Semialgebraic Proofs and Efficient Algorithm Design

Other titles in Foundations and Trends® in Theoretical Computer Science

Higher-order Fourier Analysis and Applications
Hamed Hatami, Pooya Hatami and Shachar Lovett
ISBN: 978-1-68083-592-2

On Doubly-Efficient Interactive Proof Systems
Oded Goldreich
ISBN: 978-1-68083-424-6

Coding for Interactive Communication: A Survey
Ran Gelles
ISBN: 978-1-68083-346-1

Semialgebraic Proofs and Efficient Algorithm Design

Noah Fleming
University of Toronto
noahfleming@cs.toronto.edu

Pravesh Kothari
Princeton University
kothari@cs.princeton.edu

Toniann Pitassi
University of Toronto & IAS
toni@cs.toronto.edu

now
the essence of knowledge
Boston — Delft

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
United States
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is

N. Fleming, P. Kothari and T. Pitassi. *Semialgebraic Proofs and Efficient Algorithm Design*. Foundations and Trends® in Theoretical Computer Science, vol. 14, no. 1–2, pp. 1–221, 2019.

ISBN: 978-1-68083-637-0

Foundations and Trends® in Theoretical Computer Science
Volume 14, Issue 1–2, 2019

Editorial Board

Editorial Scope

Topics

Foundations and Trends® in Theoretical Computer Science publishes survey and tutorial articles in the following topics:

- Algorithmic game theory
- Computational algebra
- Computational aspects of combinatorics and graph theory
- Computational aspects of communication
- Computational biology
- Computational complexity
- Computational geometry
- Computational learning
- Computational Models and Complexity
- Computational Number Theory
- Cryptography and information security
- Data structures
- Database theory
- Design and analysis of algorithms
- Distributed computing
- Information retrieval
- Operations Research
- Parallel algorithms
- Quantum Computation
- Randomness in Computation

Information for Librarians

Foundations and Trends® in Theoretical Computer Science, 2019, Volume 14, 4 issues. ISSN paper version 1551-305X. ISSN online version 1551-3068. Also available as a combined paper and online subscription.

Contents

Semialgebraic Proofs and Efficient Algorithm Design

Noah Fleming[1], Pravesh Kothari[2] and Toniann Pitassi[3]*

[1] *University of Toronto; noahfleming@cs.toronto.edu*
[2] *Princeton University; kothari@cs.princeton.edu*
[3] *University of Toronto & IAS; toni@cs.toronto.edu*

ABSTRACT

Over the last twenty years, an exciting interplay has emerged between proof systems and algorithms. Some natural families of algorithms can be viewed as a generic translation from a proof that a solution exists into an algorithm for finding the solution itself. This connection has perhaps been the most consequential in the context of semi-algebraic proof systems and basic primitives in algorithm design such as linear and semidefinite programming. The proof system perspective, in this context, has provided fundamentally new tools for both algorithm design and analysis. These news tools have helped in both designing better algorithms for well-studied problems and proving tight lower bounds on such techniques.

This monograph is aimed at expositing this interplay between proof systems and efficient algorithm design and surveying the state-of-the-art for two of the most important semi-algebraic proof systems: Sherali-Adams and Sum-of-Squares.

We rigorously develop and survey the state-of-the-art for Sherali-Adams and Sum-of-Squares both as proof systems,

*Research supported by NSERC.

Noah Fleming, Pravesh Kothari and Toniann Pitassi (2019), "Semialgebraic Proofs and Efficient Algorithm Design", Foundations and Trends® in Theoretical Computer Science: Vol. 14, No. 1–2, pp 1–221. DOI: 10.1561/0400000086.

as well as a general family of optimization algorithms, stressing that these perspectives are formal duals to one-another. Our treatment relies on interpreting the outputs of the Sum-of-Squares and Sherali-Adams algorithms as generalized expectation functions — a viewpoint that has been essential in obtaining both algorithmic results and lower bounds. The emphasis is on illustrating the main ideas by presenting a small fraction of representative results with detailed intuition and commentary. The monograph is self-contained and includes a review of the necessary mathematical background including basic theory of linear and semi-definite programming.

1

Introduction

Proof complexity is the study of what can be proved *efficiently*[1] in a given formal proof system. Algorithm analysis is the quest for efficient and accurate algorithms for optimization problems, that can be rigorously analyzed. Over the last twenty years, there has been an exciting interplay between proof complexity and algorithms which in a nutshell studies the proof complexity of algorithm correctness/analysis. The main focus of this monograph is on algebraic and semi-algebraic proof systems, and the story of how they became closely connected to approximation algorithms. Indeed, we will argue that proof complexity has emerged as the study of systematic techniques to obtain provably correct algorithms.

There are two high level themes underlying this connection. *The first theme is that proof system lower bounds imply lower bounds for a broad family of related algorithms.* A proof system, in a specific formal sense, corresponds to a family of efficient, provably correct algorithms. Thus, lower bounds in specific proof systems (showing hardness of proving

[1]The emphasis on efficiency, as opposed to existence, is what distinguishes proof complexity from classical proof theory, and also what links proof complexity with complexity theory and algorithms.

well-definedness or other key properties of the function) rules out large classes of algorithms for solving NP-hard optimization problems.

One of the earliest appearances of this theme was in the work of Chvátal [46], which proved almost exponential lower bounds against a promising class of algorithms for independent set by studying an associated proof system. Another influential paper from 2006, aptly titled paper "Proving Integrality Gaps without Knowing the Linear Program" explicitly demonstrated the potential of this theme [5]. This paper considered broad classes of linear relaxations for NP-optimzation problems, and proved nearly tight integrality gaps for several important problems (VertexCover, MaxSAT, and MaxCut) for *any* linear relaxation from the class. The classes that they considered correspond to the algorithms that underlie the semi-algebraic proof systems SA (Sherali-Adams) and LS (Lovász-Schrijver).

Since then, there has been a huge body of work, proving integrality gaps for large families of linear programming and semidefinite programming-based algorithms for a variety of important NP-hard optimization problems. These integrality gaps are none other than proof complexity lower bounds for specific families of formulas. Most notably are the proof systems Polynomial Calculus (PC) which is gives rise to a family of algebraic algorithms, Sherali-Adams (SA) which gives rise to a large family of linear programs, and Sum-of-Squares (SoS), which gives rise to a large family of semidefinite programs. In another exciting line of work, lower bounds for SA and SoS form the basis of exponential lower bounds on the size of extended formulations (and positive semi-definite extended formulations) for approximating MaxCut as well as for other NP-hard optimization problems.

The second theme is that (sometimes) proof system upper bounds can automatically generate efficient algorithms. More specifically, a proof system is said to be *automatizable* if there is an algorithm that can find proofs in that system efficiently, in the size of the shortest proof. (So if there is a short proof in the system, then it can be found efficiently as well.) SA is *degree* automatizable, in the sense that if there is a degree d proof, then it can be found time $n^{O(d)}$. SoS is also practically *degree* automatizable, if we assume that the coefficients have length bounded by a polynomial in n (or can be sufficiently well

approximated).[2] In an automatizable proof system, an efficient proof certifying the existence of a solution automatically implies an efficient algorithm for the problem. Using this theme, several remarkable recent papers have obtained new algorithms for unsupervised learning problems via efficient SoS proofs.

In the rest of this introduction, we give a brief tour of proof complexity including an introduction to the algebraic and semi-algebraic proof systems that we will focus on in this monograph, from the proof complexity point of view.

1.1 Proof complexity primer

Proof complexity refers to the study of nondeterministic algorithms for solving problems in *co*NP. Abstractly, let $\mathcal{L}$ be a language in *co*NP. For example, $\mathcal{L}$ could be the set of all undirected graphs that are *not* 3-colorable. The following definition of a proof system was given by Cook and Reckhow in their seminal paper introducing the key ideas behind the field [50].

Definition 1.1 (Propositional Proof System)**.** A propositional proof system for a language $\mathcal{L} \subseteq \{0,1\}^*$ is a polynomial-time function $\mathcal{P} : \{0,1\}^* \to \{0,1\}^*$ such that the following properties hold:

(1) (Soundness) For every y, $\mathcal{P}(y) \in \mathcal{L}$;

(2) (Completeness) For every $x \in \mathcal{L}$, there exists a y such that $\mathcal{P}(y) = x$.

We think of $\mathcal{P}$ as an efficient algorithm that checks to see if y encodes a legal proof that some x is in $\mathcal{L}$. If so, then $\mathcal{P}(y)$ outputs x; otherwise (if y does not code a legal proof), then $\mathcal{P}(y)$ outputs some canonical string $x \in \mathcal{L}$. The soundness property guarantees that the proof system can only produce proofs for strings in $\mathcal{L}$ and the completeness property means that every $x \in \mathcal{L}$ has a proof.

Definition 1.2 (Proof Size)**.** Let $\mathcal{L} \subseteq \{0,1\}^*$ and let $\mathcal{P}$ be a proof system for $\mathcal{L}$. For $x \in \mathcal{L}$ $\text{size}_{\mathcal{P}}(x)$ is the minimal natural number m such

[2] See the discussion in Section 3.2.3.3.

that there exists y, $|y| = m$ and $\mathcal{P}(y) = x$. In other words, $\text{size}_{\mathcal{P}}(x)$ is the length of the shortest $\mathcal{P}$-proof of x.

Definition 1.3 (Polynomially Bounded Proof System)**.** $\mathcal{P}$ is *polynomially bounded* (or p-bounded) if for sufficiently large n, for all $x \in \mathcal{L}$, $|x| \geqslant n$, $\text{size}_{\mathcal{P}}(x) \in |x|^{O(1)}$.

It is easy to see from the definitions that there exists a polynomially bounded proof system $\mathcal{P}$ for a language $\mathcal{L}$ if and only if $\mathcal{L}$ is in NP: The NP algorithm on input x simply guesses some y (of polynomial length) and accepts if and only if $\mathcal{P}(y) = x$. When $x \in \mathcal{L}$, since $\mathcal{P}$ is complete and polynomially bounded, there is some y of polynomial length such that $\mathcal{P}(y) = x$ and conversely if $x \notin \mathcal{L}$ then by soundness, for every y, $\mathcal{P}(y) \neq x$. In the other direction, any NP algorithm for $\mathcal{L}$ gives rise to a polynomially bounded proof system for $\mathcal{L}$. Since the existence of a polynomially bounded proof system for a language $\mathcal{L}$ is equivalent to saying that $\mathcal{L}$ is in NP, proving that *no* polynomially bounded proof system exists for a *co*NP-complete language (such as UNSAT) is equivalent to proving NP $\neq$ *co*NP. This is a daunting task — in particular, it implies that P $\neq$ NP.

In light of the difficulty of proving that *no* proof system for UNSAT (or for any other *co*NP-complete language) is polynomially bounded, much of the research in the field has focused on proving superpolynomial lower bounds for *standard* proof systems for the underlying *co*NP-complete language $\mathcal{L}$. For example, Resolution, Frege and Extended Frege are standard proof systems for UNSAT; the Hajos calculus is a natural proof system for proving non k-colorability of graphs; Cutting Planes, Sherali-Adams and SOS (Sum-of-Squares) are well-studied proof systems for proving that a set of linear inequalities has no integer solution; and Nullstellensatz and Polynomial Calculus are proof systems for showing that a set of polynomial equations has no common zero/one solution. In the subsequent section, we will describe many of these proof systems and give some concrete examples.

A key high-level point is that the natural proof systems are well-studied for a good reason and this is the link between proof complexity and algorithms. Namely, the best algorithms for $\mathcal{L}$, both exact and approximate, are usually associated with a natural proof system in

the sense that the transcript of the algorithm on some $x \in \mathcal{L}$ *is* a proof y in the associated proof system! Moreover, we can also associate transcripts of approximation algorithms with proofs. Therefore, proving lower bounds for well-studied proof systems for $\mathcal{L}$ is tightly connected to our understanding of how well large and natural classes of algorithms can solve or approximate the optimization problem.

The next definition allows us to compare the relative strength of different proof systems for the same language $\mathcal{L}$.

Definition 1.4 (p-Simulation)**.** Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be two propositional proof systems for $\mathcal{L}$. We say that $\mathcal{P}_1$ p-simulates $\mathcal{P}_2$ if there exists a polynomial q such that for sufficiently large n, for all $x \in \mathcal{L}$, $|x| \geqslant n$, $\text{size}_{\mathcal{P}_1}(x) \leqslant q(\text{size}_{\mathcal{P}_2}(x))$. In other words, $\mathcal{P}_1$ p-simulates $\mathcal{P}_2$ if for every $x \in \mathcal{L}$, the minimum proof length in $\mathcal{P}_1$ is at most polynomially larger than the minimum proof length in $\mathcal{P}_2$. $\mathcal{P}_1$ and $\mathcal{P}_2$ are p-equivalent if $\mathcal{P}_1$ p-simulates $\mathcal{P}_2$, and $\mathcal{P}_2$ also p-simulates $\mathcal{P}_1$.

While proof size is important, it is also important to be able to *find* a proof quickly. Given the likelihood that all proof systems for *co*NP-hard languages are not polynomially-bounded, we should measure the complexity of finding a proof with respect to the size of the shortest proof, which motivates the next definition.

Definition 1.5 (Polynomial Automatizibility)**.** A proof system $\mathcal{P}$ for $\mathcal{L}$ is *polynomially automatizable* if there exists an algorithm A that takes as input $x \in \mathcal{L}$ and returns a y such that $\mathcal{P}(y) = x$ and moreover, the runtime of A is polynomial in $\text{size}_P(x)$ — that is, the runtime is polynomial in the size of the shortest $\mathcal{P}$-proof of x.

Shortly, we define *algebraic* and *semialgebraic* proof systems for proving that a system of polynomial equations or inequalities has no integral solution. For these systems, proofs will consist of a sequence of polynomial equations/equalities. For these proof systems, we are interested not only in proof size (the total length of the proof), but also in the degree of the proof — the minimal degree d such that there is an algebraic proof where every polynomial in the proof has degree at most d. Thus, we define the following degree-based variant of automatizability.

Definition 1.6 (Degree Automatizibility)**.** An algebraic proof system $\mathcal{P}$ is *degree automatizable* if there is an algorithm A that returns a $\mathcal{P}$-refutation of f in time $n^{O(\deg_{\mathcal{P}}(f))}$, where $\deg_{\mathcal{P}}(f)$ is the minimal degree refutation of f in $\mathcal{P}$.

Derivations versus Refutations. We have defined proof systems as nondeterministic procedure for proving that $x \in \mathcal{L}$ where $\mathcal{L}$ is a language in *co*NP. What if we want to consider instead proofs of derivations, such as a proof that if $x \notin \mathcal{L}$, then $x' \notin \mathcal{L}$. Of course we can always determine if this implication is true by a reduction to our nondeterministic procedure for $\mathcal{L}$. For example, if $\mathcal{L}$ is UNSAT, and we want to prove that if x is a satisfiable Boolean formula, then x' is also satisfiable, then we can do so indirectly by obtaining a proof that $\neg(\neg x \vee x') \in$ UNSAT. However, it will often be more convenient to work directly with proofs of derivations (for example, when we want to study the proof complexity of approximation algorithms). To this end, we define a proof system for derivations as a polynomial-time function $\mathcal{P}$ from strings (encodings of proofs) to strings (encodings of implications of the form $x \to x'$), with the property that the range of $\mathcal{P}$ is exactly the set of all valid implications. (An implication $x \to x'$ is valid if $x \notin \mathcal{L}$, then $x' \notin \mathcal{L}$). The special case of refutations/proofs then corresponds to implications where x' is empty.

1.2 Proof systems for UNSAT

UNSAT is the language consisting of all unsatisfiable Boolean formulas. Thus, x is an encoding of a Boolean formula, and a proof system verifies the unsatisfiability of x, or equivalently it could verify that x is a Boolean tautology. Since any formula can be efficiently converted into an equivalent formula in conjunctive normal form (CNF), we will without loss of generality, focus our discussion on propositional proof systems for k-UNSAT — verifying the unsatisfiability of k-CNF formulas. A k-CNF formula $\mathcal{C}$ over Boolean variables $x_1, \ldots, x_n$ is a conjunction of clauses, $C_1, \ldots, C_m$, where each clause is a disjunction of k literals. We will often view a k-CNF formula as a set of clauses or constraints. A set of clauses $\{C_1, \ldots, C_m\}$ is *satisfiable* if there exists

a Boolean assignment α to the underlying variables such that every clause C_i evaluates to true under α; otherwise the set of clauses are *unsatisfiable*.

Typical propositional proof systems for UNSAT are *axiomatic*, meaning that they are described by a finite set of syntactic derivation rules, which describe how to derive new formulas from one or two previous ones. In an axiomatic system, a proof that a CNF formula $\mathcal{C}$ (over $x_1, \ldots, x_n$) is unsatisfiable will be (an encoding of) a sequence of formulas, where each formula in the sequence is either one of the initial clauses C_i, or follows from previous formulas in the sequence by one of the rules. Finally, the last formula in the sequence should be a canonical formula that is trivially unsatisfiable. (For example, the formula "0" or the formula $x \wedge \neg x$.)

Resolution. The Resolution proof system (more commonly called a refutation system since we are refuting the existence of a satisfying assignment) is one of the most well-known propositional proof systems and forms the basis for many well-known automated theorem provers and SAT solvers. There is only one rule (the Resolution rule): $(A \vee x),\ (B \vee \neg x) \to (A \vee B)$, where A and B are clauses, and by $A \vee B$ we mean the clause obtained by taking the disjunction of all literals occurring in A or B, removing duplications. For example, we can derive $(x_1 \vee x_2)$ from $(x_1 \vee x_3)$ and $(x_1 \vee x_2 \vee \neg x_3)$. A Resolution refutation of a k-CNF formula $\mathcal{C} = \{C_1, \ldots, C_m\}$ is a sequence of clauses such that every clause in the sequence is either an initial clause C_i, $i \in [m]$, or follows from two previous clauses by the Resolution rule, and such that the final clause is the empty clause. Resolution is sound and complete: a CNF formula $\mathcal{C}$ has a Resolution refutation if and only if $\mathcal{C}$ is unsatisfiable.

We will encode a Resolution refutation by encoding each clause in the sequence; since each clause has length $O(n)$, the number of clauses in the refutation is polynomially related to the bit-length encoding. Thus, for simplicity and without loss of generality we take the number of clauses to be the length of a Resolution refutation. If the initial formula is k-CNF formula for k constant, then its length is polynomial in n, the number of underlying variables. Therefore, a Resolution refutation

of a k-CNF over n variables is polynomially bounded if its length is polynomial in n.

Example. Consider the family of propositional formulas, $\{\mathsf{IND}_n,\ n \geqslant 2\}$ corresponding to the induction principle. We have n variables associated with IND_n, $x_1, \ldots, x_n$, and the following clauses: (1) (x_1); (2) For all $i < n$ $(\neg x_i \vee x_{i+1})$; (3) $(\neg x_n)$. For each n, IND_n has size polynomial in n and we say that IND_n has efficient Resolution refutations if for n sufficiently large, IND_n has a Resolution refutation of polynomial size in n. It is not hard to see that there are Resolution refutations of IND_n of linear length. More generally, any unsatisfiable Horn formula (CNF formula with at most one negated variable per clause) has efficient Resolution refutations, as does any unsatisfiable 2-CNF formula.

The well-known David-Putnam-Logemann-Loveland (DPLL) algorithms [52, 53] for satisfiability is our first example demonstrating the connection between proofs and algorithms. The DPLL algorithm is a complete, backtracking-based search algorithm for deciding the satisfiability of a CNF formula, $\mathcal{C}$. Whenever x is unsatisfiable, the transcript of the algorithm produces a decision tree over the underlying variables, where each leaf of the tree is labelled with some clause $C_i \in \mathcal{C}$ such that the partial truth assignment of the variables queried along the path to that leaf falsifies C_i. Such a decision tree, in turn, is actually a *tree-like* Resolution proof of $\mathcal{C}$. Therefore, running the DPLL algorithm on an unsatisfiable input x yields a Resolution proof that x is in Unsat. In the last decade, enormous progress has been made on practical SAT solvers, using more sophisticated backtracking

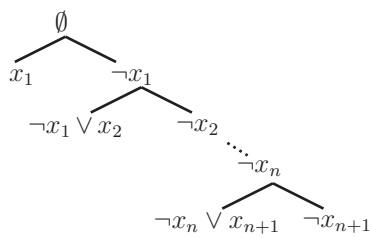

Figure 1.1: Resolution refutation of IND_n.

algorithms, incorporating caching and restarts. In particular, the CDCL algorithm (Conflict Driven Clause Learning) routinely solves very large instances of SAT (with thousands of variables) efficiently [113]. Once again, the CDCL algorithm as well as many of its extensions are based on Resolution: running CDCL on an unsatisfiable input x yields a Resolution refutation of x. Thus superpolynomial lower bounds for Resolution proves unconditionally that DPLL, and CDCL are not in P.

We will now explain how Resolution characterizes a class of *approximation* algorithms for SAT. Any instance of 3SAT has an assignment satisfying at least 7/8th's of the clauses, and such an assignment can be found efficiently — so 3SAT has a polynomial-time 7/8-approximation algorithm. In a major result, Hastad proved that no polynomial-time algorithm can do better unless $\mathsf{P} = \mathsf{NP}$ — that is, he showed that it is NP-hard to achieve an approximation ratio of $7/8 + \varepsilon$, for $\varepsilon > 0$ [72]. Proof complexity provides a framework for proving *unconditional* lower bounds on approximation algorithms. Extensions of DPLL and CDCL have been developed for solving and for approximating MaxSAT which are again based on Resolution in the following sense. If $\mathcal{C}$ is an unsatisfiable 3CNF formula, then running a Resolution-based $(7/8 + \varepsilon)$-approximation algorithm on input $\mathcal{C}$ will output a Resolution proof that C is not $(7/8 + \varepsilon)$ satisfiable. Again, known superpolynomial Resolution lower bounds on random unsatisfiable 3CNFs [47] can be invoked to prove unconditionally that *any* Resolution-based $(7/8 + \varepsilon)$ approximation algorithm is not in P.

Frege Proofs. The most well-known collection of proof systems for UNSAT, collectively referred to as Frege systems, are axiomatic systems typically presented in undergraduate logic textbooks. Lines in a Frege system are propositional formulas (usually over the standard basis $\{\wedge, \vee, \neg\}$). A Frege system is equipped with a finite set of axiom and rule schemas, and a Frege proof is a sequence of formulas, starting with axioms, and inferring new formulas from previous ones by applying these rules. Extended Frege systems are generalization of Frege systems where lines are boolean circuits (rather than formulas). Cook and Reckhow showed that standard propositional proof systems form a natural hierarchy, which mirrors the well-known circuit class hierarchy.

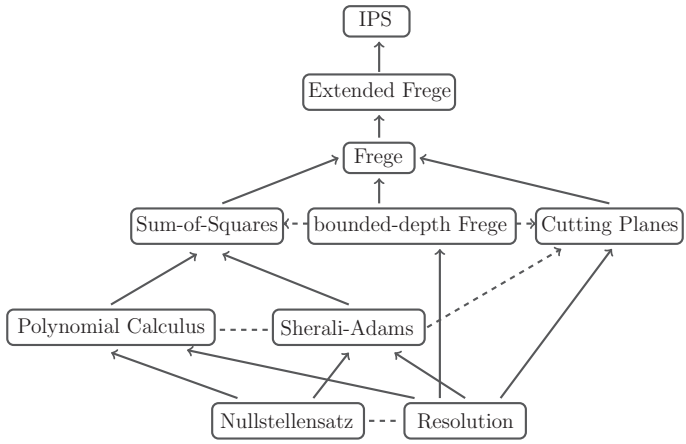

Figure 1.2: The hierarchy of standard propositional proof systems (not only those for UNSAT). An *arrow* $\mathcal{P}_1 \to \mathcal{P}_2$ indicates that $\mathcal{P}_2$ is strictly stronger than $\mathcal{P}_1$: $\mathcal{P}_2$ p-simulates $\mathcal{P}_1$ and there exists a formula which has polynomial-size proofs in $\mathcal{P}_2$ but which requires super-polynomial size to prove in $\mathcal{P}_1$. A *dashed arrow* from $\mathcal{P}_1$ to $\mathcal{P}_2$ implies that there is a formula which has short proofs in $\mathcal{P}_2$ but not in $\mathcal{P}_1$, but it is unknown whether $\mathcal{P}_2$ p-simulates $\mathcal{P}_1$. A *dashed line* between $\mathcal{P}_1$ and $\mathcal{P}_2$ indicates that $\mathcal{P}_1$ and $\mathcal{P}_2$ do not p-simulate each other.

At the bottom are Resolution proofs, where lines are clauses and thus they (roughly) correspond to depth-1 circuits; above that are bounded-depth Frege systems where lines are bounded-depth formulas, and thus they correspond to bounded-depth AC_0 circuits. Similarly, Frege systems correspond to formulas (NC_1 circuits) and Extended Frege systems correspond to polynomial-size circuits. Thus as shown by Cook and Reckhow, bounded-depth Frege p-simulates Resolution, Frege p-simulates bounded-depth Frege, and Extended Frege p-simulates Frege.

Lower Bounds for UNSAT Proof Systems. In terms of lower bounds, Haken famously proved exponential lower bounds for Resolution (using the propositional pigeonhole principle as the hard formulas) [71]. Lower bounds for the Tseitin formulas (essentially random mod 2 equations) and for random k-CNF formulas were subsequently obtained by [148] and [47]. In [31] an even more general result was proven, showing that Resolution proof size could be reduced to Resolution width. (The width of a Resolution proof is the maximum clause size

over all clauses in the proof.) Beyond Resolution, a landmark paper by Ajtai [1] proved superpolynomial lower bounds (again for the pigeonhole principle) for Frege systems of bounded depth, and in [26] this was improved to truly exponential lower bounds. It is a longstanding open problem to prove superpolynomial lower bounds for Frege systems. A comprehensive treatment of propositional proof complexity can be found in the following surveys [28, 139, 130].

Proof Search. Some non-trivial proof-search algorithms have been discovered for several weak proof systems for UNSAT. Beame and Pitassi [27] showed that any Resolution proof of size S can be found in time $n^{O(\sqrt{n \log S})}$. Ben-Sasson and Wigderson [31] noted that the same result follows from the reduction from proof size to width, by simply generating all clauses, according to the Resolution rule, of width bounded by the width of the proof. For tree-like Resolution, the size-width trade-off yields an automating algorithm which runs in quasi-polynomial time.

For stronger proof systems, a line of work has sought to rule out their automatizability under widely-believed cryptographic assumptions. This began with the work of Krajícek and Pudlák, who showed that Extended Frege is not automatizable unless RSA is not secure against polynomial size circuits [99]. Building on these ideas, Bonet *et al.* showed non-automatizability of Frege [35] and bounded-depth Frege systems [34] under the assumption that computing the Diffie-Hellman function cannot be computed by polynomial and sub-exponential size circuits respectively. For weaker proof systems, this approach seems less hopeful as it is not clear how to use the limited reasoning of these weak proof systems to break cryptographic assumptions leaving, in particular, the polynomial automatizability of Resolution as a tantalizing open problem. In an important paper, Alekhnovich and Razborov [2] showed that if Resolution, or even tree-like Resolution were polynomially automatizable, then the hierarchy of parameterized complexity classes would collapse; that is, $\mathsf{W}[\mathsf{P}] = \mathsf{FPT}$. In a recent groundbreaking paper, Atserias and Müller [12] settled the question of automating Resolution proofs by showing that Resolution is not even sub-exponentially automatizable unless $\mathsf{P} = \mathsf{NP}$.

1.3 Algebraic proof systems

In this section we consider *algebraic* proof systems for proving that a system of polynomial equations/identities has no solution. For this language, we have a fixed ambient field or ring which is typically the integers or a finite field. An instance is now (an encoding of) a set of polynomial equations over the variables $x_1, \ldots, x_n$, and we want to prove that this set of polynomial equations has no solution over the underlying field.

Any algebraic proof system can also be viewed as a proof system for UNSAT, and more generally using polynomial-time reductions we can view a proof system for one *co*NP-complete language as a proof system for any other another *co*NP language (although it may not be a natural one). In the case of UNSAT, we translate each clause into an equivalent polynomial equation. For example $(x_1 \vee \neg x_2 \vee x_3)$ becomes the equation $(1 - x_1)(x_2)(1 - x_3) = 0$, and we can also add the equations $x_i^2 - x_i = 0$ in order to force only Boolean solutions.

Nullstellensatz and Polynomial Calculus. These proof systems are based on Hilbert's Nullstellensatz which states that a system of polynomial equations $p_1(x_1, \ldots, x_n), \ldots, p_m(x_1, \ldots, x_m)$ over a field F is unsatisfiable if and only if the ideal in the ring $F[x_1, \ldots, x_n]$ generated by $p_1(x), \ldots, p_m(x)$ contains 1. In other words, if and only if there exists polynomials $q_1(x), \ldots, q_m(x)$ such that

$$p_1 q_1 + \ldots + p_m q_m = 1.$$

As mentioned above, in our standard context where the variables x_i range over the Boolean domain $\{0, 1\}$, we can enforce this by adding $x_i^2 - x_i$ to the list of polynomial equations. Alternatively we can just factor them out by working in the ring $F[x_1, \ldots, x_n]/(x_1^2 - x_1, \ldots, x_n^2 - x_n)$ of multilinear polynomials. Thus $q_1(x), \ldots, q_m(x)$ can be assumed to be multilinear and therefore of degree at most n.

A Nullstellensatz (Nsatz) refutation of $p_1(x), \ldots, p_m(x)$ is thus a set of polynomials $q_1, \ldots, q_m$ such that $\sum_i p_i q_i = 1$. It is clear that this proof satisfies the Cook-Reckhow definition: it is easy to check that $\sum_i p_i q_i = 1$, and soundness and completeness follow from Hilbert's Nullstellensatz.

The *size* of a Nullstellensatz refutation is the sum of the sizes of the $q_i(x)$'s. Another important measure for Nullstellensatz refutations is the *degree*, which is the maximal degree of the $p_i(x)q_i(x)$'s. As mentioned above, the degree is at most linear. An important property of Nullstellensatz refutations is that they are *degree automatizable*: If an initial family $p_1(x), \ldots, p_m(x)$ of polynomials has a degree d Nullstellensatz refutation, then it can be found in time $n^{O(d)}$ by simply solving a system of linear equations where the underlying variables of the equations are the coefficients of the monomials in $q_1(x), \ldots, q_m(x)$.

The Polynomial Calculus (PC) is a dynamic version of Nullstellensatz that is rule-based. The inference rules are: (i) from $f = 0$, $g = 0$ we can derive $\alpha f + \beta g = 0$, and (ii) from $f = 0$ we can derive $fg = 0$. The size of a PC proof is the sum of the sizes of all polynomials in the derivation, and the degree is the maximum degree of any line in the proof. Because this system is dynamic, it is sometimes possible (through cancellations) to obtain a much lower degree refutation than is possible using the static Nullstellensatz system. A great example is the induction principle IND_n mentioned above. It is not too hard to see that they have degree 2 PC refutations; on the other hand, it has been shown that their Nsatz degree is $\Theta(\log n)$ [38]. Clegg, Edmonds and Impagliazzo [48] proved that, like Nsatz, PC refutations are degree automatizable.

Stronger Algebraic Proof Systems. The Nullstellensatz and Polynomial Calculus proof systems witness the unsolvability of a set $\mathcal{P}$ of polynomial equations by demonstrating that 1 lies in the ideal generated by $\mathcal{P}$, where the measure of complexity of the proof is the maximal degree. More generally proofs can be viewed as directed acyclic graphs, where each line in the proof is either a polynomial from $\mathcal{P}$, or follows from two previous lines by taking a linear combination of two previous lines, or by multiplying a previous equation by a variable, and where the final line is the identically 1 polynomial. Thus, the entire proof can be viewed more compactly as an algebraic circuit, with the leaves labelled by polynomials from $\mathcal{P}$, and constants, and where internal vertices are either plus or times gates. This proof system (now called Hilbert-IPS) was introduced in [117] and is known to be quite powerful: it can efficiently simulate proofs in all standard propositional proof systems. However, it

is not known to be a Cook-Reckhow proof system since proofs are not known to be verifiable in polynomial time. Determinining if a Hilbert-IPS circuit is a proof amounts to determining if the polynomial that it computes is the identically-1 polynomial, and therefore verifying a Hilbert-IPS proof amounts to solving PIT (polynomial identity testing), a problem that admits a randomized (one-sided error) polynomial-time algorithm. A longstanding and important problem is to prove (or disprove) that PIT has a *deterministic* polynomial time algorithm.

A generalization of Hilbert-IPS called the IPS proof system (the Ideal Proof System) was introduced by Grochow and Pitassi [67]. IPS proofs have no rules — a proof of unsolvability of $\mathcal{P}$ is simply an algebraic circuit $\mathcal{C}$ with two types of inputs, $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$, and subject to the following properties (which can be verified by a PIT algorithm): (i) $\mathcal{C}$ with zero substituted for each of the y_i variables evaluates to the identically zero polynomial; (ii) $\mathcal{C}$ with $p_1(x), \ldots, p_m(x)$ substituted for $y_1, \ldots, y_m$, computes the identically 1 polynomial. As for Hilbert-IPS, IPS are not known to be deterministically verifiable in polynomial time, and can simulate all standard Frege and Extended Frege systems. Grochow and Pitassi prove that superpolynomial lower bounds for IPS for any family of unsolvable polynomials $\mathcal{P}$ would resolve the longstanding problem of separating VP from VNP, thus establishing a connection between lower bounds in proof complexity and circuit lower bounds.

In [104] Tzemeret and Wang define a noncommutative version of IPS, and quite surprisingly, they prove that it is *equivalent* to standard Frege systems. In different but related work, Grigoriev and Hirsch [65] introduce an algebraic proof system with derivation rules corresponding to the ring axioms; unlike the IPS systems, proofs in their system can be verified in polynomial-time. (See [120] for a survey of algebraic proof systems.)

Lower Bounds and Proof Search. Lower bounds for Nsatz and PC are known and will be discussed in Chapter 6. For the stronger algebraic proof systems which can efficiently represent polynomials by algebraic circuits (Hilbert-IPS and IPS), there are no nontrivial lower bounds (although lower bounds have been proven for some restricted

subsystems [54].) In terms of proof search, both Nsatz and PC are degree automatizable [49], using a modification of the Grobner basis algorithm.

1.4 Semialgebraic proof systems

In the previous section we discussed proof systems for proving that a system of polynomial identities is solvable. We can generalize to the semialgebraic setting, where now the input is a system of polynomial *inequalities*, and a semialgebraic proof should certify that the system of inequalities has no solution over the reals. This generalizes algebraic proofs (over the reals) since we can always write a polynomial equality as a set of two inequalities. And by translating clauses into a polynomial inequalities ($(x_1 \vee \neg x_2 \vee x_3)$ becomes $x_1 + x_3 \geqslant x_2$) we can also view semialgebraic proof systems as proof systems for UNSAT.

Cutting Planes. Cutting Planes is a semi-algebraic proof system over the integers, where the inequalities are *linear*. It was originally devised as a method for solving integer linear programs by relaxing them to fractional constraints (replacing $x_i \in \{0,1\}$ by $0 \leqslant x_i \leqslant 1$), and then deriving new inequalities from previous ones via the Cutting Planes rules, as a way to tighten the relaxed polytope, to whittle away at non-integral points. There are two rules for deriving new inequalities:

(i) Combination: From $\sum_i a_i x_i \geqslant \gamma$ and $\sum_i b_i x_i \geqslant \delta$, derive $\sum_i (\alpha a_i + \beta b_i) x_i \geqslant \alpha\gamma + \beta\delta$, for nonnegative integers α, β.

(ii) Division with rounding: From $\sum_i a_i x_i \geqslant \gamma$, derive $\sum_i \frac{a_i}{c} x_i \geqslant \lceil \frac{\gamma}{c} \rceil$, provided that c divides all a_i.

Sherali-Adams. Sherali-Adams (SA) will be the focus of Chapter 2. Like Cutting Planes, it is a semi-algebraic proof system over the integers, where lines are polynomial inequalities. Like Nsatz, it is a static proof system. Let $p_1(x), \ldots, p_m(x)$ be a set of polynomial inequalities that includes that polynomial inequalities $x_i^2 \geqslant x_i, x_i \geqslant x_i^2$ for all $i \in [n]$. A SA refutation of $p_1(x) \geqslant 0, \ldots, p_m(x) \geqslant 0$ is a set of polynomials $q_1(x), \ldots, q_m(x)$ such that each $q_i(x)$ is a non-negative combination of

non-negative juntas, and such that $\sum_i p_i q_i = -1$. (A non-negative junta is a polynomial that corresponds to a conjunction of Boolean literals — for example, $x_1(1-x_2)x_3$ is a non-negative junta that corresponds to the Boolean conjunction $x_1 \neg x_2 x_3$).

Sum-of-Squares. (SoS) is another static semi-algebraic proof system, and is the focus of Chapter 3. Again let $p_1(x), \ldots, p_m(x)$ be a set of polynomial inequalities that includes $x_i^2 - x_i = 0$ for all $i \in [n]$. An SoS refutation of $p_1(x) \geqslant 0, \ldots, p_m(x) \geqslant 0$ is a set of polynomials $q_1(x), \ldots, q_m(x)$ such that each q_i is a *sum-of-squares* — that is, $q_i(x)$ can be written as $\sum_j (r_{i,j}(x))^2$ for some polynomials $r_{i,j}(x)$, and such that $\sum_i p_i q_i = -1$. It is not hard to see that non-negative combinations of non-negative juntas are sum-of-squares, and thus the SoS proof system extends SA.

Lower Bounds and Proof Search. We will discuss lower bounds for semialgebraic proof systems in detail in Chapter 6. In terms of proof search, both SA and SoS are degree automatizable refutation systems (under some niceness conditions) which makes them extremely useful for designing good approximation algorithms. Indeed developing this connection is one of the main themes of this monograph.

1.5 Connection between algorithms and proofs

Having described the specific proof systems that will be the highlight of the connection between proof complexity and algorithms, we will return to the two themes underlying this connection.

Suppose that you have a correct algorithm A for solving SAT (or some other NP-hard optimization problem). By correct we mean that when run on a satisfiable instance, the algorithm will always output "satisfiable" and when run on an unsatisfiable formula, the algorithm will always output "unsatisfiable". Since the algorithm is correct, on any unsatisfiable formula f, we can view the computation of the algorithm on input f as a *proof* of the unsatisfiability of f. Furthermore, if A's run on f is efficient (say polynomial-time), then A provides us with a polynomial-length refutation that f is unsatisfiable. Moreover, the same idea applies to approximation algorithms. That is, suppose that we have

a correct 2-approximation algorithm for some optimization problem, such as Independent Set. That is, for every graph G, it outputs an independent set of size at least half of the size of the largest independent set in G. If we run the algorithm on some graph G and it returns an independent set of size s, this is a proof that G does not contain an independent set of size greater than $2s$.

Now speaking somewhat informally, if A is correct, then there is a proof of A's correctness. This proof of correctness of A, when applied to each unsatisfiable f gives us a propositional proof that f is unsatisfiable, in some proof system, let's call it P_A. Therefore, superpolynomial lower bounds for this proof system show that there can be no polynomial-time algorithm whose proof of correctness is based on P_A.

As an example of this paradigm, we explained how state-of-the-art complete algorithms for SAT are based on Resolution, and thus Resolution lower bounds imply similar impossibility results for a broad class of algorithms for SAT. For stronger proof systems such as Frege and Extended Frege, what is the corresponding class of algorithms? It turns out that Extended Frege proofs capture nearly all known provably correct algorithms! Therefore, superpolynomial lower bounds for Extended Frege systems would have far-reaching consequences: it would essentially rule out almost all known algorithms and algorithmic paradigms for efficiently solving any NP-hard problem.

Unfortunately, at present we do not know how to prove superpolynomial Extended Frege lower bounds or superpolynomial Frege lower bounds. But luckily, for many of the algebraic and semi-algebraic proof systems, we *can* prove superpolynomial lower bounds, and at the same time, the corresponding algorithms capture interesting and ubiquitous families of algorithms. In particular, we will develop the proof system Sherali-Adams (SA) from an algorithmic point of view, and see that SA captures a large class of linear programming relaxations. We will see, through the theory of linear programming duality, that SA derivations (a syntactic object) are dual to points in the corresponding LP polytopes (a semantic object), thus linking the SA proof complexity of proving that a solution exists, to the algorithmic complexity of finding such a solution. In a similar manner we will develop the SoS system, and see that it captures semi-definite programming relaxations. Again, we will

link SoS derivations to the points in the corresponding semi-definite cone, thus linking SoS complexity of proving the existence of a solution to the algorithmic complexity of finding such a solution.

In Chapter 5 we prove SoS and SA lower bounds for a particular family of 3XOR and instances. Using the above connection, this implies in a precise sense that *no* linear programming or semi-definite program based on low-degree SoS can approximate 3XOR or MaxSAT better than the trivial approximation. That is, we see an exact instance where a proof complexity lower bound implies lower bounds for a large class of approximation algorithms.

In the other direction, in Chapter 4 we will see how SoS *upper bounds* can lead to efficient algorithms.[3] The high level idea is as follows. Start with some optimization problem such as Independent Set. If we can manage to give a low degree SoS proof that for every graph G, there exists a 2-approximation, then by the degree automatizability of SoS, this implies an efficient algorithm for actually finding the solution. This idea is very powerful, and has been applied to many problems in machine learning. The general approach is to obtain low-degree SoS proofs of polynomial sample complexity bounds for the learning problem, and then by degree-automatizability of SoS, this yields an efficient learning algorithm.

As a toy example to illustrate this connection, suppose that you are given samples from an unknown Gaussian with mean μ and standard deviation one, and want to approximately recover the true mean from these samples. A necessary condition for succeeding in polynomial time are *sample complexity* bounds — that is, polynomially many samples must be enough to approximate the true mean information theoretically. Now suppose that we can formalize and prove this sample complexity bound with a low degree SoS proof. Then by degree automatizability of SoS, this automatically gives a polynomial-time algorithm for solving the learning problem! In Section 4.3, we discuss several instantiations of this approach where state-of-the-art learning algorithms are obtained for: dictionary learning, tensor decomposition, as well as learning mixtures of Gaussians.

[3] We remark that there is a long history of related results in logic, showing strong links between proofs and programs. In particular, in restricted systems of arithmetic, programs/algorithms can be extracted from proofs of existence.

Proof. Let ϕ be our random instance. For the soundness, we leave it as an exercise to the reader to use the same argument as in the 3XOR case. For completeness, we define $\phi_\oplus$ to be a 3XOR instance as follows: for each clause $C : (x_i^{e_i} \vee x_j^{e_j} \vee x_k^{e_k})$, we have a constraint $C' : x_i x_j x_k = a_{ijk}$, where $a_{ijk} = (-1)^{e_i+e_j+e_k}$. This is a random instance of 3XOR, as i, j, k were chosen i.i.d. and $(-1)^{e_i+e_j+e_k}$ is uniformly distributed over ± 1. Thus, with probability 0.99 there exists a pseudo-distribution satisifying all constraints in $\phi_\oplus$. The result follows by noting that if we transform this pseudo-distribution back to $\{0, 1\}$ valued variables via $x \to \frac{1-x}{2}$, any assignment in the support of the pseudo-distribution satisfies C', and any assignment satisfying C' also satisfies C. □

Recently there has been a surge of works for showing SoS lower bounds for *average-case* settings. [95] proved a sharp SoS lower bound to precisely characterize the number of clauses required for refuting a constraint satisfaction problem with a given predicate. Following a sequence of work [111, 75], Barak *et al.* [18] proved an optimal lower bound for the planted clique problem via the new technique of *pseudocalibration*. This technique was later used in [74] to prove strong lower bounds for optimizing random degree 3 polynomials over the unit sphere and Sparse principal component analysis (PCA).

5.3 Applications of lower bounds

So far we have focused on using SoS degree bounds and integrality gaps is to rule out LP and SDP relaxations of NP-hard optimization problems. Quite surprisingly, SA and SoS lower bounds can also be used to rule out other very general classes of algorithms. Again these proofs are reductions, albeit much more sophisticated ones. The reductions we discuss next are examples of *hardness escalation* whereby (SA or SoS) lower bounds for computing (or approximating) a function in a weaker model (LP or SDP) can be lifted via function composition to obtain lower bounds in a stronger model of computation. Some of the early examples of lifting include Sherstov's pattern matrix method [141], and Raz and McKenzie's separation of the monotone NC hierarchy [128]. In recent years, many lifting theorems have been discovered, and have in

turn resolved a large number of open problems in circuit complexity, game theory and proof complexity (i.e., [61, 60, 62, 56]). Unfortunately, the ideas and even the setup for these results are beyond the scope of this manuscript, so we will settle with at least mentioning some of the main lifting theorems that use SA or SoS lower bounds as their starting point.

First, SoS degree lower bounds, and more specifically Nullstellensatz degree bounds, have been used to prove exponential size lower bounds for monotone *circuit models*. Monotone span programs capture the power of reasoning using linear algebra in order to compute monotone functions [81]. [118] prove a lifting theorem between Nullstellensatz degree and monotone span program size that implies exponential lower bounds on the size of monotone span programs for several functions (and over all fields). By the known equivalence between monotone span programs and linear secret sharing schemes, this also implies exponential lower bounds for the latter.

Secondly, SoS degree lower bounds have been used to prove exponential lower bounds for extended formulations [150]. More specifically, SA degree bounds were shown to imply lower bounds for LP extended formulations [40, 94] and similarly, SoS lower bounds were shown to imply lower bounds for SDP extended formulations [98].

As mentioned above, all of these applications of SoS lower bounds are instantiations of *lifting theorems* whereby query complexity lower bounds for a particular search problem are *lifted* via composition with an inner gadget, in order to obtain stronger communication complexity lower bounds in the corresponding communication measure. For example, [128, 61] lift tree-like Resolution height (here the query measure is decision tree height) to deterministic communication complexity, which in turn is known to be equivalent to monotone formula size. [118] lift Nullstellensatz degree (here the query measure is polynomial degree) to its corresponding communication measure, which in turn is known to be equivalent to monotone span program size. And [94] lift SA degree (where the query measure is junta degree) to the nonnegative rank of the communication matrix, which in turn is known to be equivalent to LP extension complexity [150]. Finally, [98] proved via a lifting theorem, that SoS lower bounds imply SDP extension complexity lower bounds.

Appendices

Appendix

Section 2.2.3 Missing Proofs

Claim 2.29. *If there is a non-negative linear combination of the constraints of* $\mathsf{SA}_d(\mathcal{P})$ *equalling* $c_0 \in \mathbb{R}$*, then there exists a degree* d *SA derivation of* c_0 *from* $\mathcal{P}$*.*

Proof. Denote by $L(y)$ the SA constraint corresponding to the multilinearization of $P_i(x)J_{S,T}(x)$ over the placeholder variables y, for $P_i(x) \geqslant 0 \in \mathcal{P} \cup \{1 \geqslant 0\}$, and $J_{S,T}(x)$ a degree at most $(d - \deg(P_i))$ non-negative junta. Suppose that there exists a non-negative linear combination

$$c_i \sum_{i=1}^{\ell} L(y) = c_0. \tag{1}$$

We can translate this into a sum over the x-variables by replacing each linearized $L(y)$ by its corresponding term in the x variables, $J_{S_i,T_i}(x) \cdot P_i(x)$

$$c_i \sum_{i=1}^{\ell} P_i(x) \cdot J_{S_i,T_i}(x).$$

It may no longer be the case that this evaluates to c because terms which previously cancelled in the linearized sum may no longer cancel in this non-linearized sum. The axioms $\pm(x_i - x_i^2) \geqslant 0$ can be used

to mimic the linearization. Each term $c\prod_{i\in[k]} x_i^{a_i}$ can be linearized by introducing the following telescoping sum

$$\sum_{i=1}^{k}\left(\sum_{\ell=0}^{a_i-2} c(x_i - x_i^2)x_i^\ell \prod_{j>i} x_j^{a_j} \prod_{j<i} x_j\right).$$

Each term in this sum is of the form $(x_i^2 - x_i)\cdot J_{S,T}(x)$ for some S, T with $|S| + |T| \leqslant d-2$, and therefore is a valid inequality for SA. The degree of this proof is the maximum degree of among the constraints being linearized, and therefore is bounded above by d. □

Section 3.1 Missing Proofs

Lemma 3.41. *For any set of polynomial inequalities $\mathcal{P} = \{P_1(x) \geqslant 0, \ldots, P_m(x) \geqslant 0\}$, $\mathsf{SOS}_d(\mathcal{P})$ satisfies the following properties*:

1. $0 \leqslant y_J \leqslant y_I \leqslant 1$ *for every* $I \subseteq J \subseteq [n]$ *with* $|J| \leqslant d$.
2. $\sum_{J\subseteq T}(-1)^{|J|} y_{S\cup J} \geqslant 0$ *for every non-negative* d*-junta* $J_{S,T}(x)$.
3. *If* $\alpha \in \{0,1\}^n$ *satisfies every* $P_i(x) \geqslant 0 \in \mathcal{P}$, *then* $\alpha \in \mathrm{proj}_{[n]}(\mathsf{SOS}_d(\mathcal{P}))$ *for every* $d \geqslant \deg(\mathcal{P})$.[1]

Proof. To prove (1), we will use the fact that the diagonal entries of a symmetric PSD matrix are non-negative (Claim 3.5), and that $\mathcal{M}_d(y)$ is symmetric PSD. First, let $I \subseteq [n]$ with $|I| \leqslant d$. Observe that y_I occurs on the diagonal of $\mathcal{M}_d(y)$ and therefore $y_I \geqslant 0$. To prove that $y_I \leqslant 1$, define $u \in \mathbb{R}^{\binom{n}{\leqslant d}}$ as

$$u_K = \begin{cases} 1 & \text{if } K = \emptyset, \\ -1 & \text{if } K = I, \\ 0 & \text{otherwise.} \end{cases}$$

Then,

$$u^\top \mathcal{M}_d(y) u = \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} y_\emptyset & y_I \\ y_I & y_I \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = y_\emptyset - y_I \geqslant 0,$$

[1] Recall that $\mathrm{proj}_{[n]}(\mathcal{P}) = \left\{\alpha \restriction_{y_{\{1\}},\ldots,y_{\{n\}}} : \alpha \in \mathcal{P}\right\}$, the orthogonal projection of $\mathcal{P}$ to the first n variables.

where the final inequality follows because $\mathcal{M}_d(y) \succeq 0$. Finally, because $y_\emptyset = 1$, we have $1 - y_I \geqslant 0$. Now, for any $I \subseteq J \subseteq [n]$ with $|J| \leqslant 2d$, define $u' \in \mathbb{R}^{n^{2d}+1}$ as

$$u'_K = \begin{cases} 1 & \text{if } K = I, \\ -1 & \text{if } K = J, \\ 0 & \text{otherwise.} \end{cases}$$

Then, as before,

$$u'^{\top} \mathcal{M}_d(y) u' = \begin{bmatrix} 1 & -1 \end{bmatrix} \begin{bmatrix} y_I & y_J \\ y_J & y_J \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = y_I - y_J \geqslant 0.$$

Therefore, we have $0 \leqslant y_J \leqslant y_I \leqslant 1$.

For (2), let $\sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T}$ be the y-variable representation of some non-negative d-junta $J_{S,T}(x)$. Define the vector $v \in \mathbb{R}^{\binom{n}{\leqslant d}}$ as $v_I = 1$ if y_I occurs positively in the junta, $v_I = -1$ if y_I occurs negatively, and $v_I = 0$ if y_I is absent from the junta. We claim that

$$v^{\top} \mathcal{M}_d(y) v = \sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T}.$$

Indeed, multiplying by v is equivalent to multiplying the principal submatrix M of $\mathcal{M}_d(y)$ corresponding to rows and columns indexed by $\{I : v_I \neq 0\}$ by the vector $v' := v \restriction_{v_I \neq 0}$. First, let's look at the vector $v'^{\top} M$. The first entry of this vector, corresponding to multiplying $v'^{\top}$ by the column indexed by S, is $\sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T}$. We claim that the rest of the entries in this vector are 0. To prove this, we will use the following property of juntas $J_{S,T}(x)$: for any $i \in T$, because T is disjoint from S, we can write

$$\sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T} = \sum_{J \subseteq T : i \in J} (-1)^{|J|} y_{S \cup J} - \sum_{J \subseteq T : i \notin J} (-1)^{|J|} y_{S \cup J},$$

as well as the observation that the only non-zero entries v_L are such that $S \subseteq L \subseteq S \cup T$. Now, consider an entry, other than the first, in the vector $v'^{\top} M$. For this entry to be included in $v'^{\top} M$, it must be non-zero in $v^{\top} \mathcal{M}_d(y)$, and so by the definition of v it must correspond

to a column of $\mathcal{M}_d(y)$ indexed by $S \cup K$ for some $K \subseteq T$ with $K \neq \emptyset$. Denote this entry by $(v'^{\top} M)_{S \cup K}$, and let $i \in K \cap T$. Then,

$$\begin{aligned}(v'^{\top} M)_{S \cup K} &= \sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T \cup K}, \\ &= \sum_{J \subseteq T : i \in J} (-1)^{|J|} y_{S \cup J \cup K} - \sum_{J \subseteq T : i \notin J} (-1)^{|J|} y_{S \cup J \cup K} = 0.\end{aligned}$$

Therefore,

$$\begin{aligned}v^{\top} \mathcal{M}_d(y) v = v'^{\top} M v' &= \left(\sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T},\ 0,\ 0,\ \ldots,\ 0 \right) v' \\ &= \sum_{J \subseteq T} (-1)^{|J|} y_{S \cup T} \geqslant 0,\end{aligned}$$

where the last equality follows because the first entry of v' is the entry $v_S = 1$.

For (3), let $\alpha \in \{0,1\}^n$ such that $P_i(\alpha) \geqslant 0$ for every $P_i(x) \geqslant 0 \in \mathcal{P}$. The moment matrix corresponding to α is defined as $\mathcal{M}(\alpha)_{I,J} = \prod_{i \in I} \alpha_i \prod_{j \in J} \alpha_j$ for every $|I|, |J| \leqslant d$ with $I \cap J = \emptyset$, and $\mathcal{M}(\alpha, P_i)$ is defined analogously. Extend α to an $\binom{n}{\leqslant d}$-dimensional vector $\tilde{\alpha}$ by defining $\tilde{\alpha}_I = \prod_{i \in I} \alpha_I$. Then, for any $v \in \mathbb{R}^{\binom{n}{\leqslant d}}$,

$$v^{\top} \mathcal{M}(\tilde{\alpha}) v = v^{\top} \tilde{\alpha} \tilde{\alpha}^{\top} v = (v^{\top} \tilde{\alpha})^2 \geqslant 0,$$

and so $\mathcal{M}(\tilde{\alpha}) \succeq 0$. To see that $\mathcal{M}(\tilde{\alpha}, P_i) \succeq 0$, define the vector p where $p_I = P_i(\alpha) \prod_{j \in I} \alpha_j$ for $|I| \leqslant d - \deg(P_i)/2$, and observe that $pp^{\top} = \mathcal{M}(\tilde{\alpha}, P_i)$. Therefore $v^{\top} \mathcal{M}(\tilde{\alpha}, P_i) v = (v^{\top} p)^2 \geqslant 0$. □

Acknowledgments

The authors are grateful to Aleksandar Nikolov, Robert Robere, and Morgan Shirley for their comments and suggestions on an earlier version on this monograph. The authors would like to thank Ian Mertz for his suggestions on the presentation of the 3XOR lower bound in Section 5.1.

Finally, we are grateful to Paul Beame for correcting several historical remarks made in an earlier draft of this manuscript.

References

[1] Ajtai, M. (1994). "The Complexity of the Pigeonhole Principle". *Combinatorica*. 14(4): 417–433.

[2] Alekhnovich, M. and A. A. Razborov (2008). "Resolution Is Not Automatizable Unless W[P] Is Tractable". *SIAM J. Comput.* 38(4): 1347–1363.

[3] Alon, N. and A. Naor (2006). "Approximating the Cut-Norm via Grothendieck's Inequality". *SIAM J. Comput.* 35(4): 787–803.

[4] Arora, S., B. Barak, and D. Steurer (2010). "Subexponential Algorithms for Unique Games and Related Problems". In: *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23–26, 2010, Las Vegas, Nevada, USA*. 563–572.

[5] Arora, S., B. Bollobás, L. Lovász, and I. Tourlakis (2006). "Proving Integrality Gaps without Knowing the Linear Program". *Theory of Computing*. 2(2): 19–51.

[6] Arora, S., C. Lund, R. Motwani, M. Sudan, and M. Szegedy (1998). "Proof Verification and the Hardness of Approximation Problems". *J. ACM*. 45(3): 501–555.

[7] Arora, S., S. Rao, and U. V. Vazirani (2009). "Expander flows, geometric embeddings and graph partitioning". *J. ACM*. 56(2): 5:1–5:37.

[8] Arora, S. and S. Safra (1998). "Probabilistic Checking of Proofs: A New Characterization of NP". *J. ACM*. 45(1): 70–122.

[9] Artin, E. (1927). "Über die Zerlegung definiter Funktionen in Quadrate". *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*. 5(1): 100–115.

[10] Asano, T. and D. P. Williamson (2002). "Improved Approximation Algorithms for MAX SAT". *J. Algorithms*. 42(1): 173–202.

[11] Atserias, A. and T. Hakoniemi (2018). "Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs". *CoRR*. abs/1811.01351.

[12] Atserias, A. and M. Müller (2019). "Automating Resolution is NP-Hard". *CoRR*. abs/1904.02991.

[13] Austrin, P., S. Benabbas, and K. Georgiou (2016). "Better Balance by Being Biased: A 0.8776-Approximation for Max Bisection". *ACM Trans. Algorithms*. 13(1): 2:1–2:27.

[14] Barak, B., Z. Brakerski, I. Komargodski, and P. K. Kothari (2018). "Limits on Low-Degree Pseudorandom Generators (Or: Sum-of-Squares Meets Program Obfuscation)". In: *Advances in Cryptology — EUROCRYPT 2018 — 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29–May 3, 2018 Proceedings, Part II*. 649–679.

[15] Barak, B., F. G. S. L. Brandão, A. W. Harrow, J. A. Kelner, D. Steurer, and Y. Zhou (2012). "Hypercontractivity, sum-of-squares proofs, and their applications". In: *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19–22, 2012*. 307–326.

[16] Barak, B., S. O. Chan, and P. Kothari (2015a). "Sum of Squares Lower Bounds from Pairwise Independence". *CoRR*. abs/1501.00734.

[17] Barak, B., S. B. Hopkins, A. Jain, P. Kothari, and A. Sahai (2019). "Sum-of-Squares Meets Program Obfuscation, Revisited". In: *Advances in Cryptology — EUROCRYPT 2019 — 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19–23, 2019, Proceedings, Part I*. 226–250.

[18] Barak, B., S. B. Hopkins, J. A. Kelner, P. Kothari, A. Moitra, and A. Potechin (2016). "A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem". In: *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9–11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*. 428–437.

[19] Barak, B., J. A. Kelner, and D. Steurer (2013). "Rounding Sum-of-Squares Relaxations". *CoRR*. abs/1312.6652.

[20] Barak, B., J. A. Kelner, and D. Steurer (2015b). "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*. 143–151.

[21] Barak, B., P. K. Kothari, and D. Steurer (2017). "Quantum entanglement, sum of squares, and the log rank conjecture". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19–23, 2017*. 975–988.

[22] Barak, B. and A. Moitra (2016). "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*. 417–445.

[23] Barak, B., P. Raghavendra, and D. Steurer (2011). "Rounding Semidefinite Programming Hierarchies via Global Correlation". In: *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22–25, 2011*. 472–481.

[24] Barak, B. and D. Steurer (2014). "Sum-of-squares proofs and the quest toward optimal algorithms". *CoRR*. abs/1404.5236.

[25] Beame, P., R. Impagliazzo, J. Krajícek, T. Pitassi, and P. Pudlák (1994). "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*. 794–806.

[26] Beame, P., R. Impagliazzo, J. Krajícek, T. Pitassi, P. Pudlák, and A. R. Woods (1992). "Exponential Lower Bounds for the Pigeonhole Principle". In: *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4–6, 1992, Victoria, British Columbia, Canada*. 200–220.

[27] Beame, P. and T. Pitassi (1996). "Simplified and Improved Resolution Lower Bounds". In: *37th Annual Symposium on Foundations of Computer Science, FOCS '96, Burlington, Vermont, USA, 14–16 October, 1996*. 274–282.

[28] Beame, P. and T. Pitassi (2001). "Current Trends in Theoretical Computer Science". Ed. by B. Păun, G. Rozenberg, and A. Salomaa. River Edge, NJ, USA.

[29] Bellare, M., O. Goldreich, and M. Sudan (1998). "Free Bits, PCPs, and Nonapproximability-Towards Tight Results". *SIAM J. Comput.* 27(3): 804–915.

[30] Benabbas, S., K. Georgiou, A. Magen, and M. Tulsiani (2012). "SDP Gaps from Pairwise Independence". *Theory of Computing*. 8(1): 269–289.

[31] Ben-Sasson, E. and A. Wigderson (1999). "Short Proofs are Narrow — Resolution Made Simple". In: *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, May 1–4, 1999, Atlanta, Georgia, USA*. 517–526.

[32] Berkholz, C. (2017). "The Relation between Polynomial Calculus, Sherali-Adams, and Sum-of-Squares Proofs". *Electronic Colloquium on Computational Complexity (ECCC)*. 24: 154.

[33] Bertsimas, D. and J. N. Tsitsiklis (1997). *Introduction to Linear Optimization*. Vol. 6. Athena Scientific Belmont, MA.

[34] Bonet, M. L., C. Domingo, R. Gavaldà, A. Maciel, and T. Pitassi (2004). "Non-Automatizability of Bounded-Depth Frege Proofs". *Computational Complexity*. 13(1–2): 47–68.

[35] Bonet, M. L., T. Pitassi, and R. Raz (2000). "On Interpolation and Automatization for Frege Systems". *SIAM J. Comput.* 29(6): 1939–1967.

[36] Buresh-Oppenheim, J., M. Clegg, R. Impagliazzo, and T. Pitassi (2002). "Homogenization and the polynomial calculus". *Computational Complexity*. 11(3-4): 91–108.

[37] Buss, S. R., D. Grigoriev, R. Impagliazzo, and T. Pitassi (2001). "Linear Gaps between Degrees for the Polynomial Calculus Modulo Distinct Primes". *J. Comput. Syst. Sci.* 62(2): 267–289.

[38] Buss, S. R. and T. Pitassi (1998). "Good Degree Bounds on Nullstellensatz Refutations of the Induction Principle". *Journal of Computer and System Sciences*. 57(2): 162–171.

[39] Candès, E. J. and B. Recht (2009). "Exact Matrix Completion via Convex Optimization". *Foundations of Computational Mathematics*. 9(6): 717–772.

[40] Chan, S. O., J. R. Lee, P. Raghavendra, and D. Steurer (2016). "Approximate Constraint Satisfaction Requires Large LP Relaxations". *J. ACM*. 63(4): 34:1–34:22.

[41] Charikar, M., V. Guruswami, and A. Wirth (2017). "Clustering with Qualitative Information". In: *Encyclopedia of Machine Learning and Data Mining*. Springer US. 231.

[42] Charikar, M., K. Makarychev, and Y. Makarychev (2006). "Near-optimal algorithms for unique games". In: *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21–23, 2006*. 205–214.

[43] Charikar, M., K. Makarychev, and Y. Makarychev (2009). "Integrality gaps for Sherali-Adams relaxations". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31–June 2, 2009*. 283–292.

[44] Chawla, S., R. Krauthgamer, R. Kumar, Y. Rabani, and D. Sivakumar (2006). "On the Hardness of Approximating Multicut and Sparsest-Cut". *Computational Complexity*. 15(2): 94–114.

[45] Chlamtac, E., K. Makarychev, and Y. Makarychev (2006). "How to Play Unique Games Using Embeddings". In: *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21–24 October 2006, Berkeley, California, USA, Proceedings*. 687–696.

[46] Chvátal, V. (1977). "Determining the Stability Number of a Graph". *SIAM J. Comput.* 6(4): 643–662.

[47] Chvátal, V. and E. Szemerédi (1988). "Many Hard Examples for Resolution". *J. ACM*. 35(4): 759–768.

[48] Clegg, M., J. Edmonds, and R. Impagliazzo (1996a). "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*. 174–183.

[49] Clegg, M., J. Edmonds, and R. Impagliazzo (1996b). "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*. 174–183.

[50] Cook, S. A. and R. A. Reckhow (1979). "The Relative Efficiency of Propositional Proof Systems". *J. Symb. Log.* 44(1): 36–50.

[51] Dantchev, S. S., B. Martin, and M. N. C. Rhodes (2009). "Tight rank lower bounds for the Sherali-Adams proof system". *Theor. Comput. Sci.* 410(21–23): 2054–2063.

[52] Davis, M., G. Logemann, and D. W. Loveland (1962). "A machine program for theorem-proving". *Commun. ACM.* 5(7): 394–397.

[53] Davis, M. and H. Putnam (1960). "A Computing Procedure for Quantification Theory". *J. ACM.* 7(3): 201–215.

[54] Forbes, M. A., A. Shpilka, I. Tzameret, and A. Wigderson (2016). "Proof Complexity Lower Bounds from Algebraic Circuit Complexity". In: *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*. 32:1–32:17.

[55] Galesi, N. and M. Lauria (2010). "On the Automatizability of Polynomial Calculus". *Theory Comput. Syst.* 47(2): 491–506.

[56] Garg, A., M. Göös, P. Kamath, and D. Sokolov (2018). "Monotone circuit lower bounds from resolution". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25–29, 2018*. 902–911.

[57] Georgiou, K., A. Magen, T. Pitassi, and I. Tourlakis (2010). "Integrality Gaps of 2-o(1) for Vertex Cover SDPs in the Lov[a-acute]sz–Schrijver Hierarchy". *SIAM J. Comput.* 39(8): 3553–3570.

[58] Goemans, M. X. (1997). "Semidefinite programming in combinatorial optimization". *Math. Program.* 79: 143–161.

[59] Goemans, M. X. and D. P. Williamson (1994). ".879approximationn Algorithms for MAX CUT and MAX 2SAT". In: *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing. STOC '94*. Montreal, Quebec, Canada: ACM. 422–431.

[60] Göös, M., S. Lovett, R. Meka, T. Watson, and D. Zuckerman (2016). "Rectangles are Nonnegative Juntas". *SIAM J. Comput.* 45(5): 1835–1869.

[61] Göös, M., T. Pitassi, and T. Watson (2015). "Deterministic Communication vs. Partition Number". In: *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17–20 October, 2015*. 1077–1088.

[62] Göös, M., T. Pitassi, and T. Watson (2017). "Query-to-Communication Lifting for BPP". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15–17, 2017*. 132–143.

[63] Grigoriev, D. (1998). "Tseitin's Tautologies and Lower Bounds for Nullstellensatz Proofs". In: *39th Annual Symposium on Foundations of Computer Science, FOCS '98, November 8–11, 1998, Palo Alto, California, USA*. 648–652.

[64] Grigoriev, D. (2001). "Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity". *Theor. Comput. Sci.* 259(1–2): 613–622.

[65] Grigoriev, D. and E. A. Hirsch (2003). "Algebraic proof systems over formulas". *Theor. Comput. Sci.* 303(1): 83–102.

[66] Grigoriev, D. and N. Vorobjov (2001). "Complexity of Null-and Positivstellensatz proofs". *Ann. Pure Appl. Logic*. 113: 153–160.

[67] Grochow, J. A. and T. Pitassi (2018). "Circuit Complexity, Proof Complexity, and Polynomial Identity Testing: The Ideal Proof System". *J. ACM*. 65(6): 37:1–37:59.

[68] Grötschel, M., L. Lovász, and A. Schrijver (1981). "The ellipsoid method and its consequences in combinatorial optimization". *Combinatorica*. 1(2): 169–197.

[69] Grötschel, M., L. Lovász, and A. Schrijver (2012). *Geometric algorithms and combinatorial optimization*. Vol. 2. Springer Science & Business Media.

[70] Guruswami, V. and A. K. Sinop (2011). "Lasserre Hierarchy, Higher Eigenvalues, and Approximation Schemes for Graph Partitioning and Quadratic Integer Programming with PSD Objectives". In: *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22–25, 2011*. 482–491.

[71] Haken, A. (1985). "The Intractability of Resolution". *Theor. Comput. Sci.* 39: 297–308.

[72] Håstad, J. (2001). "Some optimal inapproximability results". *J. ACM*. 48(4): 798–859.

[73] Hilbert, D. (1888). "Ueber die Darstellung definiter Formen als Summe von Formenquadraten". *Mathematische Annalen*. 32(3): 342–350.

[74] Hopkins, S. B., P. K. Kothari, A. Potechin, P. Raghavendra, T. Schramm, and D. Steurer (2017). "The Power of Sum-of-Squares for Detecting Hidden Structures". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15–17, 2017*. 720–731.

[75] Hopkins, S. B., P. Kothari, A. H. Potechin, P. Raghavendra, and T. Schramm (2018). "On the Integrality Gap of Degree-4 Sum of Squares for Planted Clique". *ACM Trans. Algorithms*. 14(3): 28:1–28:31.

[76] Hopkins, S. B. and J. Li (2018). "Mixture models, robustness, and sum of squares proofs". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25–29, 2018*. 1021–1034.

[77] Hurwitz, A. (1891). "Ueber den vergleich des arithmetischen und des geometrischen mit-tels." *Journal fur die reine und angewandte Mathematik*.

[78] Impagliazzo, R., P. Pudlák, and J. Sgall (1999). "Lower Bounds for the Polynomial Calculus and the Gröbner Basis Algorithm". *Computational Complexity*. 8(2): 127–144.

[79] Iudin, D. and A. Nemirovskii (1976). "Informational complexity and effective methods of solution for convex extremal problems. Matekon: Translations of Russian and East European Math". *Economics*. 13: 3–25.

[80] Josz, C. and D. Henrion (2016). "Strong duality in Lasserre's hierarchy for polynomial optimization". *Optimization Letters*. 10: 3–10.

[81] Karchmer, M. and A. Wigderson (1993). "On Span Programs". In: *Proceedings of the Eigth Annual Structure in Complexity Theory Conference, San Diego, CA, USA, May 18-21, 1993*. 102–111.

[82] Karmarkar, N. (1980). "A Polynomial-Time Algorithm for Solving Linear Programs". *Math. Oper. Res.* 5(1): iv–iv.

[83] Karp, R. (1972). "Reducibility among combinatorial problems". In: *Complexity of Computer Computations*. Ed. by R. Miller and J. Thatcher. Plenum Press. 85–103.

[84] Khachiyan, L. G. (1979). "A polynomial algorithm in linear programming". In: *Doklady Academii Nauk SSSR*. Vol. 244. 1093–1096.

[85] Khachiyan, L. (1980). "Polynomial algorithms in linear programming". *USSR Computational Mathematics and Mathematical Physics*. 20(1): 53–72.

[86] Khot, S. (2002). "On the Power of Unique 2-Prover 1-Round Games". In: *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21–24, 2002*. 25.

[87] Khot, S. (2010). "On the Unique Games Conjecture (Invited Survey)". In: *Proceedings of the 2010 IEEE 25th Annual Conference on Computational Complexity. CCC '10*. Washington, DC, USA: IEEE Computer Society. 99–121.

[88] Khot, S., G. Kindler, E. Mossel, and R. O'Donnell (2004). "Optimal Inapproximability Results for Max-Cut and Other 2-Variable CSPs?" In: *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science. FOCS '04*. Washington, DC, USA: IEEE Computer Society. 146–154.

[89] Khot, S., G. Kindler, E. Mossel, and R. O'Donnell (2007). "Optimal Inapproximability Results for MAX-CUT and Other 2-Variable CSPs?" *SIAM J. Comput.* 37(1): 319–357.

[90] Khot, S., D. Minzer, and M. Safra (2018). "Pseudorandom Sets in Grassmann Graph have Near-Perfect Expansion". *Electronic Colloquium on Computational Complexity (ECCC)*. 25: 6.

[91] Khot, S. and O. Regev (2008). "Vertex cover might be hard to approximate to within 2-epsilon". *J. Comput. Syst. Sci.* 74(3): 335–349.

[92] Khot, S. and N. K. Vishnoi (2015). "The Unique Games Conjecture, Integrality Gap for Cut Problems and Embeddability of Negative-Type Metrics into l1". *J. ACM*. 62(1): 8:1–8:39.

[93] Klivans, A. R., P. K. Kothari, and R. Meka (2018). "Efficient Algorithms for Outlier-Robust Regression". In: *Conference On Learning Theory, COLT 2018, Stockholm, Sweden, 6-9 July 2018*. 1420–1430.

[94] Kothari, P. K., R. Meka, and P. Raghavendra (2017a). "Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19–23, 2017*. 590–603.

[95] Kothari, P. K., R. Mori, R. O'Donnell, and D. Witmer (2017b). "Sum of squares lower bounds for refuting any CSP". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19–23, 2017*. 132–145.

[96] Kothari, P. K. and J. Steinhardt (2017). "Better Agnostic Clustering Via Relaxed Tensor Norms". *CoRR*. abs/1711.07465.

[97] Kothari, P. K. and D. Steurer (2017). "Outlier-robust moment-estimation via sum-of-squares". *CoRR*. abs/1711.11581.

[98] Kothari, P., R. Meka, and P. Raghavendra (2016). "Approximating Rectangles by Juntas and Weakly-Exponential Lower Bounds for LP Relaxations of CSPs". *CoRR*. abs/1610.02704.

[99] Krajícek, J. and P. Pudlák (1998). "Some consequences of cryptographical conjectures for S^1_2 and EF". *Inf. Comput.* 140(1): 82–94.

[100] Krivine, J.-L. (1964). "Anneaux préordonnés." *Journal d'Analyse Mathématique*. 12(1): 307–326.

[101] Lasserre, J. B. (2001a). "An Explicit Exact SDP Relaxation for Nonlinear 0-1 Programs". In: *Integer Programming and Combinatorial Optimization*. Ed. by K. Aardal and B. Gerards. Berlin, Heidelberg: Springer Berlin Heidelberg. 293–303.

[102] Lasserre, J. B. (2001b). "Global Optimization with Polynomials and the Problem of Moments". *SIAM Journal on Optimization*. 11(3): 796–817.

[103] Lewin, M., D. Livnat, and U. Zwick (2002). "Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems". In: *International Conference on Integer Programming and Combinatorial Optimization*. Springer. 67–82.

[104] Li, F., I. Tzameret, and Z. Wang (2015). "Non-Commutative Formulas and Frege Lower Bounds: a New Characterization of Propositional Proofs". In: *30th Conference on Computational Complexity, CCC 2015, June 17-19, 2015, Portland, Oregon, USA*. 412–432.

[105] Linial, N. (2002). "Finite metric spaces: combinatorics, geometry and algorithms". In: *Proceedings of the 18th Annual Symposium on Computational Geometry, Barcelona, Spain, June 5-7, 2002*. 63.

[106] Lovász, L. (2003). "Semidefinite Programs and Combinatorial Optimization". In: *Recent Advances in Algorithms and Combinatorics*. Ed. by B. A. Reed and C. L. Sales. New York, NY: Springer New York. 137–194.

[107] Lovász, L. (1979). "On the Shannon capacity of a graph". *IEEE Trans. Information Theory*. 25(1): 1–7.

[108] Lovász, L. (1995). "Semidefinite programs and combinatorial optimization".

[109] Ma, T., J. Shi, and D. Steurer (2016). "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*. 438–446.

[110] Mathieu, C. and A. Sinclair (2009). "Sherali-adams relaxations of the matching polytope". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31–June 2, 2009*. 293–302.

[111] Meka, R., A. Potechin, and A. Wigderson (2015). "Sum-of-squares Lower Bounds for Planted Clique". In: *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14–17, 2015*. 87–96.

[112] Motzkin, T. S. (1967). "The arithmetic-geometric inequality". *Inequalities (Proc. Symposium on Inequalities, edited by O. Shisha, Academic Press*: 205–224.

[113] Nordström, J. (2015). "On the interplay between proof complexity and SAT solving". *SIGLOG News*. 2(3): 19–44.

[114] O'Donnell, R. (2017). "SOS Is Not Obviously Automatizable, Even Approximately". In: *8th Innovations in Theoretical Computer Science Conference, ITCS 2017, January 9–11, 2017, Berkeley, CA, USA*. 59:1–59:10.

[115] Parrilo, P. A. (2000). "Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization". *PhD thesis*. California Institute of Technology.

[116] Parrilo, P. A. (2003). "Semidefinite programming relaxations for semialgebraic problems". *Mathematical programming*. 96(2): 293–320.

[117] Pitassi, T. (1996). "Algebraic Propositional Proof Systems". In: *Descriptive Complexity and Finite Models, Proceedings of a DIMACS Workshop 1996, Princeton, New Jersey, USA, January 14–17, 1996*. 215–244.

[118] Pitassi, T. and R. Robere (2018). "Lifting nullstellensatz to monotone span programs over any field". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25–29, 2018*. 1207–1219.

[119] Pitassi, T. and N. Segerlind (2012). "Exponential Lower Bounds and Integrality Gaps for Tree-Like Lovász-Schrijver Procedures". *SIAM J. Comput.* 41(1): 128–159.

[120] Pitassi, T. and I. Tzameret (2016). "Algebraic proof complexity: progress, frontiers and challenges". *SIGLOG News*. 3(3): 21–43.

[121] Potechin, A. and D. Steurer (2017). "Exact tensor completion with sum-of-squares". In: *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*. 1619–1673.

[122] Putinar, M. (1993). "Positive polynomials on compact semi-algebraic sets". *Indiana University Mathematics Journal*. 42(3): 969–984.

[123] Raghavendra, P. (2008). "Optimal algorithms and inapproximability results for every CSP?" In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17–20, 2008*. 245–254.

[124] Raghavendra, P. and D. Steurer (2010). "Graph expansion and the unique games conjecture". In: *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*. 755–764.

[125] Raghavendra, P. and B. Weitz (2017). "On the Bit Complexity of Sum-of-Squares Proofs". In: *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*. 80:1–80:13.

[126] Ramana, M., L. Tunçel, and H. Wolkowicz (1997). "Strong Duality for Semidefinite Programming". *SIAM Journal on Optimization*. 7(3): 641–662.

[127] Ramana, M. V. (1997). "An exact duality theory for semidefinite programming and its complexity implications". *Mathematical Programming*. 77(1): 129–162.

[128] Raz, R. and P. McKenzie (1999). "Separation of the Monotone NC Hierarchy". *Combinatorica*. 19(3): 403–435.

[129] Razborov, A. A. (1998). "Lower Bounds for the Polynomial Calculus". *Computational Complexity*. 7(4): 291–324.

[130] Razborov, A. A. (2016). "Guest Column: Proof Complexity and Beyond". *SIGACT News*. 47(2): 66–86.

[131] Recht, B. (2011). "A Simpler Approach to Matrix Completion". *Journal of Machine Learning Research*. 12: 3413–3430.

[132] Regev, O. and A. Vijayaraghavan (2017). "On Learning Mixtures of Well-Separated Gaussians". In: *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15–17, 2017*. 85–96.

[133] Reznick, B. (1996). "Some Concrete Aspects Of Hilbert's 17th Problem". In: *In Contemporary Mathematics*. American Mathematical Society. 251–272.

[134] Rothvoß, T. (2013). "The Lasserre hierarchy in approximation algorithms". *Lecture Notes for the MAPSP*: 1–25.

[135] Sahni, S. and T. Gonzalez (1976). "P-complete approximation problems". *Journal of the ACM (JACM)*. 23(3): 555–565.

[136] Scheiderer, C. (2003). "Sums of squares on real algebraic curves". *Mathematical journal*. 245(4): 725–760.

[137] schmidt, K. (2012). "Around Hilbert's 17th problem". *Documenta Mathematica, Optimization stories, extra volume ISMP*: 433–438.

[138] Schoenebeck, G. (2008). "Linear Level Lasserre Lower Bounds for Certain k-CSPs". In: *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*. 593–602.

[139] Segerlind, N. (2007). "The Complexity of Propositional Proofs". *Bulletin of Symbolic Logic*. 13(4): 417–481.

[140] Sherali, H. D. and W. P. Adams (1994). "A Hierarchy of Relaxations and Convex Hull Characterizations for Mixed-integer Zero-one Programming Problems". *Discrete Applied Mathematics*. 52(1): 83–106.

[141] Sherstov, A. A. (2011). "The Pattern Matrix Method". *SIAM J. Comput.* 40(6): 1969–2000.

[142] Shor, N. Z. (1987). "An approach to obtaining global extremums in polynomial mathematical programming problems". *Cybernetics*. 23(5): 695–700.

[143] Shor, N. Z. (1977). "Cut-off method with space extension in convex programming problems". *Cybernetics and systems analysis*. 13(1): 94–96.

[144] Stengle, G. (1994). "A Nullstellensatz and Positivstellensatz in semialgebraic geometry". *Math. Ann.* 207: 87–97.

[145] Trevisan, L. (2011). "On Khot's Unique Games Conjecture".

[146] Trnovská, M. (2005). "Strong duality conditions in semidefinite programming". *Journal of Electrical Engineering*. 56(12): 1–5.

[147] Tulsiani, M. (2009). "CSP gaps and reductions in the lasserre hierarchy". In: *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31–June 2, 2009*. 303–312.

[148] Urquhart, A. (1987). "Hard examples for resolution". *J. ACM*. 34(1): 209–219.

[149] Wenceslas Fernandez de la Vega and Claire Kenyon-Mathieu (2007). "Linear programming relaxations of maxcut". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007, New Orleans, Louisiana, USA, January 7-9, 2007*. 53–61.

[150] Yannakakis, M. (1988). "Expressing Combinatorial Optimization Problems by Linear Programs (Extended Abstract)". In: *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*. 223–228.

Index