\tau_{\mathcal{S}}$, $\mathsf{P}^{\mathsf{dt}}(\tau_{\mathcal{S}})$, is $\Omega(k \log n)$, which by our assumption of g implies that the real communication complexity of the search problem associated with $\tau_{\mathcal{S}}^{cp}$, $\mathsf{P}^{\mathsf{rcc}}(\tau_{\mathcal{S}}^{cp})$, equals $\mathsf{P}^{\mathsf{rcc}}(\tau_{\mathcal{S}} \circ g^{n'}) = \Theta(\mathsf{P}^{\mathsf{dt}}(\tau_{\mathcal{S}})) = \Omega(k \log n)$.

Assume for contradiction that $\tau_{\mathcal{S}}^{cp}$ has a TreeCP refutation π of size $s = n^{o(k)}$. Let T be the underlying tree of π with vertices labeled with the corresponding lines of π. First we find a line $L \in \pi$ such that the subtree rooted at L contains between $\frac{s}{3}$ and $\frac{2s}{3}$ lines. Let T_1 be the subtree rooted at L and let T_2 be T with T_1 removed. Our protocol first evaluates line L in one round of real communication as follows: let $L =$ "$a_1x_1 + \ldots + a_ux_u + b_1y_1 + \ldots + b_vy_v \geq t$" where Alice has the x variables and Bob has the y variables. Alice sends the referee $a_1x_1 + \ldots + a_ux_u$ and Bob sends the referee $t - (b_1y_1 + \ldots + b_vy_v)$. If the referee returns "Alice $<$ Bob" then the line is falsified, and we repeat this procedure on T_1. Otherwise the line is satisfied and we repeat this procedure on T_2. We recurse until we reach a leaf, which must be labeled with a falsified clause since at every step the root of the current tree is labeled with a falsified line. Recursively this will find a falsified clause in $\log s = o(k \log n)$ rounds, which is a contradiction because $\mathsf{P}^{\mathsf{rcc}}(\tau_{\mathcal{S}}^{cp}) = \Omega(k \log n)$. □

We note that while a constant-sized gadget lifting theorem gives a lower bound to match Theorem 1.1 for TreeCP, we could get away with a somewhat larger gadget size. In particular, a lifting theorem with gadget size $\tilde{o}(\log \log n)$ will still be enough to refute polynomial automatizability for TreeCP under GapETH. Secondly, we note that we only need a constant-sized gadget lifting theorem for our specific search problem to get the nonautomatizability result, as opposed to a more general lifting theorem that would have to work for tautologies with much decision tree complexity, such as $\Omega(n)$.

Chapter 6

Conclusions

The automatizability problem is still far from closed, and while this work makes significant strides towards optimality for TreeRes and Nullsatz there are many further directions to take from here on out:

- as we saw in Chapter 5, the next systems worth considering for the program of [2] are SA and CP, which should pose a decent challenge for those looking to test their knowledge of this technique. SA will be a purely proof complexity problem, of finding a pseudodistribution for τ_S or possibly a variant τ'_S more amiable to proving lower bounds. CP on the other hand seems to be in the realm of lifting theorems, which has more to do with communication complexity and entropy arguments. Either one would be a significant breakthrough; SA automatizability lower bounds may help us close in on the jewel of SoS, while the lifting theorem needed for CP automatizability lower bounds is perhaps an even larger discovery in and of itself.

- to improve the results of this work, it is worth observing that the constructions in [12, 17] are not known to be optimal, and any hardness results against approximating the gap hitting set problem in time $n^{o(k)}$ for a larger value of k immediately gives a lower bound of $n^{o(k)}$ against automatizability. By the crucial fact that $k \leq \frac{\log m}{4} = \frac{\log n}{4k}$, this technique can't be strengthened past the $k = o(\log^{1/2} n)$ threshold, but with the TreeRes and Nullsatz upper bounds at $k = O(\log n)$ (and perhaps not-so-coincidentally the ability to approximate minimum hitting set size to within an $O(\log n)$ factor in polynomial time) this would be close to optimal.

- for those interested in strong automatizability lower bounds who are not attached to this particular program, to avoid this $O(\log n)$ barrier we will need to come up with a transformation such that the upper bound in the "yes" instances fails for TreeRes and Nullsatz. There are a number of examples of tautologies for which TreeRes and Res have an exponential gap, but one might even consider focusing only on much stronger systems such as SoS and thus being able to choose a much more subtle type of trapdoor. One promising direction is the recent versions of Feige's hypothesis that have evolved out of the work on SoS lower bounds. However ultimately it would be extremely practical to resolve the automatizability of Res, which is tied to SAT solvers. The only known automatizability upper bound is $n^{O(\sqrt{n \log S})}$, and our lower bound is $n^{\tilde{\Omega}(\log \log S)}$, so the door is wide open for tightening either bound.

We started this work in order to understand the technique of [2], as the first paper to prove automatizability lower bounds for a "practical" proof system. Since then there have been many connections found between proof complexity and other areas of theoretical computer science, and so the problem of automatizability has only grown in importance since then. We feel that it provides another method of attack for problems in learning theory, approximation, SAT solving, and possibly more, and while also being one of the most fundamental questions to a fundamental field in complexity theory.

Bibliography

[1] Michael Alekhnovich, Samuel R. Buss, Shlomo Moran, and Toniann Pitassi. Minimum propositional proof length is np-hard to linearly approximate. *J. Symb. Log.*, 66(1):171–191, 2001.

[2] Michael Alekhnovich and Alexander A. Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM J. Comput.*, 38(4):1347–1363, 2008.

[3] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple construction of almost k-wise independent random variables. *Random Struct. Algorithms*, 3(3):289–304, 1992.

[4] Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *J. Comput. Syst. Sci.*, 74(3):323–334, 2008.

[5] Albert Atserias, Massimo Lauria, and Jakob Nordström. Narrow proofs may be maximally long. *ACM Trans. Comput. Log.*, 17(3):19:1–19:30, 2016.

[6] Boaz Barak, Jonathan A. Kelner, and David Steurer. Dictionary learning and tensor decomposition via the sum-of-squares method. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 143–151, 2015.

[7] Boaz Barak and Ankur Moitra. Noisy tensor completion via the sum-of-squares hierarchy. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 417–445, 2016.

[8] Paul Beame, Russell Impagliazzo, Jan Krajícek, Toniann Pitassi, and Pavel Pudlák. Lower bound on hilbert's nullstellensatz and propositional proofs. In *35th Annual Symposium on Foundations of Computer Science, Santa Fe, New Mexico, USA, 20-22 November 1994*, pages 794–806, 1994.

[9] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow - resolution made simple. *J. ACM*, 48(2):149–169, 2001.

[10] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel, and Toniann Pitassi. Non-automatizability of bounded-depth frege proofs. *Computational Complexity*, 13(1-2):47–68, 2004.

[11] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for frege systems. *SIAM J. Comput.*, 29(6):1939–1967, 2000.

[12] Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From gap-eth to fpt-inapproximability: Clique, dominating set, and more. *CoRR*, abs/1708.04218, 2017.

[13] Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation theorems via pseudorandom properties. *CoRR*, abs/1704.06807, 2017.

[14] Yijia Chen and Bingkai Lin. The constant inapproximability of the parameterized dominating set problem. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 505–514, 2016.

[15] V. Chvtal. Cutting planes in combinatorics. *European Journal of Combinatorics*, 6(3):217 – 226, 1985.

[16] Matthew Clegg, Jeff Edmonds, and Russell Impagliazzo. Using the groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, pages 174–183, 1996.

[17] Karthik C.S., Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. *Fix this reference*, Fix this reference, 2017.

[18] Irit Dinur. Mildly exponential reduction from gap 3sat to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:128, 2016.

[19] Nicola Galesi and Massimo Lauria. On the automatizability of polynomial calculus. *Theory Comput. Syst.*, 47(2):491–506, 2010.

[20] Ralph E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64(5):275–278, 09 1958.

[21] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 847–856, 2014.

[22] Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for BPP. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA*, pages 132–143. IEEE Computer Society, 2017.

[23] Samuel B. Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sum-of-squares proofs. *CoRR*, abs/1507.03269, 2015.

[24] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 233–248, 2012.

[25] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[26] Jan Krajícek and Pavel Pudlák. Some consequences of cryptographical conjectures for $s^1{}_2$ and EF. *Inf. Comput.*, 140(1):82–94, 1998.

[27] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 438–446, 2016.

[28] Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense csps. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 78:1–78:15, 2017.

[29] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[30] Aaron Potechin and David Steurer. Exact tensor completion with sum-of-squares. *CoRR*, abs/1702.06237, 2017.

[31] Pavel Pudlák. Proofs as games. *The American Mathematical Monthly*, 107(6):541–550, 2000.

[32] Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999.

[33] 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.

[34] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems. *Discrete Appl. Math.*, 52(1):83–106, July 1994.

[35] Xiaodi Wu, Penghui Yao, and Henry Yuen. Raz-mckenzie simulation with the inner product gadget. *Electronic Colloquium on Computational Complexity (ECCC)*, 2017.

Chapter A

Hardness assumptions

The statement of Lemma 2.2 is not clear from the results in [12] or [17], and takes a small bit of inspection and parameter fixing as both results were proved in the setting of parameterized complexity. We provide a guide to obtaining Lemma 2.2 from the papers by tracking the optimal values of all parameters. Note that while we provide references to the theorems used, we defer the formal definitions and proofs to their respective papers.

A.1 Hardness of hitting set under GapETH

In this section we prove Lemma 2.2 in the case of GapETH. The proof follows the reduction in [12] from the *label cover* problem to the gap hitting set problem, but the definitions of the problem and details of the reduction are omitted as we only need to focus on how the parameters change at each step. We restate the lemma now for convenience.

Lemma A.1 (Lemma 2.2 under GapETH)**.** *Assuming* GapETH*, for sufficiently large n and $k = \tilde{O}(\log\log n)$ no algorithm can solve the gap hitting set problem $(\mathcal{S}, k, k^2)$ in time $n^{o(k)}$.*

Proof. Let $K(n)$ be a function such that $K^{K^{O(K)}} = 2^{O(n/K)}$, and note that $K \in \omega(\frac{\log n}{\log\log n})$. Following the proof of Theorems 4.3 and 4.4 of [12] consider an arbitrary label cover instance $\Gamma = (G = (U, V, E), \Sigma_U, \Sigma_V, \Pi)$, where:

- $|U| = n$
- $|V| = O(n)$
- $|\Sigma_U| = O(1)$
- $|\Sigma_V| = O(1)$
- $|\Pi| = O(|\Sigma_U||\Sigma_V|) = O(1)$

Assuming GapETH it is known that no $2^{o(|U|)}$-time algorithm distinguishes between a *max covering* of size $|U|$ and a max covering of size less than $(1-\epsilon)|U|$ for any sufficiently large (constant) $\epsilon > 0$. We can transform this into a new label cover instance $\Gamma' = (G' = (U', V', E'), \Sigma_{U'}, \Sigma_{V'}, \Pi')$ where

- $|U'| = \binom{|U|}{(K \ln K)/\epsilon} = n^{O(K \ln K)}$

- $|V'| = K$
- $|\Sigma_{U'}| = |\Sigma_U|^{(K \ln K)/\epsilon} = K^{O(K)}$
- $|\Sigma_{V'}| = |\Sigma_V|^{n/K} = 2^{O(n/K)}$
- $|\Pi'| = O(|\Sigma_{U'}||\Sigma_{V'}|) = 2^{O(n/K)}$
- Γ has a max covering of size $|U|$ iff Γ' has a max covering of size $|U'|$
- Γ has no max covering of size $(1-\epsilon)|U|$ iff Γ' has no max covering of size $\frac{1}{K^K}|U'|$

For our choice of K, it holds that $|\Gamma'|$ is dominated by $|\Sigma_{V'}| = 2^{O(n/K)}$. So under GapETH it is impossible to distinguish between Γ' having a max covering of size $|U'|$ and not having a max covering of size $\frac{1}{K^K}|U'|$ in time $2^{o(n)} = |\Gamma'|^{o(K)}$. Theorem 4.4 of [12] shows that distinguishing these two cases on Γ' implies distinguishing *min-right coverings* of size at most $|V'| = K$ and those of size greater than K^2 for the same label cover instance $|\Gamma'|$. Using this fact and following Theorem 5.4 of the same paper, we transform Γ' in time $\mathsf{poly}(|\Gamma'|)$ into a hitting set instance $\mathcal{H} = (\mathcal{U}, \mathcal{S})$, where

- $|\mathcal{U}| = |U'||V'|^{|\Sigma_{U'}|} = n^{O(K \ln K)} K^{K^{O(K)}} = K^{K^{O(K)}}$
- $|\mathcal{S}| = |V||\Sigma_V| = K2^{O(n/K)} = 2^{O(n/K)}$
- $\gamma(\mathcal{S})$ is equivalent to the min-right covering number of $|\Gamma'|$

Define $N = |\mathcal{H}|$, and because $K^{K^{O(K)}} = 2^{O(n/K)}$ we get $N = |\mathcal{U}||\mathcal{S}| = 2^{O(n/K)}$. We now define $k(n)$ to be such that $k(N) = K(n)$, which can be shown to be $\tilde{O}(\log\log n)$ (suppressing $\log\log\log n$ factors). Therefore under GapETH there doesn't exist any algorithm that can distinguish between $\gamma(\mathcal{H}) \leq k(N)$ and $\gamma(\mathcal{H}) > k^2(N)$ in time $N^{o(k(N))}$ for hitting set instances $\mathcal{H}$ of size N. □

A.2 Hardness of hitting set under ETH

In this section we prove Lemma 2.2 in the case of ETH. The proof follows the reduction in [17], but once again we omit the definitions and proofs of their reduction and focus on how the parameters are restricted therein.

Lemma A.2 (Lemma 2.2 under ETH). *Assuming* ETH*, for sufficiently large n and $k = O(\log^{1/5}\log n)$ no algorithm can solve the gap hitting set problem $(\mathcal{S}, k, k^2)$ in time $n^{o(k)}$.*

Proof. By Corollary 7.2 of [17] there is a $(0, O((\log m)^2), (\log m)/2k, (1/m)^{1/O(kt)})$ efficient protocol for the $\text{MULTEQ}_{m,k,t}$ problem. By Corollary 5.2 if there exists a (w, r, l, s)-efficient protocol for $\text{MULTEQ}_{m,k,t}$ such that $w + r + lk = o(m)$, and $l < (\log m)/\beta \cdot k$ for constant $\beta > 1$ then no algorithm can distinguish, for a hitting set instance $\mathcal{S}$ over the universe $[O(N)]$, whether $\gamma(\mathcal{S}) \leq k$ or $\gamma(\mathcal{S}) \geq (1/s)^{1/k} \cdot k$ in time $N^{o(k)}$. Putting these two facts together we get that no algorithm can distinguish whether $\gamma(\mathcal{S}) \leq k$ or $\gamma(\mathcal{S}) \geq m^{1/k^2 t} \cdot k$ in time $N^{o(k)}$. The proof of Corollary 5.2 relies on Definition 4.11, which defines $t = k + \binom{k}{2} + \binom{k}{3}$ and $m = t(1 + k \log N)$. Therefore we get $t = O(k^3)$ and $m = O(k^4 \log N)$, and so we get that no algorithm can distinguish whether $\gamma(\mathcal{S}) \leq k$ or $\gamma(\mathcal{S}) \geq (k^4 \log N)^{1/k^5} \cdot k$. Setting $(k^4 \log N)^{1/k^5} \cdot k = k^2$ gives us $k = O(\log\log N)^{1/5}$. □