KTH ROYAL INSTITUTE
OF TECHNOLOGY

Doctoral Thesis in Computer Science

On Long Proofs of Simple Truths

KILIAN RISSE

Stockholm, Sweden 2022

On Long Proofs of Simple Truths

KILIAN RISSE

Academic Dissertation which, with due permission of the KTH Royal Institute of Technology, is submitted for public defence for the Degree of Doctor of Philosophy on Thursday the 27th of October 2022, at 2:00 p.m. in F3, Lindstedtsvägen 26, Stockholm

Doctoral Thesis in Computer Science
KTH Royal Institute of Technology
Stockholm, Sweden 2022

ISBN 978-91-8040-354-2
TRITA-EECS-AVL-2022:55

Printed by: Universitetsservice US-AB, Sweden 2022

Abstract

Propositional proof complexity is the study of certificates of infeasibility. In this thesis we consider several proof systems with limited deductive ability and unconditionally show that they require long refutations of the feasibility of certain Boolean formulas.

We show that the depth d Frege proof system, restricted to line size M, requires proofs of length at least $\exp\left(n/(\log M)^{O(d)}\right)$ to refute the Tseitin contradiction defined over the $n \times n$ grid graph, improving upon the recent result of Pitassi et al. [PRT22]. Along the way we also sharpen the lower bound of Håstad [Hås20] on the depth d Frege refutation size for the same formula from exponential in $\tilde{\Omega}(n^{1/59d})$ to exponential in $\tilde{\Omega}(n^{1/(2d-1)})$.

We also consider the perfect matching formula defined over a sparse random graph on an odd number of vertices n. We show that polynomial calculus over fields of characteristic $\neq 2$ and sum of squares require size exponential in $\Omega(n/\log^2 n)$ to refute said formula. For depth d Frege we show that there is a constant $\delta > 0$ such that refutations of these formulas require size $\exp\left(\Omega(n^{\delta/d})\right)$.

The perfect matching formula has a close sibling over bipartite graphs: the graph pigeonhole principle. There are two methods to prove resolution refutation size lower bounds for the pigeonhole principle. On the one hand there is the general width-size tradeoff by Ben-Sasson and Wigderson [BW01] which can be used to show resolution refutation size lower bounds in the setting where we have a sparse bipartite graph with n holes and $m \ll n^2$ pigeons. On the other hand there is the pseudo-width technique developed by Razborov [Raz04] that applies for any number of pigeons, but requires the graph to be somewhat dense. We extend the latter technique to also cover the previous setting and more: for example, it has been open whether the functional pigeonhole principle defined over a random bipartite graph of bounded degree and $\text{poly}(n) \geq n^2$ pigeons requires super-polynomial size resolution refutations. We answer this and related questions.

Finally we also study the circuit tautology which claims that a Boolean function has a circuit of size s computing it. For $s = \text{poly}(n)$ we prove an essentially optimal Sum of Squares degree lower bound of $\Omega(s^{1-\varepsilon})$ to refute this claim for any Boolean function. Further, we show that for any $0 < \alpha < 1$ there are Boolean functions on n bits with circuit complexity larger than 2^{n^α} but the Sum of Squares proof system requires size $2^{\left(2^{\Omega(n^\alpha)}\right)}$ to prove this. Lastly we show that these lower bounds can also be extended to the monotone setting.

Sammanfattning

Propositionell beviskomplexitet är studerar certifikat av icke-satisfierbarhet. Vi betraktar bevissystem med begränsad deduktiv förmåga och bevisar ovillkorliga undre gränser för längden på vederläggningar av formlers satisferbarhet. Denna avhandling bevisar flera nya sådana undre gränser för bevissystemen resolution, polynomialkalkyl, kvadratsummor, och Frege-system av begränsat djup.

Vi visar att Frege-systemet av djup d, begränsat till rader av storlek M, kräver minst bevis av längd minst $\exp\left(n/(\log M)^{O(d)}\right)$ för att motbevisa Tseitin-kontradiktionen definierad över $n \times n$-rutnätet, vilket förbättrar ett nyligen visat resultat av Pitassi et al. [PRT22]. Längs vägen skärper vi även Håstads undre gräns [Hås20] för längd för Frege av djup d för samma formel från exponentiell i $\tilde{\Omega}(n^{1/59d})$ till exponentiell i $\tilde{\Omega}(n^{1/(2d-1)})$.

Vi betraktar också formeln för perfekt matchning över en gles slumpgraf med ett udda antal hörn n. Vi visar att polynomkalkyl över kroppar med karaktäristik $\neq 2$ och kvadratsummor kräver längd exponentiell i $\Omega(n/\log^2 n)$ för att motbevisa denna formel. För Frege av djup d visar vi att det finns en konstant $\delta > 0$ så att vederläggningar av dessa formler kräver storlek $\exp\left(\Omega(n^{\delta/d})\right)$.

Formeln för perfekt matchning har ett nära syskon över bipartita grafer: duvslagsprincipen över grafer. Det finns två metoder för att visa undre gränser för refutations för duvslagsprincipen. Å ena sidan finns Ben-Sasson och Wigdersons [BW01] generella avvägning mellan bredd och storlek som kan användas för att visa undre gränser för resolution i fallet där vi har en gles bipartit graf med n hål och $m \ll n^2$ duvor. Å andra sidan finns pseudo-bredd-tekniken utvecklad av Razborov [Raz04] som kan appliceras för valfritt antal duvor, men kräver att grafen är någorlunda tät. Vi utökar den senare tekniken till att även täcka det förstnämnda fallet och mer: till exempel har det varit öppet om den funktionella duvslagssprincipen definierad över en slumpmässig bipartit graf med begränsade gradtal och $\text{poly}(n) \geq n^2$ duvor kräver motbevis av superpolynomisk storlek. Vi besvarar detta och relaterade frågor.

Slutligen studerar vi också kretstautologin som hävdar att en Boolean funktion har en krets av storlek s som beräknar den. Vi bevisar en väsentligen optimal undre gräns för gradtal för kvadratsummor på $\Omega(s^{1-\varepsilon})$ för att motbevisa detta påstående för varje Boolesk funktion, för $s > \text{poly}(n)$. Vidare visar vi att det för alla $0 < \alpha < 1$ finns Booleska funktioner på n bitar med kretskomplexitet större än 2^{n^α} men där kvadratsummor kräver storlek $2^{\left(2^{\Omega(n^\alpha)}\right)}$ för att bevisa detta.

Acknowledgments

Foremost I want to thank all my advisors, in alphabetical order, Per Austrin, Johan Håstad and Jakob Nordström. They always stood by my side, never doubted me and seemingly had infinite patience to repeatedly explain the same concept to me. Without them this thesis would not exist.

Jakob, thank you for introducing me to proof complexity and for sharing your enthusiasm for this subject with me. I have fond memories of the trips to Lund and all the research discussions with you. Johan, thank you for always having an open door, patiently listening to me and all the thoughts you have shared with me. Per, I have learned a lot from you – about research but also about movies. Thank you for your good spirit and all the long afternoons discussing research.

In addition to my advisors I also have the honor to call Susanna de Rezende and Dmitry Sokolov my co-authors. It was, and is, a lot of fun to solve problems with you and I am looking forward to our continued collaboration. During my stay at KTH I further had the pleasure to collaborate with Aaron, Sagnik, Jonah, Joseph and Aleksa. Thank you for the time in front of a whiteboard with you.

Naemi, Mattias, Chris and Chris, Martin, Tom, Linda, Irena and Solomon, you guys made my stay in Sweden worthwhile – thank you! Special thanks to Naemi and Solomon who dared to share an apartment with me. Back in Switzerland I want to thank Bernhard, Adriano, Matthias, Simon, Daniel, Raphael, Leonie, Linh-Thu and Franziska for staying in touch with me even though I was mostly absent the past years. Kay, Florine I am so grateful to call you my friends. Thank you for always having an open door in Copenhagen.

My family has supported me throughout the last years despite fully understanding what I do. Thank you Mom and Dad for your generosity and patience, Anna, Pascal and Jannis for always having an empty bed for me, and Leon, Chanel, Christine, Dominique and Urs for all the discussions and the time that you chose to spend with me.

Last but not least, Patrycja, thank you for making my dark days brighter, for enduring my absent-mindedness and simply for being you.

Contents

Abstract iii

Sammanfattning v

Acknowledgments vii

Contents ix

I Thesis 1

1 Introduction 3
1.1 Proofs . . . 3
1.2 Efficient Proofs . . . 5

2 Background 11
2.1 Preliminaries . . . 11
2.2 Proof Systems . . . 16
2.3 Propositional Formulas . . . 21

3 Contributions 27
3.1 On Bounded Depth Frege Refutations of the Tseitin Formula 27
3.2 Average-Case Perfect Matching Lower Bounds . . . 29
3.3 The Circuit Tautology is Hard for Sum of Squares . . . 32
3.4 The Sparse Weak Pigeonhole Principle is Hard for Resolution 33

II Included Papers 45

A On Bounded Depth Frege Refutations of the Tseitin Formula 49
A.1 Introduction . . . 50
A.2 Preliminaries . . . 54
A.3 Properties of assignments on the grid and some games . . . 55

A.4 Restrictions . . . 65
A.5 Decision trees . . . 72
A.6 Basics for t-evaluations . . . 74
A.7 Proofs of the main theorems . . . 77
A.8 The improved standard switching lemma . . . 82
A.9 The multi-switching lemma . . . 101
A.10 Conclusion . . . 105

B Average-Case Perfect Matching Lower Bounds **111**
B.1 Introduction . . . 112
B.2 Preliminaries . . . 122
B.3 Lower Bounds on Average . . . 128
B.4 Proof of the Partition Lemma . . . 133
B.5 Embedding Theorem . . . 140
B.6 Concluding Remarks . . . 152
B.7 Worst-Case Lower Bounds . . . 155
B.8 Embedding Algorithm . . . 159

C The Circuit Tautology is Hard for Sum of Squares **173**
C.1 Introduction . . . 174
C.2 Preliminaries . . . 179
C.3 On Circuits and Restrictions . . . 185
C.4 Lower Bounds for General Circuits . . . 189
C.5 Lower Bounds for Monotone Circuits . . . 193
C.6 Degree Upper Bound . . . 198
C.7 On Encodings of the $\text{Circuit}_s(f)$ Tautology . . . 202
C.8 Concluding Remarks . . . 206

D The Sparse Weak Pigeonhole Principle is Hard for Resolution **213**
D.1 Introduction . . . 214
D.2 Preliminaries . . . 222
D.3 Two Key Technical Tools . . . 226
D.4 Lower Bounds for Weak Graph FPHP Formulas . . . 230
D.5 Lower Bounds for Perfect Matching Principle Formulas . . . 240
D.6 Concluding Remarks . . . 259

Part I

Thesis

Introduction

This chapter is a non-technical introduction to the thesis intended for readers with little or no mathematical background. All notions informally introduced in this chapter are revisited in Chapter 2.

1.1 Proofs

This thesis is about proofs and contains proofs proving properties of proofs. But before talking more about proofs (and proofs about proofs) it might be worthwhile to take a step back and think about what a proof really is – after all it does seem to be a central concept of this thesis.

Broadly speaking, a proof is an object that, hopefully, convinces others of some claim. Depending on the situation a proof may have very different form: sometimes a paper from an authority may be sufficient, while in other situations, like in court, a proof has to be convincing enough so that the judge believes it "beyond reasonable doubt". And even in mathematics itself there are several notions of a proof. For example a proof published in an article intended to convince other mathematicians of the validity of a theorem is usually not very formal. Well-known steps may be skipped, some statements may be left to prove by the reader and sometimes there are even oversights by the authors. As such it is not surprising that, once in a while, mathematicians make mistakes and therefore have to retract articles.

Mathematicians could prevent this from happening – since the late 19th century we know of proof systems that produce machine verifiable proofs. These proofs are great to verify statements and make sure that they are indeed correct. However, fully formalizing a proof is very time consuming, they are usually not human readable and cannot convey the intuition of a proof. As such these formal proofs are an interesting object to study but not very useful to communicate proofs to other human beings.

The first formal proof system has been put forward by Frege [Fre79] in 1879 and these systems have later been popularized in the 1920s by Hilbert when he proposed his program to base mathematics on a solid

foundation. Hilbert's ultimate goal was to base mathematics on a logic basis: he wanted to reprove all mathematical statements from a finite number of assumptions (also known as *axioms*) such that (i) these assumptions do not contradict each other, and (ii) all true statements can be proven from these assumptions. This is a seemingly desirable goal if one believes that mathematics describes universal truth.

However, this turns out to be impossible: in 1931 Gödel [Göd31] proved that strong enough proof systems, as studied by Hilbert and others, are either self-contradicting or they cannot prove the statement that the system itself can prove all correct statements. In other words, we cannot hope for a single set of non-contradicting axioms $\mathcal{A}$ that can prove the statement "all true consequences can be derived from $\mathcal{A}$". This theorem, known as Gödel's incompleteness theorem, left Hilbert's original program in shambles.

On the upside, this led other mathematicians to study related problems and ask more refined questions about the existence of proofs. We just learned that we cannot prove everything. Can we at least determine what statements can be proven? That is, given some statement P, can we decide whether the claim P can be proven from our favorite set of axioms? Even this turns out to be too much to ask for. But before discussing this we should take a small detour explaining what we mean by "we can tell" that a statement has a proof – what kind of machine are we allowed to use to determine whether there is a proof of a statement? Mathematicians put forward different formalisms in the 30s of the last century, one of them being the Turing machine. This formal machine, proposed by Turing, intends to be able to perform any task that could also be done "by pen and paper", or, in more modern terms, by an algorithm. In 1936 Turing [Tur37], and independently Church [Chu36] using a different albeit equivalent formalism, showed that no algorithm running in finite time can determine whether a statement can be proven from a given set of axioms.

To summarize, there are correct statements with no proof and, even worse, we cannot tell whether a given statement has a proof. Now, I cannot blame you if your mind has started to wander and you find this a quite lofty and dry discussion. So let me give you a very concrete real-life consequence. We all use computers, smartphones or even smartwatches in our daily lives. Would it not be great if we could write an algorithm that ensures that all these devices never crash? No need to ever reboot your computer because — reasons? Well, you may have guessed, this is unfortunately one of these so-called undecidable statements. Bear this in mind the next time you call your computer person – they are trying hard to solve an undecidable problem. Have some patience, get a coffee, and, in the mean time, reboot

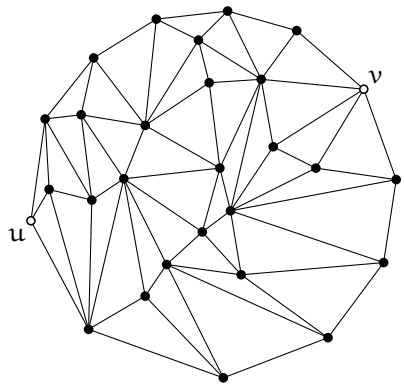

Figure 1.1: A graph with two labeled nodes u and v

your computer.

But joking aside, it really seems like we have to give up on these undecidable statements. Maybe there is more to discover in the set of decidable statements? For example, do such statements always have small proofs? Can small proofs be found efficiently? These, and related questions, are the foundations of theoretical computer science.

1.2 Efficient Proofs

From now on we only study claims that are decidable, i.e., have a proof, though it may be long. Let us start with a simple example of the kind of statements that we want to consider. Suppose we are given a graph with two labeled nodes u, v and the claim that these two nodes are connected by a path of length 5. An illustration of an example graph can be found in Figure 1.1.

Is this claim correct? Staring at the graph for a bit one can actually find a path of length 5, as illustrated in Figure 1.2 on the following page. This illustration is a pretty convincing proof – that was not so hard to prove. What about the claim that there is no path of length at most 4 connecting u to v? Can we convince ourselves that this holds? We could try to enumerate all paths between u and v and check that there is no shorter path. This would indeed give us a valid proof, but it seems cumbersome as there are so many paths connecting u to v.

Let us try to create a more concise proof, as follows. Find all nodes at distance at most one from u. This is not difficult: this is u and all neighbors $N(u)$ of u. Once we have found the nodes V_1 at distance at most 1 from u,

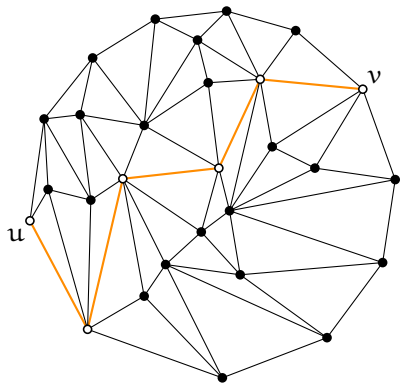

Figure 1.2: A path of length 5

we can now find all nodes at distance at most two from u: these are the nodes of V_1 as well as the neighborhood $N(V_1)$ of it. Iterating this idea, we see that v is not in the set of nodes V_4 at distance at most 4 from u. Thus we proved the claim. This proof is illustrated in Figure 1.3 on the following page.

Observe that this argument gives us an algorithm to determine the length of the shortest path between u and v: simply run the described procedure until v is, for the first time, in the set V_i and report that the shortest path between u and v is of distance i. Claims that have an efficient algorithm, as the one just described, make up the class P. That is, P consists of all claims that can be efficiently proven *and*, furthermore, this short proof can also be found efficiently.

Let us consider a claim that we suspect is of a different nature: the claim that a graph contains a cycle visiting every node exactly once (this is known as a Hamiltonian cycle). If you have the stamina, you can stare at the graph in Figure 1.1 for a while and you will notice that there is indeed a Hamiltonian cycle – one example is highlighted in Figure 1.4 on the following page. Again, this illustration is a short, efficient proof similar to the proof of the existence of a path of length 5 in Figure 1.2. So far there is no evidence that would justify calling this claim of a different nature.[1] It gets more interesting if we consider the negation of the previous claim, namely the claim that "there is no Hamiltonian cycle". Suppose our graph does not contain a Hamiltonian cycle. How would we prove this? Of course there is the brute-force proof: enumerate each ordering of nodes

[1] Here we ignore the question whether such proofs can be found efficiently. We discuss this question in detail later on.

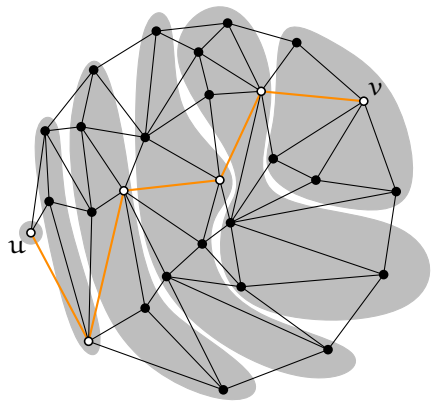

Figure 1.3: The nodes at distance $0, 1, 2, 3, 4$ and 5 from u

and rule out that this ordering gives rise to a Hamiltonian cycle. If we consider a graph on 28 nodes, as in Figure 1.1, then this brute-force proof would consist of about $28! \approx 10^{29}$ many orderings. To get a feeling for this number, suppose that we are really quick at checking orderings, say, we can check a million orderings a second. In this case it would take us a mere 10^{15} years to check all orderings – as the universe is only about 10^{10} years old this is not an awfully useful proof.

If the naïve proof of such a small graph is already this long, we have to investigate whether there are more efficient proofs of this claim. So far we have not succeeded in this endeavor and it is generally believed that there are no proofs significantly shorter than our naïve proof.[2] And this is by far the only statement for which we do not have short proofs – there is an entire cluster of statements that seem impossible to prove efficiently. In order to discuss this in the language of theoretical computer science we need to introduce two classes of statements.

The first class is called NP and consists of all statements that have an efficient proof, e.g., "there is a Hamiltonian cycle" or "there is no path of length 4". The second class is coNP which consists of all statements whose negation has an efficient proof: for example, "there is *no* Hamiltonian cycle" or "there is no path of length 4". Already from the given examples we see that NP and coNP have certain statements in common. The big open question is whether *all* claims of coNP are also in NP, or in words, whether the negation of efficiently verifiable statements also have short proofs. It is widely believed that this is not the case: it is believed that there are claims in

[2] It should be mentioned that there are *slightly* more efficient proofs. If a graph has n nodes, then proofs can be made of length roughly 2^n instead of the $n! \approx 2^{n \log n}$ length our naïve proof has.

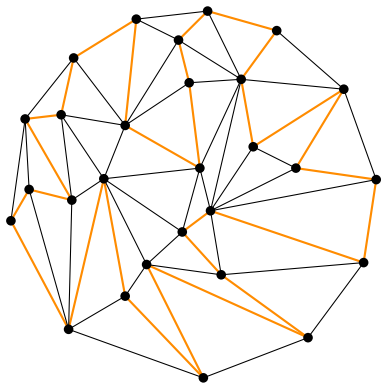

Figure 1.4: A Hamiltonian cycle

coNP, for example, "there is no Hamiltonian cycle", that inherently require long proofs.

This thesis continues a line of work that eventually hopes to separate NP from coNP. The program, put forward by Cook and Reckhow [CR79], suggests to study increasingly more powerful proof systems and prove that these systems require long proofs for some claims in coNP. Starting with "weak" proof systems, i.e., proof systems of limited deductive ability, we hope to develop a toolbox of lower bound techniques that can be applied to increasingly stronger proof systems. Apart from its intrinsic interest, separating NP from coNP would have some interesting consequences. Let us explain.

By definition we have that P is contained in NP, as all claims in P have short proofs. Also, it is not so hard to convince yourself that a claim C is in P if and only if the claim "not C" is also in P: as in the path example, we can run the efficient algorithm that finds a proof of C. By definition, this algorithm is complete, meaning that it always outputs a proof if one exists. Thus running the efficient algorithm and obtaining no proof from it is itself a short proof of the claim "not C". As such we conclude that P lies in the intersection of coNP and NP. Equivalently we can write this in symbols as $\mathsf{P} \subseteq \mathsf{NP} \cap \mathsf{coNP}$.

Now suppose that we can separate NP from coNP. We claim that this implies that P is a strict subset of NP: because P is closed under complementation but NP is not, there has to be some statement that is in NP but not in P. To determine whether $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$, or equivalently whether a claim with a short proof also has an efficient algorithm that recovers the short proof, is one of the seven millennium problems put forward in 2000

by the Clay Mathematics Institute [CJW06].

As previously mentioned, this thesis continues the program of Cook and Reckhow with the eventual goal to separate NP from coNP. We prove several new proof size lower bounds for different formulas and proof systems. We achieve this by adapting and extending known lower bound techniques, thereby expanding the current toolbox of lower bound techniques for proof complexity. In Chapter 2 we cover the necessary background to then discuss our contributions in Chapter 3. No formal proofs are covered in the first part of the thesis. The proofs of the mentioned theorems can be found in Part II which contains all the discussed papers.

Background

In the first section of this chapter we revisit some well-known notions from complexity theory as well as some basic graph theory. In Section 2.2 we introduce the relevant proof systems and in Section 2.3 the propositional formulas that we intend to study.

2.1 Preliminaries

For an integer $n \in \mathbb{N}$ we let $[n] = \{1, \dots, n\}$ denote the set of integers from 1 through n, and we let $\log n$ denote the logarithm of n to the base 2.

Let $f, g : \mathbb{R} \to \mathbb{R}$ be two functions. We write $f(n) \in O\big(g(n)\big)$, respectively $f(n) \in \Omega\big(g(n)\big)$, if there is a constant c and an n_0 such that for all $n \geq n_0$ it holds that $f(n) \leq c \cdot g(n)$, respectively $f(n) \geq c \cdot g(n)$. We say that f is *poly-bounded* if there is a constant c such that $f \in O(n^c)$. We further introduce the notation $f(n) \in o\big(g(n)\big)$ to mean that for all constants $c > 0$ there is an n_0 such that $f(n) \leq c \cdot g(n)$, for all $n \geq n_0$.

Similarly we sometimes want to supress dependencies on constants and write $f(n, \varepsilon) \in O_\varepsilon\big(g(n, \varepsilon)\big)$, respectively $f(n, \varepsilon) \in \Omega_\varepsilon\big(g(n, \varepsilon)\big)$, to mean that there exists a function $c(\varepsilon) > 0$ and a constant n_0 such that for all $n \geq n_0$ it holds that $f(n, \varepsilon) \leq c(\varepsilon) \cdot g(n, \varepsilon)$, respectively $f(n, \varepsilon) \geq c(\varepsilon) \cdot g(n, \varepsilon)$. In a similar vein we sometimes even want to suppress logarithmic dependencies: we write $f(n) \in \tilde{O}\big(g(n)\big)$ to mean that there are constants c and n_0 such that for all $n \geq n_0$ it holds that $f(n) \leq \log^c(n) \cdot g(n)$, and $f(n) \in \tilde{\Omega}\big(g(n)\big)$ if there are constants c and n_0 such that $f(n) \geq \log^c(n) \cdot g(n)$, for all $n \geq n_0$.

2.1.1 Graph Theory

Let us recall some standard graph terminology. A *graph* $G = (V, E)$ is a 2-tuple that consists of a set of vertices $V = V(G)$ and a set of edges $E = E(G) \subseteq V \times V$, and a *bipartite graph* $G = (U, V, E)$ is a 3-tuple that consists of two disjoint vertex sets U and V, with $V(G) = U \cup V$, and an edge set $E = E(G) \subseteq (U \times V) \cup (V \times U)$.

We only consider simple and undirected graphs; all graphs contain no self-loops, i.e., edges (u, u), and if $(u, v) \in E$, then also $(v, u) \in E$ and we may thus think of the edge set E as containing sets of size 2.

The neighborhood of a vertex $u \in V(G)$ is $N(u) = \{v \mid \{u, v\} \in E\}$, the neighborhood of a set of vertices $U \subseteq V(G)$ is $N(U) = \bigcup_{u \in U} N(u)$ and for sets $U, W \subseteq V(G)$ the neighborhood of U in W is $N(U, W) = N(U) \cap W$. We denote by $\deg(v) = |N(v)|$ the degree of a vertex $v \in V$, by $\Delta(G)$ the maximum degree, $\delta(G)$ the minimum degree and by $d(G)$ the average degree of G. For a set of vertices $U \subseteq V(G)$ we let $G[U]$ denote the graph induced by U, i.e., $G[U] = (U, E_U)$, where $E_u = \{e \in E(G) \mid e \subseteq U\}$.

A graph $G = (V, E)$ on n vertices is an *α-expander* (has *vertex expansion* α) if for all sets $U \subseteq V$ of size $|U| \leq n/2$ it holds that $|N(U, V \setminus U)| \geq \alpha|U|$.

The *grid graph* (more commonly *torus*) $G = (V, E)$ of dimension n consists of vertices $V = \{(i, j) \mid 0 \leq i, j < n\}$ and the vertex (i, j) is connected by edges to the four neighbors at distance 1, i.e., where one coordinate is identical and the other changes by ± 1 modulo n.

We denote the *uniform distribution over* d*-regular graphs on* n *vertices* by $\mathcal{G}(n, d)$ and tacitly assume that nd is even. A graph G contains H as a *topological minor* if there is an injective map $\sigma : V(H) \to V(G)$ and for every $\{u, v\} \in E(H)$ there is a path $p_{uv} \subseteq G$ from $\sigma(u)$ to $\sigma(v)$ that is pairwise vertex-disjoint from all other paths except in the endpoints. The paths p_{uv} are the *edge embeddings* of the minor.

2.1.2 Languages, Formulas and Circuits

An alphabet Σ is a finite set of symbols and we let Σ^* be the set of finite length strings that can be formed over Σ. For a string $w \in \Sigma^*$ we let $|w|$ denote the length of w. A *language* $L \subseteq \Sigma^*$ is a set of finite length strings over the alphabet Σ. We may usually assume that the alphabet is $\{0, 1\}$ but it is often convenient to work with larger alphabets.

Let us say that the *depth* of a string is the number of changes of symbols in a string: the depth is 0 if the string s is empty, and otherwise, if $s = (s_1, \ldots, s_k)$, then the depth is defined to be $1 + \text{Depth}(s_i, \ldots, s_k)$, where i is the largest integer such that $s_{i-1} = s_1$.

Formulas The language of *DeMorgan formulas* consists of all strings defined recursively over the alphabet consisting of variables $x_1, \ldots, x_n$, the connectives $\vee, \wedge, \neg$, the symbols $\top, \bot$, and the brackets $(,)$ as follows.

1. The symbols $\top$ and $\bot$ denote formulas,

2. any variable x_i is a formula,

3. if F is a formula, then so is $\neg F$,

4. if F_1 and F_2 are formulas, then so are $(F_1 \wedge F_2)$ and $(F_1 \vee F_2)$.

We use $\bar{x}_i$ as a shorthand for the formula $\neg x_i$ and call the formulas x_i and $\bar{x}_i$ *literals*. A *subformula* of a formula F is a substring of F that is also a formula and the *size* of a formula is the length of the string representing it.

We can think of a formula F as a binary tree T_F where each internal node is either a node with 2 children and labeled with an $\vee$ or an $\wedge$ or a node with 1 child labeled with $\neg$, and each leaf node is either labeled with a variable or one of the symbols $\top$ or $\bot$. For a branch b in T_F, let $L(b) = (\ell_1, \ldots, \ell_k)$ denote the string of internal labels encountered on b if traversed from root to leaf. We define the *depth* of the branch b as $\text{Depth}\big(L(b)\big)$, and the depth of a formula F as the maximum depth of any branch in T_F.

An *assignment* is a mapping $\alpha : \{x_1, \ldots, x_n\} \to \{\textsc{True}, \textsc{False}\}$ that sets each variable to either True or False. We often identify True with 1 and False with 0 and call a variable that may be True or False a *bit*. A formula F evaluates to True under an assignment α if

1. F is of the form $\top$, or

2. F is of the form x_i and $\alpha(x_i) = \textsc{True}$, or

3. F is of the form $\neg G$ and G does *not* evaluate to True under α, or

4. F is of the form $(G_1 \wedge G_2)$ and both G_1 and G_2 evaluate to True under α, or

5. F is of the form $(G_1 \vee G_2)$ and at least one G_i evaluates to True under α.

If F does not evaluate to True under α, then we say that it evaluates to False under α and write $F(\alpha)$, or $\alpha(F)$, to denote the (unique) value F evaluates to under α. When writing formulas we usually ignore most brackets. In particular we write $x_1 \vee x_2 \vee \cdots \vee x_n$ as a shorthand for $(x_1 \vee (x_2 \vee (\cdots (x_{n-1} \vee x_n) \cdots)$ and similarly for $\wedge$. A formula G is *logically implied* by the formulas $F_1, \ldots, F_k$, written as $F_1, \ldots, F_k \models G$, if for all assignments α such that $\alpha(F_1 \wedge \cdots \wedge F_k) = \textsc{True}$ it also holds that $G(\alpha) = \textsc{True}$.

A formula F is *satisfiable* if there is an assignment α such that $F(\alpha) = \textsc{True}$, a formula F is *unsatisfiable*, or a *contradiction*, if $F(\alpha) = \textsc{False}$ for all assignments α and similarly a formula F is a *tautology* if $F(\alpha) = \textsc{True}$ under all assignments. We denote the language that consists of all satisfiable formulas by SAT, the language of unsatisfiable formulas by UNSAT and the language of tautologies by TAUT.

A disjunction of literals, e.g., $C = (x_3 \vee \bar{x}_{42} \vee \cdots \vee \bar{x}_7)$, is called a *clause*, the *width* of C, denoted by $\text{Width}(C)$, is equal to the number of literals in C, and a k-*clause* is any clause of width at most k. A formula F is in *conjunctive normal form (CNF)* if it is a conjunction of clauses, i.e., of the form $\bigwedge_{i=1}^{m} C_i$, where each C_i is a clause. The width of F, denoted by $\text{Width}(F)$ is the maximum width of any clause occurring, the formula F is a k-*CNF* if each clause in F is a k-clause and F is of *bounded width* if there is a constant k such that F is a k-CNF.

A Boolean function $f : \{0,1\}^n \to \{0,1\}$ on n bits is represented by a formula F over n variables if for all assignments α it holds that $f(\alpha) = F(\alpha)$.

Proposition 2.1.1. *Every Boolean function* $f : \{0,1\}^n \to \{0,1\}$ *on* n *bits can be represented as a CNF of size* $O(n \cdot 2^n)$.

Proof. We construct the CNF F as follows. For each assignment α such that $f(\alpha) = \textsc{False}$, we add the negation, written as a clause, of the unique n variable conjunction that is satisfied by α to F. It is readily verified that $f(\alpha) = F(\alpha)$ for all assignements α. □

The language that consists of all satisfiable CNFs is denoted by CNF-SAT, and we let the set of unsatisfiable CNFs be denoted by CNF-UNSAT.

Circuits Note that when defining formulas, we may be forced to write down the exact same subformula several times. Circuits allow the re-use of such subformulas, thus decreasing the size of the representation.

We define the language of Boolean *circuits* over the DeMorgan basis as follows. A string C is a circuit if it is a sequence $(F_1, \ldots, F_s)$ of DeMorgan formulas over the variables $x_1, \ldots, x_n, y_1, \ldots, y_s$, where each F_i is of the following form:

1. one of the symbols $\top$ or $\bot$, or
2. x_j, for $j \in [n]$, or
3. $\neg y_j$, for $j < i$, or
4. $(y_j \circ y_k)$, for $j, k < i$ and $\circ \in \{\wedge, \vee\}$.

We determine whether a circuit $C = (F_1, \ldots, F_s)$ evaluates to $\textsc{True}$ under an assignment α (to the variables $x_1, \ldots, x_n$) as follows. Starting with $i = 1$ and $\alpha_0 = \alpha$, we sequentially evaluate the formula F_i under the assignment α_{i-1}, and extend α_{i-1} to the variable y_i to obtain α_i defined as

$$\alpha_i(z) = \begin{cases} \alpha_{i-1}(z), & \text{if } z \neq y_i, \\ \alpha_{i-1}(F_i), & \text{otherwise.} \end{cases}$$

We say that C evaluates to TRUE under α if $\alpha_s(y_s) = \text{TRUE}$ and C evaluates to FALSE otherwise. As for formulas we write $C(\alpha)$ to denote the unique value that C evaluates to under α. Finally, we say that a Boolean function f is *represented* by the circuit C, or *computed* by the circuit C, if $C(\alpha) = f(\alpha)$ for all assignments α.

Note that we can view a circuit C as a directed acyclic graph: for each $i \in [s]$ we have a node that is connected by an incoming edge to all nodes j such that y_j occurs in the formula F_i. We label each internal node by the function it computes, i.e. $\wedge$, $\vee$, or $\neg$, and add an additional outgoing edge from the node s. The *depth* of a circuit is defined analogously to formulas: the depth of a source to sink path is $\text{Depth}(L(p))$, where the function L is defined as for formulas, and the depth of the circuit is the maximum depth of any source to sink path.

Let us recall that most Boolean functions require large circuits.

Theorem 2.1.2 ([Sha49]). *Asymptotically almost surely as $n \to \infty$ a random Boolean function $f : \{0,1\}^n \to \{0,1\}$ needs a circuit of size at least $\Omega(2^n/\log n)$.*

2.1.3 The Complexity Classes P, P/poly, NP and coNP

A language $L \subseteq \Sigma^*$ is computable in polynomial time if there is an algorithm A that takes a poly-bounded number of steps in the input length, such that $A(\ell) = 1$ if and only if $\ell \in L$. We denote by P all languages that are computable in polynomial time.

Similarly, we say that a language $L \subseteq \{0,1\}^*$ is in P/poly if there is a polynomial p and a family of circuits $(C_n)_{n\in\mathbb{N}}$, each circuit C_n of size at most $p(n)$, such that $C_n(\alpha) = 1$ if and only if $\alpha \in L \cap \{0,1\}^n$.

The class NP consists of all languages $L \subseteq \Sigma_1^*$ for which there is an algorithm A, taking a poly-bounded number of steps in the input length, such that for all $\ell \in L$ there is an $x \in \Sigma_2^*$ such that $|x| = \text{poly}(|\ell|)$ and $A(\ell, x) = 1$. The class coNP consists of all languages whose complement is in NP, i.e., all languages $L \subseteq \Sigma^*$ such that the language $\bar{L} = \Sigma^* \setminus L$ is in NP.

A language L is complete for a class C if L is in C and for all other problems $L' \in C$ there is an algorithm A, taking a poly-bounded number of steps in the input length, such that $A(\ell') \in L$ if and only if $\ell' \in L'$. It is well-known that the languages SAT and CNF-SAT are NP-complete while the languages TAUT, UNSAT and CNF-UNSAT are coNP-complete.

Note that $P \subseteq NP \cap coNP$. Literally the $1 million question is whether $P \neq NP$ [CJW06]. It is widely believed that P is a proper subset of NP but this statement seems to be out of reach for current techniques. A question of similar flavor is whether $NP \neq coNP$. This is also open but widely believed

to be incomparable. Note that NP $\neq$ coNP implies that P $\neq$ NP, as P is closed under the complement.

2.2 Proof Systems

The following definition is due to Cook and Reckhow [CR79]. A *proof system* P for a language $L \subseteq \Sigma_1^*$ is a language in $\Sigma_1^* \times \Sigma_2^*$ such that

1. P is poly-time computable, i.e., there is an algorithm A such that $A(\ell, \pi) = 1$ if and only if $(\ell, \pi) \in P$, and A runs in time $\text{poly}(|\ell|, |\pi|)$,
2. for each $\ell \in L$ there is a $\pi \in \Sigma_2^*$ such that $(\ell, \pi) \in P$, and
3. for all $\ell \in \Sigma_1^* \setminus L$ and any $\pi \in \Sigma_2^*$ it holds that $(\ell, \pi) \notin P$.

A proof system is said to be *complete* if it satisfies Item 2, and *sound* if it satisfies Item 3. Throughout the thesis we only consider complete and sound proof systems. We say that $\pi \in \Sigma_2^*$ is a P proof of $\ell \in L$ if $(\ell, \pi) \in P$. The *size* of a proof π is $|\pi|$, also denoted by $\text{Size}(\pi)$, and the size of refuting ℓ in P is $\text{Size}_P(\ell) = \min_\pi \text{Size}(\pi)$, where the minimum ranges over all P proofs of ℓ. The proof system P for L is *poly-bounded* if there is a polynomial p such that for all $\ell \in L$ it holds that $\text{Size}_P(\ell) \leq p(|\ell|)$. A *propositional proof system* is a proof system for the language CNF-UNSAT. Note that we could also consider proof systems for TAUT but it is customary to work with proof systems for CNF-UNSAT and we follow the crowd. A proof in a propositional proof system is also called a *refutation* and we use these terms interchangeably.

Proposition 2.2.1 ([CR79])**.** *There is a poly-bounded propositional proof system if and only if* NP = coNP*.*

This proposition suggest the following program to separate NP from coNP: prove for stronger and stronger proof systems that they are not poly-bounded. In the past 30 years there has been a lot of work on proving exponential lower bounds for "weak" propositional proof systems such as resolution [Tse68; Hak85; BW01; Raz04a; ABRW04; IOSS16; ABdR+18; AM19], bounded depth Frege [Ajt94; PBI93; KPW95; BP96; Ben02; PRST16; Hås20], polynomial calculus [Raz98; BGIP01; BI10; AR03; MN15; LN17], sum of squares [Gri01; MPW15; BHK+16; KMOW17; Pot17; AH19], or cutting planes [Pud97; BPR97; FPPR17; HP17]. Ultimately the hope is that we understand lower bound strategies well enough so that we can start tackling "strong" propositional proof systems like Frege [Fre79], extended Frege [CR79] or the ideal proof system [GP18].

For now we settle to prove lower bounds for "weak" proof systems in order to extend the current toolbox of lower bound techniques. This thesis fits in there and proves several new lower bounds by extending and adapting lower bounds techniques to our needs.

Before defining the proof systems that we use throughout this thesis we need a way to compare the strength of two proof systems. Let $P \subseteq \Sigma_1^* \times \Sigma_2^*$ and $Q \subseteq \Sigma_1^* \times \Sigma_3^*$ be proof systems for a language L. We say that P *poly-simulates* Q if there is a function $f : \Sigma_3^* \to \Sigma_2^*$, computable in polynomial time in the input length, such that for all $\ell \in L$ and $\pi \in \Sigma_3^*$ it holds that if $(\ell, \pi) \in Q$, then $(\ell, f(\pi)) \in P$. Put in words, the function f translates Q-proofs of ℓ to P-proofs of ℓ with at most a polynomial increase in size, for all $\ell \in L$. It is readily seen that if Q is poly-bounded and P poly-simulates Q, then P is poly-bounded as well.

Furthermore, some of the propositional proof systems are based on (semi-)algebraic reasoning. As such we need to discuss how to translate a CNF $F = \wedge_{i=1}^{m} C_i$ over n variables $x_1, \ldots, x_n$ into a system of polynomials $\mathcal{P}_F$. We define $\mathcal{P}_F$ over the $2n$ variables $x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n$ and say that the system is satisfied under an assignment α if and only if all polynomials evaluate to zero under α. In order to enforce that each variable only takes Boolean values, we add the Boolean axioms

$$y(1 - y)$$

to $\mathcal{P}_F$ for each variable y. We also want to ensure that $\bar{x}_i$ is the negation of x_i and thus also add the negation axioms

$$x_i + \bar{x}_i - 1$$

to $\mathcal{P}_F$. Finally we add for each clause C_i the following polynomial to $\mathcal{P}_F$. Assuming that the clause C_i can be written as $\vee_{j=1}^{w_i} v_{ij}$, for the appropriate literals v_{ij}, we translate C_i into the monomial

$$\prod_{i=1}^{w_i} \bar{v}_{ij} \ ,$$

where we use the convention that $\bar{\bar{x}}_k = x_k$. Assuming that the Boolean axioms and the negation axioms are honored it should be evident that above polynomial is equal to 0 if and only if one literal v_{ij} is set to 1 and thus the clause C_i is satisfied. For a CNF F, let us denote by $\mathrm{Deg}(F)$ the maximum degree of any polynomial occurring in $\mathcal{P}_F$.

2.2.1 Resolution

Resolution is arguably the most studied proof system and has been first defined by Blake [Bla37] in the 30ies of the last century. Resolution is

a propositional proof system with refutations of the following form. A resolution refutation π of an unsatisfiable CNF F is a sequence of clauses $(C_1, \ldots, C_L)$ such that $C_L = \bot$ is the empty clause and each clause C_i either occurs in F or is derived from come clauses C_j and C_k, for $j, k < i$, by the *resolution rule*

$$\frac{B \vee x \qquad C \vee \overline{x}}{B \vee C} \ .$$

The *length* of π, denoted $\text{Length}(\pi)$, is L and the *width* of π is the maximum width of any clause occurring in π. We denote by $\text{Width}_R(F)$ the minimum width of any resolution refutation of F and, similarly, let $\text{Length}_R(F) = \min_\pi \text{Length}(\pi)$ where π ranges over all resolution refutations of F.

The most common way to prove resolution size lower bounds is by proving a lower bound on the width required to refute a formula and then applying the following width-length trade off due to Ben-Sasson and Wigderson.

Theorem 2.2.2 ([BW01])**.** *For any CNF F over n variables it holds that*

$$\text{Length}_R(F) = \exp\left(\Omega\left(\frac{(\text{Width}_R(F) - \text{Width}(F))^2}{n}\right)\right) \ .$$

Thus, assuming that the CNF F is of constant width, if we manage to show a width lower bound linear in the number of variables, then we obtain a $2^{\Omega(n)}$ length lower bound on any resolution refutation of F. It is worth noting that this is essentially optimal as there is always a resolution refutation of length 2^n.

2.2.2 Bounded Depth Frege

Let us first define the more general Frege proof system that can derive any consequence from a set of axioms. We then specialize the system to a propositional proof system. A k-ary Frege rule is any rule of the form

$$\frac{A_1 \qquad A_2 \qquad \ldots \qquad A_k}{B}$$

such that $A_1, \ldots, A_k \models B$. Let $\mathcal{R}$ be a set of Frege rules. A Frege proof of the formula G from the formulas $F_1, \ldots, F_m$ over $\mathcal{R}$ is a sequence of formulas $(H_1, \ldots, H_L)$ such that $H_L = G$ and each line H_i is either equal to an axiom F_j or is derived from $H_{j_1}, \ldots, H_{j_k}$, for $j_1, \ldots, j_k < i$, by a k-ary Frege rule in $\mathcal{R}$.

The width of a proof is the size of the largest formula occurring, the length is the number of formulas in the proof and the depth of a proof is the maximum depth of any formula appearing.

A Frege proof system $\mathcal{F}^p$ is a finite set of Frege rules $\mathcal{R}$ that are implicationally complete, i.e., if the formulas $F_1, \ldots, F_m$ logically imply G, then there is a Frege proof of G from the fromulas $F_1, \ldots, F_m$ over $\mathcal{R}$. The following theorem due to Reckhow states that any two Frege systems poly-simulate each other.

Theorem 2.2.3 ([Rec75]). *Any two Frege propositional proof systems poly-simulate each other. Furthermore, the simulation preserves the depth of the proof up to an additive constant, while the size of the proof increases by at most a multiplicative constant.*

Reckhow in fact proved a more general statement, which also allows a certain freedom in the choice of the basis over which the formulas are defined. We stick to the DeMorgan basis and this extension is thus not central to the following discussion. The interested reader is encouraged to consult [Kra19] for a more complete treatment.

As different Frege systems poly-simulate each other, we may consider any such system. Fix an arbitrary Frege proof system $\mathcal{F}^p$ as defined above. We define the Frege propositional proof system $\mathcal{F}$ to be the language that contains all tuples (F, π), where $F = \bigwedge_{i=1}^{m} C_i$ is an unsatisfiable CNF and π is an $\mathcal{F}^p$ proof of $\bot$ from the formulas $C_1, \ldots, C_m$.

Similarly we define the depth d Frege refutational proof system $\mathcal{F}_d$: it is the language that contains all tuples (F, π) from $\mathcal{F}$ where π is of depth at most d.

Note that by Theorem 2.2.3 there is a constant d_0, such that for all $d \geq d_0$, the proof systems $\mathcal{F}_d$ is complete. We only consider such d in this thesis. Note that when the depth is reduced, then the size of a proof may blow up substantially.

2.2.3 Polynomial Calculus

The polynomial calculus propositional proof system, introduced in [CEI96] though we use the slightly stronger version introduced in [ABRW02], is based on algebraic reasoning. A polynomial calculus refutation π of the CNF F over a field $\mathbb{F}$ consists of an ordered sequence of polynomials $(p_1, \ldots, p_L)$ such that p_L is the constant 1 polynomial and each polynomial p_i occurs either in $\mathcal{P}_F$ or is derived by one of the following two polynomial calculus rules from p_j and p_k, where $j, k < i$,

$$\frac{q_1 \qquad q_2}{\alpha q_1 + \beta q_2} \qquad\qquad \frac{q}{xq} \ ,$$

for some variable x and constants $\alpha, \beta \in \mathbb{F}$. We denote the polynomial calculus propositional proof system over $\mathbb{F}$ by $\mathrm{PC}_{\mathbb{F}}$.

The *degree* of π is the maximum degree of any polynomial that occurs in π and the length of π is L. The degree of refuting F in $\mathrm{PC}_{\mathbb{F}}$ is denoted by $\mathrm{Deg}_{\mathrm{PC}_{\mathbb{F}}}(F)$ and defined as the minimum degree of any polynomial calculus refutation over $\mathbb{F}$ of the CNF F. Finally, let us also define the *monomial size* of a refutation, which simply counts the number of monomials occurring in the proof and the monomial size of refuting F in $\mathrm{PC}_{\mathbb{F}}$, denoted by $\mathrm{MSize}_{\mathrm{PC}_{\mathbb{F}}}(F)$, is the minimum monomial size of any $\mathrm{PC}_{\mathbb{F}}$ refutation.

While for resolution we had a width-length trade off, for polynomial calculus we have a degree-monomial-size trade off.This trade off in fact predates the celebrated resolution trade off. Recall that $\mathrm{Deg}(F)$ denotes the maximum degree of any polynomial occurring in $\mathcal{P}_F$.

Theorem 2.2.4 ([CEI96; IPS99])**.** *For any unsatisfying CNF F over n variables and any field $\mathbb{F}$ it holds that*

$$\mathrm{MSize}_{\mathrm{PC}_{\mathbb{F}}}(F) = \exp\left(\Omega\left(\frac{(\mathrm{Deg}_{\mathrm{PC}_{\mathbb{F}}}(F) - \mathrm{Deg}(F))^2}{n}\right)\right) \ .$$

Let us remark that polynomial calculus over any field $\mathbb{F}$ simulates resolution with respect to size as well as degree [ABRW02], i.e., a resolution proof of size s and width w can be translated into a $\mathrm{PC}_{\mathbb{F}}$ proof of size $O(s)$ and degree $O(w)$.

2.2.4 Sum of Squares

Sum of Squares (SoS) is a propositional proof system based on semi-algebraic reasoning. In contrast to the other proof systems SoS is static meaning that a proof is a single line. In fact, it is a single polynomial identity.

A Sum of Squares (SoS) refutation of a CNF F consists of a sequence of polynomials $(r_1, \ldots, r_m, s_1, \ldots, s_t)$ such that

$$\sum_{i\in[m]} r_i p_i + \sum_{i\in[t]} s_i^2 = -1 \ ,$$

for $\mathcal{P}_F = \{p_1, \ldots, p_m\}$. As for polynomial calculus one can define the degree and monomial size of a refutation. For SoS there is also a degree-monomial-size trade off.

Theorem 2.2.5 ([AH19])**.** *For any unsatisfying CNF F over n variables it holds that*

$$\mathrm{MSize}_{\mathrm{SoS}}(F) = \exp\left(\Omega\left(\frac{(\mathrm{Deg}_{\mathrm{SoS}}(F) - \mathrm{Deg}(F))^2}{n}\right)\right) \ .$$

It is not so hard to show that SoS poly-simulates resolution. Quite surprisingly, Berkholz [Ber18] showed that SoS also poly-simulates $\mathrm{PC}_{\mathbb{R}}$. It should be mentioned that there is a separation for finite fields and this is thus the best one could hope for.

2.3 Propositional Formulas

We are interested in several simple principles which, as we show in the following, turn out to be hard for different proof systems.

2.3.1 Tseitin

The Tseitin formula is defined over a graph $G = (V, E)$. At the heart of the formula is the following idea: count the number of edges by a double sum over the vertices and a sum over the edges incident to a single vertex

$$\frac{1}{2} \sum_{v \in V} \sum_{e : v \in e} 1 = m \ ,$$

where we divide by 2 as each edge is summed over twice. This implies in particular that the sum

$$\sum_{v \in V} \sum_{e : v \in e} 1 = 2m$$

is equal to an even number. Moreover, if we associate each edge e with a Boolean variable x_e, no matter the assignment $\alpha : \{x_e \mid e \in E\} \to \{0, 1\}$ this sum is still even:

$$\sum_{v \in V} \sum_{e : v \in e} \alpha(x_e) = 2|\alpha^{-1}(1)| \ . \tag{2.1}$$

Suppose we are given a charge function $\tau : V \to \{0, 1\}$ that assigns each vertex either a charge of 0 or 1 and, furthermore, that we enforce that at each vertex v the incident edge variables sum to the charge $\tau(v)$ modulo 2. That is, for each vertex $v \in V$ we have the constraint

$$\sum_{e : v \in e} x_e = \tau(v) \mod 2 \ . \tag{2.2}$$

All these constraints together imply that the double sum from Equation (2.1) is equal to

$$\sum_{v \in V} \sum_{e : v \in e} x_e = \sum_{v \in V} \tau(v) \mod 2 \ .$$

Thus if $|\tau^{-1}(1)|$ is odd, then there is no satisfying assignment. This defines the Tseitin contradiction: fix a charge function that assigns an odd charge to the graph and claim that there is an assignment as described above.

The CNF associated with above system of equations modulo 2 is defined as follows. For each vertex $v \in V$ and a fixed τ note that the vertex axiom of v (Axiom 2.2) can be used to define a Boolean function, mapping from $\deg(v)$ many inputs to True, False, depending on whether the constraint is satisfied by a given assignment, identifying True with 1 and False with 0. By Proposition 2.1.1 we may obtain a CNF $F_{v,\tau}$ of size $O(\deg(v) \cdot 2^{\deg(v)})$ that is satisfied if and only if the corresponding vertex axiom is satisfied. Then, the CNF associated with the Tseitin contradiction is defined as

$$\text{Tseitin}(G, \tau) = \bigwedge_{v \in V} F_{v,\tau} \ .$$

Observe that this formula is of exponential size in the maximum degree of the graph. As such this formula is usually only considered in combination with graphs of constant degree.

2.3.2 Perfect Matching

The perfect matching formula is also defined over a graph $G = (V, E)$. The formula claims that there is a subset of edges $M \subseteq E$ such that every edge is in precisely one edge of M. In other words, we want a set $M \subseteq E$ such that for every vertex $v \in V$ it holds that

$$|\{e \in M \mid v \in e\}| = 1 \ .$$

A moment of reflection reveals that a matching always contains an even number of vertices: every edge in M consists of two vertices, each of which is in precisely one edge and hence there is an even number of vertices matched. Thus if we start with a graph G defined over an odd number of vertices, there cannot be a perfect matching in this graph. This defines the perfect matching contradiction.

We have a variable per edge $\{x_e \mid e \in E\}$ and for each vertex $v \in V$ we add the following clauses to the CNF formula $\text{PM}(G)$:

$$\bigvee_{e : v \in e} x_e \ , \quad \text{and} \qquad \bar{x}_e \vee \bar{x}_{e'} \ ,$$

for any $e, e' \in E$ such that $e \cap e' = \{v\}$.

2.3.3 Pigeonhole Principle

One encoding of the pigeonhole principle (PHP) is the perfect matching formula $\text{PM}(G)$ defined over a bipartite graph $G = (U, V, E)$ with $|U| > |V|$.

This is the strongest encoding of the principle: there are subformulas of PM(G) that are contradictions. Let us define two subformulas of interest to us.

Recall that we have a variable for each edge in the graph $\{x_e \mid e \in E\}$. The ordinary pigeonhole principle PHP(G), for $G = (U, V, E)$, has a clause per vertex $u \in U$

$$\bigvee_{e : u \in e} x_e \ , \tag{2.3}$$

ensuring that the vertex u is matched to some vertex in the neighborhood, and for any two distinct edges $\{u, v\}, \{u', v\} \in E$, where $v \in V$ and $u, u' \in U$, we add the axiom

$$\bar{x}_e \vee \bar{x}_{e'} \ . \tag{2.4}$$

These axioms ensure that each vertex in V is matched to at most one vertex in U. This is a contradiction if $|U| > |V|$.

We can add further axioms, making the formula more and more constrained and thus easier to refute. The functional pigeonhole principle FPHP(G) is defined by also adding Axiom 2.4 for distinct edges $\{u, v\}, \{u, v'\} \in E$, where $u \in U$ and $v, v' \in V$. Finally, we can recover the perfect matching principle PM(G) by adding Axiom 2.3 for every vertex V.

2.3.4 The Truthtable Formula

The truthtable formula is a bit more involved to define. We want to encode the claim that a Boolean function $f : \{0,1\}^n \to \{0,1\}$, given as a binary string of length 2^n, has a circuit of size at most s. By Theorem 2.1.2 most Boolean functions require circuits of size $\Omega(2^n/\log n)$ and thus for $s \ll 2^n/\log n$, this formula is a contradiction for most Boolean functions f.

The formula $\text{Circuit}_s(f)$ consists of two parts. The first part of the formula defines a circuit, and the second part of the formula ensures that the circuit encoded by the first part indeed computes f. In the following we describe a polynomial encoding of the truthtable formula. This encoding conveys enough of an idea for the following discussion and is already cumbersome to explain – the CNF translation consists of even more axioms and variables. For the formal encoding used in Paper C we recommend the interested reader to consult Section C.2.3 as well as Section C.7.

Let us first describe the *structure variables* which are used in the first part of the formula to describe the circuit.

Each of the s gates is indexed from 1 to s, with the output gate being gate s. This labeling will be topological, in a sense that each gate $v \in [s]$

has no input from a gate $u > v$. Each gate $v \in [s]$ has three variables isNeg(v), isOr(v) and isAnd(v) associated with it, indicating the operation computed at v. Similarly, for a gate $v \in [s]$ and a wire $a \in \{1,2\}$ we have variables isFromConst(v,a), isFromInput(v,a) and isFromGate(v,a) indicating whether the input wire a of v is connected to a constant, a variable or a gate.

Finally we have the variables indicating to what constant, variable or gate an input wire is connected to: We have variables constantValue(v,a), isInput(v,a,i) and isGate(v,a,u), for $a \in \{1,2\}$, $i \in [n]$ and $u < v$. The corresponding variables indicate the constant value, the input x_i or the gate u that the input wire a of v is connected to – assuming a is indeed connected to the corresponding kind.

The structure variables come along with a set of axioms that we refer to as the *structure axioms*. The first axioms ensure that every wire is connected to a single kind

$$\text{isFromConst}(v,a) + \text{isFromInput}(v,a) + \text{isFromGate}(v,a) = 1 \quad \forall v \in [s] \ ,$$

and the second group of axioms makes sure that each gate is of precisely one kind

$$\text{isNeg}(v) + \text{isOr}(v) + \text{isAnd}(v) = 1 \quad \forall v \in [s] \ .$$

The final group of structure axioms ensures that the variables that indicate to what input or gate a fixed wire is connected to always sum to one, except at gate 1 as it cannot have any inputs from other gates

$$\sum_{i=1}^{n} \text{isInput}(v,a,i) = 1 \quad \forall v \in [s] \ , \text{ and}$$
$$\sum_{u=1}^{v-1} \text{isGate}(v,a,u) = 1 \quad \forall v \in [s] \setminus \{1\} \ .$$

This completes the description of the first part of the formula.

The second part of the formula is defined over the so-called *evaluation variables* which describe what value is computed at each gate v on input $\alpha = \alpha_1, \ldots, \alpha_n$.

Each gate has $3 \cdot 2^n$ evaluation variables associated with it. These are $\text{out}_\alpha(v)$, indicating the Boolean value computed at gate $v \in [s]$ on input $\alpha \in \{0,1\}^n$, and the variables $\text{in}_\alpha(v,a)$ which indicate the value brought to vertex $v \in [s]$ on wire $a \in \{1,2\}$ on input $\alpha \in \{0,1\}^n$.

Note that we have $\Theta(s^2 + sn)$ structure variables and $3s2^n$ evaluation variables, for a total of $\Theta(s^2 + s2^n)$ variables in $\text{Circuit}_s(f)$.

The evaluation variables are accompanied by the *evaluation axioms* ensuring that the evaluation variables indeed compute the intended values. The first set of axioms ensures that the wires carry the value intended by the structure axioms. If a wire is connected to a constant, then the evaluation variable associated with that wire should always be equal to the constant

$$\mathrm{isFromConst}(v, a) \cdot \left(\mathrm{in}_\alpha(v, a) - \mathrm{constantValue}(v, a)\right) = 0 \ ,$$

and similarly if a wire is connected to an input or a gate

$$\begin{aligned} \mathrm{isFromInput}(v, a) \cdot \mathrm{isInput}(v, a, i) \cdot \left(\mathrm{in}_\alpha(v, a) - \alpha_i\right) &= 0 \ , \\ \mathrm{isFromGate}(v, a) \cdot \mathrm{isGate}(v, a, u) \cdot \left(\mathrm{in}_\alpha(v, a) - \mathrm{out}_\alpha(u)\right) &= 0 \ . \end{aligned}$$

The final set of evaluation axioms makes sure that the output evaluation variable of a gate is correctly related to the input evaluation variables:

$$\begin{aligned} \mathrm{isNeg}(v) \cdot \mathrm{out}_\alpha(v) &= \mathrm{isNeg}(v) \cdot \overline{\mathrm{in}_\alpha(v, 1)} \ , \\ \mathrm{isOr}(v) \cdot \mathrm{out}_\alpha(v) &= \mathrm{isOr}(v) \cdot \left(1 - \overline{\mathrm{in}_\alpha(v, 1)} \cdot \overline{\mathrm{in}_\alpha(v, 2)}\right) \ , \\ \mathrm{isAnd}(v) \cdot \mathrm{out}_\alpha(v) &= \mathrm{isAnd}(v) \cdot \mathrm{in}_\alpha(v, 1) \cdot \mathrm{in}_\alpha(v, 2) \ . \end{aligned}$$

Last but not least we have the axioms that ensure that the circuit outputs the function specified by the truthtable

$$\mathrm{out}_\alpha(s) = f(\alpha) \ .$$

Contributions

In this chapter we highlight the main results of the included papers. Each paper is discussed in a separate section. The sections first give some context how our results fit into the literature, followed by the main theorem and a brief discussion about the employed proof techniques.

3.1 On Bounded Depth Frege Refutations of the Tseitin Formula

Johan Håstad and Kilian Risse, *"On Bounded Depth Proofs for Tseitin Formulas on the Grid; Revisited"*, accepted to FOCS'22 [HR22]

Paper A concerns bounded depth Frege refutations of the Tseitin contradiction defined over grid graphs.

The study of bounded depth Frege refutations was initiated by Ajtai [Ajt94] who proved that the PHP cannot be refuted in polynomial size for any constant depth Frege system. This pioneering result was followed up by several papers in the 1990s, first improving Ajtai's result to hold up to depth $O(\log \log n)$ [PBI93; KPW95], and then extending it to the Tseitin contradiction defined over complete [UF96], as well as expander graphs [Ben02].

These developments followed previous work where the computational power of the class of bounded depth circuits[1] was studied [Sip83; FSS84; Yao85; Hås86; Raz88; Smo87]. It should not be surprising that it is simpler to argue about the computational power of a single circuit rather than a sequence of formulas forming a proof. This is exemplified by the lower bounds achieved by the end of the 1990s: while the bounded depth Frege lower bounds remained stuck at depth $O(\log \log n)$, the results for circuit size extended to almost logarithmic depth.

This gap was recently closed in two steps. First Pitassi et al. [PRST16] obtained super-polynomial bounded depth Frege lower bounds up to depth

[1] As in the bounded depth setting there is no major difference between circuits and formulas we gloss over the difference between these.

$o(\sqrt{\log n})$ and then Håstad [Hås20] managed to extend these results up to depth $\Theta\left(\frac{\log n}{\log\log n}\right)$, which matches the result for circuits up to constants.

All these previous bounded depth Frege lower bounds considered the total size of a proof. The total size is composed of the length (number of steps) of a refutation and the size of each line. For some proof systems, such as resolution, each line is bounded in size and hence any super-polynomial lower bound on proof size also implies a lower bound on the number of proof steps. This is not necessarily true for bounded depth Frege – the line size may grow and it is thus an interesting question to study the number of lines required, given that each line is of bounded size.

This line of investigation was recently initiated by Pitassi et al. [PRT22]. They consider the Tseitin contradiction defined over the grid of size $n \times n$ and showed that if each line of the refutation is limited to size M and depth d, then a Frege refutation must consist of at least $\exp\left(n/2^{O(d\sqrt{\log M})}\right)$ many lines. For most interesting values of M this greatly improves the bounds implied by the results for total proof size of Håstad [Hås20]. In particular, if M is a polynomial, then the lower bounds are of the form $\exp(n^{1-o(1)})$, as long as $d = o(\sqrt{\log n})$, in contrast to the total size lower bounds of the form $\exp(n^{\Omega(1/d)})$. Pitassi, Ramakrishnan, and Tan [PRT22] rely on the restrictions introduced by Håstad [Hås20] but analyze them using the methods of Pitassi et al. [PRST16].

We study the same Tseitin contradiction on the grid but analyze the restrictions using the machinery set up by Håstad [Hås20]. This allows us to improve the result of Pitassi et al. [PRT22] to obtain the lower bound conjectured by them.

Theorem 3.1.1. *For any Frege refutation of the Tseitin principle defined over the $n \times n$ grid graph the following holds. If each line of the refutation is of size M and depth d, then the length of the refutation is*

$$\exp\left(\frac{n}{\left((\log n)^{O(1)} \log M\right)^{2d}}\right) .$$

Along the way we also improve the parameters of the refutation size lower bound due to Håstad [Hås20] from exponential in $\tilde{\Omega}(n^{1/59d})$ to exponential in $\tilde{\Omega}(n^{1/(2d-1)})$.

Theorem 3.1.2. *For $d \leq O\left(\frac{\log n}{\log\log n}\right)$ the following holds. Any depth-d Frege refutation of the Tseitin contradiction defined on the $n \times n$ grid requires size*

$$\exp\left(\Omega(n^{1/(2d-1)}(\log n)^{O(1)})\right) .$$

We achieve the improvements on total size by revisiting Håstad's original proof and carefully eliminating some undesired dependencies on the depth

in the switching lemma. This forces us to use slightly more general restrictions for book-keeping but the over all proof remains unchanged.

We then use this improved proof of the switching lemma to obtain a multi-switching lemma with which we are able to prove Theorem 3.1.1. In order to prove the multi-switching lemma we need to analyze a new combinatorial game played on the grid graph. Already Håstad [Hås20] needed to analyze such a combinatorial game. This new game is quite a bit more complicated and requires an entirely new amortized analysis.

3.2 Average-Case Perfect Matching Lower Bounds

Per Austrin and Kilian Risse, *"Perfect Matching in Random Graphs is as Hard as Tseitin"*, SODA'22, to appear in TheoretiCS [AR22a]

This paper studies the power (or lack thereof) of the SoS, PC and bounded depth Frege proof systems when it comes to refuting the perfect matching formula PM(G) defined over sparse random graphs G on an odd number of vertices. Apart from being a natural and well-studied problem on its own, the perfect matching formula is interesting because of its close relation to two other widely studied families of formulas, namely the pigeonhole principle (PHP) and the Tseitin formula.

While most variants of the PHP are hard for PC [Raz98; MN15], the perfect matching variant is in fact easy to refute over any field [Rii93] and in SoS all variants of the PHP are easy to refute [GHP02]. On the other hand the Tseitin formula is (almost) always hard: for $\mathrm{PC}_{\mathbb{F}}$ over fields $\mathbb{F}$ of characteristic distinct from 2 [BGIP01; AR03] and SoS [Gri01] these formulas require linear degree if the underlying graph G is a good expander.[2]

Hence the perfect matching formula lies somewhere in between the easy PHP formula and the hard Tseitin formula and it is natural to wonder whether SoS or PC requires large degree to refute the perfect matching formula over non-bipartite graphs.

This is well understood if the perfect matching principle is defined over a *complete graph* on an odd nuber of vertices (also know as "MOD 2 principle"): the proof systems SoS and PC require degree $\Omega(n)$, except for PC defined over fields of characteristic 2 [BGIP01; Gri01]. Less is known for sparse graphs: Buss et al. [BGIP01] obtained worst-case lower bounds for PC showing that there exist bounded degree graphs on n vertices requiring $\Omega(n)$ degree refutations. This is obtained by a reduction from Tseitin formulas and while the work of Buss et al. predates the current interest in

[2] Observe that we cannot hope to prove degree lower bounds over fields of characteristic 2 as the constraints become linear and we can thus refute the Tseitin formula using Gaussian elimination. As the perfect matching formula PM(G) implies the Tseitin formula, PC over fields of characteristic 2 can easily refute PM(G), if G has an odd number of vertices.

the SoS system, it is not hard to see that the same reduction yields a similar $\Omega(n)$ degree lower bound for SoS.

However, for random graphs G little is known about the hardness of the perfect matching formula and, e.g., Razborov [Raz17] asked whether it is true that the Lovász-Schrijver hierarchy [LS91] (a proof system polynomially simulated by SoS) requires n^{ε} rounds to refute the perfect matching principle on a random sparse regular graph with high probability. We answer this question by proving the following theorem.

Theorem 3.2.1. *There is a constant d_0 such that for all constants $d \geq d_0$ the following holds asymptotically almost surely for $G \sim \mathcal{G}(n, d)$.*

1. $\mathrm{Deg}_{\mathrm{PC}_{\mathbb{F}}}\big(\mathrm{PM}(G)\big) = \Omega(n/\log n)$ *for any fixed field $\mathbb{F}$ with* $\mathrm{char}(\mathbb{F}) \neq 2$.
2. $\mathrm{Deg}_{\mathrm{SoS}}\big(\mathrm{PM}(G)\big) = \Omega(n/\log n)$.
3. *There is a $\delta > 0$ such that* $\mathrm{Size}_{\mathcal{F}_d}\big(\mathrm{PM}(G)\big) = \exp\big(\Omega(n^{\delta/D})\big)$, *for all* $D \leq \frac{\delta \log n}{\log\log n}$.

Using the known degree-monomial-size tradeoffs for Polynomial Calculus [IPS99; CEI96] and Sum of Squares [AH19], the degree lower bounds from Theorem 3.2.1 imply near-optimal monomial size lower bounds of $\exp\big(\Omega(n/\log^2 n)\big)$.

We obtain these lower bounds by a worst-case to average-case reduction. We achieve this by using the embedding technique as introduced to proof complexity by Pitassi et al. [PRST16]: for, say, the SoS lower bound, our starting point is the $\Omega(n)$ *worst-case* degree lower bound in sparse graphs, and we then prove that these hard instances can be embedded in a random d-regular graph in such a way that the hardness of refuting the formula is preserved.

There are two main components to this argument. One of them is a new graph embedding theorem which may be of independent interest. Very loosely speaking, we show that any bounded-degree graph with $O(n/\log n)$ edges can be embedded as a *topological minor* into any bounded-degree α-expander on n vertices. But this does not quite suffice: for our application we also need to be able to control the parities of the path lengths used in the topological embedding. We show that as long as every large linear-sized subgraph contains an odd cycle of length $\Omega(1/\alpha)$, this is indeed possible. The following is a quite informal statement of our embedding theorem.

Theorem 3.2.2 (Informal)**.** *Let G be a constant degree α-expander on n vertices. If H is a graph with at most $\frac{\varepsilon n}{\log n}$ edges and $\Delta(H) \ll \alpha^2 \cdot d(G)$, then G contains H as a topological minor. Furthermore, if all large vertex induced subgraphs of*

G *contain an odd cycle of length* $\Omega(1/\alpha)$*, then one can choose the parities of the length of all the edge embeddings in the minor.*

This generalizes various classical results of a similar flavor (e.g. [KR96; KN19; CN19; Kri19]). See Paper B for a discussion comparing these (and other) existing embedding results to Theorem 3.2.2.

To motivate the second component of our worst-case to average-case, we need to look the reduction in a bit more detail. A quite naïve attempt to obtain average-case lower bounds from a sparse worst-case instance H on n vertices is to topologically embed the worst-case instance into a random regular graph G on $O(n \log n)$ vertices using Theorem 3.2.2. One would then like to argue that $\mathrm{PM}(G)$ is hard.

Suppose each path p_{uv} in the embedding of H in G corresponding to some edge $\{u, v\} \in E(H)$ is of odd length. Then it is straightforward to verify that the perfect matching formula defined over the embedding is at least as hard to refute as the worst-case instance $\mathrm{PM}(H)$: map each variable y_e, for $e \in p_{uv}$, alternatingly to x_{uv} or $\bar{x}_{uv}$ such that the first and last edges of p_{uv} are mapped to x_{uv} (using that p_{uv} is of odd length). This simple projection maps the perfect matching formula defined over the embedding of H to $\mathrm{PM}(H)$ and thus shows that the hardness of $\mathrm{PM}(H)$ should be inherited.

But having such a worst-case instance as a topological minor is *not* sufficient to conclude that $\mathrm{PM}(G)$ is hard. For instance G may contain an isolated vertex and it is then trivial to refute $\mathrm{PM}(G)$. On the other hand if we could guarantee that there is a perfect matching M in the subgraph of G induced by the vertices *not* used in the embedding of H, we can conclude that $\mathrm{PM}(G)$ is hard: hit the formula with the restriction corresponding to the matching M and by the argument from the previous paragraph we are basically left with the worst-case formula.

Thus if we can ensure that H is a topological minor of G with the two additional properties that (i) every path used in the embedding of H has odd length, and (ii) there exists a perfect matching in the subgraph of G induced by the vertices *not* used in the embedding of H, then we obtain average-case lower bounds for the perfect matching formula $\mathrm{PM}(G)$.

Let us elaborate a bit further on the properties required from the topological minor of H in G. As mentioned previously, our embedding theorem can ensure that all paths are of odd length. To ensure the second property we in fact do not embed H directly into G but rather into a suitably chosen vertex induced subgraph $G[T]$ with the crucial property that for any set of vertices $U \subseteq T$ of odd cardinality the induced subgraph $G[V \setminus U]$ has a perfect matching. As the embedding of H will consist of an odd number of vertices we then obtain property (ii) above. Since we now want

to apply Theorem 3.2.2 not to G but to $G[T]$, we have to ensure that $G[T]$ satisfies all the conditions of that theorem. We prove what we refer to as the Partition Lemma, which asserts that an induced subgraph $G[T]$ exists that satisfies both the perfect matching property described above, as well as all conditions of Theorem 3.2.2. The proof of the Partition Lemma relies primarily on the Lovász Local Lemma and spectral bounds to obtain the desired properties.

3.3 The Circuit Tautology is Hard for Sum of Squares

Per Austrin and Kilian Risse, *"The Minimum Circuit Size Problem is Hard for Sum of Squares"*, in submission [AR22b]

The minimum circuit size problem (MCSP), is central to complexity theory: given the truthtable of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and a parameter s, the MCSP problem asks whether there is a Boolean circuit of size at most s computing f. The MCSP is clearly in NP: given a circuit of size at most s we can easily check in time $O(s \cdot 2^n)$ whether this circuit computes f.

Despite decades of research it is not known whether the MCSP problem is NP-hard. In fact establishing this has been shown to be a difficult itself [KC00; MW15; Hir18]. As such it is an important goal to at least rule out that certain classes of efficient algorithms solve the MCSP problem.

Despite the intrinsic motivation to study MCSP, there are further good arguments from a proof complexity viewpoint to study this problem. For starters, the MCSP problem is believed to be a source of hard formulas even for strong proof systems. There are not many formulas that are conjectured hard for strong proof systems and as such it is important to at least establish this claim for weak proof systems. There are some lower bounds for resolution [Raz04a; Raz04b], polynomial calculus [Raz98] and resolution over low width CNFs [Raz15], but due to the meta complexity flavor of this problem it seems difficult to prove strong lower bounds. It is worth mentioning that it has been stated as an explicit open problem [Raz22] to prove SoS degree lower bounds for the $\text{Circuit}_s(f)$ formula.

Another proof complexity angle that motivates the study of this formula is that it tells us how hard it is to prove circuit lower bounds: consider the formula $\text{Circuit}_s(\text{SAT})$, for $s = n^{\omega(1)}$. Proving that a proof system cannot refute this formula is essentially showing that this proof system cannot efficiently refute that problems in NP possess polynomial size circuits, i.e., the proof system cannot efficiently prove $\text{NP} \not\subseteq \text{P/poly}$.

The main result of this paper is an essentially optimal[3] degree lower

[3]There is an SoS refutation of degree s: see Section C.6 for details.

bound for any Boolean function.

Theorem 3.3.1. *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For $n \in \mathbb{N}$, all $s \geq n^d$ and any Boolean function $f : \{0,1\}^n \to \{0,1\}$ on n bits, SoS requires degree $\Omega_\varepsilon(s^{1-\varepsilon})$ to refute* $\text{Circuit}_s(f)$.

From this lower bound one can also extract a monomial size lower bound for a restricted class of circuits. Furthermore, it can also be shown that SoS requires large degree to refute the claim that monotone slice functions have small monotone circuits. For details on these results we refer the interested reader to Paper C.

The idea of the proof is to restrict the structure part of the formula $\text{Circuit}_s(f)$ in such a manner that the remaining satisfying assignments to the structure axioms (ignoring all other axioms) correspond to a well-behaved class of circuits $\mathcal{C}$. In a bit more detail we want that each circuit $C_\gamma \in \mathcal{C}$, where $\gamma \in \{0,1\}^m$, on input $\alpha \in \{0,1\}^n$ computes $C_\gamma(\alpha) = \oplus_{i \in N(\alpha)} \gamma_i$, for a bipartite graph $G = (\{0,1\}^n, [m], E)$.

Once we have this family of circuits $\mathcal{C}$, SoS simply has to show that none of the circuits in $\mathcal{C}$ compute the given truthtable, i.e., SoS has to show that there is no $\gamma \in \{0,1\}^m$ such that

$$\bigoplus_{i \in N(\alpha)} \gamma_i = f(\alpha) \ ,$$

for all $\alpha \in \{0,1\}^n$. But note that this is an xor constraint satisfaction problem and Grigoriev [Gri01] proved that if G is a good expander, then SoS requires linear degree in m to refute this. Thus by setting G to be a good expander we obtain the desired degree lower bound. There are some details to be filled in but this is the general gist of the argument.

3.4 The Sparse Weak Pigeonhole Principle is Hard for Resolution

Susanna F. de Rezende, Jakob Nordström, Kilian Risse, and Dmitry Sokolov, *"Exponential Resolution Lower Bounds for Weak Pigeonhole Principle and Perfect Matching Formulas over Sparse Graphs"*, CCC'20 [dRNRS20]

As previously mentioned, the main aim of proof complexity is to prove superpolynomial lower bounds for stronger and stronger proof systems to establish that NP $\neq$ coNP. A slightly different strand of research has been to study different combinatorial principles and investigate what kinds of arguments are needed to efficiently establish these principles. This quantifies, in a way, the mathematical "depth" of these statements in terms of how strong a proof system is required to prove them.

In this work we consider the resolution proof system and the pigeonhole principle (PHP). This is one of the simplest, and yet most useful, combinatorial principles in mathematics and it has been widely studied in proof complexity. We consider the somewhat unorthodox setting when m is a super-polynomial function of n. This setting has been useful to establish that resolution refutations of the claim $\mathsf{NP} \not\subseteq \mathsf{P/poly}$ are of doubly exponential size in $n^{O(1)}$ [Raz04a; Raz04b] by a reduction from the weak pigeonhole principle to the $\mathrm{Circuit}_s(f)$ formula.

These just mentioned lower bounds break with the general paradigm of proving resolution lower bounds. As mentioned in Section 2.2.1, the most common way to prove resolution refutation size lower bounds is to prove a width lower bound to then apply the width-length trade off to obtain a length (and thus size) lower bound. As the PHP can always be refuted by a resolution proof of width at most linear in number of holes n, independent of the number of pigeons $m > n$, we see that the width-length tradeoff stops giving useful lower bounds once $m \geq n^2$, as the number of variables increase as we increase the number of pigeons. However, there are resolution size lower bounds for the setting when there are $m > n^2$ pigeons: these can be shown by the seemingly ad-hoc arguments due to Raz [Raz04a] and subsequently Razborov [Raz04b].

A further peculiarity about the latter lower bounds is that they only apply to fairly dense graphs (recall that the PHP is defined over a bipartite graph $G = (U, V, E)$), while up to $m \ll n^2$ the lower bounds also hold for constant degree graphs. As such it is natural to wonder whether (i) the lower bounds for the setting $m \geq n^2$ can be strengthened to also holds for sparse graphs, and (ii) whether there is a single framework in which all these lower bounds can be proven.

We answer the latter in the affirmative and show how to generalize the pseudo-width method, devised by Razborov [Raz01; Raz03; Raz04b] in a series of 3 papers, to also apply in the sparse case.

Let us state three examples of the kind of lower bounds we obtain – the full, formal statements can be found in Paper D. The first theorem is an average-case lower bound for perfect matching formulas over a bipartite graph with a slightly superpolynomial number of pigeons.

Theorem 3.4.1 (Informal)**.** *Let* G *be a randomly sampled bipartite graph with* n *right vertices,* $m = n^{o(\log n)}$ *left vertices, and left degree* $\Theta(\log^2 m)$*. Then refuting the perfect matching formula over* G *in resolution requires length* $\exp(\Omega(n^{1-o(1)}))$ *asymptotically almost surely.*

Note that as the number of pigeons grow larger, it is clear that the left degree also has to grow – otherwise the birthday paradox will yield a small unsatisfiable subformula that can easily be refuted by brute force.

If m increases further to weakly exponential, then randomly sampled graphs no longer have good enough expansion for our techniques. However, there are explicit constructions of unbalanced expanders for which we can still get lower bounds.

Theorem 3.4.2 (Informal)**.** *There are explicitly constructible bipartite graphs* G *with* n *right vertices,* $m = \exp(O(n^{1/16}))$ *left vertices, and left degree* $\Theta(\log^4 m)$ *such that refuting* $\mathrm{PM}(G)$ *requires length* $\exp(\Omega(n^{1/8-\varepsilon}))$ *in resolution.*

Finally, for functional pigeonhole principle formulas we can also prove an exponential lower bound for *constant* left degree even if the number of pigeons is a large polynomial.

Theorem 3.4.3 (Informal)**.** *Let* G *be a randomly sampled bipartite graph with* n *right vertices,* $m = n^k$ *left vertices, and left degree* $\Theta((k/\varepsilon)^2)$*. Then refuting the functional pigeonhole principle formula over* G *in resolution requires length* $\exp(\Omega(n^{1-\varepsilon}))$ *asymptotically almost surely.*

As already mentioned, we heavily build on the pseudo-width technique devised by Razborov. In order to handle sparse bipartite graphs, we join this technique with the idea of a "closure", as introduced to proof complexity by [AR03; ABRW04]. Consider a good bipartite expander $G = (U, V, E)$. Then, the closure of a set of vertices W is a set $\mathrm{cl}(W) \supseteq W$ of vertices such that if we remove this set from G, then the resulting graph is still a fairly good expander. The maybe at first somewhat surprising fact is that for the correct setting of parameters, it can be shown that the size of the closure is linearly related to the size of W.

This notion allows us to build a matching in an iterative fashion such that the remainder of the graph is always a good expander and thus, by Hall's condition, can be extended to any small set of vertices – as if we were on a complete bipartite graph. Combining this idea with the pseudo-width technique turns out to be fairly involved and we recommend the interested reader to consult the introduction of Paper D for a more detailed proof overview.

References

[Ajt94] M. Ajtai, "The complexity of the pigeonhole principle", *Combinatorica*, vol. 14, no. 4, pp. 417–433, 1994, Preliminary version in *FOCS '88* (cit. on pp. 16, 27)

[ABRW02] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov and A. Wigderson, "Space complexity in propositional calculus", *SIAM Journal on Computing*, vol. 31, no. 4, pp. 1184–1211, Apr. 2002, Preliminary version in *STOC '00* (cit. on pp. 19, 20)

[ABRW04] ——, "Pseudorandom generators in propositional proof complexity", *SIAM Journal on Computing*, vol. 34, no. 1, pp. 67–88, 2004. DOI: 10.1137/S0097539701389944 (cit. on pp. 16, 35)

[AR03] M. Alekhnovich and A. A. Razborov, "Lower bounds for polynomial calculus: Non-binomial case", *Proceedings of the Steklov Institute of Mathematics*, vol. 242, pp. 18–35, 2003. [Online]. Available: http://people.cs.uchicago.edu/~razborov/files/misha.pdf (cit. on pp. 16, 29, 35)

[ABdR+18] A. Atserias, I. Bonacina, S. F. de Rezende, M. Lauria, J. Nordström and A. Razborov, "Clique is hard on average for regular resolution", in *Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18)*, Jun. 2018, pp. 866–877 (cit. on p. 16)

[AH19] A. Atserias and T. Hakoniemi, "Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs", in *34th Computational Complexity Conference (CCC 2019)*, A. Shpilka, Ed., ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 137, Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 24:1–24:20, ISBN: 978-3-95977-116-0. DOI: 10.4230/LIPIcs.CCC.2019.24. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2019/10846 (cit. on pp. 16, 20, 30)

[AM19] A. Atserias and M. Müller, "Automating resolution is NP-hard", in *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS '19)*, Nov. 2019, pp. 498–509 (cit. on p. 16)

[AR22a] P. Austrin and K. Risse, "Perfect matching in random graphs is as hard as tseitin", in *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, J. (Naor and N. Buchbinder, Eds., SIAM, 2022, pp. 979–1012. DOI:

10.1137/1.9781611977073.43. [Online]. Available: https://doi.org/10.1137/1.9781611977073.43 (cit. on p. 29)

[AR22b] ——, "The minimum circuit size problem is hard for sum-of-squares", Submitted Manuscript, 2022 (cit. on p. 32)

[BHK+16] B. Barak, S. B. Hopkins, J. Kelner, P. Kothari, A. Moitra and A. Potechin, "A nearly tight sum-of-squares lower bound for the planted clique problem", in *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, 2016, pp. 428–437 (cit. on p. 16)

[BP96] P. Beame and T. Pitassi, "An exponential separation between the parity principle and the pigeonhole principle", *Annals of Pure and Applied Logic*, vol. 80, no. 3, pp. 195–228, Aug. 1996, Preliminary version in *LICS '93* (cit. on p. 16)

[Ben02] E. Ben-Sasson, "Hard examples for the bounded depth Frege proof system", *Computational Complexity*, vol. 11, no. 3-4, pp. 109–136, 2002 (cit. on pp. 16, 27)

[BI10] E. Ben-Sasson and R. Impagliazzo, "Random CNF's are hard for the polynomial calculus", *Computational Complexity*, vol. 19, no. 4, pp. 501–519, 2010, Preliminary version in *FOCS '99* (cit. on p. 16)

[BW01] E. Ben-Sasson and A. Wigderson, "Short proofs are narrow—resolution made simple", *Journal of the ACM*, vol. 48, no. 2, pp. 149–169, Mar. 2001, Preliminary version in *STOC '99* (cit. on pp. 16, 18)

[Ber18] C. Berkholz, "The relation between polynomial calculus, Sherali-Adams, and sum-of-squares proofs", in *Proceedings of the 35th Symposium on Theoretical Aspects of Computer Science (STACS '18)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 96, Feb. 2018, 11:1–11:14 (cit. on p. 21)

[Bla37] A. Blake, "Canonical expressions in Boolean algebra", Ph.D. dissertation, University of Chicago, 1937 (cit. on p. 17)

[BPR97] M. Bonet, T. Pitassi and R. Raz, "Lower bounds for cutting planes proofs with small coefficients", *Journal of Symbolic Logic*, vol. 62, no. 3, pp. 708–728, Sep. 1997, Preliminary version in *STOC '95* (cit. on p. 16)

[BGIP01] S. R. Buss, D. Grigoriev, R. Impagliazzo and T. Pitassi, "Linear gaps between degrees for the polynomial calculus modulo distinct primes", *Journal of Computer and System Sciences*, vol. 62, no. 2, pp. 267–289, Mar. 2001, Preliminary version in *CCC '99* (cit. on pp. 16, 29)

[CJW06] J. Carlson, A. Jaffe and A. Wiles, Eds., *The Millennium Prize Problems*. Providence, RI: American Mathematical Society, Jun. 2006 (cit. on pp. 9, 15)

[Chu36] A. Church, "An unsolvable problem of elementary number theory", *American Journal of Mathematics*, vol. 58, no. 2, pp. 345–363, 1936 (cit. on p. 4)

[CN19] J. Chuzhoy and R. Nimavat, *Large minors in expanders*, 2019. arXiv: 1901.09349 [cs.DS] (cit. on p. 31)

[CEI96] M. Clegg, J. Edmonds and R. Impagliazzo, "Using the Groebner basis algorithm to find proofs of unsatisfiability", in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, May 1996, pp. 174–183 (cit. on pp. 19, 20, 30)

[CR79] S. A. Cook and R. Reckhow, "The relative efficiency of propositional proof systems", *Journal of Symbolic Logic*, vol. 44, no. 1, pp. 36–50, Mar. 1979, Preliminary version in *STOC '74* (cit. on pp. 8, 16)

[dRNRS20] S. F. de Rezende, J. Nordström, K. Risse and D. Sokolov, "Exponential Resolution Lower Bounds for Weak Pigeonhole Principle and Perfect Matching Formulas over Sparse Graphs", in *35th Computational Complexity Conference (CCC 2020)*, S. Saraf, Ed., ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 169, Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2020, 28:1–28:24, ISBN: 978-3-95977-156-6. DOI: 10.4230/LIPIcs.CCC.2020.28. [Online]. Available: https://drops.dagstuhl.de/opus/volltexte/2020/12580 (cit. on p. 33)

[FPPR17] N. Fleming, D. Pankratov, T. Pitassi and R. Robere, "Random ^(log n)-CNFs are hard for cutting planes", in *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '17)*, Oct. 2017, pp. 109–120 (cit. on p. 16)

[Fre79] G. Frege, *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Halle: Verlag von Louis Nebert, 1879. [Online]. Available: http://resolver.sub.uni-goettingen.de/purl?PPN538957069 (cit. on pp. 3, 16)

[FSS84] M. Furst, J. Saxe and M. Sipser, "Parity, circuits and the polynomial-time hierarchy", *Mathematical Systems Theory*, vol. 17, pp. 13–27, 1984 (cit. on p. 27)

[Göd31] K. Gödel, "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I", *Monatshefte für Mathematik und Physik*, vol. 38, no. 1, pp. 173–198, 1931 (cit. on p. 4)

[Gri01] D. Grigoriev, "Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity", *Theoretical Computer Science*, vol. 259, no. 1–2, pp. 613–622, May 2001 (cit. on pp. 16, 29, 33)

[GHP02] D. Grigoriev, E. A. Hirsch and D. V. Pasechnik, "Complexity of semi-algebraic proofs", in *STACS 2002*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 419–430 (cit. on p. 29)

[GP18] J. A. Grochow and T. Pitassi, "Circuit complexity, proof complexity, and polynomial identity testing: The ideal proof system", *J. ACM*, vol. 65, no. 6, Nov. 2018, ISSN: 0004-5411. DOI: 10.1145/3230742. [Online]. Available: https://doi.org/10.1145/3230742 (cit. on p. 16)

[Hak85] A. Haken, "The intractability of resolution", *Theoretical Computer Science*, vol. 39, no. 2-3, pp. 297–308, Aug. 1985 (cit. on p. 16)

[Hås86] J. Håstad, "Almost optimal lower bounds for small depth circuits", in *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, ser. STOC '86, Berkeley, California, United States: ACM, 1986, pp. 6–20 (cit. on p. 27)

[Hås20] ——, "On small-depth frege proofs for tseitin for grids", *Journal of the ACM*, vol. 68, pp. 1–31, 2020 (cit. on pp. 16, 28, 29)

[HR22] J. Håstad and K. Risse, "On bounded depth proofs for tseitin formulas on the grid; revisited", Accepted to FOCS'22, 2022 (cit. on p. 27)

[Hir18] S. Hirahara, "Non-black-box worst-case to average-case reductions within np", in *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, 2018, pp. 247–258. DOI: 10.1109/FOCS.2018.00032 (cit. on p. 32)

[HP17] P. Hrubeš and P. Pudlák, "Random formulas, monotone circuits, and interpolation", in *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '17)*, Oct. 2017, pp. 121–131 (cit. on p. 16)

[IPS99] R. Impagliazzo, P. Pudlák and J. Sgall, "Lower bounds for the polynomial calculus and the Gröbner basis algorithm", *Computational Complexity*, vol. 8, no. 2, pp. 127–144, 1999 (cit. on pp. 20, 30)

[IOSS16] D. Itsykson, V. Oparin, M. Slabodkin and D. Sokolov, "Tight lower bounds on the resolution complexity of perfect matching principles", *Fundamenta Informaticae*, vol. 145, no. 3, pp. 229–242, Aug. 2016, Preliminary version in *CSR '15* (cit. on p. 16)

[KC00] V. Kabanets and J.-Y. Cai, "Circuit minimization problem", in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, ser. STOC '00, Portland, Oregon, USA: Association for Computing Machinery, 2000, pp. 73–79, ISBN: 1581131844. DOI: 10.1145/335305.335314. [Online]. Available: https://doi.org/10.1145/335305.335314 (cit. on p. 32)

[KR96] J. Kleinberg and R. Rubinfeld, "Short paths in expander graphs", in *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, ser. FOCS '96, USA: IEEE Computer Society, 1996, p. 86 (cit. on p. 31)

[KMOW17] P. K. Kothari, R. Mori, R. O'Donnell and D. Witmer, "Sum of squares lower bounds for refuting any csp", in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2017, Montreal, Canada: Association for Computing Machinery, 2017, pp. 132–145, ISBN: 9781450345286. DOI: 10.1145/3055399.3055485. [Online]. Available: https://doi.org/10.1145/3055399.3055485 (cit. on p. 16)

[Kra19] J. Krajíček, *Proof Complexity*, ser. Encyclopedia of Mathematics and Its Applications. Cambridge University Press, Mar. 2019, vol. 170 (cit. on p. 19)

[KPW95] J. Krajíček, P. Pudlák and A. R. Woods, "An exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle", *Random Structures and Algorithms*, vol. 7, no. 1, pp. 15–40, 1995, Preliminary version in *STOC '92* (cit. on pp. 16, 27)

[Kri19] M. Krivelevich, "Expanders – how to find them, and what to find in them", in *Surveys in Combinatorics 2019*, ser. London Mathematical Society Lecture Note Series. Cambridge University Press, 2019, pp. 115–142. DOI: 10.1017/9781108649094.005 (cit. on p. 31)

[KN19] M. Krivelevich and R. Nenadov, "Complete Minors in Graphs Without Sparse Cuts", *International Mathematics Research Notices*, May 2019, rnz086, ISSN: 1073-7928. DOI: 10.1093/imrn/rnz086. eprint: https://academic.oup.com/imrn/article-pdf/doi/10.1093/imrn/rnz086/28672004/rnz086.pdf. [Online]. Available: https://doi.org/10.1093/imrn/rnz086 (cit. on p. 31)

[LN17] M. Lauria and J. Nordström, "Graph colouring is hard for algorithms based on Hilbert's Nullstellensatz and Gröbner bases", in *Proceedings of the 32nd Annual Computational Complexity Conference (CCC '17)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 79, Jul. 2017, 2:1–2:20 (cit. on p. 16)

[LS91] L. Lovász and A. Schrijver, "Cones of matrices and set-functions and 0-1 optimization", *SIAM Journal on Optimization*, vol. 1, no. 2, pp. 166–190, 1991 (cit. on p. 30)

[MPW15] R. Meka, A. Potechin and A. Wigderson, "Sum-of-squares lower bounds for planted clique", in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, Jun. 2015, pp. 87–96 (cit. on p. 16)

[MN15] M. Mikša and J. Nordström, "A generalized method for proving polynomial calculus degree lower bounds", in *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 33, Jun. 2015, pp. 467–487 (cit. on pp. 16, 29)

[MW15] C. D. Murrayand and R. R. Williams, "On the (Non) NP-Hardness of Computing Circuit Complexity", in *30th Conference on Computational Complexity (CCC 2015)*, D. Zuckerman, Ed., ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 33, Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015, pp. 365–380, ISBN: 978-3-939897-81-1. DOI: 10.4230/LIPIcs.CCC.2015.365. [Online]. Available: http://drops.dagstuhl.de/opus/volltexte/2015/5074 (cit. on p. 32)

[PRT22] T. Pitassi, P. Ramakrishnan and L. Tan, "Tradeoffs for small-depth frege proofs", in *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, Los Alamitos, CA, USA: IEEE Computer Society, Feb. 2022, pp. 445–456. DOI: 10.1109/FOCS52979.2021.00052. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/FOCS52979.2021.00052 (cit. on p. 28)

[PBI93] T. Pitassi, P. Beame and R. Impagliazzo, "Exponential lower bounds for the pigeonhole principle", *Computational Complexity*, vol. 3, pp. 97–140, 1993, Preliminary version in *STOC '92* (cit. on pp. 16, 27)

[PRST16] T. Pitassi, B. Rossman, R. Servedio and L.-Y. Tan, "Polylogarithmic Frege depth lower bounds", in *Proceedings of the 48th Annual ACM Symposium on Theory of Computing (STOC '16)*, Jun. 2016, pp. 644–657 (cit. on pp. 16, 27, 28, 30)

[Pot17] A. Potechin, "Sum of squares lower bounds from symmetry and a good story", *CoRR*, vol. abs/1711.11469, 2017. arXiv: 1711.11469. [Online]. Available: http://arxiv.org/abs/1711.11469 (cit. on p. 16)

[Pud97] P. Pudlák, "Lower bounds for resolution and cutting plane proofs and monotone computations", *Journal of Symbolic Logic*, vol. 62, no. 3, pp. 981–998, Sep. 1997 (cit. on p. 16)

[Raz04a] R. Raz, "Resolution lower bounds for the weak pigeonhole principle", *Journal of the ACM*, vol. 51, no. 2, pp. 115–138, Mar. 2004, Preliminary version in *STOC '02* (cit. on pp. 16, 32, 34)

[Raz88] A. A. Razborov, "Bounded-depth formulae over the basis {and, xor} and some combintorial problems (in russian)", *Problems of Cybernetics. Complexity Theory and Applied Mathematical Logic*, pp. 149–166, 1988 (cit. on p. 27)

[Raz98] ——, "Lower bounds for the polynomial calculus", *Computational Complexity*, vol. 7, no. 4, pp. 291–324, Dec. 1998 (cit. on pp. 16, 29, 32)

[Raz01] ——, "Improved resolution lower bounds for the weak pigeonhole principle", Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR01-055, Jul. 2001 (cit. on p. 34)

[Raz03] ——, "Resolution lower bounds for the weak functional pigeonhole principle", *Theoretical Computer Science*, vol. 1, no. 303, pp. 233–243, Jun. 2003 (cit. on p. 34)

[Raz04b] ——, "Resolution lower bounds for perfect matching principles", *Journal of Computer and System Sciences*, vol. 69, no. 1, pp. 3–27, Aug. 2004, Preliminary version in *CCC '02* (cit. on pp. 32, 34)

[Raz15] ——, "Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution", *Annals of Mathematics*, vol. 181, no. 2, pp. 415–472, Mar. 2015 (cit. on p. 32)

[Raz17] ——, "On the width of semialgebraic proofs and algorithms", *Math. Oper. Res.*, vol. 42, no. 4, pp. 1106–1134, Nov. 2017, ISSN: 0364-765X. DOI: 10.1287/moor.2016.0840. [Online]. Available: https://doi.org/10.1287/moor.2016.0840 (cit. on p. 30)

[Raz22] ——, *Open problems*, 2022. [Online]. Available: https://people.cs.uchicago.edu/~razborov/teaching/index.html (visited on 05/04/2022) (cit. on p. 32)

[Rec75] R. A. Reckhow, "On the lengths of proofs in the propositional calculus", Ph.D. dissertation, University of Toronto, 1975. [Online]. Available: https://hdl.handle.net/1807/100390 (cit. on p. 19)

[Rii93] S. Riis, "Independence in bounded arithmetic", Ph.D. dissertation, University of Oxford, 1993 (cit. on p. 29)

[Sha49] C. E. Shannon, "The synthesis of two-terminal switching circuits", *The Bell System Technical Journal*, vol. 28, no. 1, pp. 59–98, 1949. DOI: 10.1002/j.1538-7305.1949.tb03624.x (cit. on p. 15)

[Sip83] M. Sipser, "Borel sets and circuit complexity", in *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, ser. STOC '83, New York, NY, USA: ACM, 1983, pp. 61–69, ISBN: 0-89791-099-0 (cit. on p. 27)

[Smo87] R. Smolensky, "Algebraic methods in the theory of lower bounds for boolean circuit complexity", in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, ser. STOC '87, New York, New York, United States: ACM, 1987, pp. 77–82, ISBN: 0-89791-221-7 (cit. on p. 27)

[Tse68] G. Tseitin, "On the complexity of derivation in propositional calculus", in *Structures in Constructive Mathematics and Mathematical Logic, Part II*, A. O. Silenko, Ed., Consultants Bureau, New York-London, 1968, pp. 115–125 (cit. on p. 16)

[Tur37] A. M. Turing, "On computable numbers, with an application to the Entscheidungsproblem", *Proceedings of the London mathematical society*, vol. 2, no. 1, pp. 230–265, 1937 (cit. on p. 4)

[UF96] A. Urquhart and X. Fu, "Simplified lower bounds for propositional proofs", *Notre Dame Journal of Formal Logic*, vol. 37, no. 4, pp. 523–544, 1996 (cit. on p. 27)

[Yao85] A. C. Yao, "Separating the polynomial-time hierarchy by oracles", in *Foundations of Computer Science, 1985., 26th Annual Symposium on*, Oct. 1985, pp. 1–10. DOI: 10.1109/SFCS.1985.49 (cit. on p. 27)

Part II

Included Papers