Expansion in Proof Complexity

Thesis submitted for the degree

Doctor of Philosophy

by

Eli Ben-Sasson

Accepted by the Hebrew University
July 22 2002

Expansion in Proof Complexity

Thesis submitted for the degree

Doctor of Philosophy

by

Eli Ben-Sasson

Accepted by the Hebrew University
July 22 2002

This work was carried out under the supervision of

Professor Avi Wigderson

Acknowledgements

It is a great pleasure to thank the many people who helped and supported me through my studies. First and foremost are my parents, **Zami** and **Yehudit**, who encouraged my curiosity from the first day I can remember myself, and probably even before that. All that is mine - is theirs.

Within the academic world, above all I thank my great adviser, **Avi Wigderson**. His capabilities as a researcher need no advertising, and the same goes for his didactic and tutorial abilities. In addition to brilliant professional advice, his optimism, personal care, passion for research, and proper attitude to life and work, are things I can only hope to implement in my career.

The **Hebrew University**, where I studied in the past eight years, is the stimulating intellectual environment in which I grew and matured. Of the many amazing teachers here, special thanks are due to the other theorists I spoke and consulted with, and whose door when needed, was always open: **Michael (Miki) Ben-Or** and **Nati Linial**.

One cannot mention the supporting environment without thanking the smart and happy theory bunch, whom I will miss greatly: **Yonatan Bilu, Danny Gutfreund, Tamir Hazan, Shlomo Hoory, Boaz Leslau, Avner Magen, Ahuva Mu'alem, Eyal Rozenman, Ronen Shaltiel** and **Amir Shpilka**.

During my Graduate studies, I had the fortune of working with many excellent scholars: **Alexander A. (Sasha) Razborov**, **Russell Impagliazzo**, **Ran Raz**, **Toni Pitassi**, **Maria Luisa Bonet**, **Mikhail (Misha) Alekhnovich**, **Nicola Galesi, Jan Krajíček**, and **Pavel Pudlák**. With some of them I had the fortune of publishing papers, and with all of them I had many exciting conversations about our common research theme, in addition to good answers to my questions.

Last but not least, I thank **Ayelet Sompolinsky**, my past girlfriend, current fiancee, and future wife, for the love, encouragement and support, through bad and good, close and far, easy and hard.

Abstract

In this thesis we explore several well-known proof systems for **co-SAT** - the language of unsatisfiable Boolean circuits. We prove lower bounds for important complexity measures, such as proof size, and proof space. Our main contribution is presenting quite general, and very natural, combinatorial conditions on the input circuit, that ensure high proof complexity. More explicitly, we propose the following paradigm:

If C is an unsatisfiable circuit that is a *conjunction* of Boolean constraints, and

1. C is *expanding*, i.e. every small subset of the constraints has many variables.

2. Each constraint can be easily satisfied, e.g. is a clause.

Then proving the claim "$C \in \mathbf{co-SAT}$" is *hard* for simple proof systems such as *resolution*, the *polynomial calculus*, and (in some cases) *bounded depth Frege*.

We demonstrate this principle by proving *size* lower bounds for the three systems mentioned above, as well as *space* lower bounds for the *resolution* system.

At the core of our lower bound technique for *resolution* and the *polynomial calculus* lies the *width method*, which is the main technical contribution of this thesis. This method reduces *proof size* lower bounds to *proof width* lower bounds (the width of a proof is the maximal number of variables appearing in a clause of the proof). It turns out that proving *width* lower bounds is much simpler than proving *size* lower bounds directly. Using this method we greatly simplify almost all previously known size lower bounds for resolution, and present many interesting new ones for both *resolution* and the *polynomial calculus*.

The *width method* also gives rise to an automated theorem proving algorithm that finds a relatively short proof for C whenever a short proof indeed exists. We explore this algorithm and show that sometimes it outperforms some commonly used algorithms by an exponential factor. To this end we present the best gap between *general* resolution size and *treelike* resolution size.

Our thesis starts with a general introduction to proof complexity, which is intended to be readable by any person with basic knowledge of computational complexity. This is followed by our analysis of the three proof systems. We start with *resolution*, the weakest, followed by the *polynomial calculus*, and end with *bounded depth Frege*, the strongest of the three.

Contents

Chapter 1

Introduction

1.1 Proofs and Computations

Proofs and *computations* are the two most important professional activities mathematicians carry out. One of the most influential mathematical discoveries of the past 200 years was that both these activities can be automated. Proofs can be formalized into logical theories, and then verified in an automated fashion, and computations can be done mechanically by computing devices - computers. This discovery, and the subsequent formalization of proofs and computations, made it clear that these mathematical activities, can be investigated using mathematical methods. Thus, mathematics is used to understand the mathematical process itself, as in one of the self-referential drawings of M. C. Escher. In the beginning of the last (20th) century, the self-referential nature of logic and computation, gave rise to some of the deepest philosophical and mathematical insights of the 20th century. Gödel's *Incompleteness theorem* says that some natural sentences are true, yet cannot be proved, and Turing's *Undecidability theorem* says that some natural functions are well defined, yet cannot be computed. Both proofs use essentially the same methods, and show a basic idea that underlies our thesis: understanding *proofs*, and understanding *computations*, are related problems. Let us start by defining more rigorously the notion of a *proof.*

A language $\mathbf{L}$ is a set of strings over some finite alphabet, a *claim* is some string, and so is a *proof.* A *proof system* for $\mathbf{L}$ is a method (algorithm) for verifying proofs of claims of the form "$x \in \mathbf{L}$". Naturally, we would like such a method to be *sound* and *complete*, meaning that x will have a proof iff $x \in \mathbf{L}$. Many classical first order theories taught in introductory courses in mathematical logic have these properties.

We wish to emphasize one *computational* aspect of proof systems. A proof should be verifiable *efficiently* from a computational point of view. In the standard interpretation, *efficient* means *decidable in deterministic polynomial time*, and indeed, all classical proof systems (such as ZFC) are efficient, under this definition[1].

Here is a formal definition of the proof systems we investigate in this thesis.

[1] One may define *efficient* in alternative ways, e.g. *decidable in randomized polynomial time*, or *decidable using an interactive protocol between a limited verifier and unlimited prover.* These definitions and similar ones, that introduce interaction and randomness into the proof process, have proved very useful in computational complexity research in the past 20 years. See section 1.4 for more information on this subject.

Definition 1.1 (Deterministic Proof System) *Let* $\mathbf{L} \subseteq \Sigma^\star$ *be a language over the finite alphabet* Σ. *A* deterministic proof system *for* $\mathbf{L}$ *is a deterministic two-tape Turing machine* $\mathbf{A_L}(x, \pi)$ *with the following properties (think of "*$x \in \mathbf{L}$*" as the claim being proved,* π *is the proof of this claim, and* $\mathbf{A_L}$ *is the verification algorithm).*

Soundness: *If for some* $\pi \in \Sigma^\star$, $\mathbf{A_L}$ *accepts* (x, π), *then* $x \in \mathbf{L}$.

Completeness: *If* $x \in \mathbf{L}$ *there exists* $\pi \in \Sigma^\star$ *such that* $\mathbf{A_L}$ *accepts* (x, π).

Efficiency: $\mathbf{A_L}$ *runs in polynomial time in the size of its inputs* $|x| + |\pi|$.

The *efficiency* requirement is where computational complexity comes in.

1.2 Proof Systems for Languages in Co-NP

Some classes of languages, such as **NP**, have very good deterministic proof systems. Let $\phi(x_1, \ldots, x_n)$ be a Boolean circuit over some complete Boolean basis (say $\{\neg, \vee, \wedge\}$). If ϕ is satisfiable, the following claim has a short proof

$$\exists x_1, \ldots, x_n \quad (\phi(x_1, \ldots, x_n) = 1)$$

Its proof is simply a *satisfying assignment* $\alpha \in \{0, 1\}^n$. It is not hard to see that any language in **NP** has short proofs, because by definition $\mathbf{L} \in \mathbf{NP}$ iff there exists a constant c and a deterministic polytime Turing machine $\mathbf{A}$ such that

$$\mathbf{L} = \{x : \exists y \quad |y| \leq |x|^c \mathbf{A}(x, y) = \text{ accept }\}.$$

Thus, the proof of the claim "$x \in \mathbf{L}$" will simply be the corresponding y, such that $\mathbf{A}$ accepts (x, y).

This thesis concentrates on deterministic proof systems for **co-NP** complete languages such as $\mathbf{co-SAT}$:

$$\mathbf{co-SAT} \stackrel{\text{def}}{=} \{\phi(x_1, \ldots, x_n) : \quad n \in \mathbb{N}, \quad \forall x_1, \ldots, x_n \quad \phi(x_1, \ldots, x_n) = 0\}.$$

Such systems are also called *propositional proof systems*, because the language is that of Boolean contradictions over *propositional variables.* From here on, whenever we mention the term *proof system*, we will mean a *propositional* proof system, i.e. a *deterministic proof system* for some **co-NP** complete language.

Proof systems have been defined and investigated for several **co-NP** complete problems such as **co-3-Colorability** [Pitassi and Urquhart ('92)], and unsatisfiability of integer constraints (see e.g. [Beame and Pitassi ('98)] for a survey of related results). All our systems, except for the polynomial calculus described in chapter 5, will be for the language $\mathbf{co-SAT}$, which has been investigated most intensively.

A member of $\mathbf{co-SAT}$, also called a *contradiction*, is just a fancy Boolean circuit computing the constant **0** function. A natural proof system for **co-SAT** is the following. A proof will be a concatenation of all possible assignments $\alpha \in \{0, 1\}^n$, and $\phi(x_1, \ldots, x_n)$ will be accepted iff all such α do not satisfy ϕ. The main disadvantage of this proof system is the huge length of its proofs. If we wish to avoid looking through all possible assignments, we must investigate the structure of the circuit ϕ, and hope to prove some general properties of ϕ, without having to verify them on all possible inputs. A simple example will help clarify this intuition. If $\phi = (g \wedge \neg g) \wedge h$, we may argue loosely that no assignment can satisfy g and $\neg g$ at the same time, and hence, regardless of the identity and structure of g and h, and regardless of their input size, ϕ is unsatisfiable. It turns out, as we shall shortly see, that this informal argument can be made into a formal one, verifiable in an automated fashion by a Turing machine. Unfortunately, in the general case the unsatisfiability of ϕ will not be so conspicuous.

1.3 A Note on Syntax and Semantics

Before continuing, we make two remarks about syntax and semantics.

- A *tautology* $\phi(x_1, \ldots, x_n)$ is a Boolean circuit that computes the constant $\mathbf{1}$ function. Clearly ϕ is a tautology iff $\neg\phi$ is a contradiction, and hence a short proof of $\neg\phi$ is also a short proof of ϕ. Proof systems for the language of *contradictions* and for the language of *tautologies* are dual. For the sake of clarity, in this thesis we discuss only proof systems for contradictions.

- Our proof systems deal with *syntactical* objects, such as formulas and circuits. The Boolean function computed by such a circuit is a *semantic* object, it is the "proper interpretation" of the circuit. Thus, in order to speak of the function computed by a circuit one needs to define a mapping from the syntactical world of circuits, to the semantical one of functions. We omit these definitions, and assume the standard interpretation which can be found in any introductory book of mathematical logic. E.g. the interpretation of 1 is *True*, 0 is *False*, and the interpretation of the circuit $\neg x$ is the Boolean function which outputs *True* on input *False* and *False* on input *True*. **In particular, under this convention, a Boolean circuit will be viewed also as the Boolean function it computes.**

1.4 Probabilistic Proofs

In this subsection we briefly discuss non-classical verification methods that have been very productive in computational complexity, and are not part of this thesis. Investigation of various proof methods is today an important part of computational complexity. In many settings, a proof is defined to be a two player game, played by a *prover*, and a *verifier*. The prover wants to convince the verifier that ϕ is true, and the verifier wants to be sure he isn't being cheated. In this context, the *proof* is the interaction between these two players, i.e. the messages they send. A proof system for $\mathbf{L}$ is an interaction protocol promising for all a, if $a \in \mathbf{L}$ then there is some proof that convinces the verifier with high probability, and if $a \notin \mathbf{L}$, then all proofs will fail to convince him with high probability.

Some of the deepest results in computational complexity and cryptography in the past decades have been found during the investigation of these proofs. This includes the PCP theorem [Arora *et al.*('98)], [Arora and Safra ('98)], IP=PSPACE [Lund *et al.* ('92), Shamir ('92)], and zero-knowledge proofs [Goldwasser *et al.* ('89), Goldreich *et al.* ('91)], to mention a few. Most of these results depend heavily on the existence of *randomness* in the proof process: The verifier is willing to give up some of his confidence in the result, and in return can get very efficient (short and simple) proofs for very deep claims.

In this thesis, we will discuss proofs that are without randomness, and are totally error free. Thus, a proof will simply be a string that convinces us that $a \in \mathbf{L}$. In this situation, if the verifier is convinced by a proof, then the claim is indeed true, and if the claim is true, then there exists some proof the verifier accepts.

1.5 Proof Complexity and Computational Complexity

The central aim of computational complexity is to understand the question of **P vs. NP**. The central aim of proof complexity is to gain similar understanding of **NP vs. co − NP**. There are many connections between proof and circuit complexity, and we wish to emphasize some of them, as well as an important difference between the two fields.

The problem of **P vs. NP** is that of efficiently *finding* a short witness to a polynomial predicate, if such a witness exists. The problem of **NP vs. co − NP** is that of the *existence* of short witnesses (proofs) for *universally quantified* statements.

If A is an algorithm deciding an **NP**-complete language **L**, it gives rise to a natural proof system for $\bar{\mathbf{L}}$: a proof of the claim "$x \in \bar{\mathbf{L}}$" is simply the description of the running of A on input x. Thus, a lower bound on the proof size of x in the corresponding proof system, immediately gives worst-case lower bounds on the running time of the algorithm A. For example, the lower bounds on treelike resolution *proof size* for random CNF formulas (section 2.6), give similar lower bounds on the *running time* of the important class of DLL-algorithms for the Satisfiability problem (see chapter 4 for a definition of these algorithms).

Let **Q** be a proof system for the language **L**. A natural computational question, that is also of practical importance for formal verification and artificial intelligence, is that of *automated theorem proving*: given $x \in \mathbf{L}$, efficiently find a **Q**-proof of x. When **L** is some **NP**-complete language, we get exactly the question of **P vs. NP**. A similar question might be asked about a **co − NP** complete language, as we shall indeed do later on (chapter 4). A lower bound on the proof size of x is also a trivial lower bound on the running time of any corresponding automated theorem prover.

The connection between *proof* and *circuit* complexity is also strong. The basic problem of circuit complexity is the following: given a Boolean function $F : \{0,1\}^n \rightarrow \{0,1\}$, and a circuit class $\mathbb{C}$ (e.g. constant depth circuits, formulas etc.) what is the minimal size of a circuit in $\mathbb{C}$ computing F ? The basic problem of proof complexity is: given a Boolean contradiction ϕ, and a proof system **Q**, what is the shortest **Q**-proof of ϕ ?

We must point out a big difference between the two questions. In circuit complexity, simple counting arguments show that most functions require large circuits. The "only" problem is to point out one specific natural function that is hard. In proof complexity, we don't even have the counting argument. Thus, it is totally conceivable that a proof system exists in which every contradiction has a *linear size* proof.

Because of the lack of counting arguments, most lower bounds for proof systems are unconditional, and must exploit the underlying structure of the proof system, and of the claim being proved. Thus, our lower bounds will resemble in techniques and scope the unconditional lower bounds of circuit complexity. Our success will be also limited to those proof systems that correspond to simple circuit classes, whereas for strong systems, corresponding to complicated circuit classes (e.g. extended Frege), we have no lower bounds.

We end this section with an intriguing connection between proof and circuit complexity, that was our original motivation for researching this subject. We know empirically that proving circuit lower bounds is extremely hard, and no explicit lower bounds for general circuits are known. Could it be that proving circuit lower bounds is hard, from a formal and logical point of view ? Initial positive evidence in this direction was given by Razborov and Rudich, in their paper about *natural proofs* [Razborov and Rudich ('94)]. They showed that all circuit lower bounds have certain *natural* properties (under the formal definition of *natural* they give there), and that modulo some cryptographic assumptions, there can be no *natural* proof of general circuit lower bounds. Thus, a circuit lower bound for arbitrary circuits might require some totally new proof techniques. The lower bounds of [Alekhnovich *et al.* A ('00)] for generator contradictions (presented in sections 2.10 and 5.4), are a step in the direction of formalizing the *natural proof* ideology and showing that circuit lower bounds should require long proofs, and hence could be considered as *hard problems*, from a theoretical, as well as practical, point of view. The ultimate goal in this direction, which currently seems well beyond our reach, is to prove the *independence* of the claim $\mathbf{P} \neq \mathbf{NP}$ from some strong first order theory, such as ZFC.

1.6 Fundamental Problems of Propositional Proof Complexity

1.6.1 Proof Size

The most interesting parameter of a proof is its *size*, measured by the number of symbols in it, and the relation of the proof size to the size of the input contradiction ϕ. Any true mathematical claim about a *finite* world can be formulated as a tautology $\phi(x_1, \dots, x_n)$. A natural question is the following: given such a claim ϕ, how *hard* is it to prove ϕ ? If we interpret *hard* as *long*, we arrive at the question of proof size. The trivial proof presented in the previous section always has size $n \cdot 2^n$. Thus, we shall compare the proof size of a claim ϕ with $|\phi|$ on the one hand, and with 2^n on the other. If the minimal proof of ϕ is much larger than $|\phi|$, we call ϕ a *hard* claim (typically we shall consider ϕ with $|\phi(x_1, \dots, x_n)| = n^{O(1)}$). For a proof system $\mathbf{P}$ and ϕ a circuit, we define the proof size of ϕ in $\mathbf{P}$, denoted $S_{\mathbf{P}}(\phi)$, to be the minimal size of a proof of ϕ, if such a proof exists, and ∞ otherwise. Listed are a few more important questions regarding proof size.

1. A proof system for the language $\mathbf{L}$ is called *super* if all members of the language have *short proofs*, i.e. there exists a constant c, such that for any $x \in \mathbf{L}$, there exists a proof of x of length $|x|^c$. for $\mathbf{L}$,

   The basic theorem of Cook and Reckhow states that a super proof system exists iff **NP=co-NP** [Cook and Reckhow ('79)]. Proving super-polynomial lower bounds for *all* proof systems is the most important problem of propositional proof complexity, which currently seems beyond our power.

2. Proof systems can be ordered with respect to their relative efficiency. **P** *polynomially simulates* **Q** if for some constant c, and every contradiction $\mathcal{F}$, $S_{\mathbf{P}}(\mathcal{F}) \leq (S_{\mathbf{Q}}(\mathcal{F}))^c$. **P** is said to be *optimal* if it polynomially simulates all other proof systems. The existence of an optimal proof system, could be stated as second in importance only to the previous question. Not surprisingly, this question remains open and seems to be also beyond our reach. We do have however a partial ordering of some of the commonly used proof systems (see e.g. [Krajíček ('95)] and [Urquhart ('95)]). The existence of an optimal proof system is connected to other complexity related questions, and we refer the interested reader to [Krajíček and Pudlák ('89)] for more information on these connections.

3. Given a specific proof system **P**, we ask whether **P** is super. This question is especially interesting for those natural proof systems that correspond to various methods of mathematical reasoning (logic, algebra, etc.). Most of this thesis deals with proving lower bounds on proof size for some simple and natural proof systems, showing that they are far from super, by proving exponential lower bounds.

4. Given a specific proof system **P**, we try to understand as much as we can the mathematical properties of claims that are easy/hard to prove in **P**. Once again, for some simple proof systems we give in this monograph a characterization of some general classes of hard claims. Naturally, our understanding decreases as the power of the proof system increases.

1.6.2 Proof DAG Structure

All proof systems we currently know of, except the trivial one presented earlier, are *sequential*. A sequential proof of (the unsatisfiability of) ϕ is a *sequence* of circuits, starting with ϕ, as an assumption, and each circuit in the sequence is derived from earlier formulas in the sequence, by a *sound derivation rule*. The main properties of derivation rules are:

1. Given a set of circuits $\{\phi_1, \dots \phi_k, \psi\}$ and a rule, one can decide in polynomial time whether ψ can be deduced from $\{\phi_1, \dots \phi_k\}$ by the rule. $\phi_1, \dots \phi_k$ are the *premises* of the rule, and ψ is its *consequence*. A derivation rule with no premises is called an *axiom scheme*, and an instance of this rule is an *axiom*.

2. The rules are sound: if ψ is deduced from $\phi_1, \dots \phi_k$ then any assignment that satisfies all of $\phi_1, \dots \phi_k$ satisfies ψ as well. We denote this by $\{\phi_1, \dots \phi_k\} \models \psi$.

If we derive $\mathbf{0}$ from some formula ϕ, via repeated applications of sound derivation rules, then by the transitivity of property 2, we have shown $\phi \models \mathbf{0}$, and hence $\phi \in \mathbf{co-SAT}$. By property 1, such a proof can be verified in polynomial time. The only missing property we need in order to make this system a proof system, is *completeness*, which in practice can be achieved by allowing sufficiently (but finitely) many derivation rules.

A *sequential proof system* is defined by a finite set of sound derivation rules. A *derivation* of a formula h from a formula ϕ is a sequence $\{\psi_1, \dots \psi_s\}$ where $\psi_1 = \phi$, $\psi_s = h$ and for $i = 2 \dots s$, ψ_i is derived from some $\mathcal{G} \subseteq \{\psi_1, \dots \psi_{i-1}\}$ via one of the derivation rules. A *proof* of ϕ is a derivation of $\mathbf{0}$. The proof system accepts a pair (ϕ, π) iff π is a proof of ϕ.

A sequential proof gives rise to a DAG (Directed Acyclic Graph) called the *proof-DAG*. The nodes of the DAG are the circuits of the proof, and a node has one edge coming in from each one of the premises it was derived from. Thus, the *sources* of the proof-DAG are either the assumption ϕ, or axioms of the proof system, and wlog the DAG has a single *sink* node, which is labeled $\mathbf{0}$. A proof is said to be *tree-like* if its DAG is a tree.

Almost as important as the proof size is the *proof length* of ϕ in a sequential proof system $\mathbf{P}$, which is the minimal number of circuits in a proof of ϕ, if a proof exists, and ∞ otherwise. In some simple proof systems, such as *resolution*, these two measures are polynomially related. Additional questions about proof DAG structure are:

1. What is the minimal treelike proof of ϕ, and what is the relation of this proof to the minimal general DAG proof of ϕ. A large gap displays a different form of complexity of the claim ϕ: any proof of ϕ has many lines that must be reused many times in the course of the proof. Such a complex structure also has implications to *automated theorem proving* (see section 1.6.3). In chapter 4 we show that this gap can be exponentially large for resolution.

2. What are the combinatorial properties of the proof DAG: depth of the DAG ? Is there a Size-Depth trade-off ? What is the minimal space needed to follow a proof of ϕ, where each line of the proof requires a unit of space ? This question of proof space for resolution is discussed in chapter 3.

1.6.3 Automated Theorem Proving

The proof size question is that of the *existence* of a short proof for a true claim ϕ. Tightly connected to it, and interesting in its own right is the question of *efficiently finding* a short proof of ϕ. This problem, called *automated theorem proving* is also of great practical importance. For instance, suppose we design a chip C that computes efficiently some Boolean function $g(x_1, \ldots, x_n)$. Surely, we would like to be sure that $C \equiv g$, i.e. that the chip we designed indeed operates correctly. One way to do this is by finding a proof of the tautology $C(x_1, \ldots, x_n) \equiv g(x_1, \ldots, x_n)$ in some proof system. A different motivation arises from *artificial intelligence* problems. In this context we want a machine to deduce automatically some correct statements about the world. One way to do this is to use a formal logical proof system, and search for a proof of the proper statement.

Let $\mathbf{Q}$ be a proof system. Clearly, the run time of any automated theorem prover that searches for $\mathbf{Q}$-proofs, is lower bounded by $\mathbf{S_Q}(\phi)$. Thus, it is interesting to compare the running time of such an algorithm with the proof size of ϕ. For $f : \mathbb{N} \to \mathbb{N}$ a function, we call a proof system *f-automatizable* if there is an algorithm that finds a proof of ϕ in time $f(S_{\mathbf{Q}}(\phi))$. In section 2.4 we show that *treelike resolution* is $O(S^{\log S})$-automatizable.

We end this section with a letter from Gödel to von-Neumann, from 1956, where the question of *finding* a short proof of first order formulas is initially raised. It is interesting to notice that the common belief today contradicts Gödel's intuition that there is an efficient method for deciding the existence of short proofs.

"One can obviously easily construct a Turing machine, which for every formula F in first order predicate logic and every natural number n, allows one to decide if there is a proof of F of length n (length = number of symbols). Let $\psi(F,n)$ be the number of steps the machine requires for this and let $\varphi(n) = \max_F \psi(F,n)$. The question is how fast $\varphi(n)$ grows for an optimal machine. One can show that $\varphi(n) \geq k \cdot n$. If there really were a machine with $\varphi(n) \sim k \cdot n$ (or even $\sim k \cdot n^2$), this would have consequences of the greatest importance. Namely, it would obviously mean that in spite of the undecidability of Entscheidungsproblem, the mental work of a mathematician concerning Yes-or-No questions could be completely[2] replaced by a machine. After all, one would simply have to choose the natural number n so large that when the machine does not deliver a result, it makes not sense to think more about the problem. Now it seems to me, however, to be completely within the realm of possibility that $\varphi(n)$ grows that slowly. Since (1) it seems that $\varphi(n) \geq k \cdot n$ is the only estimation which one can obtain by a generalization of the proof of the undecidability of the Entscheidungsproblem; and (2) after all $\varphi(n) \sim k \cdot n$ (or $\sim k \cdot n^2$) only means that the number of steps as opposed to trial and error can be reduced from N to $\log N$ (or $(\log N)^2$). However, such strong reductions appear in other finite problems, for example in the computation of the quadratic residue symbol using repeated application of the law of reciprocity. It would be interesting to know, for instance, the situation concerning the determination of primality of a number and how strongly <u>in general</u> the number of steps in finite combinatorial problems can be reduced with respect to simple exhaustive search." [3]

1.7 Proof Systems Discussed in the Thesis

We will discuss three proof systems:

1. Resolution.
2. The polynomial calculus.
3. Bounded depth Frege.

Each system is stronger than (polynomially simulates) the preceding one, and we shall deal with them in order of increasing power. As one might expect, the stronger the proof system, the weaker our lower bounds.

Frege systems are proof systems for the $\mathbf{co-NP}$ complete language $\mathbf{co-SAT}$. These systems are based on logical derivation rules, the most important being the *cut* rule (see section 6.3 for a definition). A *bounded-depth Frege* system restricts all lines in the proof to be bounded depth formulas.

Resolution is actually *depth one Frege*, since all lines are restricted to be *clauses*, i.e. disjunctions of literals, that have depth 1. This system has received the most

[2] except for the setting up of axioms

[3] We thank the Institute for Advanced Studies, copyright holder of the German original, and Oxford University Press, copyright holder of English translation, for allowing us to include this quotation. The translation is by Peter Clote [Clote and Krajíček ('93)]; the footnote and the underlining are Gödel's.

attention in proof complexity, because of its simplicity and connection to automated theorem proving algorithms.

The *polynomial calculus* is an algebra based proof system for the different **co − NP** complete language **co − POLY − SAT** which is the language of sets of polynomials that have no common root (see chapter 5 for definitions). This language is closely related to **co − SAT** since any circuit C can be encoded as a set of polynomials $\mathcal{P}$ such that C is satisfiable iff $\mathcal{P}$ has a common root.

Resolution is a proof system for CNF formulas, i.e. for *sets* of clauses, and the polynomial calculus is a proof system for *sets* of polynomials. For this reason it will be convenient for us to think of our input contradiction as a *set of constraints*, where a constraint is simply a *Boolean function*. Throughout this monograph, we will reserve calligraphic letters for *sets*, and in particular, the letter $\mathcal{C}$ will always denote a CNF formula, i.e. a set of clauses. A set of constraints $\mathcal{F}$ is said to be satisfied by an assignment α iff $f(\alpha) = 1$ for all constraints $f \in \mathcal{F}$, and this happens iff $\wedge\{f \in \mathcal{F}\}(\alpha) = 1$. As mentioned earlier, the notation $\mathcal{F} \models g$ will denote that $\mathcal{F}$ implies g, i.e.

$$\forall\alpha \quad ((\mathcal{F}(\alpha) = 1) \Rightarrow (g(\alpha) = 1).$$

A derivation of ψ from a set of constraints $\mathcal{F}$ is a sequence $\{\phi_1 \dots \phi_s\}$, where $\phi_s = \psi$, and for all $i = 1 \dots s$, ϕ_i is either some constraint $f \in \mathcal{F}$ or is derived from some $\mathcal{G} \subseteq \{\phi_1 \dots \phi_{i-1}\}$ by a derivation rule. Thus, the sources of the proof-DAG of $\mathcal{F}$ are labeled by constraints $f \in \mathcal{F}$ instead of the single formula $\wedge\mathcal{F}$.

The question of the exact formulation of a (semantical) Boolean constraint as a (syntactical) circuit, will be dealt with later on. Our lower bounds will be fairly robust to the various formulations, and will depend mainly on properties of the set of constraints $\mathcal{F}$, as presented in the next section.

1.8 Main Result: Expansion + Sensitivity ⇒ Hardness

The main result of this thesis appears in the header of this section. Here are the necessary definitions.

Expansion For f a function, let $Vars(f)$ be the set of variables f depends on, and let $Vars(\mathcal{F}) = \cup\{Vars(f) : f \in \mathcal{F}\}$. $\mathcal{F}$ is called (r, c)-*expanding* if for every $\mathcal{F}' \subset \mathcal{F} \ \ |\mathcal{F}| \leq r \ \ |Vars(\mathcal{F}')| \geq (1 + c)|\mathcal{F}'|$.

Sensitivity A function is called *sensitive* if any assignment to it can be changed to a satisfying one by changing only a small number of bits in the assignment. E.g. a *clause* (disjunction of literals), is highly sensitive.

We restate the main result more explicitly:

If $\mathcal{F}$ is an *expanding* set of *sensitive* constraints, then proving $\mathcal{F}$ is *hard*.

Notice that we do not care too much about the exact *syntactic* encoding of the constraints, and both requirements (sensitivity and expansion) are *semantic* properties of the functions $\mathcal{F}$.

The principle stated above has been used implicitly in some of the previous lower bounds in proof complexity, but never has it appeared in such a general form. We will show that many of the previously known lower bounds, as well as many new ones, can be derived naturally from the sensitivity and expansion of the input contradiction $\mathcal{F}$. Let us demonstrate this general principal. For concreteness, assume $\mathcal{F}$ is a contradictory set of clauses over n variables, with good expansion properties (say $r = \Omega(n)$ and c a large constant). We will prove the following results:

- The minimal size of a *treelike resolution* proof is exponential in $c \cdot r$.
- The minimal size of a *general resolution* proof is exponential in $\frac{(c \cdot r)^2}{n}$. If $c \cdot r >> \sqrt{n}$ we get non-trivial lower bounds.
- The minimal *space* required for verifying a resolution proof is linear in $c \cdot r$.
- The minimal degree of a *polynomial calculus* proof is linear in $c \cdot r$ (whenever the characteristic of the underlying field is greater than 2).
- If $\mathcal{F}$ is a Tseitin formula of an expander graph (this is a special case of an expanding set of sensitive functions), then the minimal size of a *depth d Frege* proof is exponential in n^{δ}, where $\delta > 0$ depends on d, as well as on the expansion properties of the expander graph.

Many of these lower bounds (but certainly not all) come from the *width method* described in the next section. This simple method, that gives in a unified manner almost all of the currently known lower bounds for resolution proof size, is another important message of our thesis.

Before continuing, let us remark that throughout the monograph, we will meet three variants of expansion: bipartite expansion (def. 2.12), bipartite boundary expansion (def. 2.28), and general expansion (def. 2.36). All are related, and formalize the same intuition: a graph is expanding if any subset of its vertices has many neighbors/outgoing edges.

1.9 The Width Method

The main technical tool we present in this thesis is the *width method*, based on the *size-width relations*, which we now state informally. The importance of this method comes from its simplicity and generality. The *width* of a clause is the number of literals in it, the width of a proof is the maximal width of a clause in it, and for $\mathcal{C}$ a CNF, the width of $\mathcal{C}$ (denoted $W(\mathcal{C})$), is the minimal width of a proof of $\mathcal{C}$. Denote treelike resolution by $\mathbf{T}$, and resolution by $\mathbf{R}$. The basic size-width relations are:

$$\mathbf{S_T}(\mathcal{C}) \geq \exp(W(\mathcal{C}))$$

$$\mathbf{S_R}(\mathcal{C}) \geq \exp(\frac{W(\mathcal{C})^2}{|Vars(\mathcal{C})|})$$

Using these relations, one only needs to show that any proof of $\mathcal{C}$ must include a wide clause, and get immediately size lower bounds. It turns out that in practice, this indirect approach is much easier than proving size lower bounds directly. Indeed, almost all of the previously known size lower bounds for resolution, that are quite complicated, can be simplified by our techniques, and derived in a uniform manner that exposes the connection of width to expansion and sensitivity. Additionally, this technique helps us derive many interesting new lower bounds. Some cases, such as random CNF formulas, follow immediately from the size-width relations. Other formulas (i.e. Tseitin contradictions, generator contradictions, and the pigeonhole principle, for definitions see chapter 2), require a more subtle analysis, but still remain within the main paradigm - *"prove width lower bounds, and deduce size lower bounds"*.

1.10 Summary of Results+Credits

We now list more explicitly our results, classified by the chapters of the thesis, and give the due credit to our co-authors.

1.10.1 Resolution Size Lower Bounds (chapter 2)

In this chapter we introduce the *size-width* relations and give four important lower bounds as their consequences (definitions appear in chapter 2).

1. Random CNF formulas.
2. Tseitin graph based contradictions.
3. Various contradictions based on generators.
4. Variants of the pigeonhole principle.

All lower bounds, except for one variant of the pigeonhole principle, are the best known, many are optimal (a matching upper bound exists), and some are novel.

The size-width relations are not just good for proving lower bounds. They also motivate a simple automated theorem proving algorithm, which we introduce in this chapter, and further investigate in chapter 4.

Credits: The basic size-width relations and their consequences are joint work with Avi Wigderson, and initially appeared in [Ben-Sasson and Wigderson ('99)]. The generalization of the size-width relations to explicit and implicit encodings is joint work with Mikhail Alekhnovich, Avi Wigderson and Alexander A. Razborov, and appeared initially in [Alekhnovich *et al.* A ('00)].

1.10.2 Resolution Space Lower Bounds (chapter 3)

In this chapter we prove that any proof of an expanding CNF requires large memory space in order to verify it. As a result we get optimal lower bounds for the space of proving a random CNF. These results are novel, answering an open question of [Esteban and Toran ('99)] and [Alekhnovich *et al.* B ('00)]. The lower bound is via an elegant reduction to a two-player *matching game* played on a graph that is naturally induced by the input CNF.

The space required for verifying a proof is just the *pebbling measure* of the corresponding proof-DAG (definition 4.5). The pebbling measure captures the notion of circuit computation space, and has been extensively used for proving space lower bounds, and time-space tradeoffs for Turing machines. We also meet this measure in the next chapter, though in a surprising and unexpected context.

Credits: The results of this chapter are joint work with Nicola Galesi, and appeared initially in [Ben-Sasson and Galesi ('01)].

1.10.3 Separating Treelike and General Resolution (chapter 4)

In this chapter we take a small diversion from our main theme, and analyze the gap between *treelike* and *general* resolution size. Our near-optimal, exponential separation of the two systems is displayed by a set of contradictions based on graphs that require *large pebbling measure.* We show a surprising connection between the treelike proof size of the contradictions, and the *pebbling measure* of the underlying graphs. The strange connection of this measure to treelike proof size uses a reduction of the treelike proof to a two-player game (unrelated to the matching game mentioned above).

One interesting consequence of this gap is that in some natural cases, our width-based automated theorem prover (introduced in chapter 2), is exponentially more efficient than commonly used provers.

Credits: The results of this chapter are joint work with Russell Impagliazzo and Avi Wigderson, and appeared initially in [Ben-Sasson *et al.* ('00)].

1.10.4 Polynomial Calculus Degree Lower Bounds (chapter 5)

In this chapter we show a reduction of polynomial calculus proof degree, to resolution width. This reduction allows us to obtain immediately novel and optimal degree lower bounds for interesting classes of contradictions such as random CNF formulas, and generator based contradictions. Our proof shows that in many interesting cases, the polynomial calculus is not essentially more efficient than resolution. The simplicity of our reduction allows us to witness how expansion and sensitivity give rise to proof hardness in the polynomial calculus.

Credits: The introduction of the *Gaussian width* and its connection to *width* is the contribution of Mikhail Alekhnovich, done during joint work on the generator contradictions [Alekhnovich *et al.* A ('00)]. The reduction of PC degree to width and the lower bound for random CNF is joint work with Russell Impagliazzo, and appeared initially in [Ben-Sasson and Impagliazzo ('99)]. The lower bounds for expanding contradictions are joint work Mikhail Alekhnovich, Avi Wigderson and Alexander A. Razborov, and appeared initially in [Alekhnovich *et al.* A ('00)].

1.10.5 Bounded Depth Frege Size Lower Bounds (chapter 6)

In our final chapter, we show that a special class of expanding contradictions are hard for the relatively strong system of bounded depth Frege. Our proof is a novel reduction of graph based Tseitin contradictions to the well known pigeonhole principle. We show that a short proof for the Tseitin contradiction implies a short proof for the pigeonhole principle. Thus we apply the well-known lower bounds for this latter principle [Krajíček *et al.* ('95), Pitassi *et al.* ('93)], and derive novel lower bounds for the former contradiction. The reduction itself is non-trivial, and seems interesting in its own right.

Chapter 2

Resolution Size Lower Bounds

The simplest proof system we consider is *resolution*. This system is complete for the **co-NP** complete language of *unsatisfiable CNF formulas*, has essentially one derivation rule, and all lines in a proof are clauses. Following some basic definitions, the results of this chapter will be described.

2.1 Resolution - Definition

A *literal* over x is either x, denoted also x^1, or $\bar{x}$, denoted also x^0. A *clause* is a disjunction of literals. A variable x *appears* in C, denoted $x \in C$, if a literal over x appears in C. A CNF formula is a set of clauses. The *resolution rule* is the following derivation rule:

Resolution Rule: Derive $E \vee F$ from $\{E \vee x, F \vee \overline{x}\}$, where E, F are any clauses, and x is any variable.

For the sake of convenience, we will also allow use of the weakening rule, in a resolution derivation.

Weakening Rule: Derive $E \vee F$ from $\{E\}$ for any clauses E, F.

The weakening rule does not make resolution more efficient, as the following lemma states, it only simplifies some of our proofs.

Lemma 2.1 *If π is a derivation of a clause A from a CNF $\mathcal{C}$, using the resolution and weakening rules, then there exists a sub-clause $A' \subseteq A$ and a derivation π' such that π' is a derivation of A' from $\mathcal{C}$ using only the resolution rule, and the size of π' is at most the size of π.*

The *resolution proof system* is the sequential proof system based on the resolution and weakening rules. Notice that by the definition of these rules, all lines in a resolution proof must be clauses. For the sake of clarity, let us specify exactly what a resolution proof looks like. Let $\mathcal{C} = \{C_1, C_2 \ldots C_m\}$ be a CNF formula over n variables. A *resolution derivation* of a clause A from $\mathcal{C}$ is a sequence of clauses $\pi = \{D_1, D_2 \ldots D_S\}$ such that the last clause is A and each formula D_i is either some initial clause $C_j \in \mathcal{C}$, or is derived from previous clauses using the resolution or weakening rule. A *resolution proof*, sometimes also called a *refutation*, is a resolution derivation of the empty clause, $\mathbf{0}$. The minimal size of a proof of $\mathcal{C}$, denoted $\mathbf{S_R}(\mathcal{C})$, is the minimal number of clauses in a proof of $\mathcal{C}$. Similarly, $\mathbf{S_T}(\mathcal{C})$ is the minimal size of a treelike resolution proof. The following fundamental theorem of resolution implies that $\mathbf{S_T}(\mathcal{C}) = \mathbf{S_R}(\mathcal{C}) = \infty$ iff $\mathcal{C} \in \mathbf{SAT}$.

Theorem 2.2 (Completeness of Resolution) *Resolution is a complete proof system for CNF formulas. In other words, for any CNF $\mathcal{C}$, $\mathcal{C} \in \mathbf{co-SAT}$ iff there exists a resolution proof of $\mathcal{C}$.*

Resolution is implicationally complete *for CNF formulas, meaning for any CNF $\mathcal{C}$, and any clause A, there is a derivation of A from $\mathcal{C}$ iff $\mathcal{C} \models A$.*

2.2 Survey of Results

2.2.1 History and Previous Lower Bounds

The *resolution* proof system was initially introduced by [Blake ('37)], and further developed in the influential work of [Robinson ('65)] and [Davis and Putnam ('60)]. This system is one of the weakest in the world of propositional proof systems, and therein lies its strength. Because of its simplicity, and because all lines in a proof are clauses, this system yields easily to proof-search algorithms. Indeed, many automated theorem proving algorithms used in practice actually search for a treelike resolution proof. The simplicity and practical usefulness of resolution made it the most investigated propositional proof system.

This simplicity is misleading, especially when trying to prove size lower bounds. The first non-trivial proof size lower bound was given in the seminal work of [Tseitin ('68)]. In this paper he constructed graph based contradictions that formulate the basic fact that the sum of degrees of any graph must be even. Tseitin showed that in a weaker variant of resolution, called *regular resolution*, for certain graphs the contradictions require proofs that are exponentially long in the size of the input contradiction. It took nearly 20 years until the first non-trivial lower bounds for *general resolution* was presented. In 1985, Haken proved the breakthrough exponential lower bound for the well-known *pigeonhole principle* [Haken ('85)]. This principle states that there is no matching from a set of size $n+1$ to a set of size n, and can be naturally formulated as a CNF of size $O(n^2)$ (see definition 2.48). Haken showed that this CNF requires resolution proof size that is exponential in n, i.e. exponential in square root the size of the CNF. Shortly after, [Urquhart ('87)] proved that Tseitin contradictions of *expander graphs* (a concept not known at the time Tseitin published his work) require resolution proofs that are exponential in the size of the input CNF. Chvátal and Szemerédi gave the final blow to resolution by showing that with high probability, a random CNF requires exponential size proofs, and thus the weakness of resolution as a proof system is not restricted to specially tailored formulas, rather it is the rule [Chvátal and Szemerédi ('88)]. All these proofs used combinatorial methods borrowed from circuit complexity, such as random restrictions, and had to be custom tailored to the contradiction in hand.

2.2.2 Summary of our Lower Bounds

Unfortunately, in all lower bounds mentioned above, it seemed hard to pinpoint the exact property of the CNFs that make them hard to prove, and it was not clear whether there is a common denominator to all of them. These lower bounds involved proving the existence of "complicated clauses", and then deducing, using random restrictions, that exponentially many such clauses must exist. Our main contribution, described in this chapter, is a simplification of these lower bounds, by a unified technique that easily gives all these lower bounds, as well as many new ones. Our contribution is that instead of proving *size* lower bounds directly, it is enough to prove *width* lower bounds, i.e. that any proof must include a clause with many literals. If the proof width is large, then so is the proof size. Finally, it turns out that proving width lower bounds is much easier than proving size lower bounds directly.

Our simplification and better understanding came as usual from an unexpected direction. When investigating the polynomial calculus, Clegg, Edmonds and Impagliazzo proved that a short resolution proof implies a small degree polynomial calculus proof [Clegg, *et al.* ('96)]. Ben-Sasson and Wigderson then discovered (using similar proof techniques) that this relation applies to resolution as well [Ben-Sasson and Wigderson ('99)]. The *size width relation* at the core of this chapter, was found to yield *all lower bounds* known at the time[1], except for those using the interpolation method in a simple and unified manner. Since its discovery, the width method has also yielded many new size lower bounds, such as those for generator based contradictions which we discuss in section 2.10, and various generalizations of the pigeonhole principle (section 2.11). Since its introduction, the width measure has been recognized as a key parameter in determining properties of resolution proofs, and its applications are the focus of much research.

Recall that a Boolean *constraint* is a Boolean function. Intuitively, a contradictory set of constraints should be *easy to prove* when there is some *local contradiction*, like a small number of constraints, each involving only a few variables, that are contradictory. Following this intuition, a contradiction should be *hard to prove* if it is very "non-local", meaning that any small subset of it is satisfiable, and the unsatisfiability arises from subtle interactions between large subsets of constraints. Surprisingly, for resolution we can give very clear and formal definitions of this intuition, and use these definitions to prove many lower bounds on proof size. There are two important requirements from our set of constraints $\mathcal{F}$ for our lower bounds to work.

Expansion $\mathcal{F}$ has to be *expanding* in the following sense. Think of the bipartite graph that has on the left side the set of constraints, on the right side the set of variables, and every constraint is connected by edges to all variables it depends on. This bipartite graph, denoted $\mathbb{G}(\mathcal{F})$, should be a good *expander*, i.e. every small subset on the left side should have many neighbors on the right side.

Sensitivity Each constraint $f \in \mathcal{F}$ should be highly satisfiable, meaning that any falsifying assignment to f can be made into a satisfying one, by changing only a few input bits. E.g. the OR and PARITY functions are very sensitive, since even a change of a single bit suffices to satisfy them.

Roughly speaking, whenever $\mathcal{F}$ is an *expanding*, contradictory set of *sensitive* constraints, and each constraint is represented naturally by a CNF, the conjunction

[1] Currently this is no longer true, and see section 2.2.3 for more details.

of these CNFs, itself also a CNF, requires very long resolution proofs, and the exact length depends on the expansion and sensitivity parameters, as well as on the specific formulation of the constraints as CNFs.

Having defined the necessary ingredients for obtaining exponential size lower bounds, we still remain with the actual problem of proving these lower bounds. Our main contribution is presenting an indirect approach for proving such bounds. While previous, direct lower bounds were quite tricky, and used probabilistic analysis of random restrictions, our method avoids these technicalities and proceeds as follows.

Size Lower Bound:

1. If the minimal *width* of proving $\mathcal{F}$ is large, (i.e. any proof must include a wide clause), then the proof *size* of $\mathcal{F}$ must be large. Thus we need only prove width lower bounds.

2. If $\mathcal{F}$ is an *expanding* set of *sensitive* constraints then any proof of $\mathcal{F}$ must have a *wide clause* (with many literals). Width lower bounds are proved along these lines:

   (a) If $\mathcal{F}$ is an *expanding* set of *sensitive* constraints then any small subset of the constraints is *satisfiable*. Hence any proof must have a clause that is derived from a large subset $\mathcal{F}' \subset \mathcal{F}$, and *cannot* be derived from a smaller subset.

   (b) If a clause is derived from a large *expanding* set of *sensitive* constraints, it must be easily satisfiable, and hence must have many literals.

After proving part 1 of the above strategy (in section 2.3), our focus will be on proving *width* lower bounds in various settings. Our first width lower bounds will be for expanding CNF formulas, where each constraint is a *clause* (section 2.5). Lower bounds for random CNF formulas, will follow immediately from the observation that with high probability, a random CNF is expanding (section 2.6). After this warm-up, the width lower bounds will be generalized to apply to *arbitrary* sets of sensitive constraints, under various natural encodings (section 2.7). These width lower bounds will allow us to prove size lower bounds for Tseitin contradictions (section 2.9) and generator based contradictions (section 2.10). Finally, we will show how to derive lower bounds for the pigeonhole principle using our techniques (section 2.11).

2.2.3 Other Lower Bounds

For the sake of completeness, we mention two other lower bound techniques, not directly connected to the width method.

The *interpolation method* reduces proof complexity to circuit complexity. This method, well known in first order logic [Craig ('57)], says that if there is a short proof a certain contradiction, then there is a short circuit computing a related predicate. More specifically, let $\mathcal{C}(\vec{x},\vec{y},\vec{z}) = \mathcal{C}_1(\vec{x},\vec{y}) \wedge \mathcal{C}_2(\vec{x},\vec{z})$ be a CNF contradiction, where the only common variables of $\mathcal{C}_1$ and $\mathcal{C}_2$ are the x variables. Since $\mathcal{C}$ is unsatisfiable, any assignment α to the x variables must make one of $\mathcal{C}_1, \mathcal{C}_2$ unsatisfiable. An *interpolant* for $\mathcal{C}$ is a circuit $D(\vec{x})$, such that for all assignments α to $\vec{x}$:

1. $D(\alpha) = 1 \rightarrow \mathcal{C}_1(\alpha, \vec{y}) \in \mathbf{co-SAT}$.

2. $D(\alpha) = 0 \rightarrow \mathcal{C}_2(\alpha, \vec{z}) \in \mathbf{co-SAT}$.

In the context of resolution, Krajíček showed if there is a proof of $\mathcal{C}$ of size S, then there is an interpolant for $\mathcal{C}$ of size S [Krajíček ('97)]. Moreover, if $\mathcal{C}$ is *monotone*, i.e. the $\vec{x}$ variables appear in $\mathcal{C}_1$ only as positive literals, then the interpolant is a monotone circuit. Krajíček presented natural CNFs that are monotone, and such that any interpolant for them separates the set of $k-1$ colorable graphs, from the set of graphs with a k-clique. The famous monotone-circuit lower bound of Razborov for this problem [Razborov ('85)], gives a proof size lower bound via the interpolation method. Thus, for these special formulas, one can reduce proof size to circuit size.

An important open problem, that did not follow from previous results, nor from our techniques, was that of the *weak pigeonhole principle*. This contradiction formulates the simple idea that there is no matching from a set of size m to a set of size $n << m$. As m grows larger with respect to n, the contradiction becomes more "obvious", and hence, easier to prove. All previous lower bounds, including ours, became trivial when $m = n^2$. Recently, Raz solved this open problem, proving exponential lower bounds for the weak pigeonhole principle, independent of n [Raz ('01)]. Even more recently, Razborov improved the lower bound of Raz, placing it in a far reaching generalization of our basic width method [Razborov ('01)] (for more information, see section 2.11).

2.2.4 Automated Theorem Proving

The width method is not only a useful technique for obtaining lower bounds, it also gives better understanding of an automated theorem proving algorithm. This algorithm, that tries to find a *minimal width* proof, turns out to be relatively efficient in terms of its running time. We describe this algorithm, and prove upper bounds on its running time, using the size-width relations (section 2.4). In chapter 4 we will recall this algorithm, and show that sometimes it outperforms other commonly used automated theorem provers by an exponential factor.

2.3 Lower Bounds on Width give Lower Bounds on Size, or Short Proofs are Narrow

After the introduction, we go into the details of the width method, starting with the basic size-width relations, proved in this section. We state the main relations, followed by the necessary definitions, and then prove the basic relations.

Definition 2.3 (Width) *The* width *of a clause C, denoted $|C|$, is the number of literals in it. A set, or sequence of clauses is called a k-CNF if the maximal width of a clause in it is k. For $\mathcal{C}$ a CNF and C a clause, the* derivation width *of C from $\mathcal{C}$, denoted $W(\mathcal{C} \rightarrow C)$, is the minimal width of a derivation of the clause C from $\mathcal{C}$, and ∞ if no such derivation exists. The* proof width *of $\mathcal{C}$, denoted $W(\mathcal{C})$, is the minimal width of a proof of $\mathcal{C}$, and ∞ if no such proof exists, i.e. if $\mathcal{C} \in \mathbf{SAT}$.*

Theorem 2.4 (Size-width relations for *treelike resolution*) *For $\mathcal{C}$ a k-CNF:*

$$W(\mathcal{C}) \leq \log \mathbf{S_T}(\mathcal{C}) + k.$$

Exponentiating both sides:

$$\mathbf{S_T}(\mathcal{C}) \geq 2^{W(\mathcal{C})-k}.$$

Theorem 2.5 (Size-width relations for *resolution*) *For $\mathcal{C}$ a k-CNF:*

$$W(\mathcal{C}) = O\left(\sqrt{n \ln \mathbf{S_R}(\mathcal{C})}\right) + k.$$

Exponentiating both sides:

$$\mathbf{S_R}(\mathcal{C}) = \exp\left(\Omega(\frac{(W(\mathcal{C})-k)^2}{n})\right).$$

If the input clauses are large, i.e. $k \sim |Vars(\mathcal{C})|$, theorems 2.4,2.5 are useless for obtaining lower bounds. This will not be a problem in practice, since all the formulas we shall consider, either have constant initial width (e.g. Tseitin formulas, random 3-CNF's), or can be reduced to such formulas (e.g. the pigeonhole principle).

2.3.1 Tightness of the bounds

How large can be the gap between size and width be ? For treelike resolution, we shall meet in chapter 4 a family of contradictions that can be proved in constant width, but the tree-like size is exponential in the size of the input CNF, hence the gap can be arbitrarily large.

For general resolution, Bonet and Galesi showed that the trade-off of theorem 2.5 is as tight as one can hope for [Bonet and Galesi ('99)]. This is done by presenting a natural family of graph-based k-CNF contradictions (constant k), over n variables and $O(n)$ clauses, that have proofs of size $n^{O(1)}$, but width $\Omega(\sqrt{n})$.

2.3.2 Notation - Restrictions

A restriction of a formula is a new formula obtained by a partial assignment to it: replace every literal by it's value, and then simplify the formula in the natural way. Here is a formal definition of a restriction to clauses, CNFs, and proofs.

For C a clause, x a variable and $a \in \{0,1\}$, the *restriction* of x on a is:

$$C|_{x=a} \stackrel{\text{def}}{=} \begin{cases} C & \text{if } x \text{ does not appear in } C \\ 1 & \text{if the } \textit{literal } x^a \text{ appears in } C \\ C \setminus \{x^{1-a}\} & \text{otherwise} \end{cases}$$

Similarly, $\mathcal{F}|_{x=a} \stackrel{\text{def}}{=} \{C|_{x=a} : C \in \mathcal{F}\}$. For $\pi = \{C_1, \ldots C_s\}$ a derivation of C_s from $\mathcal{F}$ and $a \in \{0,1\}$, let $\pi|_{x=a} = \{C'_1, \ldots C'_s\}$ be the restriction of π on $x = a$, defined inductively by:

$$C'_i = \begin{cases} C_i|_{x=a} & C_i \in \mathcal{F} \\ C'_{j_1} \vee C'_{j_2} & C_i \text{ was derived from } C_{j_1} \vee y \text{ and} \\ & C_{j_2} \vee \bar{y} \text{ via a resolution step,} \\ & \text{for } j_1 < j_2 < i \\ C'_j \vee A|_{x=a} & C_i = C_j \vee A \text{ via the weakening rule,} \\ & \text{for } j < i \end{cases}$$

The consequence of resolving a clause B with 1 is defined to be B. Assume w.l.o.g. that $\pi|_{x=a}$ does not contain the clause 1, by removing all such clauses from $\pi|_{x=a}$.

Definition 2.6 (Falsifying Restriction) *For C a clause, the* falsifying restriction *of C, denoted $\rho_{C=0}$ is the minimal size restriction ρ that sets $C|_{\rho_{C=0}} = 0$:*

$$\rho_{C=0}(x) = \begin{cases} 1-a & x^a \text{ is a literal of } C \\ \star & \text{otherwise} \end{cases}$$

2.3.3 Proof of Theorems 2.4, 2.5

The theorems and proof techniques in this subsection are an adaptation of the size-degree relations of [Clegg, *et al.* ('96)], to the framework of resolution. The main idea of the proof is to use induction on the following claim: *If there is a short proof then there is a proof with small width.* Given a short proof π, find a literal ℓ that appears in many wide clauses. Setting $\ell = 1$ removes all these clauses, and $\pi|_{\ell=1}$ is a proof of $\mathcal{C}|_{\ell=1}$. Hence by induction, there is a small width proof of $\mathcal{C}|_{\ell=1}$. On the other hand, we apply induction on the *number of variables* to bound the width of $\mathcal{C}|_{\ell=0}$. Finally, we claim that small width proofs of $\mathcal{C}|_{\ell=1}$ and of $\mathcal{C}|_{\ell=0}$ can be combined into a small width proof of $\mathcal{C}$. We start by proving formally that the two small width restricted proofs, can be combined into one small width non-restricted proof. Recall the definition 2.3 of width.

Lemma 2.7 *for $a \in \{0,1\}$, A a clause and $\mathcal{C}$ a CNF, if $W(\mathcal{C}|_{x=a} \to A) \le d$ then $W(\mathcal{C} \to A \vee x^{1-a}) \le d+1$.*

Proof: $\mathcal{C}|_{x=a}$ is created from $\mathcal{C}$ by disposing of all initial clauses that include the literal x^a and removing the literal x^{1-a} from all other initial clauses where it appears. Let $\mathcal{C}'$ be the set of initial clauses containing the literal x^{1-a}, and let π be a width w derivation of A from $\mathcal{C}_{x=a}$. Add the literal x^{1-a} to all clauses in π and call the new derivation π'. We claim that π' is a legal resolution derivation. If $C \in \mathcal{C}'$ then $C \vee x^{1-a}$ is an initial clause of $\mathcal{C}$. If $C \in \mathcal{C} \setminus \mathcal{C}'$ then $C \vee x^{1-a}$ can be derived from C by a single weakening step. Finally, if C was derived from D, E via a resolution step, then $C \vee x^{1-a}$ is the resolution consequence of $D \vee x^{1-a}, E \vee x^{1-a}$. It is easy to see that the width of each clause in π' is larger by 1 than the matching clause in π. □

Lemma 2.8 *For $a \in \{0,1\}$, define $\mathcal{C}_{x^a}$ as the set of all clauses in $\mathcal{C}$ containing the literal x^a, and let k be the maximal width of a clause in $\mathcal{C}_{x^a}$. If $W(\mathcal{C}|_{x=a}) \le d-1$ and $W(\mathcal{C}|_{x=1-a}) \le d$ then $W(\mathcal{C}) \le \max\{d,k\}$.*

Proof: According to lemma 2.7, if $W(\mathcal{C}|_{x=a}) \le d-1$ then $W(\mathcal{C} \to x^{1-a}) \le d$. Resolve the clause x^{1-a} with all clauses in $\mathcal{C}_{x^a}$, to derive $\mathcal{C}|_{x=1-a}$. This part will have width k. Finally, by the assumption we can prove $\mathcal{C}|_{x=1-a}$ with width d. □

In the case of treelike resolution, the variable we restrict is simply the last one to be resolved in the proof. Restricting on this variable removes at least half of the proof, because the proof-DAG is a tree. Thus the size-width relation we obtain is logarithmic.

Theorem 2.4 (Size-width relations for *treelike resolution*) For $\mathcal{C}$ a k-CNF:

$$W(\mathcal{C}) \leq \log \mathbf{S_T}(\mathcal{C}) + k.$$

Proof: Prove by induction on b and n, the number of variables, that if $\mathbf{S_T}(\mathcal{C}) \leq 2^b$ then $W(\mathcal{C}) \leq k + b$. If $b = 0$ then $\mathbf{0} \in \mathcal{C}$, and we're done. Otherwise, the last derivation is

$$\frac{x \quad \overline{x}}{\mathbf{0}},$$

where $x, \overline{x}$ were derived by tree-like derivations $T_x, T_{\overline{x}}$ of sizes $S_x, S_{\overline{x}}$, respectively, and $\mathbf{S_T}(\mathcal{C}) = S_x + S_{\bar{x}} + 1$. $T_x|_{x=0}$ ($T_x|_{x=1}$) is a tree-like proof of $\mathcal{C}|_{x=0}$ ($\mathcal{C}|_{x=1}$) of size at most S_x ($S_{\bar{x}}$). Assume w.l.o.g. $S_x \leq 2^{b-1}$. By induction on b, $W(\mathcal{C}|_{x=0}) \leq k+b-1$ and by induction on n, $W(\mathcal{C}|_{x=1}) \leq k+b$. Applying lemma 2.8 completes the proof. $\square$

In the case of general resolution, we are not guaranteed to have a literal that appears in a constant fraction of the wide clauses. Thus, we use the fact that a wide clause has many literals, and hence there should be a literal appearing in many wide clauses. This will result in a relation that is not as good as for treelike resolution, but as we mentioned earlier (section 2.3.1), it is the best one can hope for.

Theorem 2.5 (Size-width relations for *resolution*) For $\mathcal{C}$ a k-CNF:

$$W(\mathcal{C}) = O\left(\sqrt{n \ln \mathbf{S_R}(\mathcal{C})}\right) + k.$$

Proof: Let π be a minimal size proof of $\mathcal{C}$ of size $\mathbf{S_R}$. If $\mathbf{S_R} = 1$ then $\mathbf{0} \in \mathcal{C}$ and we're done. Otherwise, set $d \stackrel{\text{def}}{=} \lceil \sqrt{2n \ln \mathbf{S_R}(\mathcal{C})} \rceil$, and $a \stackrel{\text{def}}{=} (1 - \frac{d}{2n})^{-1}$. Let π^* be the set of *fat* clauses in π, having width greater than d. Prove by induction on b and n, that if $|\pi^*| < a^b$ then $W(\mathcal{C}) \leq d + b + k$. The base case ($b = 0$) is trivially true. For the induction step, there are $2n$ literals, so by the Pigeonhole Principle there is some literal (w.l.o.g.) x appearing in $\geq \frac{d}{2n} \cdot |\pi^*|$ fat clauses. Restricting $x = 1$ removes all the clauses in which x appears, and leaves us with a proof of $\mathcal{C}|_{x=1}$ with at most $(1 - \frac{d}{2n}) \cdot |\pi^*| < a^{b-1}$ fat clauses. Hence by induction on b:

$$W(\mathcal{C}|_{x=1}) \leq d + b + k - 1.$$

By induction on n:

$$W(\mathcal{C}|_{x=0}) \leq d + b + k.$$

Applying lemma 2.8 completes the proof. $\square$

2.4 A Width Based Automated Theorem Prover

Before showing the use of the width method for obtaining proof size lower bounds, we discuss its importance and application to automated theorem proving. A natural approach to finding a small size resolution proof is to seek a *minimal width* proof. Algorithms for finding such proofs are well-known in the AI community [Selman ('95)], and our size-width relations give them theoretical motivation (a similar analysis was given by [Beame and Pitassi ('96)]). For concreteness we describe such an algorithm, which we call the *restricted width* (RW) algorithm.

```
RW (C)
    fix w = 0
    Repeat {
        If 0 ∈ C end
        Else {
            Increase w
            Repeatedly derive from C all clauses of
            width ≤ w and add to C
            }
        }
```

This algorithm has running time bounded by $(2n)^{W(\mathcal{C})}$, because this is an upper bound on the number of different clauses that will be encountered (there are $2n$ literals, and $\binom{n}{k} \leq n^k$). Plugging in the size-width relations, we get the following non-trivial results on the automatizability of resolution and treelike resolution. Let $\mathbf{TIME}(A(x))$ denote the running time of the algorithm A on input x.

Theorem 2.9 *Treelike resolution has quasi-polynomial automatization, i.e. for $\mathcal{C}$ a 3-CNF with $\mathbf{S_T}(\mathcal{C}) = S$:*

$$\mathbf{TIME}(RW(\mathcal{C})) = O(S^{\log S}).$$

Proof: Clearly, the number of literals in a proof of $\mathcal{C}$ is at most S. By theorem 2.4 there is a proof of $\mathcal{C}$ of width $\log S + 3$, and hence the overall number of clauses $RW(\mathcal{C})$ might generate is $O(S^{\log S})$. □

Theorem 2.10 *Resolution has sub-exponential automatization, i.e for $\mathcal{C}$ a 3-CNF with n variables and $\mathbf{S_R}(\mathcal{C}) = S$:*

$$\mathbf{TIME}(RW(\mathcal{C})) = n^{O(\sqrt{n \log S})}.$$

Proof: Similar to the previous theorem, using theorem 2.5 instead of theorem 2.4. □

Our automatization of general resolution is not as good as for the treelike case, because of the different size-width relations we use. Comparing our upper bound with the trivial upper bound, which is 2^S, we see that it is a nice improvement. For instance, if $\mathbf{S_R}(\mathcal{C}) = n^{O(1)}$ we can find a proof in time $2^{O(\sqrt{n \cdot \log^3 n})}$, which is substantially smaller than $2^S = 2^{n^{O(1)}}$.

In chapter 4 we will see that sometimes, the restricted width algorithm outperforms the commonly used automated theorem proving procedures, by an exponential factor (section 4.6).

2.5 Width Lower Bounds for Expanding CNFs

Our first width lower bound will be for the simple case, where the contradiction is an *expanding* CNF, i.e. each constraint is a clause. Recall that a clause, being an *OR* of literals, is *sensitive*, and thus our contradiction answers the two requirements needed for proving the width lower bound, stated in section 2.2.2. Following the definition of an expanding CNF we state the main theorem of this section. The theorem is proved in the last part of the section.

Definition 2.11 *For $\mathcal{C}$ a CNF, the* graph *of $\mathcal{C}$, denoted $\mathbb{G}(\mathcal{C})$, is the following bipartite graph $\mathbb{G}(\mathcal{C}) =< V \dot{\cup} U, E >$.*

1. *V is the set of clauses.*
2. *U is the set of variables.*
3. *$(C, x) \in E$ iff x appears in C (as a positive or negative literal).*

Definition 2.12 (Bipartite Expanders) *A bipartite graph $\mathbb{G} =< V \dot{\cup} U, E >$ is called an* (r, c)-expander *if*

$$\forall V' \subset V \quad |V'| \leq r, \quad |N(V')| \geq (1 + c)|V'|,$$

where $N(V')$ is the set of neighbors of V'.

Remark: Notice that c might be greater than 1.

Definition 2.13 (Expanding CNFs) *A CNF formula is called (r, c)-expanding if its graph is an (r, c)-expander.*

The main theorem of this section is:

Theorem 2.14 (Width Lower Bound for Expanding CNFs) *For any (r, c)-expanding CNF $\mathcal{C}$:*

$$W(\mathcal{C}) \geq c \cdot r/2.$$

2.5.1 Hall's theorem, Tarsi's theorem

We start with some preliminary results, that will be needed for the proof of theorem 2.14. The following theorem is perhaps folklore, but we know it as "Tarsi's theorem" [Aharoni and Linial ('86)], and that is the name we give it throughout this monograph. A *minimal unsatisfiable* CNF is an unsatisfiable CNF such that any proper subset of it *is satisfiable*, and $Vars(\mathcal{C})$ is the set of variables appearing in $\mathcal{C}$. The proof uses the famous matching theorem of Hall, which we quote here without proof.

Theorem 2.15 (Hall's Matching Theorem) *For a bipartite graph $G =< V \cup U, E >$, there exists a perfect matching of $V' \subseteq V$ into U iff $\forall V'' \subseteq V', |N(V'')| \geq |V''|$, where $N(V'')$ is the set of neighbors of V''.*

Theorem 2.16 (Tarsi's theorem) *For $\mathcal{C}$ a minimal unsatisfiable CNF:*

$$|\mathcal{C}| > |Vars(\mathcal{C})|$$

Proof: Consider the graph $\mathbb{G}(\mathcal{C})$ from definition 2.11. There is no matching from $\mathcal{C}$ to $Vars(\mathcal{C})$, because the formula is unsatisfiable. Hence, by Hall's theorem, there must be some set $S \subset \mathcal{C}$ such that $|N(S)| < |S|$. Let S be such a set of maximal size. By the maximality of S, for any $V \subset \mathcal{C} - S$ we have $|N(V) \cap Vars(\mathcal{C}) - N(S)| \geq |V|$. Thus there is a matching from $\mathcal{C} - S$ into $Vars(\mathcal{C}) - N(S)$. By the minimal unsatisfiability of $\mathcal{C}$, if $S \neq \mathcal{C}$ then S is satisfiable, and $\mathcal{C} - S$ is satisfiable by its matching into $Vars(\mathcal{C}) - N(S)$. Conclude that $S = \mathcal{C}$ and hence $|S| = |\mathcal{C}| > |N(S)| = |Vars(\mathcal{C})|$. $\square$

Definition 2.17 *A CNF $\mathcal{C}$* minimally implies *C if $\mathcal{C} \models C$, but any proper subset $\mathcal{C}' \subset \mathcal{C}$ does not imply C.*

An immediate generalization of Tarsi's theorem is the following.

Theorem 2.18 *For $\mathcal{C}$ a CNF minimally implying a clause C:*

$$|C| > |Vars(\mathcal{C})| - |\mathcal{C}|.$$

Proof: Let ρ be the falsifying restriction of C from definition 2.6, and notice that $|\rho| = |C|$. $\mathcal{C}|_\rho$ is unsatisfiable, since $\mathcal{C} \models C$. Moreover, $\mathcal{C}|_\rho$ is *minimal* unsatisfiable, for otherwise there would be some $\mathcal{C}' \subset \mathcal{C}$ such that $\mathcal{C}'|_\rho = \mathbf{0}$, meaning $\mathcal{C}' \models C$. By Tarsi's theorem 2.16:

$$\begin{aligned} |\mathcal{C}|_\rho| &> |Vars(\mathcal{C}|_\rho)| \\ &= |Vars(\mathcal{C})| - |\rho| \\ &= |Vars(\mathcal{C})| - |C| \end{aligned}$$

□

2.5.2 Proof of Theorem 2.14

The following definition and lemma are crucial, as all our width lower bounds will use *proper width measures.* Let *Clauses* be the set of all clauses.

Definition 2.19 (Proper Width Measure) *A function $\sigma : Clauses \rightarrow \mathbb{N}$ is called a* proper width measure *for $\mathcal{C}$ (a CNF formula) if the following hold.*

1. *For any $C \in \mathcal{C}$, $\sigma(C) \leq 1$.*
2. *If C can be derived from D, E by a single resolution step, then $\sigma(C) \leq \sigma(D) + \sigma(E)$.*

Lemma 2.20 *If μ is a proper width measure for the CNF $\mathcal{C}$, then any proof of $\mathcal{C}$ must have a clause C with*

$$\frac{1}{2}\mu(\mathbf{0}) \leq \mu(C) < \mu(\mathbf{0}).$$

Proof: Let us look at the value μ gives to the nodes of the proof DAG. At the sources, μ equals 1. At the root we get $\mu(\mathbf{0})$. At any node in the middle, μ is sub-additive. The lemma is proven. □

Theorem 2.14 (Width Lower Bound for Expanding CNFs) *For any (r, c)-expanding CNF $\mathcal{C}$:*

$$W(\mathcal{C}) \geq c \cdot r/2.$$

Proof: Assume wlog $c, r > 0$ (otherwise the proof is trivial). Define the following proper width measure for $\mathcal{C}$.

$$\mu(C) = \mu_{\mathcal{C}}(C) \stackrel{\text{def}}{=} \min\{|\mathcal{C}'| : \mathcal{C}' \subseteq \mathcal{C}, \mathcal{C} \models C\}$$

Claim 2.21 *μ is a proper width measure for $\mathcal{C}$.*

Proof: Property 1 is trivial. Property 2 follows from the soundness of resolution. Indeed, let $\mathcal{C}_D, \mathcal{C}_E$ be minimal size sub-formulas of $\mathcal{C}$ implying D, E respectively. Clearly $\mathcal{C}_D \cup \mathcal{C}_E \models C$, and hence $\mu(C) \leq |\mathcal{C}_D| + |\mathcal{C}_E| = \mu(D) + \mu(E)$. □

Notice that $\mu(\mathbf{0})$ is the minimal size of an unsatisfiable sub-formula of $\mathcal{C}$. Such a sub-formula is also minimal unsatisfiable, by definition. Hence, by Tarsi's theorem 2.16, and the fact that $c, r > 0$, it must be that $\mu(\mathbf{0}) > r$. By lemma 2.20 the proof must include a clause C with $r/2 \leq \mu(C) < r$. Fix such a C and fix some $\mathcal{C}' \subset \mathcal{C}$ minimally implying C. By the expansion of $\mathbb{G}(\mathcal{C})$ and the fact that $|\mathcal{C}'| \leq r$ we get:

$$|Vars(\mathcal{C}')| \geq (1 + c) \cdot |\mathcal{C}'|.$$

By the previous theorem 2.18 we conclude

$$|C| > |Vars(\mathcal{C}')| - |\mathcal{C}'| \geq c \cdot |\mathcal{C}'| \geq c \cdot r/2.$$

□

2.6 Expanding Contradictions Part 1: Random CNFs

Our first concrete example of a hard contradiction for resolution is that of a random CNF. Stated simply, a random CNF is hard for resolution, because it is expanding.

It is well-known in circuit complexity that simple counting arguments show that a random function is hard to compute. In studying the complexity of a given proof system it is natural to ask what is the proof complexity of a contradiction taken at random. There is no good definition of what constitutes a random contradiction, but there is an interesting and natural distribution that with high probability is unsatisfiable. An easy calculation shows that for a high enough constant Δ, with high probability (i.e. with probability $1 - o(1)$) a random 3-CNF formula with n variables and Δn clauses is unsatisfiable (Δ is called the *clause density*) [2]. Let us introduce the definition of a random CNF.

Definition 2.22 (Random CNFs) *Let $\mathbb{C}_m^{k,n}$ be the probability distribution obtained by selecting m clauses uniformly at random from the set of all $2^k \cdot \binom{n}{k}$ clauses of size k over n variables. $\mathcal{C} \sim \mathbb{C}_m^{k,n}$, means that $\mathcal{C}$ is selected at random from this distribution. A random k-CNF formula is a formula $\mathcal{C} \sim \mathbb{C}_m^{k,n}$.*

Size proof complexity of unsatisfiable random CNFs has been widely studied. Chvatál and Szmerédi in their seminal paper showed that with high probability, any random 3-CNF over n variables and Δn clauses for $\Delta = O(1)$, requires exponentially long *resolution proofs* [Chvátal and Szemerédi ('88)]. The importance of their work was in showing that in fact resolution is a very weak proof system, because in some sense almost all unsatisfiable 3-CNF require exponential size proofs. Their lower bound was later improved and simplified by [Beame and Pitassi ('96)],

[2]The existence and value of a *satisfiability threshold constant* is an important combinatorial question, outside the scope of this paper. We refer the interested reader to [Achlioptas ('00), Janson *et al.* ('00), Dubois, *et al.* ('00)] for bounds on the satisfiability threshold.

and finally improved up to a ratio $\Delta = o(\sqrt{n})$ by [Beame *et al.* ('98)] and by [Ben-Sasson and Wigderson ('99)]. Following is a simple proof of the best known lower bound, based on the expansion of a random CNF. The following lemma states that a random CNF is expanding. This lemma is proved by calculating a standard union bound, and its proof is deferred to appendix A.1.

Lemma 2.23 *For each integer $k \geq 3$ and $0 < \epsilon < 1/2$, there exists a constant $\kappa = \kappa(k, \epsilon)$ such that the following holds. With high probability $\mathcal{C} \sim \mathbb{C}^{k,n}_{\Delta \cdot n}$ is an $\left(\kappa \cdot n \cdot \Delta^{-\frac{1+\epsilon}{k-2-\epsilon}}, \epsilon\right)$-expanding CNF.*

The previous lemma and theorems 2.14, 2.4 and 2.5 give immediately:

Theorem 2.24 (Lower Bounds for Random CNFs) *For any $k \geq 3$, and any $0 < \epsilon < 1/2$, with high probability $\mathcal{C} \sim \mathbb{C}^{k,n}_{\Delta \cdot n}$ has:*

$$W(\mathcal{C}) = \Omega\left(\frac{n}{\Delta^{1/(k-2)+\epsilon}}\right)$$

$$\mathbf{S_T}(\mathcal{C}) = \exp\left(\Omega\left(\frac{n}{\Delta^{1/(k-2)+\epsilon}}\right)\right)$$

$$\mathbf{S_R}(\mathcal{C}) = \exp\left(\Omega\left(\frac{n}{\Delta^{2/(k-2)+\epsilon}}\right)\right)$$

For $k = 3$, and constant Δ we get linear width lower bounds, which translate to exponential lower bounds for both treelike and general resolution. When Δ is not constant, but a function of n, we get good lower bounds for treelike resolution for clause density $n^{1-\delta}$ for any $\delta > 0$. Notice that this lower bound is almost optimal, because at clause density $n/O(\log n)$, with high probability there are short treelike proofs [Beame and Pitassi ('96)]. For general resolution we get good lower bounds for clause density $n^{\frac{1}{2}-\delta}$, and thus the proof size for clause density above $\sqrt{n}$ remains open.

2.7 Width of Explicit Encodings

This section starts to generalize the techniques of the previous one, obtaining lower bounds for contradictions that are expanding sets of *arbitrary* sensitive *constraints.* Before giving the actual lower bounds, we have to clarify the question of encoding a constraint as a CNF formula, because resolution deals only with such formulas. Two possible encodings of a constraint will be considered:

Explicit Encoding The constraint is encoded as a CNF over the set of its "essential" variables.

Implicit Encoding The constraint is represented by any circuit that computes it.

The latter encoding may be shorter, but in turn it uses new "nonessential" variables. The introduction of these *auxiliary variables* will slightly decrease the efficiency of our lower bounds, but will allow them to deal with any reasonable representation of constraints as circuits. We shall deal with implicit encodings in the next section. In this section we prove width lower bounds for explicit encodings. We shall need a stronger definition of expansion, called *boundary expansion*. Our main theorem, stated informally, says that for $\mathcal{F}$ a boundary expanding set of sensitive constraints, any explicit encoding of $\mathcal{F}$ will require large-width proofs (theorem 2.30). We start with the necessary definitions, followed by this theorem and its proof.

2.7.1 Definitions

For $f(x_1, \ldots, x_n) : \{0,1\}^n \to \{0,1\}$ a constraint, let $Supp(f)$ be the set of variables on which f depends, i.e. it is the minimal subset $X \subseteq \{x_1, \ldots, x_n\}$ such that any assignment to $\{0,1\}^X$ fixes f. For f a circuit, let $Vars(f)$ denote the set of variables appearing in f. Notice that $Supp(f)$ does not necessarily equal $Vars(f)$, e.g. for any contradiction $Supp(f)$ is the empty set, where as $Vars(f)$ may be arbitrarily large. Here is the generalization of definition 2.11 to an arbitrary set of constraints (notice that for $\mathcal{C}$ a CNF, the definitions coincide).

Definition 2.25 *For $\mathcal{F} = \{f_1, \ldots f_m\}$ a set of constraints, the* graph *of $\mathcal{F}$, denoted $\mathbb{G}(\mathcal{F})$, is the following bipartite graph $\mathbb{G}(\mathcal{F}) = < V \dot\cup U, E >$.*

1. *V is the set of constraints.*
2. *U is the set of variables.*
3. *$(f_i, x_j) \in E$ iff $x_j \in Supp(f_i)$.*

Definition 2.26 (Sensitivity) *A constraint f is* ℓ-sensitive *if for any assignment α such that $f(\alpha) = 0$, and any $X \subset Supp(f)$ $|X| \leq \ell$, there is an assignment β such that $f(\beta) = 1$ and β agrees with α outside of X.*

Intuitively, a constraint is ℓ-sensitive if it can be satisfied by changing any ℓ of its variables. The OR and $\bigoplus$ functions are 1-sensitive, and the ℓ-threshold function, that outputs 1 iff at least ℓ of its inputs are 1, is ℓ-sensitive.

Definition 2.27 (Explicit Encoding of Boolean Constraints) *An* explicit encoding *of a Boolean constraint $f(x_1, \ldots x_n) : \{0,1\}^n \to \{0,1\}$, denoted $\mathcal{C}(f)$, is some CNF formula over $Supp(f)$ that is equivalent to f, i.e.*

$$\forall \alpha \in \{0,1\}^n \quad f(\alpha) = \mathcal{C}(f)(\alpha).$$

For $\mathcal{F} = \{f_1, \ldots f_m\}$ a set of Boolean constraints over variable set X, an explicit encoding *of $\mathcal{F}$ is*

$$\mathcal{C}(\mathcal{F}) \stackrel{\text{def}}{=} \bigcup_{i=1}^{m} \{\mathcal{C}(f_i)\}$$

where $\mathcal{C}(f_i)$ is an explicit encoding of f_i, for $i = 1 \ldots m$.

Our lower bounds will require a stronger form of expansion, called *boundary expansion*.

Definition 2.28 (Boundary Expansion) *Let $G =< V \dot\cup U, E >$ be a bipartite graph. The* boundary *of $V' \subseteq V$ is*

$$\partial V' \stackrel{\text{def}}{=} \{u \in U : |N(u) \cap V'| = 1\}$$

In words: the boundary *of V' is the set of* unique neighbors *of V', such that every boundary element is a neighbor of exactly one vertex of V'.*

*G is called an (r, c)-*boundary expander *if*

$$\forall V' \subseteq V, \quad |V'| \leq r \quad |\partial V'| \geq c \cdot |V'|$$

A set of constraints $\mathcal{F}$ is said to be (r, c)-boundary expanding, if $\mathbb{G}(\mathcal{F})$ is an (r, c)-boundary expander.

2.7.2 Expansion and Boundary Expansion

As one might expect, there is a connection between expansion and boundary expansion. A very good expander is also a decent boundary expander.

Lemma 2.29 *If $G =< V \dot\cup U, E >$ is an (r, c)- expander of maximal degree d on the V side, then it is an $(r, 2 + 2c - d)$-boundary expander.*

Proof: By definition of expansion, for any $V' \subseteq V \;\; |V'| \leq r$, we have

$$|N(V')| \geq (1 + c)|V'|$$

On the other hand, since every non boundary element must have at least two neighbors in V', we get

$$|\partial V'| + \frac{1}{2}(d|V'| - |\partial V'|) \geq |N(V')|$$

Combining these two constraints completes the proof. □

2.7.3 Width Lower Bound

We are ready to state and prove the main theorem of this section.

Theorem 2.30 (Width Lower Bounds for Explicit Encodings) *For $\mathcal{F}$ an (r, c)-boundary expanding set of ℓ-sensitive constraints, and $\mathcal{C}(\mathcal{F})$ an explicit encoding of it as a CNF:*

$$W(\mathcal{C}(\mathcal{F})) > \frac{r \cdot (c - \ell + 1)}{2}$$

For the proof of our main theorem, we need the following analog of theorem 2.18, saying that a clause that is minimally implied by a expanding set of sensitive constraints, has large width. A set of constraints is said to *minimally imply* a clause C, if $\mathcal{F}$ implies C but any proper subset of it does not. Notice that one cannot hope to prove theorem 2.18 for arbitrary sensitive constraints, as the following counter example shows. Let $\mathcal{F} = \{\bigoplus_{i=1}^{n} x_i, \;\; \bigoplus_{i=1}^{n} x_i \oplus 1\}$. $|\mathcal{F}| = 2, \;\; |Supp(\mathcal{F})| = n$ and $\mathcal{F}$ minimally implies $\mathbf{0}$.

Theorem 2.31 *If $\mathcal{F}$ is a $(|\mathcal{F}|, c)$-boundary expanding set of ℓ-sensitive constraints, minimally implying a clause C, then:*

$$|C| > |\mathcal{F}| \cdot (c - \ell + 1)$$

Proof: Let ρ be the falsifying restriction of C from definition 2.6. Since $|\rho| = |C|$ we get:

$$|\partial \mathcal{F}| \leq |\partial(\mathcal{F}|_\rho)| + |C| \tag{2.1}$$

By the minimality of $\mathcal{F}$, $\mathcal{F}|_\rho$ is minimal unsatisfiable, for otherwise it could be replaced by some smaller $\mathcal{F}' \subset \mathcal{F}$ such that $\mathcal{F}'|_\rho$ is unsatisfiable, and hence $\mathcal{F}' \models C$. We claim that for any $f \in \mathcal{F}$

$$|Supp(f|_\rho) \cap \partial(\mathcal{F}|_\rho)| \leq \ell - 1 \tag{2.2}$$

If this is not the case, there exists a set of ℓ boundary variables $X \subseteq Supp(f|_\rho) \cap \partial(\mathcal{F}|_\rho)$ that is not fixed by ρ. Let α be an assignment that extends ρ and fixes $\mathcal{F} \setminus \{f\}$ to 1. Such an assignment must exist by the minimal unsatisfiability of $\mathcal{F}|_\rho$. By the ℓ-sensitivity of f, we can change α on X, satisfy f, and not change the value of $\mathcal{F} \setminus \{f\}$, since X consists of boundary variables appearing only in $Supp(f)$. $\mathcal{F}|_\rho$ is satisfiable, contradiction, and equation (2.2) is proved. Hence $|\partial(\mathcal{F}|_\rho)| \leq (\ell - 1) \cdot |\mathcal{F}|$, and now we sum up, using inequality (2.1):

$$\begin{aligned} (\ell - 1) \cdot |\mathcal{F}| &\geq |\partial(\mathcal{F}|_\rho)| \\ &\geq |\partial \mathcal{F}| - |C| \\ &\geq c \cdot |\mathcal{F}| - |C| \end{aligned}$$

The theorem is proved. □

Proof [Theorem 2.30]: Assume $c > \ell - 1$, otherwise the claim is trivial. Define the following proper width measure (recall definition 2.19):

$$\mu(C) = \mu_{\mathcal{F}}(C) \stackrel{\text{def}}{=} \min\{|\mathcal{F}'| : \mathcal{F}' \subseteq \mathcal{F}, \mathcal{F} \models C\}$$

Claim 2.32 *μ is a proper width measure.*

Proof: Clearly, for any $C \in \mathcal{C}(\mathcal{F}), \mu(C) \leq 1$. μ is sub-additive, because if C was derived in a single step from clauses D, E, then for any sets $\mathcal{F}_D, \mathcal{F}_E$ such that $\mathcal{F}_D \models D$, $\mathcal{F}_E \models E$, we get (by the soundness of resolution) $\mathcal{F}_D \cup \mathcal{F}_E \models C$, and hence $\mu(C) \leq \mu(D) + \mu(E)$. □

Theorem 2.31 implies that every $\mathcal{F}' \subset \mathcal{F}$ of size $|\mathcal{F}'| \leq r$ is satisfiable. To see this, notice that $|\mathbf{0}| = 0$, and $\mathcal{F}'$ is by definition an $(|\mathcal{F}'|, c)$-boundary expander. Hence $\mu(\mathbf{0}) > r$, and so by lemma 2.20, any proof of $\mathcal{C}(\mathcal{F})$ must include a clause C with $r/2 \leq \mu(C) < r$. Using theorem 2.31 (again), we get

$$|C| > \mu(C) \cdot (c - \ell + 1) \geq \frac{r \cdot (c - \ell + 1)}{2}$$

The theorem is proved. □

2.8 Width of Implicit Encodings

This section offers our best generalization of theorem 2.14, giving width lower bounds for *implicit encodings* of sets of constraints. Let us give some intuition about implicit encodings. The famous proof of the NP-completeness of 3-SAT is a reduction of the general satisfiability problem to that of satisfiability of 3-CNFs. This reduction says that for any circuit $\mathbf{C}$ with n inputs, and m gates of fan-in 2, there exists a 3-CNF $\mathcal{C}$, with $m+n$ variables, and $O(m)$ clauses, that *encodes* $\mathcal{C}$, i.e.

$$\forall \alpha \in \{0,1\}^n \quad (\mathbf{C}(\alpha) = 1) \quad \text{iff} \quad (\mathcal{C}|_\alpha \in \mathbf{SAT}\ ).$$

Moreover, the reduction is efficiently computable. The following definition is a generalization of this encoding.

Definition 2.33 (Implicit Encoding) *For $f(x_1, \ldots, x_n)$ a Boolean constraint, and a 3-CNF formula $\mathcal{I}(x_1, \ldots, x_n, y_1, \ldots, y_m)$, we say $\mathcal{I}$ is an* implicit encoding *of f if*

$$\forall \alpha \in \{0,1\}^n \qquad (f(\alpha) = 1 \iff \exists \beta \in \{0,1\}^m \ \ \mathcal{I}(\alpha, \beta) = 1)$$

$y_1, \ldots, y_m$ are called the auxiliary variables *of $\mathcal{I}$ with respect to f, and $x_1, \ldots, x_n$ are called the* essential variables.

For $\mathcal{F} = \{f_1 \ldots f_\ell\}$ a set of Boolean constraints over $x_1, \ldots, x_n$, an implicit encoding *of $\mathcal{F}$ is*

$$\mathcal{I}(\mathcal{F}) = \bigcup_{i=1}^{\ell} \{\mathcal{I}_i\},$$

where $\mathcal{I}_i$ is an implicit encoding of f_i, using unique auxiliary variables, *i.e. for any $i \neq j, \ \ i,j \in [\ell]$*

$$Vars(\mathcal{I}_i) \cap Vars(\mathcal{I}_j) \subseteq \{x_1, \ldots, x_n\}.$$

The main theorem of this section is the following lower bound on proof width of implicit encodings of expanding sets of sensitive constraints.

Theorem 2.34 (Width Lower Bounds for Implicit Encodings) *For $\mathcal{F}$ any (r,c)-boundary expanding set of ℓ-sensitive constraints, each of support size $\leq d$, and any implicit encoding $\mathcal{I}(\mathcal{F})$ of $\mathcal{F}$:*

$$W(\mathcal{I}(\mathcal{F})) \geq \frac{r \cdot (c - \ell + 1)}{2(d - \ell + 1)}$$

Remark: Theorem 2.34 can be proved for even more general encodings of constraints, where one allows using *all* possible circuits computing f_i, each with distinct extension variables. The exact definition for this *functional encoding*, as well as motivation for it, can be found in definition 2.4 of [Alekhnovich *et al.* A ('00)].

Proof: We would like to use the measure μ used in the proof of the previous theorem 2.30. The problem with this measure is the existence of auxiliary variables, that may succinctly encode a large number of essential variables. To deal with this problem we need a variant of μ. Define $\mu'(C)$ to be the size of the minimal $\mathcal{F}' \subset \mathcal{F}$ such that the following pair of conditions hold:

1. If $y \in C$ is an auxiliary variable used in the implicit encoding of f_i, then there is some $f_j \in \mathcal{F}'$ such that $Supp(f_i) \subseteq Supp(f_j)$.

2. $\mathcal{F}' \models C$.

Claim 2.35 *μ' is a proper width measure.*

Proof: Clearly $\mu'(A) \leq 1$ for any axiom $A \in \mathcal{I}(\mathcal{F})$. μ' is sub-additive, by the same reasoning showing that μ is sub-additive. □

$\mu'(\mathbf{0}) > r$, because $\mu(\mathbf{0}) > r$ and $\mu' \geq \mu$. Hence, once again we conclude that there must exist a clause C with $r/2 < \mu'(C) \leq r$. Fix this C, and some set $\mathcal{F}'$ minimally implying C. We claim that C has large width.

Let us partition $\mathcal{F}'$ into $\mathcal{F}_1$, any minimal subset satisfying property 1, and $\mathcal{F}_2 = \mathcal{F}' \setminus \mathcal{F}_1$. Notice that by the minimality of $\mathcal{F}'$, removing any constraint f_j from $\mathcal{F}_2$ will ruin property 2. We claim that for any $f_j \in \mathcal{F}_2$, f_j has small intersection with $\partial\mathcal{F}'$. Namely,

$$|Supp(f_j) \cap \partial\mathcal{F}'| \leq \ell - 1 \tag{2.3}$$

Indeed, as noticed above, $\mathcal{F}_2 \setminus \{f_j\} \not\models C$. Let α be an assignment such that $\mathcal{F}_2 \setminus \{f_j\}(\alpha) = 1$ but $C(\alpha) = 0$. Let ρ be the restriction that does not assign values to $Supp(f_j) \cap \partial\mathcal{F}'$ and agrees with α elsewhere. Since ρ sets all variables of $\{Supp(f) : f \in \mathcal{F}' \setminus \{f_j\}\}$, and also to all variables of C (by property 1 and because $f_j \notin \mathcal{F}_1$), we get $\mathcal{F}' \setminus \{f_j\}|_\rho = 1$ and $C|_\rho = 0$. Hence, by property 2 we conclude that $f_j|_\rho = 0$. Since f_j is ℓ-sensitive, we get the desired inequality 2.3. Now we may sum up:

$$\begin{aligned} c \cdot |\mathcal{F}'| &\leq |\partial\mathcal{F}'| \\ &\leq (\ell-1)|\mathcal{F}_2| + d \cdot |\mathcal{F}_1| \\ &= (\ell-1)|\mathcal{F}'| + (d-\ell+1) \cdot |\mathcal{F}_1| \\ &\leq (\ell-1)|\mathcal{F}'| + (d-\ell+1) \cdot |C| \end{aligned}$$

Recalling that $|\mathcal{F}'| > r/2$, we get our bound. □

2.9 Expanding Contradictions Part 2: Tseitin Formulas

In this section we discuss a major class of contradictions for which we obtain width and size lower bounds via theorems 2.30 and 2.34. These contradictions, introduced by [Tseitin ('68)] for proving the first exponential lower bounds for a variant of resolution (regular resolution), formalize the basic property of graphs: **"For G a connected graph, the sum of degrees of G is even".** We shall see that if G has good expansion, then the natural formulation of this property will require large proofs. We start by giving the necessary definitions, and then prove our lower bound.

Definition 2.36 (Graph Expansion) *Let $G = <V, E>$ be a graph on n vertices. for S a subset of V, let $E(S, \bar{S})$ denote the set of edges connecting some vertex in S to some vertex in $V - S$. The* expansion constant *of G is:*

$$e(G) \stackrel{\text{def}}{=} \min\{\frac{|E(S,\bar{S})|}{|S|} : S \subset V,\ |S| \leq |\bar{S}|\}$$

Remark: Good expanders exist, i.e. there exist infinite families $\{G_n\}_{\mathbb{N}}$, $|G_n| = n$, of maximal degree 3, and constant expansion $e > 0$ (for a discussion of expanders see e.g. [Alon, Spencer and Erdős ('92)]).

2.9.1 Tseitin Formulas - Definition

Fix $G =< V, E >$ a finite connected graph (V is the vertex set, and E is the edge set), with $|V| = n$. A labeling of G is some Boolean function $\sigma : V \rightarrow \{0, 1\}$. A labeling σ is said to have *odd weight* if $\Sigma_{v\in V}\sigma(v) \equiv 1 \pmod 2$, and otherwise it is said to have *even weight*. Assign a distinct variable x_e to each edge $e \in E$. For $v \in V$ let $X(v)$ be the set of variables corresponding to edges touching v, i.e. $X(v) \stackrel{\text{def}}{=} \{x_e : v \in e\}$. For X a set of variables, let $\bigoplus X \stackrel{\text{def}}{=} \oplus_{x\in X} x$.

Definition 2.37 *For f a function, and b a variable, or a bit, let $EQ(f, b)$ be the function that is 1 iff $f = b$:*

$$EQ(f, b) \stackrel{\text{def}}{=} f \oplus b \oplus 1$$

By definition 2.27, an explicit encoding of $EQ(\bigoplus X, b)$ is the unique CNF $\mathcal{C}(EQ(\bigoplus X, b))$ over X, that computes $EQ(\bigoplus X, b)$. It is easy to see that this CNF has $2^{|X|-1}$ clauses of size $|X|$ each. By definition 2.33, an implicit encoding of $EQ(\bigoplus X, b)$ is some CNF $\mathcal{I}(EQ(\bigoplus X, b))$ over a variable set $X \cup Y$. Notice the difference between EQ, $\mathcal{C}(EQ)$ and $\mathcal{I}(EQ)$: the first is a function, the second is a CNF formula over X that computes this function, and the third is a CNF formula over $X \cup Y$ that computes this function implicitly, in the sense that an assignment α to the X variables satisfies EQ iff there exists an assignment β to the Y variables such that $< \alpha, \beta >$ satisfies $\mathcal{I}(EQ)$.

Definition 2.38 (Tseitin Formulas) *The* explicit Tseitin formula *of $G =< V, E >$ and σ is:*

$$\mathcal{C}[G, \sigma] \stackrel{\text{def}}{=} \bigcup_{v\in V} \{\mathcal{C}(EQ(\bigoplus X(v), \sigma(v)))\}$$

where $\mathcal{C}(EQ(\bigoplus X(v), \sigma(v)))$ is the explicit encoding of $EQ(\bigoplus X(v), \sigma(v))$.

An implicit Tseitin formula *of G and σ is:*

$$\mathcal{I}[G, \sigma] \stackrel{\text{def}}{=} \bigcup_{v\in V} \{\mathcal{I}(EQ(\bigoplus X(v), \sigma(v)))\}$$

where $\mathcal{I}(EQ(\bigoplus X(v), \sigma(v)))$ is some implicit encoding of $EQ(\bigoplus X(v), \sigma(v))$, using unique auxiliary variables Y_v.

If the maximal degree of G is small, then so is the size of $\mathcal{C}[G, \sigma]$. Additionally, $\bigoplus X$ has a circuit of size $O(|X|)$, hence for any G, σ there exists an implicit Tseitin formula of size $O(|E|)$. A Tseitin formula is satisfied iff for every vertex v, the weight of the assignment to the edges incident with v is equal to $\sigma(v)$. The proof of the following lemma can be found in [Urquhart ('95)].

Lemma 2.39 *[Urquhart ('95)] For G a connected graph, $\mathcal{C}[G, \sigma]$ is a contradiction iff σ has odd weight.*

Thus, whenever we talk about a *Tseitin contradiction* we mean a Tseitin formula of an odd labeled graph.

2.9.2 Lower Bounds for Tseitin Formulas

The main results of this section are the following lower bounds.

Theorem 2.40 *For G a graph with expansion constant $e(G)$, and maximal degree d, and σ a labeling of its vertices, let $\mathcal{C}[G,\sigma]$ be the explicit Tseitin contradiction based on G and σ, and let $\mathcal{I}[G,\sigma]$ be an explicit Tseitin contradiction based on G and σ.*

$$W(\mathcal{C}[G,\sigma]) \geq \frac{e(G)\cdot n}{4}$$

$$W(\mathcal{I}[G,\sigma]) \geq \frac{e(G)\cdot n}{4d}$$

Let us give a short list of corollaries, all of them following immediately from the previous theorem and the basic size width relations (theorems 2.4 and 2.5). The size lower bounds for explicit encodings were first proved by [Urquhart ('87)], and the generalization to implicit encodings was first presented by [Alekhnovich *et al.* A ('00)].

Corollary 2.41 *For G a graph with expansion constant $e(G)$, and maximal degree d, and σ a labeling of its vertices, let $\mathcal{C}[G,\sigma]$ be the explicit Tseitin contradiction based on G and σ, and let $\mathcal{I}[G,\sigma]$ be an explicit Tseitin contradiction based on G and σ.*

- $\mathbf{S_T}(\mathcal{C}[G,\sigma]) \geq 2^{\frac{e(G)\cdot n}{4}-d}$
- $\mathbf{S_R}(\mathcal{C}[G,\sigma]) = \exp(\Omega((\frac{e(G)\cdot n}{4}-d)^2/dn)) = \exp(\Omega(n\cdot\frac{e(G)^2}{16d}-\frac{d}{2}))$

For $\{G_n\}_{\mathbb{N}}$ a family of graphs of constant maximal degree d, and constant graph expansion $e>0$,

- $\mathbf{S_T}(\mathcal{C}[G,\sigma]) \geq \mathbf{S_T}(\mathcal{I}[G,\sigma]) = \exp(\Omega(n))$
- $\mathbf{S_R}(\mathcal{C}[G,\sigma]) = \exp(\Omega(n))$

If $\mathcal{I}[G,\sigma]$ uses at most a constant number of auxiliary variables per vertex, then

- $\mathbf{S_R}(\mathcal{I}[G,\sigma]) = \exp(\Omega(n))$

2.9.3 Proof of Theorem 2.40

Tseitin contradictions are known to have large proofs in Resolution, as long as G is a good expander. This was proved initially by Tseitin for regular resolution in his important paper [Tseitin ('68)]. Later, Urquhart improved the lower bounds basing them on expander graphs (which were not known at the time Tseitin published his paper), and proved them for general resolution as well, making this the first proof of a resolution size lower bound that is exponential in the size of the input contradiction [Urquhart ('87)].

Proof [Theorem 2.40]: For $S\subset V$, the boundary of $\{EQ(\bigoplus X(v),\sigma) : v\in S\}$ is exactly $\{x_e : e\in E(S,\bar{S})\}$, and hence $\mathcal{C}[G,\sigma]$ is an $(n/2,e(G))$-boundary expanding contradiction. Additionally, the parity function is 1-sensitive. Applying theorems 2.30, and 2.34 respectively, completes the proof. $\square$

2.10 Expanding Contradictions Part 3: Generator Contradictions

Researchers in theoretical computer science know all too well, that proving non-trivial claims about complicated Boolean functions, is a very hard problem[3]. In this section we offer an initial explanation of this hardness, in the limited context of resolution. The hardness we show is not a property of a *specific contradiction*, but rather a property of many contradictions, emerging from one hard *function*. In particular, we show that proving basic properties of a generator (= a circuit that stretches few bits to many bits) is hard for resolution. This will be shown for a class of Nisan-Wigderson style generators. The result itself is a direct application of theorems 2.30 and 2.34 to these generators, and does not introduce any new mathematical techniques. The main purpose of this section is to illustrate that our previous lower bounds imply the hardness of reasoning about complicated Boolean functions.

We start with a different view of Tseitin formulas. For G a connected graph with n vertices and m edges, look at the multi-output, Boolean function $F : \{0,1\}^m \to \{0,1\}^n$, defined by $F(\vec{x}) = A \cdot \vec{x}$, where A is the $n \times m$ matrix such that $a_{ij} = 1$ iff the j'th edge is incident with the i'th vertex, and all calculations are done modulo 2. Theorem 2.40 showed that if G is an expander, then for *any* $\sigma \in \{0,1\}^n$, proving the following claim:

$$\sigma \notin Im(F)$$

is very hard for resolution.[4] Notice that any graph G gives rise to many hard formulas, one for each $\sigma \in \{0,1\}^n$. When viewed as a function, the natural formulation of the claim $\sigma \notin Im(F)$ is hard, regardless of the identity of σ. Generalizing this example, in [Alekhnovich *et al.* A ('00)] we proposed the following paradigm: a function $F : \{0,1\}^n \to \{0,1\}^m$ is hard for some propositional proof system P if and only if for *any* string $b \in \{0,1\}^m$ there is no efficient P-proof of the properly encoded statement $F(x_1, \ldots, x_n) \neq b$. Thus, even the basic fact that F is not a surjection, does not have a P-proof. A similar suggestion was independently made by [Krajíček ('99)].

Let us present our result in comparison with *hard* functions for circuits, called *pseudorandom generators*. A *generator* is a function that stretches bits, i.e. $G : \{0,1\}^n \to \{0,1\}^m$, where $m > n$. Pseudorandom generators, originally introduced by [Yao ('82)] have become by now one of the most important concepts in theoretical computer science penetrating virtually all its sub-areas. The mapping G is called *pseudorandom* for the circuit class $\mathbb{C}$ if no "small" circuit $C(y_1, \ldots, y_m) \in \mathbb{C}$ can distinguish between the two probabilistic distributions $G(\boldsymbol{x})$ and $\boldsymbol{y}$ in the sense that $|\mathbf{Pr}[C(G(\boldsymbol{x})) = 1] - \mathbf{Pr}[C(\boldsymbol{y}) = 1]|$ is small ($\boldsymbol{x}$ is picked at random from $\{0,1\}^n$, and $\boldsymbol{y}$ is picked at random from $\{0,1\}^m$).

One of the most useful pseudorandom generators is the class of Nisan-Wigderson generators [Nisan and Wigderson ('94)]. Let A be an $(m \times n)$ 0-1 matrix, $f_1, \ldots, f_m$

[3] E.g. the problem of $P \neq NP$ is that of proving that the SAT function does not have efficient algorithms

[4] Recall that if σ has even weight, then $\sigma \in Im(F)$ and hence the claim has no proof, i.e. the proof size is ∞.

be functions such that $Supp(f_i) \subseteq \{x_j : a_{ij} = 1\}$ and $NW : \{0,1\}^n \to \{0,1\}^m$ be given by

$$NW(x_1, \ldots, x_n) \stackrel{\text{def}}{=} (f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n)).$$

[Nisan and Wigderson ('94)] proved that if A satisfies the following combinatorial conditions:

1. For all $i \in [m]$ $\quad |Supp(f_i)| \leq d$, for small enough d.
2. For any $1 \leq i < i' \leq m$, $|Supp(f_i) \cap Supp(f_{i'})| \leq k$, for small enough k.
3. The functions f_i are computationally hard, i.e. cannot be computed by a small circuit.

Then NW is a good pseudorandom generator in the computational sense. Our theorem 2.44 and corollary 2.47, show something quite similar: if $\mathcal{F} = \{f_1 \ldots f_m\}$ is an (r, c)-boundary expanding set of functions of small support, and all f_i's are hard for resolution, i.e. *sensitive*, then the generator $F : \{0,1\}^n \to \{0,1\}^m$ defined by

$$F(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \{f_1(x_1, \ldots, x_n), \ldots f_m(x_1, \ldots, x_n)\},$$

is hard for resolution. Before presenting our results formally, let us mention two other connections of computational hardness to proof hardness:

Hardness in presence of Feasible Interpolation Let $G_n : \{0,1\}^n \to \{0,1\}^m$ be an arbitrary pseudorandom generator that is hard w.r.t. poly-size (in $m+n$) circuits, and let $n < m/2$. Following [Razborov B ('95)], let us take bit-wise XOR of two independent copies of this generator $G'_n : \{0,1\}^{2n} \to \{0,1\}^m$; $G'_n(x_1, \ldots, x_n, x'_1, \ldots, x'_n) \stackrel{\text{def}}{=} G_n(x_1, \ldots, x_n) \oplus G_n(x'_1, \ldots, x'_n)$. Then G'_n is hard for any propositional proof system P which has the property of feasible interpolation (for a definition see e.g. [Beame and Pitassi ('98)]).

Natural Proofs Let $F_n : \{0,1\}^{n^k} \to \{0,1\}^{2^n}$ be any pseudorandom *function generator* that stretches n^k random bits to a Boolean function in n variables viewed as a string of length 2^n in its truth-table representation. Assume that F_n is hard w.r.t. $2^{O(n)}$-sized circuits. [Razborov and Rudich ('94)] proved that for *any* function f_n, viewed as its truth table (i.e. a string of size 2^n) there is no "natural" (in the strict sense also defined in that paper) proof of the fact $f_n \notin Im(F_n)$.

If we manage to increase significantly the number of output bits and construct a poly-time computable function generator $G_n : \{0,1\}^{n^k} \to \{0,1\}^{2^n}$ that is hard for P, we, similarly to [Razborov and Rudich ('94), Razborov B ('95)], can conclude that in the framework proposed in [Razborov A ('95)] there are no efficient P-proofs of $\mathbf{NP} \not\subseteq \mathbf{P}/poly$.

Here is a formal definition of a "proper encoding" of the statement $F(x_1, \ldots, x_n) \neq b$, followed by the main results of the section.

Definition 2.42 (Parametrized Formulas) *A* parametrized function (circuit, CNF) *is a function (circuit, CNF) over two disjoint variable sets, one called the* variables, *and the other called the* parameters. *The* parametrized proof width *of a parametrized CNF $\mathcal{C}$ is*

$$W(\mathcal{C}) \stackrel{\text{def}}{=} \min_{\rho}\{W(\mathcal{C}|_{\rho})\}$$

Where ρ ranges over all restrictions setting all the parameters, and none of the variables. The parametrized proof size *and* parametrized treelike proof size *of $\mathcal{C}$ are similarly defined.*

Definition 2.43 (Parametrized Formulations of Functions) *For $F : \{0,1\}^n \rightarrow \{0,1\}^m$, an* explicit parametrized formulation *of F is a CNF:*

$$\mathcal{C}(F) \stackrel{\text{def}}{=} \bigcup_{i=1}^{m}\{(\mathcal{C}(EQ(f_i, z_i)))\}$$

where f_i is the projection of F on the i'th bit, and z_i is a unique output parameter *of the i'th function, appearing only in $\mathcal{C}(EQ(f_i, z_i))$.*

An implicit parametrized formulation *of F is some 3-CNF*

$$\mathcal{I}(F) \stackrel{\text{def}}{=} \bigcup_{i=1}^{m}\{(\mathcal{I}(EQ(f_i, z_i)))\}$$

where $\mathcal{I}(EQ(f_i, z_i))$ is an implicit formulation of the function $EQ(f_i, z_i)$, and z_i is a unique parameter appearing only in $\mathcal{I}(EQ(f_i, z_i))$.

F is called an (r, c)-*boundary expanding function*, if $\mathcal{F} = \{f_1, \ldots f_m\}$ is an (r, c)-boundary expanding set of functions. F has *maximal support* d, if for all $i \in [m]$ f_i has maximal support d. F is ℓ-sensitive if for all $i \in [m], b \in \{0,1\}$, the function $EQ(f_i, b)$ is ℓ-sensitive.

The main result of this section is the following theorem. It is merely a reformulation of theorems 2.30, 2.34, in the language of parametrized functions, and thus its proof is identical. The theorem states that if F is an expanding function with m output bits, then for *any* binary string α of size m, the claim "$\alpha \notin Range(F)$" has no short resolution proof.

Theorem 2.44 *Let $F : \{0,1\}^n \rightarrow \{0,1\}^m$ be an ℓ-sensitive, (r, c)-boundary expanding generator, of maximal support d. Let $\mathcal{C}(F)$ be an explicit parametrized formulation of F, and let $\mathcal{I}(F)$ be an implicit parametrized formulation of F.*

$$W(\mathcal{C}(F)) \geq \frac{r \cdot (c - \ell + 1)}{2}$$

$$W(\mathcal{I}(F)) \geq \frac{r \cdot (c - \ell + 1)}{2(d - \ell + 1)}$$

Corollary 2.45 *For $F, \mathcal{C}(F), \mathcal{I}(F)$ as in theorem 2.44:*

- $\mathbf{S_T}(\mathcal{C}(F)) \geq 2^{\frac{r\cdot(c-\ell+1)}{2}}$
- $\mathbf{S_R}(\mathcal{C}(F)) = \exp(\Omega(\frac{r^2}{n} \cdot \frac{(c-\ell+1)^2}{4(d-\ell+1)^2}))$

The following claim shows the existence of extremely strong boundary expanders. Its proof (a simple union bound) appears in appendix A.2.

Theorem 2.46 *For any parameters $m > n > 2$ there exists a bipartite graph $G =< V \dot\cup U, E >$, $|V| = m, |U| = n$, of maximal degree $d = O(\log m)$ on the V side, which is an $(\Omega(n/d), \frac{3d}{4})$-boundary expander.*

Using the previous theorem 2.46, with constant ℓ, we get the existence of:

- Explicit hard generators for resolution for $m = 2^{n^\epsilon}$.
- Implicit hard generators for treelike resolution for $m = 2^{n^\epsilon}$.
- Implicit hard generators for general resolution for the weaker parameter $m = n^{2-\epsilon}$.

Corollary 2.47 *For any parameters $m > n > 2$ there exist generators $F : \{0,1\}^n \to \{0,1\}^m$ such that for any explicit formulation $\mathcal{C}(F)$, and implicit formulation $\mathcal{I}(F)$:*

- $W(\mathcal{C}(F)) = \Omega(n)$.
- $W(\mathcal{I}(F)) = \Omega(n/\log m)$.
- $\mathbf{S_T}(\mathcal{C}(F)) \geq \mathbf{S_R}(\mathcal{C}(F)) = 2^{\Omega(n)}$
- $\mathbf{S_T}(\mathcal{I}(F)) = 2^{\Omega(n/\log m)}$

If the implicit encoding of F uses $O(m)$ variables, and $m = n^{2-\delta}, \delta \in (0,1)$:

- $\mathbf{S_R}(\mathcal{I}(F)) = \exp(\Omega(n^\delta))$.

2.11 Expanding Contradictions Part 4: The Pigeonhole Principle

Our final examples of hard contradictions in this chapter, are two variations of the *pigeonhole principle*, which was the first contradiction to be proven hard for resolution [Haken ('85)]. We will define this principle, and then state and prove our lower bounds.

The Pigeonhole Principle with m pigeons and n pigeon-holes, states that there is no 1-1 map from m to n, as long as $m > n$. This can be stated by a formula on $n \cdot m$ variables $x_{ij}, 1 \leq i \leq m, 1 \leq j \leq n$, where $x_{ij} = 1$ means that i is mapped to j.

Definition 2.48 ($\mathbf{PHP^m_n}$) $PHP^m_n \stackrel{\text{def}}{=} \mathcal{P}^m_n \cup \mathcal{H}^m_n$, *where*

Pigeon Axioms $\mathcal{P}^m_n$ *is the set of* pigeon axioms P_i, $i \in m$, *and* $P_i \stackrel{\text{def}}{=} \bigvee_{1\leq j\leq n} x_{ij}$.

Hole Axioms $\mathcal{H}^m_n$ *is the set of* hole axioms $H^j_{i,i'}$, $1 \leq i < i' \leq m, 1 \leq j \leq n$, *and* $H^j_{i,i'} \stackrel{\text{def}}{=} \bar{x}_{ij} \vee \bar{x}_{i'j}$

Whenever $m > n$, PHP_n^m is an unsatisfiable CNF with $m \cdot n > n^2$ variables, $O(m^2)$ Clauses and initial width n. This formula was the first for which exponential resolution size lower bounds were proved, in the breakthrough result of Haken.

Theorem 2.49 *[Haken ('85)]* $\mathbf{S_R}(PHP_n^{n+1}) = 2^{\Omega(n)}$

The previous lower bound was generalized for by Buss and Turan for larger m. Late we shall show a simple proof of their result:

Theorem 2.50 *[Buss and Turán ('88)]* $\mathbf{S_R}(PHP_n^m) = 2^{\Omega(\frac{n^2}{m \log m})}$

The previous lower bound becomes trivial when $m = n^2$. The question of the weak pigeonhole complexity (i.e. for $m \geq n^2$) was open until the recent breakthrough of [Raz ('01)]:

Theorem 2.51 *[Raz ('01)] There exists an $\epsilon > 0$ such that for all $m > n$:*

$$\mathbf{S_R}(PHP_n^m) \geq 2^{n^\epsilon}.$$

Even more recently, [Razborov ('01)] presented a simplification of Raz's proof, that also gives a better constant than that presented by Raz (there $\epsilon \sim 1/10$). Interestingly, this latter proof uses a generalization of our width method.

Theorem 2.52 *[Razborov ('01)] For any $m > n$:*

$$\mathbf{S_R}(PHP_n^m) \geq 2^{\Omega(n^{1/3})}.$$

2.11.1 Width Upper Bounds

The large initial width of PHP_n^m and its large number of variables does not allow the derivation of size lower bounds directly from width lower bounds, as the following lemma shows (recall that PHP_n^m has clauses of width n).

Lemma 2.53 *For any $m > n \geq 2$, $W(PHP_n^m) = n$.*

Proof: By induction on n. For $n = 2$ notice that PHP_2^m is a 2-CNF, and for any unsatisfiable 2-CNF $\mathcal{C}$, $W(\mathcal{C}) \leq 2$ because a resolution of two clauses of width 2 results in another clause of width ≤ 2. For the induction step, restrict PHP_n^m to $x_{11} = 1$, and call this restriction ρ. $H_{1,i}^1|_\rho = \bar{x}_{i1}$ for all $1 < i \leq m$. Use these singleton clauses to remove x_{i1} from all clauses P_i for $1 < i \leq m$. We have reduced $PHP_n^m|_\rho$ in width n to PHP_{n-1}^{m-1}. By the induction hypothesis $W(PHP_{n-1}^{m-1}) = n - 1$, and applying lemma 2.7 completes the proof. $\square$

We are ready to prove the width lower bounds for the variants of the pigeonhole principle. The first variant will give as a corollary optimal lower bounds for treelike proof size, and the second will give lower bounds for the general proof size.

2.11.2 Implicit Encodings of the Pigeonhole Principle

Recall that an implicit encoding of PHP^m_n is a set of 3-CNFs, one for each clause of PHP^m_n. The main theorem of this subsection is the following lower bound on implicit encodings of the pigeonhole principle.

Theorem 2.54 *For $\mathcal{I}(PHP^m_n)$ an implicit encoding of PHP^m_n,*

$$W(\mathcal{I}(PHP^m_n)) \geq n/3$$

Recall that any implicit encoding is a 3-CNF, and hence applying theorem 2.4 gives:

Corollary 2.55 *For $\mathcal{I}(PHP^m_n)$ an implicit pigeon encoding of PHP^m_n,*

$$\mathbf{S_T}(\mathcal{I}(PHP^m_n)) \geq 2^{(n-1)/3}$$

Corollary 2.56 *For all m, n $\mathbf{S_T}(PHP^m_n) \geq 2^{(n-1)/3}/n$.*

Proof: There exists a implicit encoding $\mathcal{I}(P_i)$ such that $\mathbf{S_T}(\mathcal{I}(P_i) \rightarrow P_i) = n$. Hence, a proof of PHP^m_n of treelike size s can be converted to a proof of some $\mathcal{I}(PHP^m_n)$ of treelike size $s \cdot n$. $\square$

Proof [Theorem 2.54]:

Remark: Since $H^j_{i,i'}$ is a clause of width 2, it makes little sense to introduce an implicit encoding for it. Generally we may assume that the implicit encoding of $H^j_{i,i'}$ is the trivial one ($H^j_{i,i'}$), although our lower bound will still be true even for implicit encodings of the Hole axioms.

Define $\mu(C) = \min\{|\mathcal{P}'| : \mathcal{P}' \subset \mathcal{P}^m_n, \mathcal{I}(\mathcal{P}') \bigwedge \mathcal{I}(\mathcal{H}^m_n) \models C\}$. Notice that any assignment satisfying $\mathcal{P}' \bigwedge \mathcal{H}^m_n$ must be $1-1$ on $\mathcal{P}'$. It is easy to see that μ is a proper width measure (see definition 2.19). Hence in every proof π there must be a clause C with $n/3 \leq \mu(C) < 2n/3$. Fix such a C and let $\mathcal{P}' \subset \mathcal{P}$ be a minimal set such that $\mathcal{I}(\mathcal{P}') \bigwedge \mathcal{I}(\mathcal{H}^m_n) \models C$, and denote $I = \{i : P_i \in \mathcal{P}'\}$. Assume for the sake of contradiction that $|C| < n/3$. In this case there must exist a pigeon axiom $P_i \in \mathcal{P}$ such that no literal of $\mathcal{I}(P_i)$, the implicit encoding of P_i, appears in C. Take any assignment α such that $\mathcal{H}^m_n \bigwedge \mathcal{I}(\mathcal{P}') \setminus \{P_i\} = 1$, and $P_i(\alpha) = C(\alpha) = 0$, which must exist by the minimality of $\mathcal{P}'$. By our earlier observation, α is $1-1$ on $\mathcal{P}' \setminus \{P_i\}$, and sets all variables of P_i to 0. Since $|C| < n/3$, and $|\mathcal{P}'| < 2n/3$ there is some $j \in [n]$ such that $x_{i'j}(\alpha) = 0$ for all $i' \in I$, and C has no literal from $\mathcal{I}(H^j_{i',i''})$, for all $1 \leq i' < i'' \leq m$. Make the following changes to α, to obtain the assignment α'.

1. Set $x_{ij} = 1$, and correct the assignment to the auxiliary variables of $\mathcal{I}(P_i)$ so that $\mathcal{I}(P_i)(\alpha') = 1$. Such a correction must exist because $P_i(\alpha) = 1$. Notice that this change affects only $Vars(\mathcal{I}(P_i))$.

2. Set $x_{i'j} = 0$, and correct the assignments to $\mathcal{I}(H^j_{i,i'})$ so that $\mathcal{I}(H^j_{i,i'}) = 1$ for all $i' \in [m] \setminus \{i\}$. Such an assignment must exist because $H^j_{i,i'}(\alpha') = 1$. Notice that this change affects only $\cup_{i' \in [m] \setminus \{i\}} Vars(\mathcal{I}(H^j_{i,i'}))$.

Since α' agrees with α on $Vars(C)$ and on $Vars(\mathcal{I}(P_{i'}))$ for all $i' \in I \setminus \{i\}$, and satisfies $\mathcal{I}(P_i)$, we get that α' satisfies $\mathcal{I}(\mathcal{P}') \bigwedge \mathcal{I}(\mathcal{H}^m_n)$ while falsifying C. Contradiction, and we conclude $|C| \geq n/3$. $\square$

Width lower bound for General Resolution

The second variant of the pigeonhole principle limits every pigeon to a small number of possible holes. In this situation, proving that no matching exists should become *easier*, but we still get essentially the same lower bounds as for the usual pigeonhole principle. An added benefit of limiting each pigeon to a small number of holes, is that the number of variables decreases, and so we can use the width lower bound to obtain size lower bounds for general resolution. Notice that the width lower bound of theorem 2.54 is not useful for general resolution because $|Vars(PHP_n^m)| \geq n^2$. We start with the proper definition and then prove our main theorem.

Definition 2.57 (G – PHP) *Let $G = ((V\dot{\cup}U), E)$ be a bipartite graph, $|V| = m$, $|U| = n$. Assign each edge a distinct variable x_e. $G - PHP$ is the conjunction of the following clauses:*

$P_v \stackrel{\text{def}}{=} \bigvee_{v\in e} x_e$ *for* $v \in V$.

$H^u_{v,v'} \stackrel{\text{def}}{=} \bar{x}_e \vee \bar{x}_{e'}$ *for* $e = (v,u), e' = (v',u), \quad v, v' \in V,$
$v \neq v', \quad u \in U$.

$G - PHP$ is a natural generalization of the Pigeonhole Principle, because $PHP_n^m = K_{m,n} - PHP$. This generalization will prove useful by the following observation, presented here without a proof:

Lemma 2.58 *For any two bipartite graphs $G =< V, E >, G' =< V', E' >$ over* the same vertex set, *if $E' \subseteq E$ then $\mathbf{S_R}(G' - PHP) \leq \mathbf{S_R}(G - PHP)$.*

The main theorem of this section is:

Theorem 2.59 *For every bipartite graph G that is an (r,c)-boundary expander, $W(G - PHP) \geq \frac{r\cdot c}{2}$.*

Proof: Similar to the previous proof of theorem 2.54. For $V' \subset V$ Denote $\mathcal{P}(V') = \{P_v : v \in V'\}$. Define $\mu(C) = \min\{|V'| : V' \subset V, \mathcal{P}(V') \bigwedge \mathcal{H}_n^m \models C\}$. Again, it is not hard to see that μ is a proper width measure (notice that $\mu'(\mathbf{0}) > r$ because every V' of size $|V'| \leq r$ has a matching into U). Hence in every proof there must be a clause C with $r/2 \leq \mu(C) < r$. Fix such a C and fix a minimal $V' \subset V$ such that $\mathcal{P}(V') \bigwedge \mathcal{H}_n^m \models C$.

We claim that for each $u \in \partial V'$, there must appear in C some variable $x_{(\hat{v},u)}$, for some $\hat{v} \in V$ (but not necessarily in V'). Indeed, for such a boundary element u, let v be its only neighbor in V'. Assume for the sake of contradiction that C has no variable $x_{(\hat{v},u)}$. Let α be the assignment satisfying $\mathcal{P}(V' \setminus \{v\})$ and falsifying P_v and C. Clearly α assigns zero to $x_{v,u}$. Assume wlog that α sets to zero all variables $x_{(\hat{v},u)}$, because this cannot falsify any axiom $P_{v'}$ for $v' \in V'$ (recall that $x_{(v,u)}$ is a boundary variable). Thus, we may flip α on $x_{(v,u)}$, and get an assignment satisfying $\mathcal{P}(V')$, without changing the value of C, contradiction. Hence the width of C is at least the size of its boundary, and the theorem is proven. $\square$

The previous theorem immediately yields the best size lower bounds that were known until the recent lower bound of Raz (theorem 2.51).

Proof [Theorems 2.49, 2.50]: For $m = n + 1$, theorem 2.46 shows that there exist bipartite graphs which are $(\Omega(n), c)$-boundary expanders, of constant degree and $c > 0$, and together with lemma 2.58, this gives a simple, alternative proof of theorem 2.49. Theorem 2.46 also promises the existence of $(\Omega(\frac{n}{\log m}), \frac{3}{4}\log m)$-boundary expanders of degree $\log m$, and using these expanders theorem 2.59 gives a simple proof of theorem 2.50. $\square$

2.12 Open Questions

What is the resolution size for random CNF formulas with clause density $\Delta > \sqrt{n}$?

Chapter 3

Resolution Space Lower Bounds

3.1 Introduction

This chapter introduces and discusses another measure of sequential proofs, called the *proof space*. We shall prove lower bounds on the space of proving $\mathcal{C}$ in resolution whenever $\mathbb{G}(\mathcal{C})$ is a good expander (recall definition 2.12). We start with an intuitive explanation of the proof space measure.

Suppose we are given a proof of $\mathcal{C}$, as a directed acyclic graph. We wish to verify this proof, using a minimal amount of memory. The proof π, as well as the contradiction $\mathcal{C}$, are written on a read-only memory tape. We keep in our *working memory* a subset of the clauses in the proof (starting with the empty set), and at each time step we either add a clause $C \in \mathcal{C}$ to our memory, *or* apply a single resolution derivation to two clauses *in the memory*, and add the resulting clause to it, *or* remove an unnecessary clause from the memory. The *space* of the proof π is the maximal number of clauses we need to keep at some time during the verification of the proof. Of course, there are many ways to verify a single proof (e.g. we may keep all clauses in the proof, in which case the space will be $|\pi|$), and we naturally define the space to be the minimal one over all possible verifications of π. Finally, the proof space of $\mathcal{C}$ is the minimal space of a proof π of $\mathcal{C}$ (if such a proof exists, and ∞ if no proof exists). For those familiar with the pebbling complexity measure (definition 4.5), the space of π is exactly the pebbling measure of the DAG of π.

Aside from being a natural complexity measure, space crucially influences the practical appeal of various resolution based automated theorem provers (ATP). All commonly used resolution based ATPs produce *treelike* resolution proofs. Clearly, a minimal size general resolution proof is no longer than a treelike one, and actually, as we shall see in chapter 4, the difference between the two can be exponential in size. Why aren't there a lot of efficient general resolution ATPs used in practice ? The answer is *space complexity*. If one wishes to make full use of general resolution in a proof, one needs to save in memory a subset of the clauses derived, in order to reuse them time and again in the proof. The selection and maintenance of this set of clauses is a formidable task, that so far no ATP has managed to overcome (for more detail on this subject, see chapter 4.

The actual space consumption of an ATP is highly specific, but obtaining strong space lower bounds apply to any resolution based algorithm. The lower bounds presented in this chapter (as well as in previous work) may serve as a starting point for understanding the actual space requirements of such ATPs, as well as other important issues like time-space tradeoffs.

In this chapter we present a novel connection of expansion to space. We prove that if $\mathcal{C}$ is an expanding CNF, then the space required to prove $\mathcal{C}$ is large, and increases as the expansion parameters of $\mathcal{C}$ get better (see theorem 3.5 for the exact statement). Our proof uses new techniques, and relies on analysis of a natural two player game played on $\mathbb{G}(\mathcal{C})$.

Before continuing our description of the previous and current results, we pause briefly to give a formal definition of the *clause space*. We start with a different representation of a resolution proof, that exposes the clause space naturally. Comparing this definition with our previous one (section 2.1), one sees they are equivalent in size, up to a polynomial factor.

Definition 3.1 (Resolution Derivation - Alternative Definition) *A* configuration *is a set of clauses. A* proof π *of a CNF* $\mathcal{C}$*, is a sequence of configurations* $\mathcal{C}_0, \ldots, \mathcal{C}_s$ *such that* $\mathcal{C}_0 = \emptyset$, $\mathcal{C}_s = \{\mathbf{0}\}$ *(the empty clause) and for all* $t \in [s]$, $\mathcal{C}_t$ *is obtained from* $\mathcal{C}_{t-1}$ *by one of the following rules:*

AXIOM DOWNLOAD $\mathcal{C}_t := \mathcal{C}_{t-1} \cup C$ *for some clause* $C \in \mathcal{C}$*;*

MEMORY ERASING $\mathcal{C}_t := \mathcal{C}_{t-1} - \mathcal{C}'$ *for some* $\mathcal{C}' \subseteq \mathcal{C}_{t-1}$*;*

INFERENCE ADDING $\mathcal{C}_t := \mathcal{C}_{t-1} \cup C$*, for some* C *obtained by a single application of the resolution rule to two clauses in* $\mathcal{C}_{t-1}$*.*

Definition 3.2 (Clause Space) *The* clause space *of a set of configurations* $\pi = \{\mathcal{C}_0, \ldots, \mathcal{C}_s\}$ *is*

$$Space(\pi) \stackrel{\text{def}}{=} \max\{|\mathcal{C}_i| : i \in [s]\}$$

The resolution clause space *of a CNF* $\mathcal{C}$*, is*

$$Space(\mathcal{C}) \stackrel{\text{def}}{=} \min\{Space(\pi)\}$$

where the minimum is taken over all proofs of $\mathcal{C}$*, and is defined to be* ∞ *if no such proof exists (i.e.* $\mathcal{C} \in \mathbf{SAT}$ *).*

3.2 History

The question of proof space was first introduced by A. Haken during a workshop on "Complexity Lower Bounds", held at the Fields Institute in Toronto, 1998. [Esteban and Toran ('99)] introduced the clause space measure, showed its connection to the DAG-proof structure, and proved other basic properties of this measure, such as the following:

Theorem 3.3 *[Esteban and Toran ('99)] Let* $\mathcal{C}$ *be an unsatisfiable CNF formula with tree-like resolution proofs of size* S*. Then* $\mathcal{C}$ *has a resolution proof of clause space* $\lceil \log S \rceil + 1$*.*

Thus, lower bounds on clause space give lower bounds on treelike size, and we shall recall this theorem later on. Another basic result is the following:

Theorem 3.4 *[Esteban and Toran ('99)] For any unsatisfiable CNF $\mathcal{C}$ over n variables there is a proof of clause space $n + O(1)$.*

[Torán ('99)] further investigated the clause space measure for resolution, and obtained the following results:

1. The clause space of the pigeonhole principle PHP_n^m is $\Omega(n)$.

2. The clause space of a Tseitin formula of an expander graph is linear in the number of variables (edges).

[Alekhnovich *et al.* B ('00)] introduced the symbol space measure, that measures the total number of symbols in the memory (not just the number of clauses). They also generalized the clause space measure to the stronger polynomial calculus proof system (see chapter 5 for definitions), and extended the results of Esteban and Torán to a range of contradictions called *wide contradictions*. Among their results are the following:

1. The clause space of the pigeonhole principle PHP_n^m in the polynomial calculus is $\Omega(n)$.

2. The symbol space of the pigeonhole principle PHP_n^m in resolution and the polynomial calculus is $\Omega(n^2)$.

3. Any contradiction $\mathcal{C}$ can be proved in any Frege system with constant line space and symbol space $O(|\mathcal{C}|)$.

3.3 Main Results

In this chapter we show that expansion yields lower bounds on clause space. This result does not follow from our previous knowledge, and the proof uses new techniques as well. Recall the definition of an expanding CNF (definition 2.12). The main theorem of this chapter is:

Theorem 3.5 *If $\mathcal{C}$ is an (r, c)-expanding CNF, then*

$$Space(\mathcal{C}) \geq \frac{c \cdot r}{2 + c}$$

Applying theorem 2.23 we get good lower bounds for random CNFs, answering an open problem in both [Torán ('99)] and [Alekhnovich *et al.* A ('00)].

Theorem 3.6 *For any $k \geq 3$, any $0 < \epsilon \leq 1/2$ and any Δ, with high probability a random k-CNF with n variables and $\Delta \cdot n$ clauses requires clause space $\Omega\left(n \cdot \Delta^{-\frac{1+\epsilon}{k-2-\epsilon}}\right)$*

For instance, setting Δ to be a constant we get linear lower bounds. Since the clause space of any formula is at most $n + 1$ [Esteban and Toran ('99)], this lower bound is optimal up to a multiplicative constant.

Corollary 3.7 [Constant Δ] *For any constant Δ, a random k-CNF with n variables and Δn clauses requires $\Omega(n)$ clause space to prove.*

Another interesting corollary is for large clause density, which we state for concreteness for $k = 3$.

Corollary 3.8 *For any constant $1 > \delta > 0$, there exists an $\epsilon > 0$ such that with high probability, a random 3-CNF with n variables and $n^{2-\delta}$ clauses requires clause space $\Omega(n^{\epsilon})$ to prove.*

[Esteban and Toran ('99)], proved that the treelike proof size is exponential in the clause space, and thus we obtain another proof of theorem 2.24.

3.4 Proof Outline

For the proof of our main theorem, theorem 3.5, we introduce a 2-player *matching game*, to be played on the bipartite graph $\mathbb{G}(\mathcal{C})$ (recall definition 2.11). The aim of the first player is to prove that there is no perfect matching. The second player is an opponent that instead tries to force a perfect matching. The first player should complete his task "remembering" as few as possible of his moves in the game.

Theorem 3.5 follows immediately from this couple of claims:

1. The space required to prove a CNF $\mathcal{C}$, is at least the space required for winning the matching game on $\mathbb{G}(\mathcal{C})$ (theorem 3.9).

2. If G is an (r, c)-bipartite expander, then the space required for winning the matching game is at least $\frac{c \cdot r}{2+\epsilon}$ (theorem 3.11).

It turns out that our lower bound is quite general, and can be extended to the case of the $G - PHP$ (recall definition 2.57). We prove that the clause space of the $G - PHP$ based on the bipartite graph G can be reduced to the space complexity of the matching game played on G.

We now define the matching game and reduce the proof space to the matching game space(section 3.5), then we prove lower bounds on the matching space (section 3.6), and finally we prove similar lower bounds for the $G - PHP$ formulas (section 3.7).

3.5 The Matching Game

For $G =< (V \cup U), E >$ a bipartite graph, if $|V| > |U|$ then there is no matching of V into U. We wish to prove this claim, using "limited space". For this purpose let us define a two player game. The players are Pete (Prover) and Dana (Disprover). Pete tries to prove that there is no matching from V to U, and Dana tries to prove that such a matching exists. Pete has k fingers, numbered $\{1, \ldots, k\}$, and Dana has k fingers, numbered identically. We start with all vertices of G uncovered, and on each round one of the following occurs:

1. Pete Places a finger j on some uncovered $v \in V$, and Dana must answer by placing her finger j on some uncovered $u \in U$ that is a neighbor of v.

2. Pete removes a finger j from a covered $v \in V$, and Dana answers by removing her finger j from its covered neighbor $u \in U$.

Notice that the set of fingers placed on the graph corresponds naturally to a partial matching in G : each v covered by a finger j of Pete is matched to the u that is covered by Dana's finger j in reply. Pete wins the game when he places a finger on some vertex such that all its neighbors are already covered by Dana. If Pete cannot win the game, then Dana wins. We define $MSpace(G)$, the *matching space* to be the minimal number of fingers that Pete needs in order to win the game. Clearly $MSpace(G) \leq |U| + 1$. We now prove the main claim of this section.

Theorem 3.9 *$Space(\mathcal{C}) \geq MSpace(\mathbb{G}(\mathcal{C}))$.*

Proof: Recall that the left side of $\mathbb{G}(\mathcal{C})$ is the set of clauses, and the right side is the set of variables. For $m = \{(C_{i_1}, x_{i_1}), \ldots (C_{i_k}, x_{i_k})\}$ a partial matching in $\mathbb{G}(\mathcal{C})$ of size k, define $\rho(m)$ to be the restriction of size k that sets the variable x_{i_j} to the value that satisfies the clause C_{i_j}, for $j = 1 \ldots k$, and leaves all other variables unassigned.

Assume Dana has a winning strategy when the matching game is played on $\mathbb{G}(\mathcal{C})$ using k fingers. We will use this strategy to show that every set of clauses derivable in clause space k is satisfiable. Let $\mathcal{C}_0, \ldots \mathcal{C}_\ell$ be a derivation from $\mathcal{C}$, of space k. We construct inductively a sequence of partial matchings in $\mathbb{G}(\mathcal{C})$, $m_0, \ldots m_\ell \quad m_t \subset E \quad t = 0 \ldots \ell$, that maintains the following properties for all $t = 0 \ldots \ell$:

1. m_t is the matching obtained by playing the matching game with k fingers for $t' \leq t$ rounds.
2. $|m_t| \leq |\mathcal{C}_t|$.
3. $\mathcal{C}_t|_{\rho(m_t)} = 1$

m_0 is the empty matching. For the induction step, we prove the claim according to the type of rule used at time t:

1. **Axiom Download:** If the new axiom C is satisfied by the restriction, we do nothing, and clearly all properties are maintained. Otherwise, the number of fingers Pete has at time $t-1$ is at most $|C_{t-1}| \leq k-1$, and thus, when Pete places a finger on C in $\mathbb{G}(\mathcal{C})$, Dana can respond by placing a finger on some uncovered x appearing in C. We set $m_t = m_{t-1} \cup \{(C, x)\}$. All properties are maintained: the new matching corresponds to playing one more round of the matching game. The memory size and the matching size are both incremented by one. $\mathcal{C}_t|_{\rho(m_t)} = 1$, because $\mathcal{C}_{t-1}|_{\rho(m_{t-1})} = 1$, and $C|_{\rho(\{(C,x)\})} = 1$.
2. **Inference:** Set $m_t = m_{t-1}$. By the soundness of resolution, we know that $\mathcal{C}_{t-1} \models \mathcal{C}_t$ and hence $\mathcal{C}_t|_{\rho(m_t)} = \mathcal{C}_t|_{\rho(m_{t-1})} = 1$. Since $|\mathcal{C}_t| > |\mathcal{C}_{t-1}|$ it also follows that $|m_t| \leq |\mathcal{C}_t|$.
3. **Memory Erasure:** $\mathcal{C}_t = \mathcal{C}_{t-1} - C'$ for some clauses C'. Any assignment satisfying $\mathcal{C}_{t-1}$ also satisfies $\mathcal{C}_t$. We set $m_t \subseteq m_{t-1}$ to be some minimal size submatching such that $\mathcal{C}_t|_{\rho(m_t)} = 1$. Since $\rho(m)$ is in 1-1 correspondence with m, for any m, we have that $|m_t| \leq |\mathcal{C}_t|$ using the locality lemma for Resolution from [Alekhnovich *et al.* A ('00)] (the proof of which follows immediately).

Lemma 3.10 (Locality lemma) *[Alekhnovich et al. A ('00)] Let ρ be a restriction and $\mathcal{C}$ be a set of clauses, such that $(\bigwedge_{C\in\mathcal{C}} C|_\rho) \equiv 1$. Then there exists a sub-restriction ρ' of ρ, such that $(\bigwedge_{C\in\mathcal{C}} C|_{\rho'}) \equiv 1$, and $|\rho'| \le |\mathcal{C}|$.*

Thus we get that the clause space required to prove $\mathcal{C}$ is at least $MSpace(G(\mathcal{C}))$ and the theorem is proved. $\square$

Proof [Locality Lemma 3.10]: For each clause $C \in \mathcal{C}$, there exists some literal $\ell \in C$ such that $\ell|_\rho = 1$. For each C, fix ℓ_C to be one such literal (arbitrarily). Set

$$\rho'(x) = \begin{cases} \rho(x) & \exists C \in \mathcal{C}, \ell_C \text{ is a literal over } x \\ \star & \text{otherwise} \end{cases}$$

Notice that if ℓ_C and ℓ_D are literals over the same variable x, then $\ell_C = \ell_D$, because $\ell_C|_\rho = \ell_D|_\rho = 1$, so ρ' is well defined. By definition, $\mathcal{C}|_{\rho'} = 1$, and $|\rho'| \le |\mathcal{C}|$. $\square$

3.6 Lower Bound on the Matching Game

We shall now prove lower bounds on the Matching Space when G is a good expander. In the proof we extensively use Hall's theorem 2.15. We call V' *minimal unmatchable* into U' if V' is unmatchable into U', and any proper subset of V' is matchable into U'. By Hall's theorem, this occurs iff $|N(V')| < |V'|$ but for all $V'' \subset V'$ $|N(V'')| \ge |V''|$.

Theorem 3.11 (Matching Space Lower Bound) *If a graph $G =< (V \cup U), E >$ is an (r,c)-bipartite expander, then*

$$MSpace(G) > \frac{c \cdot r}{2+c}$$

Proof: Suppose the game is played until at time T, Pete wins. Thus, at time $0 \le t < T$ the set of covered vertices corresponds to a partial matching in G. For any $0 \le t < T$, let $E_t \subseteq E$ be the matching at time t, let $s_t = |E_t|$ be the matching space at time t, and let V_t (resp. U_t) be the set of uncovered vertices in V (resp. U). We define a strategy for Dana.

Strategy 3.12 *Answer trying to maintain the property:*

$$\forall V' \subseteq V_t, |V'| \le r - s_t,\ V' \text{ can be matched into } U_t. \tag{3.1}$$

At $t = 0$ $s_t = 0$, and the property holds, by the definition of expansion and Hall's theorem. Let t be the first time property (3.1) does not hold. We claim that $s_t \ge \frac{cr}{2+c}$. The proof of this claim is divided into two cases, according to the step taken at time t.

1. **Pete removes a finger from v:** Let u be the vertex matched to v at time $t-1$ (i.e. $(v,u) \in E_{t-1}$). $V_t = V_{t-1} \cup \{v\}$, $U_t = U_{t-1} \cup \{u\}$ and $s_t = s_{t-1} - 1$. There exists some $V' \subset V_t$ of size at most $r - s_t$, that is minimal unmatchable into U_t.

   Claim 3.13 $|V'| = r - s_t$.

Proof: By definition, $|V'| \leq r - s_t$. Assume, for the sake of contradiction, that $|V'| < r - s_t$. By the minimality of t, every set of size $(r - s_t) - 1 = r - s_{t-1}$ which *does not* include v, is matchable into U_t because it is even matchable into $U_{t-1} \subset U_t$. A set of size $r - s_{t-1}$ which *does* includes v can be matched into U_t in the following way: match v to u, and use the matching of the remaining set (of size $r - s_{t-1} - 1$) into U_{t-1}, which must exist, by the minimality of t. We conclude that V' is matchable into U_t, contradiction. Thus $|V'| \geq r - s_t$. $\square$

Let us calculate the size of the set of neighbors of V' in the original graph G. On the one hand, $|V'| \leq r$, and hence by the definition of expansion:

$$|N(V')| \geq (1 + c)|V'| \tag{3.2}$$

On the other hand, V' is *minimal* unmatchable into U_t, and hence by Hall's matching theorem $|V'| > |N(V') \cap U_t|$. The only other possible neighbors of V' in G are in the matching E_t, which has size s_t. Thus we get

$$|V'| + s_t > |N(V')| \tag{3.3}$$

Combining the two inequalities and setting $|V'| = r - s_t$ we get:

$$\begin{aligned} |V'| + s_t &> (1 + c)|V'| \quad \Rightarrow & (3.4)\\ s_t &> c \cdot |V'| \qquad\quad \Rightarrow & (3.5)\\ s_t &> \frac{c \cdot r}{1 + c} > \frac{c \cdot r}{2 + c} & (3.6) \end{aligned}$$

Case 1 is proven.

2. **Pete places a finger on v:** Let $u_1, \ldots u_d$ be the neighbors of v in U_{t-1} (for some $d > 0$). For any choice u_i that Dana makes, there is some $V^i \subset V_t$, $|V^i| \leq r - s_t$ that is minimally unmatchable into $U_{t-1} \setminus \{u_i\}$.

   Claim 3.14 *There is no matching of $\tilde{V} = \cup_{i=1}^{d} V_i \cup \{v\}$ into U_{t-1}.*

   Proof: Assume for the sake of contradiction that $\tilde{V}$ is matchable into U_{t-1}, and fix such a matching. Let $U^i \subset U_{t-1}$ be the image of V^i under this matching. Recalling that V^i is minimal unmatchable into $U_{t-1} \setminus \{u_i\}$, we conclude that $u_i \in U^i$. This is true for any $i = 1 \ldots d$, and hence all neighbors of v are already taken by the matching on $\cup_{i=1}^{d} V^i$. Thus v cannot be matched, contradiction. $\square$

   By the claim, and the minimality of t, $|\tilde{V}| > r - s_{t-1}$, and since $s_t = s_{t-1} + 1$, we get $| \cup_{i=1}^{d} V^i| > r - s_t$.

   Claim 3.15 *For $i = 1 \ldots d$, $|N(V^i) \cap U_{t-1}| = |V^i|$.*

   Proof: $|V^i| \leq r - s_t < r - s_{t-1}$, so V^i is matchable into U_{t-1}, and hence $|N(V^i) \cap U_{t-1}| \geq |V^i|$. V^i is minimal unmatchable into $U_{t-1} \setminus \{u_i\}$, so $|N(V^i) \cap U_{t-1} \setminus \{u_i\}| < |V^i|$, and hence $|N(V^i) \cap U_{t-1}| \leq |V^i|$ $\square$

   For all i, $|V_i| \leq r - s_t$, whereas, $| \cup_{i=1}^{d} V^i| > r - s_t$. There must exist some subset $I \subseteq [d]$ such that $\frac{r - s_t}{2} < | \cup_{i \in I} V^i| \leq r - s_t$. Fix such an I, and denote $V' = \cup_{i \in I} V^i$.

Claim 3.16 *Suppose $V^1, \dots V^d$ are sets such that for all $i \in [d]$*

(a) $|N(V^i)| = |V^i|$.

(b) V^i *is matchable into* U.

Then $|N(\cup_{i\in[d]} V^i)| \leq |\cup_{i\in[d]} V^i|$.

Proof: By induction on d. For $d = 1$ the claim is simply condition a. Let $V^d_{NEW} = V^d \setminus (\cup_{i=1}^{d-1} V^i)$ be the set of "new" vertices added by V^d, and $V^d_{OLD} = V^d \setminus V^d_{NEW}$. If $V^d_{NEW} = \emptyset$ there is nothing to prove. Otherwise, assume for the sake of contradiction that $|N(V^d_{NEW}) \setminus N(\cup_{i=1}^{d-1} V^i)| > |V^d_{NEW}|$, i.e. V^d adds more neighbors than its size, when it is added to $\cup_{i=1}^{d-1} V^i$. By property 2 we know that $|N(V^d_{OLD})| \geq |V^d_{OLD}|$. Thus

$$
\begin{aligned}
|N(V^d)| &= |N(V^d_{OLD})| + |N(V^d_{NEW}) \setminus N(V^d_{OLD})| \\
&\geq |N(V^d_{OLD})| + |N(V^d_{NEW}) \setminus N(\cup_{i=1}^{d-1} V^i)| \\
&> |V^d_{OLD}| + |V^d_{NEW}| \\
&> |V^d|
\end{aligned}
$$

The second inequality follows from the fact that $V^d_{OLD} \subseteq \cup_{i=1}^{d-1} V^i$. We have reached a contradiction to property (a), and the claim is proven. □

Let us calculate once again the size of the set of neighbors of V' in the initial graph G. On the one side $s_{t-1} + |N(V') \cap U_{t-1}| \geq |N(V')|$. Applying claims 3.15, 3.16 to $\{V^i\}_{i\in I}$ gives us

$$s_{t-1} + |V'| \geq |N(V')|.$$

On the other hand, the expansion property and the fact $|V'| \leq r$ give us

$$|N(V')| \geq (1 + c)|V'|.$$

Combining the two together, we get:

$$
\begin{aligned}
|V'| + s_{t-1} &\geq (1 + c)|V'| \quad &\Rightarrow \quad &(s_t > s_t - 1) \\
s_t &> c|V'| \quad &\Rightarrow \quad &(|V'| \geq \frac{r - s_t}{2}) \\
s_t &> \frac{c \cdot r}{2 + c}
\end{aligned}
$$

Case 2 is proven, and with it the theorem. □

3.7 Space Lower Bounds for $G - PHP$

An optimal space lower bound for proof of the pigeonhole principle in Resolution was proved by [Esteban and Toran ('99)], and even extended to the *polynomial calculus* by [Alekhnovich *et al.* A ('00)]. The *graph pigeonhole principle*, $G - PHP$, introduced in definition 2.57, is a generalization for which size lower bounds still apply (theorem 2.59). In this section we reduce the space complexity of proving the $G - PHP$ to the matching game on G, in a generalization of theorem 3.9. This immediately gives optimal space lower bounds for the $G - PHP$, whenever G is a good expander.

Recall the definition of the $G - PHP$ (definition 2.57), and denote by $\mathcal{H}$ the conjunction of $H^u_{v,v'}$, for all $e = (v, u), e' = (v', u)$, $v, v' \in V$, $v \neq v'$, $u \in U$. For $\mathcal{C}$ a set of clauses over $Vars(G - PHP)$, and ρ a partial restriction that does not falsify $\mathcal{H}$, we say $\mathcal{C}|_\rho \equiv 1 (\text{mod } \mathcal{H})$ if $\mathcal{H} \models \mathcal{C}|_\rho$, i.e. any assignment that satisfies $\mathcal{H}$, satisfies $\mathcal{C}|_\rho$ as well.

Lemma 3.17 *A restriction falsifies $\mathcal{H}$ iff there exist a pair of edges $e_1 = (v, u)$, $e_2 = (v', u)$ such that $v \neq v'$ and $e_1, e_2 \in \{e : x_e|_\rho = 1\}$.*

Theorem 3.18 *For any graph G, $CS(G - PHP) \geq Space(G)$.*

Proof: For $m = \{(v_{i_1}, u_{i_1}), \dots (v_{i_k}, u_{i_k})\}$ a partial matching in G of size k, we define its corresponding restriction to set 1's to all edges in m, 0's to all edges (v, u) such that u is in matched in m to some $v' \neq v$, and leave all other variables unassigned. Formally:

$$x_{(v,u)}|_{\rho(m)} = \begin{cases} 1 & (v,u) \in m \\ 0 & (v,u) \notin m \text{ And } \exists v' \in V \quad (v', u) \in m \\ \star & \text{otherwise} \end{cases}$$

As in the proof of theorem 3.9, suppose Dana can win when the matching game over G is played with k pebbles. Let $\mathcal{C}_0, \dots \mathcal{C}_\ell$ be a derivation from $G - PHP$ (in the form of sequence of configurations) of space k. We construct inductively a sequence of matchings $m_0, \dots m_\ell$ that maintains the following properties for all $t = 0 \dots \ell$:

1. $|m_t| \leq |\mathcal{C}_t|$.
2. $\mathcal{C}_t|_{\rho(m_t)} \equiv 1 \ (\text{mod } \mathcal{H})$.

Notice that for a matching m, $\rho(m)$ does not falsify $\mathcal{H}$, and can be easily extended to an assignment that satisfies $\mathcal{H}$ (by setting all unassigned variables to 0). Thus, condition 2 implies that $\mathcal{C}_t$ is satisfiable for all $t = 0 \dots \ell$.

m_0 is the empty matching. For the induction step, we prove the claim according to the type of step taken:

1. **Axiom Download:** Suppose we download the axiom C. If $C \in \mathcal{H}$, we set $m_t = m_{t-1}$, and it is easy to see that both conditions are maintained. If $C = P_v$, Pete places a blue pebble on v in the graph G, and Dana places a red pebble on some $u \in N(v)$ (this is possible because $|m_{t-1}| < k$ and Dana wins when the game is played with k pebbles). We set $m_t = m_{t-1} \cup \{(v, u)\}$, and once again, it is easy to verify that both conditions are maintained.

2. **Inference:** By the soundness of resolution, $\mathcal{C}_{t-1} \models \mathcal{C}$, and hence $\mathcal{C}_t|_{\rho(m_{t-1})} \equiv 1$ (mod $\mathcal{H}$). Additionally, $|\mathcal{C}_t| = |\mathcal{C}_{t-1}| + 1$, so setting $m_t = m_{t-1}$ maintains both conditions.

3. **Memory Erasure:** Follows from the following locality claim, which is similar to the locality lemma of [Alekhnovich *et al.* A ('00)]

Lemma 3.19 *Let $\mathcal{C}$ be a set of clauses over $Vars(G-PHP)$, satisfiable* $\pmod{\mathcal{H}}$. *For all matching m in G such that $\mathcal{C}|_{\rho(m)} \equiv 1 \pmod{\mathcal{H}}$ and for all clauses $C \in \mathcal{C}$, there exists an edge $e_C \in m$, such that*

- $C|_{\rho(\{e_C\})} \equiv 1 \pmod{\mathcal{H}}$ *and*
- *for all $C' \in \mathcal{C}$, $C' \neq C$ $C'|_{\rho(\{e_{C'}\})} \not\equiv 0 \pmod{\mathcal{H}}$.*

Proof: Fix any $C \in \mathcal{C}$. Let m be a matching in G such that $\mathcal{C}|_{\rho(m)} \equiv 1 \pmod{\mathcal{H}}$. We have that in particular $C|_{\rho(m)} \equiv 1 \pmod{\mathcal{H}}$. Now look at C. If $\rho(m)$ makes C true fixing some positive literal x_e, then we fix $e_C = e$. Obviously $C|_\rho(\{e_C\}) \equiv 1 \pmod{\mathcal{H}}$ and $\rho(\{e_C\})$ cannot falsify any other clauses in $\mathcal{C}$ since $e \in m$.

Otherwise $\rho(m)$ makes true C satisfying some negated literals $\bar{x}_e$. Assume $e = (v, u)$, then by definition of $\rho(m)$, there is a $v' \in V$ such that $e' = (v'u) \in m$. Fix $e_C = e'$. As before this edge is good to prove our claim. □

Thus if $\mathcal{C}_t \subset \mathcal{C}_{t-1}$ we pick from m_{t-1} according to the previous lemma one edge per clause in $\mathcal{C}_t$, and get m_t that satisfies both properties. □

3.8 Open Questions

1. The *variable space* of a CNF formula $\mathcal{C}$ is $VSp(\mathcal{C}) \stackrel{\text{def}}{=} \sum_{C\in\mathcal{C}} |C|$, the variable space of a proof is the maximal variable space of a configuration in the proof, and the variable space of proving a formula is the minimal variable space of a proof. For any $\mathcal{F}$ over n variables, $VSpace(\mathcal{F}) \leq n^2$, because $CSpace(\mathcal{F}) \leq n$. [Alekhnovich *et al.* A ('00)] proved $\Omega(n^2)$ lower bounds for a certain formula with initial width n. Can one find a 3 CNF for which $VSpace(\mathcal{F}) = \Omega(n^2)$? Is this true for a random 3-CNF with Δn clauses (constant Δ) ? We believe the answer is positive.

2. What is the clause space complexity of proving a random CNF formula in the Polynomial Calculus ? We suspect that one should get essentially the same lower bounds as for resolution.

3. For many hard tautologies we get linear lower bounds on the *width* and on the *clause space*. This is true for Tseitin graph formulas, the Pigeonhole principle, and random formulas. Notice that width is also a space measure : it is the maximal space of a single clause in the proof. What is the relationship between the two measures ? At least in one aspect width is "harder" than space. A width lower bound yields a size lower bound for treelike *and* general resolution, whereas a space lower bound yields a size lower bound only for treelike resolution. For this reason we conjecture that $CSpace(\mathcal{C}) \geq width(\mathcal{C})$. Is this true ? can one find a counter example ?

4. The following question was raised by Ron Lavi. One may view the graph matching game as an online problem. Let G be a fixed bipartite graph, with $|V| > |U|$. One receives "matching requests" online, and wishes to keep the set matched. The strategy we presented for Dana requires her to compute on each

request the matching properties for an exponential number of subsets of V, and doing this in the trivial manner is inefficient. Can one find a polynomial time algorithm that would operate as well as Dana's strategy ?

Chapter 4

Separating Treelike and General Resolution

4.1 Introduction

This chapter is a small detour from our main theme, of expansion and proof complexity. We investigate the connection between minimal *treelike* and *general resolution* size complexities, answering the following question.

Question: What is the largest possible gap between the minimal number of lines (=clauses) in a proof, and the minimal number of lines in a *treelike* proof ?

We shall give the best known separation of the two size measures, and additionally show that the separation we present is nearly optimal. The separation will be proved by showing a somewhat surprising connection between *treelike proof size* and the *pebbling measure* of a DAG.

The question we answer in this chapter is interesting in its own right, but also has implications on the relative efficiency of various automated theorem proving algorithms. Recall the automated theorem proving problem (section 1.6.3): Given an unsatisfiable CNF $\mathcal{C}$, find a proof of $\mathcal{C}$ efficiently, in terms of time and space. Compare this couple of natural methods for finding such a proof.

Recursion: pick a variable x, and recursively try to prove $\mathcal{C}|_{x=0}$ and $\mathcal{C}|_{x=1}$. Such methods, introduced by [Davis *et al.* ('62)], are called *DLL-Procedures.*

Dynamic Programming: Start from the axioms $\mathcal{C}$, arbitrarily derive new clauses $\mathcal{C}'$, using the resolution rule, and set $\mathcal{C} \leftarrow \mathcal{C} \cup \mathcal{C}'$. Repeat this procedure until the empty clause is derived. If one picks a variable x and performs all possible resolutions on it, the method, introduced by [Davis and Putnam ('60)], is called a *DP-procedure.*

Let us briefly describe the pros and cons of these methods, and their natural connection to resolution. The recursive procedure has recursion depth that is bounded by the number of variables, thus it is extremely *space-efficient* - the space required is at most linear in the input size. It's main flaw is that it corresponds to a *decision-tree* for the problem of finding a clause falsified by an assignment to the variables. It turns out that such a decision-tree is a tree-like resolution proof of $\mathcal{C}$

(see lemma 4.4). The recursive method will be *time-costly* on inputs that do not have short tree-like resolution proofs.

The dynamic programming method, on the other hand, produces general resolution proofs. Thus, if $\mathcal{C}$ has a short proof, one might hope to find a short proof by using such a method. Indeed, we shall see that for certain contradictions, there exist dynamic programming proof search methods that are extremely efficient (polynomial time). The main disadvantage of the method is the large space it might use. If we wish to save all possible resolution consequences we might require space that is exponential in n. As we shall promptly see, a large gap between general and tree-like proof size will have implications on the relative efficiency of these two proof search methods.

4.2 History and Main Results

For a given CNF $\mathcal{C}$, we are interested in measuring the gap between $\mathbf{S_R}(\mathcal{C})$ (minimal resolution proof size) and $\mathbf{S_T}(\mathcal{C})$ (minimal treelike resolution proof size). A super-polynomial gap follows easily from [Buss and Pitassi ('97)], but improving on this requires some additional insight. A better separation was given by [Bonet *et al.* ('98)], who used a contradiction based on the Pyramid graph, to obtain a $\exp(\sqrt{n})$ gap, building on the construction and analysis of [Raz and McKenzie ('98)]. The analysis given by [Bonet *et al.* ('98)] is specifically tailored to the Pyramid graph and the specific contradiction based on it. It was not clear what property of the contradiction, and the underlying graph, give rise to this gap. Hence, it was also not clear how to improve the gap, or generalize the result to contradictions based on different graphs.

In this chapter we solve these difficulties. Motivated by [Raz and McKenzie ('98), Bonet *et al.* ('98)], we give a surprising, yet simple, reduction of the gap between treelike and general resolution, to the *pebbling* measure of the underlying graph. Our simpler characterization allows us to base our contradictions on graphs with a higher pebbling measure than the Pyramid graph (for which the pebbling measure is $\sqrt{n}$), and thus obtain a nearly optimal separation of treelike and general resolution, improving on the results of [Bonet *et al.* ('98)]. We deliver the following couple of messages. The first is that the gap between general and treelike resolution can be very large:

Theorem 4.1 (Lower Bound) *There exists an infinite family of explicitly constructible contradictions $\{\mathcal{C}_n\}$ such that $|\mathcal{C}_n| = O(n)$, $\mathbf{S_R}(\mathcal{C}_n) = O(n)$, $W(\mathcal{C}) = O(1)$ and $\mathbf{S_T}(\mathcal{C}) = \exp(\Omega(\frac{n}{\log n}))$.*

It is easy to see that $\mathbf{S_T} \leq 2^{\mathbf{S_R}}$. Our second message is that there is a non-trivial upper bound on the gap, and it is almost tight:

Theorem 4.2 (Upper Bound) *For any CNF formula $\mathcal{C}$:*

$$\mathbf{S_T}(\mathcal{C}) = \exp(O(\frac{\mathbf{S_R}(\mathcal{C}) \log\log \mathbf{S_R}(\mathcal{C})}{\log \mathbf{S_R}(\mathcal{C})}))$$

Theorem 4.1 is proved by giving a construction which associates with every fan-in 2 DAG G on n edges, a contradiction $\mathcal{C}(G)$ with the following properties.

1. $\mathcal{C}(G)$ has $O(n)$ variables and $O(n)$ clauses.

2. $\mathcal{C}(G)$ has general resolution proofs of length $O(n)$.

3. $\mathcal{C}(G)$ has resolution proofs of width $O(1)$ (i.e. all clauses in the proof have a constant number of variables).

4. Every tree-like resolution proof of $\mathcal{C}(G)$ has length at least $\exp(P(G))$, where $P(G)$ is the classical pebbling number of the circuit G.

The pebbling measure of a circuit G is the minimal memory size needed to carry out the computation described in the circuit G, on an input x, (both G and x are given as inputs), assuming each gate output costs one memory unit. The surprising connection of treelike resolution size to the pebbling measure is a novel one, and its proof is the main technical contribution of this chapter. We show that the pebbling measure of the graph G is an exponential lower bound on the tree-like size of proving $\mathcal{C}(G)$. The connection of pebbling to tree-like size allows us to use graphs that have a high pebbling measure. Specifically, [Celoni *et al.* ('77)] explicitly construct for every n a graph G_n of size $O(n)$ satisfying $P(G_n) = \Omega(n/\log n)$. This, combined with (1) and the upper bound (2), gives a truly exponential separation between general and tree-like resolution systems. Finally, (3) means that for these contradictions, the natural restricted-width dynamic programming algorithm, that searches for a minimal width proof (section 2.4), runs in polynomial time, and is exponentially faster than any recursive method.

Theorem 4.2 uses techniques from [Celoni *et al.* ('77)], who showed that if a Boolean function has a circuit of size S then it has a circuit of depth $S/\log S$. We use similar techniques to show that one can "cut" any general proof roughly in half, by removing some of the middle lines. Then we construct a small tree-like proof, depending on the relative size of the "middle part" that was removed.

This chapter is organized as follows. After presenting the essential definitions in section 4.3, we start by proving the lower bound (section 4.4), followed by the upper bound (section 4.5). The chapter ends with a discussion of the applications of these bounds to automated theorem proving (section 4.6).

4.3 Definitions

4.3.1 Decision Trees

Let $\mathcal{C}$ be a CNF formula over n variables and m clauses. A *search problem* for $\mathcal{C}$ is the following: given an assignment $\alpha \in \{0,1\}^n$, find a clause $C_i, 1 \leq i \leq m$ such that $C_i(\alpha) = 0$, if there is such a clause, and otherwise (i.e. $\mathcal{C}(\alpha) = 1$) answer 1.

Definition 4.3 (Decision Trees for CNF Search Problems) *A* decision tree *is a binary tree, with internal vertices labeled by variables, edges labeled by* 0 *or* 1*, and leafs labeled with the possible outputs. Every assignment to the variables defines a path through the tree in the natural way, and the label at the end of the path is said to be the output of the decision tree on that assignment.*

We say that D is a decision tree for the Search problem for $\mathcal{C}$ if it correctly solves it on every input.

for $\mathcal{C}$ a CNF formula, Let $\mathbf{S_D}(\mathcal{C})$ denote the minimal size of a decision tree solving the CNF Search Problem for $\mathcal{C}$.

Decision trees for CNF search problems are closely related to tree-like resolution, as the following lemma shows.

Lemma 4.4 *For $\mathcal{C}$ an unsatisfiable CNF, $\mathbf{S_T}(\mathcal{C}) = \mathbf{S_D}(\mathcal{C})$*

Proof: The DAG of the resolution proof is a decision tree, where each internal vertex is labeled by the variable resolved upon at that step. Hence $\mathbf{S_T}(\mathcal{C}) \leq \mathbf{S_D}(\mathcal{C})$.

For the opposite direction, we claim that given a decision tree, we can derive from it a tree like proof without increasing it's size. Notice that if $\mathcal{C}$ is unsatisfiable, every leaf is labeled by a clause, since 1 is not a legitimate answer. We look at two leaves labeled C_i, C_j, with their father v labeled x. If x does not appear in one of the two clauses (w.l.o.g. C_i), then we may label v with C_i, erase it's sons, and make the tree smaller. Otherwise, w.l.o.g. x must appear in C_i in positive form and in C_j in negative form. In this case we may label v with the consequence of resolving C_i, C_j on x. Continuing in this way up through the decision tree we conclude that $\mathbf{S_T}(\mathcal{C}) \geq \mathbf{S_D}(\mathcal{C})$. $\square$

4.3.2 Pebbling

The pebbling measure of a circuit is, intuitively, the space needed for simulating the computation of the circuit on a Turing machine. For a thorough introduction to results regarding the pebbling game, see the excellent survey [Pippenger]. In this section we briefly state the essential definitions and facts that will be used later on in our discussion.

Definition 4.5 (Pebbling) *Let G a circuit, and let S and T be subsets of the vertices. The pebble game on G with* sources *S and* targets *T is the following 1-player game. At any point in the game, some vertices of G will have pebbles on them (1 pebble per vertex), while the remaining vertices will not. A* configuration *is a subset of vertices, comprising just those vertices that have pebbles on them. The rules of the pebble game are as follows.*

1. *At any time, a pebble may be placed on any vertex in S.*
2. *If all immediate predecessors of a vertex have a pebble on them, a pebble may be placed on that vertex.*
3. *A pebble may be removed from any vertex.*
4. *If a pebble is placed on a vertex in T, the player wins and the game ends.*

A legal *pebbling of T on G from S is a sequence of configurations, the first being the empty set, the last containing some element of T, and in which each configuration follows from the previous configuration by one of the rules. The number of pebbles used in such a pebbling is the maximum number of pebbles in any configuration. The pebbling price of T on G from S, denoted $P_G(S,T)$, is the minimal number of pebbles needed in any legal pebbling of T from S on G.*

Notice that $P_G(S,T) = 1$ iff S and T intersect, and that $P_G(S,T) = P_G(S,\{t\})$ for some $t \in T$. Also, without loss of generality, we will assume that every node with no predecessors is in S, since the rules allow us to pebble such nodes at any time. We will only be considering DAGs G where every node either has in-degree 0 or 2.

4.3.3 A game for proving lower bounds on tree-like resolution

Lower bounds for size of tree-like resolution proofs can be given in terms of a 2-player game; this description is due to [Pudlák and Impagliazzo ('00)]. Any small tree-like proof will give a good strategy for one of the players, so a good strategy for the other player yields a corresponding lower bound on proofs.

Let $\mathcal{C}$ be an inconsistent set of clauses. Consider the following game between two players, the *prover*, and the *delayer*. In each round, the prover chooses a variable to be assigned a value. Then the delayer chooses one of 0,1, or $*$. If 0 or 1 is chosen, no points are scored, the variable is set to the chosen bit, and the next round begins. If $*$ is chosen, then the delayer scores one point, but the prover then can choose the value for the variable. The game ends when one of the clauses in $\mathcal{C}$ is forced to false by the assigned values, i.e., when all the literals in the clause are assigned false.

Lemma 4.1 *If $\mathcal{C}$ has a tree-like resolution proof of size S, then the prover has a strategy where the delayer can win at most $\lceil \log S \rceil$ points.*

Proof: Fix a proof of size S. The prover will maintain the following invariant after each round: If the delayer has scored t points, then the partial assignment will falsify a clause in the proof, and the sub-tree rooted at this clause will be of size at most $S/2^t$. In particular, this will show the claimed bound on the number of points the delayer can score.

Let C be the clause from the invariant at the previous round. If C is a leaf, then the game halts, since a clause in $\mathcal{C}$ has been falsified. Otherwise, the prover picks the variable x resolved on to derive C from C_0 and C_1 in the proof. If the delayer assigns the variable a value, at least one of the two clauses C_0, C_1 is falsified, and this is the new clause for the invariant. If the delayer chooses $*$, and scores a point, then the prover chooses the value to falsify the clause with the smaller sub-tree. This is at most half the size of the sub-tree for the current clause, so the invariant is maintained. □

Corollary 4.2 *If the delayer has a strategy which always scores r points, then $\mathbf{S_T}(\mathcal{C}) \geq 2^r$.*

4.4 Lower Bound

In this section we prove the lower bound of theorem 4.1, by explicitly presenting a family of contradictions that achieves the lower bound. The family of contradictions is a generalization of [Raz and McKenzie ('98)] and [Bonet *et al.* ('98)]. For these contradictions we expose a connection between pebbling and tree-like resolution. We start by defining these contradictions.

Definition 4.6 (Pebbling Contradictions) *Let $G =< V, E >$ be a circuit, and S and T subsets of vertices. Associate a pair of Boolean variables $x(v)_0, x(v)_1$ with every vertex $v \in V$. $Peb_{G,S,T}$, the* pebbling contradiction *of G is the conjunction of:*

Source Axioms: $x(s)_0 \vee x(s)_1$ *for each* $s \in S$.

Target Axioms: $\bar{x}(t)_0$ *and* $\bar{x}(t)_1$ *for each* $t \in T$

Pebbling Axioms: $(x(u_1)_a \wedge x(u_2)_b) \to (x(v)_0 \vee x(v)_1)$ *for u_1, u_2 the two predecessors of v, and all $a, b \in \{0, 1\}$.*

Notice that $Peb_{G,S,T}$ is a non-satisfiable 4-CNF over $2|V|$ variables, with $O(|V|)$ clauses. It is also easy to see that $Peb_{G,S,T}$ has a short, constant width resolution proof:

Lemma 4.7 *For G a circuit, $\mathbf{S_R}(Peb_{G,S,T}) = O(|V|)$ and $W(Peb_{G,S,T}) \leq 6$.*

Proof: Fix a topological sort of G. In order of this sort, we derive $x(v)_0 \vee x(v)_1$ for each $v \in V$. If v has no predecessors, $v \in S$, so this is an axiom. If v has two predecessors, u_1 and u_2, we have inductively derived $x(u_1)_0 \vee x(u_1)_1$, and $x(u_2)_0 \vee x(u_2)_1$. Together with the four Pebbling Axioms for v, these imply the clause $x(v)_0 \vee x(v)_1$. By the implicational completeness of resolution (theorem 2.2), there is a derivation of $x(v)_0 \vee x(v)_1$ from the above mentioned clauses. This derivation has constant width and size, since it involves at most 6 variables. Hence, starting from the Source Axioms, one can infer $x(t)_0 \vee x(t)_1$, for a target $t \in T$, in constant width and size $O(|V|)$. Then, resolving with the Target Axioms, one derives 0. $\square$

The following theorem is the main result presented in this section, and it immediately implies Theorem 4.1.

Theorem 4.8 $\mathbf{S_T}(Peb_{G,S,T}) = 2^{\Omega(P_G(S,T))}$.

Proof [Theorem 4.1]: [Celoni *et al.* ('77)] present an infinite family of explicitly constructible circuits $\{G_n\}$, with $|G_n| = n$ and $P_G(S,T) = \Omega(n/\log n)$. Taking the Pebbling contradictions of these graphs, Theorem 4.7 and Lemma 4.8 imply Theorem 4.1. $\square$

The rest of this section is devoted to the proof of theorem 4.8. We use Corollary 4.2; we give a strategy for the delayer that achieves at least $P_G(S,T) - 3$ points on $Peb_{G,S,T}$.

The strategy is as follows: Set $S' = S, T' = T$. On each round, the prover proposes a variable $x(v)_i$.

The delayer's response is as follows:

Case 1 If $v \in T'$, assign the variable 0.

Case 2 If $v \in S'$ assign the variable 1.

Case 3 If $v \notin S' \cup T'$, and if $P_G(S', T' \cup \{v\}) = P_G(S', T')$, assign the variable 0 and add v to T'

Case 4 If $v \notin S' \cup T'$, and if $P_G(S', T' \cup \{v\}) < P_G(S', T')$, respond $*$ (letting the prover assign a value) and add v to S'.

We prove that the strategy above earns the delayer at least $P_G(S, T) - 3$ points by showing that $P_G(S', T')$ only decreases by the number of points earned, and is at most 3 at the end of the game.

Lemma 4.3 *When the game terminates, $P_G(S', T') \leq 3$.*

Proof: Note that, if any variable $x(v)_i$ is set to 1, it happened either in Case 2 or Case 4, and in either case $v \in S'$. Similarly, if both $x(v)_0$ and $x(v)_1$ are assigned 0, the first to be assigned was either via Case 1 or Case 3, and so $v \in T'$. (If the first to be assigned fell under Case 4, v would be put in S', and the second variable would have been assigned 1 via Case 2.)

So a source axiom will never be violated for $s \in S \subseteq S'$, because both variables will not be set to 0, and similarly, a target axiom will never be violated for $t \in T \subseteq T'$. So at the end of the game, a a pebbling axiom is violated, say for node v with predecessors u and u'. To be violated, both $x(v)_0$ and $x(v)_1$ must be set to 0, so $v \in T'$. Also, at least one of $x(u)_0$ and $x(u)_1$ must be set to 1, so $u \in S'$. Similarly, $u' \in S'$. So to pebble T' from S' using three pebbles, simply place a pebble on both u and u', and then on t. □

Lemma 4.4 *For any $v \in V$ and any sets S, T, $P_G(S, T) \leq max\{P_G(S, T \cup \{v\}), P_G(S \cup \{v\}, T) + 1\}$*

Proof: One way to pebble T from S is to first pebble $T \cup \{v\}$ from S, using $P_G(S, T \cup \{v\})$ pebbles. If the result is a pebble in T, stop, otherwise the final configuration has a pebble on v. Keeping that pebble on v, remove the other pebbles, then simulate a pebbling of T from $S \cup \{v\}$. This second stage uses a total of $P_G(S \cup \{v\}, T) + 1$ pebbles. □

Lemma 4.5 *After any round, if the delayer has scored p points, $P_G(S', T') \geq P_G(S, T) - p$.*

Proof: At the beginning of the game, both sides are $P_G(S, T)$. Note that the only case when $P_G(S', T')$ changes is in Case 4. In this case, $P_G(S', T' \cup \{v\}) < P_G(S', T')$ at the beginning of the round. By Lemma 4.4, this means $P_G(S' \cup \{v\}, T') \geq P_G(S', T') - 1$. Since in Case 4, the delayer scores a point, and v is added to S', this preserves the invariant. □

Corollary 4.6 *Using the strategy described, the delayer scores at least $P_G(S, T) - 3$ points.*

The previous corollary and corollary 4.2 imply Theorem 4.8. □

4.5 Upper Bound

In this section we prove that the gap presented in the previous section cannot be "too large". Specifically, we prove theorem 4.2. This shows that the lower bound stated in theorem 4.1 is nearly optimal.

The theorem will be proved by explicitly constructing a small size tree like proof of $\mathcal{C}$ given a small size general proof. We shall limit our attention only to "hard" contradictions, for which the gap between tree-like and general size is maximal, and bound from above the proof size for these inputs.

In order to prove theorem 4.2, we shall work with a different size measure, hereby defined. This measure, inspired by [Hopcroft *et al.* ('77), Celoni *et al.* ('77)], counts the number of internal edges in the graph of a minimal proof, and is closely connected to the standard size measure.

Definition 4.9 (Internal Size) *An* internal vertex *of a DAG is any vertex other than a source or sink. An* internal edge *in a DAG connects two internal vertices. For G a DAG, $e(G)$ is the number of internal edges in G. For $\mathcal{C}$ a CNF contradiction, define the internal size of proving $\mathcal{C}$ to be:*

$$e(\mathcal{C}) \stackrel{\text{def}}{=} \min\{e(G_\pi) : \pi \textit{ is a proof of } \mathcal{C}\}$$

We define $f(e)$ to be the maximal tree-like size of proving $\mathcal{C}$, given that $\mathcal{C}$ has a proof with internal size e. Formally:

$$f(e) \stackrel{\text{def}}{=} \max\{\mathbf{S_T}(\mathcal{C}) : e(\mathcal{C}) \leq e\}$$

$\mathcal{C}$ is called e-maximal *if $e(\mathcal{C}) = e$, $\mathbf{S_T}(\mathcal{C}) = f(e)$, and removing any clause from $\mathcal{C}$ enlarges e or makes $\mathcal{C}$ satisfiable.*

We start by proving two lemmas that will be used in the proof of the main theorem.

Lemma 4.10 *$f(e+1) \leq 2f(e)$.*

Proof [Lemma 4.10]: Let $\mathcal{C}$ be an $e+1$-maximal contradiction, and let π be a proof of $\mathcal{C}$ having $e+1$ internal edges. Let C be a clause of π that is a consequence of resolving two axioms $A, B \in \mathcal{C}$, and is involved in at least one internal edge. (The proof has at least one internal edge. Repeatedly backing from the current node to any internal predecessor, starting at the source of this edge, we eventually get to a minimally internal vertex, which must be labeled by such a C.) Let $\mathcal{C}' = \mathcal{C} \cup \{C\}$. $\mathcal{C}'$ has a tree like proof T' of size at most $f(e)$, since deleting the derivation of C from π gives a proof of $\mathcal{C}'$ with $\leq e$ internal edges. Adding the derivation $\frac{A \quad B}{C}$ to T' whenever C appears as an axiom in T', will be a proof of $\mathcal{C}$ which is at most double the size of T'. $\square$

Let $\binom{n}{\leq m} \stackrel{\text{def}}{=} \sum_{i=0}^{m} \binom{n}{i}$.

Lemma 4.11 *Given a CNF formula with m clauses and n variables, there is a decision tree solving the CNF search problem, with size at most $\binom{n}{\leq m}$.*

Proof [Lemma 4.11]: Consider the following strategy for the CNF search problem. In every step, we have a partial assignment ρ , we query a variable, and add its value to ρ. If at the beginning of a step, ρ has forced any clause to be false, we stop and output that clause. If ρ has forced all clauses to be true, we stop and output "true". Otherwise, we choose the first clause whose value is not forced, and query the first variable in that clause whose value is not assigned.

In each step of the above strategy, there is one value for the query that satisfies the current clause. Along any path of the decision tree, the falsifying value can be chosen at most m times before all clauses are forced to true, and the decision tree terminates. Thus, a path in the tree can be described by a sequence of n bits with at most m 1's, where 1 means, assign the satisfying value, and 0, assign the other value. Therefore, there are at most $\binom{n}{\leq m}$ such paths. $\square$

Next, we wish to show that any G_π can be split "in half", into two derivations, each having roughly half the number of internal edges appearing in the original G_π.

Definition 4.12 (Topological Partition) *Let $v_1 \ldots v_S$ be a topological ordering of the vertices of a circuit G. For $0 \leq i \leq S$, let $V_0(i) = \{v_1, \ldots, v_i\}$, and let $V_1(i) = \{v_{i+1}, \ldots v_S\}$. Let $e_0(i)$ ($e_1(i)$ resp.) be the number of internal edges in the graph induced by $V_0(i)$ ($V_1(i)$ resp.). Let M_i be the set of* internal *vertices of $V_0(i)$ that connected to vertices in $V_1(i)$, and define $m_i = |M_i|$.*

Notice that $\pi = \{C_1, \ldots, C_S\}$ is a topological ordering of G_π. If $V_0(i), V_1(i)$ is a topological partition of G_π, then $V_0(i)$ is a resolution derivation from $\mathcal{C}$, and $V_1(i)$ is a resolution proof of $\mathcal{C} \cup M_i$. The following lemma is taken from [Celoni *et al.* ('77)].

Lemma 4.13 (Existence of an Equal Topological Partition) *For $v_1 \ldots v_S$ a topological ordering of the vertices of a single-target circuit G, there exists a partition of G such that $|e_0(i) - e_1(i)| \leq 2$. Such a partition is called* equal*, and for an equal partition, $e_0(i) \leq \frac{e-m}{2} + 1$.*

Proof: $e_0(0) = 0$, $e_0(S) = e$ and $e_0(i+1) \leq e_0(i) + 2$, since adding the vertex v_{i+1} to $V_0(i)$ increases the number of internal edges in $V_0(i)$ by at most two. Hence, there exists some i such that $|e_0(i) - e_1(i)| \leq 2$.

G has a single target, so every vertex in M_i is connected to at least one vertex in $V_1(i)$. Hence $e(G) \geq e_0(i) + e_1(i) + m_i$, and $e_0(i) \leq \frac{e-m_i}{2} + 1$. $\square$

We are ready to prove the main theorem of this section:

Theorem 4.14 *There exists a constant $c > 0$, such that for all integers $k \geq 4$, $f(k2^k) \leq 2^{c \cdot 2^k \cdot \log k}$*

Proof: By induction on k. Let $c \geq 8$ be large enough that the claim is true for all $k < 4$, and assume the claim holds for all values smaller than some fixed $k \geq 4$. Let $e = k2^k$. We assume $\mathcal{C}$ is a CNF in n variables which has a proof with at most e internal edges, and show how to construct a tree-like proof of size $2^{c \cdot 2^k \cdot \log k}$.

First, if $n > e$, there is some variable x never resolved on in the proof. Deleting x and any clause containing x from the proof makes it only smaller, and keeps it a proof. So without loss of generality, we assume $n \leq e$.

Let $\pi = \{C_1, C_2, \dots C_S\}$ be a proof of $\mathcal{C}$, such that G_π has e internal edges. Let π_0, π_1 be the equal partitioning of π at some $1 \le i \le S$, denote $e_0 = e_0(i), e_1 = e_1(i)$ and $m = |M_i|$.

Given the partition, we present two alternative methods for constructing tree-like proofs of $\mathcal{C}$. We choose our method according to the size of m:

$m \ge 2^k$ **- The Brute Force method** Each clause $C \in M_i$ was derived by a derivation with at most $e_0 \frac{e-m}{2} + 1 \le (k-1)2^{k-1} + 1$ internal edges, hence it has a tree like derivation of size at most $f((k-1)2^{k-1}+1) \le 2f((k-1)2^{k-1})$ (by lemma 4.10). We replace π_1 with a tree-like derivation, that, similarly, has size at most $2f((k-1)2^{k-1})$, and then replace each axiom $C \in M_i$ by the tree-like derivation of size at most $2f((k-1)2^{k-1})$. The total size of this derivation is at most $4f^2((k-1)2^{k-1})$. Using induction we get:

$$4f^2((k-1)2^{k-1}) \le 2^{2+c2^k \log(k-1)} \le 2^{2+c2^k \log k - \Omega(c2^k/k)} \le 2^{c2^k \log k}$$

, for suitably large c and $k \ge 4$. (One can check that $c \ge 8$ suffices.)

$m < 2^k$ **- The Intelligent method** By lemma 4.11, there is a decision tree D of size at most $\binom{n}{\le m}$ solving the CNF search problem over M_i (M_i may be satisfiable). Look at a leaf v of D. If v is labeled $C \in M_i$, it can be derived from $\mathcal{C}$ by a tree like derivation of size at most $f(\frac{e-m}{2})$, and we plug into v this "small" tree-like derivation. Otherwise v must be labeled 1. This means that the restriction ρ defined by the path leading to v, satisfies M_i, and hence $\pi_1|_\rho$ is a proof of $\mathcal{C}|_\rho$ with at most $\frac{e-m}{2}$ internal edges. Hence, there is a tree like proof of $\mathcal{C}_\rho$ of size at most $f(\frac{e-m}{2})$. This can be transformed into a derivation from $\mathcal{C}$ of C_ρ, the clause which contains all literals set to 0 by ρ, without increasing its size. Plugging the tree-like derivation of C_ρ into v, for every v labeled 1 in D, transforms D into a tree-like proof of $\mathcal{C}$. We need only show that the new tree-like proof is not too large. Its size is bounded by:

$$\binom{n}{\le m} \cdot f(\frac{k2^k - m}{2} + 1) \tag{4.1}$$

$$\le \binom{n}{\le m} \cdot 2f(\frac{k2^k - m}{2}) \tag{4.2}$$

(by lemma 4.10)

$$\le \binom{k2^k + 1}{\le m} \cdot 2f(\frac{k2^k - m}{2}) \tag{4.3}$$

(since $n \le e$)

$$\le 2^{k+1} \binom{k2^k}{m} \cdot f(\frac{k2^k - 2^k}{2} + \frac{2^k - m}{2}) \tag{4.4}$$

(since $k \ge 4$ and $m < 2^k$)

$$\le 2^{k+1} \binom{k2^k}{m} \cdot 2^{\frac{2^k - m}{2}} f((k-1)2^{k-1}) \tag{4.5}$$

(using lemma 4.10 again)

$$\le 2^{k+1} \binom{k2^k}{2^k} f((k-1)2^{k-1}) \tag{4.6}$$

(because $\binom{r}{m}2^{\frac{-m}{2}}$ is monotonically increasing in m for $m \leq r/4$, and $m < 2^k \leq 1/4k2^k$, we can substitute $m = 2^k$ to upper bound the quantity)

$$\leq 2^{k+1}(4k)^{2^k} \cdot 2^{c \cdot 2^{k-1} \cdot \log k - 1} \tag{4.7}$$

$$\leq 2^{k+1+2\cdot 2^k + 2^k \cdot (\log k) + (1/2)c(2^k \log k)} \tag{4.8}$$

$$\leq 2^{2^k \cdot (\log k)(2+(1/2)c+2/(\log k)+(k+1)/2^k)} \leq 2^{c2^k \cdot \log k} \tag{4.9}$$

since $k \geq 4$ and $c \geq 8$.

□

We are now set to prove theorem 4.2.

Proof [Theorem 4.2]: Let $\mathcal{C}$ be a contradiction, and π be it's minimal size proof, with $|\pi| = \mathbf{S_R}(\mathcal{C}) = S$. $e(G_\pi) \leq 2 \cdot S$, because at most two internal edges enter a vertex of G_π. Set $k \stackrel{\text{def}}{=} \lceil \log \frac{2e}{\log e} \rceil$. Clearly $e \leq k2^k$, and recall that $f(e)$ is monotonically increasing, hence:

$$\mathbf{S_T}(\mathcal{C}) \leq f(e) \leq f(k2^k) = 2^{O(\frac{S \log\log S}{\log S})}$$

□

4.6 Applications to Automated Theorem Proving

Some of the most extensively used methods for proving unsatisfiability of CNF formulas, are called Davis-Putnam procedures after [Davis and Putnam ('60)]. Actually, these procedures are derived from a system devised by [Davis *et al.* ('62)], and hence we will refer to them as *DLL Procedures*. A DLL procedure relies on choosing a variable x, and trying to prove $\mathcal{C}|_{x=T}$ and $\mathcal{C}|_{x=F}$ recursively.

If $\mathcal{C}$ is unsatisfiable, $DLL(\mathcal{C})$ terminates providing a tree-like resolution proof of $\mathcal{C}$. The DLL procedure is actually a family of algorithms, diversified by the different strategies for picking the next variable to be resolved upon.

Recall our restricted width algorithm, described in section 2.4, and our trivial observation that this algorithm will operate well on inputs with small width proofs. A simple consequence of this observation, is that the pebbling contradictions provide concrete examples for which the restricted width algorithm exponentially outperforms any DLL-procedure. Let $\mathbf{TIME}(A(x))$ denote the running time of the algorithm A on input x.

Theorem 4.15 *Let A be any implementation of a DLL procedure. There exists an infinite family of unsatisfiable CNF formulas $\mathcal{C}$ such that $\mathbf{TIME}(A(\mathcal{C}))$ is exponentially larger than $\mathbf{TIME}(RW(\mathcal{C}))$.*

If $T_A = \mathbf{TIME}(A(\mathcal{C}))$, and $T_{RW} = \mathbf{TIME}(RW(\mathcal{C}))$, then:

$$T_A = \exp(\Omega(T_{RW}/\log T_{RW})).$$

Proof: We use the notation of section 4.4. Take $\mathcal{C} = Peb_{G,S,T}$ where G is a circuit with high pebbling measure $P_G(S,T) = |V|/\log|V|$. By lemma 4.7

$$\mathbf{TIME}(RW(\mathcal{C})) = |V|^{O(1)}$$

By theorem 4.1, any tree-like proof of $\mathcal{C}$, must have size at least

$$2^{\Omega(\frac{|V|}{\log|V|})}$$

because the run time of any DLL procedure is bounded from below by the size of its output, which is a tree-like proof. The theorem follows. $\square$

4.7 Open Questions

The result of this chapter is

$$\exp(\frac{\mathbf{S_R}}{\log\mathbf{S_R}}) \leq \mathbf{S_T} \leq \exp(\frac{\mathbf{S_R}\log\log\mathbf{S_R}}{\log\mathbf{S_R}})$$

Can this gap be closed ?

Chapter 5

Polynomial Calculus Degree Lower Bounds

After investigating various aspects of resolution, we turn our attention to stronger proof systems. The first one, considered in this chapter, is the *polynomial calculus*, and in the next chapter we shall encounter *bounded depth Frege*.

The *polynomial calculus* is a proof system using algebraic reasoning, most notably properties of polynomials over a fixed field. This system is strictly stronger than resolution, as any resolution proof can be carried out in the polynomial calculus, with minor modifications. In particular, the size and space measures in the polynomial calculus are at most those of resolution.

For this stronger system, we can still exhibit our main theme, and show that expansion of the input formula implies large proof complexity. We start with a formal definition of this new proof system, followed by the discussion of previous and current results. The proof of our main claim occupies the rest of the chapter.

5.1 The Polynomial Calculus

Let $\mathcal{P} = \{P_i(\vec{x})\}_{i\in[m]}$ be a set of polynomials over a field $\mathbb{F}$. Hilbert's Nullstellensatz theorem states that $\mathcal{P}$ does not have a common root in the algebraic closure of $\mathbb{F}$ iff the polynomial 1 belongs to the ideal generated by $\mathcal{P}$. This theorem provides a basis for a proof system, called the *polynomial calculus* (PC), which will be the focus of this chapter. The polynomial calculus is a proof system for the **co-NP** complete language of sets of polynomials that have no $\{0,1\}$ solutions. [1] Let $\mathbb{F}[x_1, x_2, \ldots]$ denote the ring of multivariate polynomials over $\mathbb{F}$. The ideal generated by $\mathcal{P}$, denoted $I(\mathcal{P})$, is the closure of $\mathcal{P}$ under linear combinations and multiplication by polynomials. Thus, the two derivation rules of the polynomial calculus are:

Addition: Derive $\alpha P_1 + \beta P_2$ from $\{P_1, P_2\}$, for any $P_1, P_2 \in \mathbb{F}[x_1, x_2, \ldots]$, $\alpha, \beta \in \mathbb{F}$.

Multiplication: Derive $x_i \cdot P$ from P for any $P \in \mathbb{F}[x_1, x_2, \ldots]$ and any variable x_i.

[1] One can generalize this proof system to the language of sets of polynomials that have no common roots over a field $\mathbb{F}$. This language is not exactly **co-NP** complete, because it is not clear how to succinctly describe a solution to an arbitrary set of polynomials over an arbitrary field.

There exist representations of sets of Boolean constraints $\mathcal{F}$ as sets of polynomials $\mathcal{P}(\mathcal{F})$ such that $\mathcal{F}$ is satisfiable iff $\mathcal{P}(\mathcal{F})$ has a common root (see definition 5.3). An algebraic derivation starting from $\mathcal{P}(\mathcal{F})$ and ending with the polynomial 1, using the two derivation rules defined above, convinces us that $1 \in I(\mathcal{P}(\mathcal{F}))$, and hence by Hilbert's Nullstellensatz, that "$\mathcal{F} \in \mathbf{co-SAT}$". Thus we define a *PC-derivation* of a polynomial Q from a set of polynomials $\mathcal{P}$ to be a sequence of polynomials $\pi = \{P_1 \dots P_S\}$ such that $P_S = Q$ and every $P_i,\ i \in [S]$ is either some axiom $P \in \mathcal{P}$, or is derived from previous polynomials by one of the two derivation rules. A *PC proof of* $\mathcal{P}$ is a derivation of 1 from $\mathcal{P}$. The fundamental theorem of the polynomial calculus is just a rephrasing of Hilbert's Nullstellensatz.

Theorem 5.1 (Completeness of the Polynomial Calculus) *A set of polynomials $\mathcal{P} \subset \mathbb{F}[x_1, x_2, \dots]$ has no common root in the algebraic closure of $\mathbb{F}$ iff there is a polynomial calculus proof of $\mathcal{P}$.*

The main complexity measure of a PC derivation is its *degree*, defined to be the maximal degree of a polynomial in the proof. For P a polynomial, let $d(P)$ be the degree of P. The proof degree of $\mathcal{P}$, denoted $D(\mathcal{P}) = D_{\mathbb{F}}(\mathcal{P})$, is defined to be the minimal degree of a proof of $\mathcal{P}$, if such a proof exists, and ∞ otherwise. Notice the resemblance between proof degree and resolution width. Namely, if we represent the clause C in the natural way as the polynomial $P(C)$ (see definition 5.3), then $|C| = d(P(C))$. Indeed, the width measure is a direct adaptation of the PC-degree measure, and the main size-width theorems of resolution (theorems 2.4, 2.5) are a direct translation of the following basic results of Clegg, Edmonds and Impagliazzo, to the framework of resolution.

Theorem 5.2 *[Clegg, et al. ('96)] For $\mathcal{C}$ a k-CNF over n variables, $\mathcal{P}(\mathcal{C})$ its corresponding set of polynomials,*

$$D(\mathcal{P}(\mathcal{C})) \leq k + \log \mathbf{S_T}(\mathcal{C})$$

$$D(\mathcal{P}(\mathcal{C})) = O(k + \sqrt{n \log \mathbf{S_R}(\mathcal{C})})$$

When discussing resolution, the main complexity measure we investigated was the proof *size*. When discussing PC proof size we must first decide how the polynomials in the proof will be encoded syntactically. If we allow polynomials to be represented as *algebraic circuits*, Pitassi showed that this system can polynomially simulate extended Frege ([Pitassi ('96)], theorem 5). Currently we do not know of any lower bounds for extended Frege, and hence we do not have any non-trivial lower bounds for proof size using this representation. If we demand polynomials to be represented as an explicit *sum of monomials*, then the clause $\vee_{i=1}^{n} x_i$ is represented by the polynomial $\prod_{i=1}^{n}(1 - x_i)$ which has 2^n monomials. In this case the polynomial calculus becomes even weaker than resolution. A reasonable compromise raised by [Alekhnovich *et al.* B ('00)], which we adopt in this chapter, uses a distinct variable for every *literal*, (i.e. x and $\bar{x}$ are two different variables ranging over $\mathbb{F}$), and we force x to be equal to $1 - \bar{x}$. This is done by factoring $\mathbb{F}[x_1, x_2, \dots]$ by

$$\{x_i + \bar{x}_i - 1\}_{i \in \mathbb{N}} \tag{5.1}$$

Additionally, we factor $\mathbb{F}[x_1, x_2, \ldots]$ by

$$\{x_i^2 - x_i\}_{i \in \mathbb{N}} \tag{5.2}$$

This second factoring creates the ring of multi-linear polynomials, and forces Boolean solutions to $\mathcal{P}$. We denote by $\bar{\mathbb{F}}[x_1, x_2, \ldots]$ the *Boolean ring over* $\mathbb{F}$, which is simply $\mathbb{F}[x_1, x_2, \ldots]$ factored by (5.1), (5.2). Next we give a natural representation of clauses as polynomials. This representation can be easily generalized to arbitrary Boolean circuits.

Definition 5.3 *For C a clause over (Boolean) variables $\mathbf{x_1} \ldots \mathbf{x_n}$, its representing polynomial is defined by*

$$P(C) \stackrel{\text{def}}{=} \begin{cases} 1 & C = \mathbf{0} \\ \bar{x}_i & C = \neg \mathbf{x}_i \\ x_i & C = \mathbf{x}_i \\ \prod_{i=1}^n P(\ell_i) & C = \vee_{i=1}^n \ell_i \end{cases}$$

where $x_i, \bar{x}_i$ are distinct variables ranging over $\mathbb{F}$. For $\mathcal{C}$ a CNF, $\mathcal{P}(\mathcal{C}) \stackrel{\text{def}}{=} \{P(C) : C \in \mathcal{C}\}$.

When working in the Boolean ring, the polynomial calculus is as least as good as resolution, with respect to proof *size* and *width*.

Theorem 5.4 *If a CNF $\mathcal{C}$ has a resolution proof of size S and width w, then $\mathcal{P}(\mathcal{C})$ has a PC proof of size $O(S)$ and degree $w + O(1)$ over the Boolean ring $\bar{\mathbb{F}}[x_1, x_2, \ldots]$.*

Impagliazzo, Pudlák and Sgall, based on the work of Clegg, Edmonds and Impagliazzo, showed that in the Boolean ring the polynomial calculus proofs displays the same size-degree relation that resolution has for size and width. Once again, this size degree relation predates the analogous relation for resolution.

Theorem 5.5 *[Impagliazzo et al. ('97)] For $\mathcal{P}$ a set of polynomials of degree at most d in n variables,*

$$D(\mathcal{P}) = O(d + \sqrt{n \log M(\mathcal{P})}).$$

We end this section by quoting the completeness of the polynomial calculus with respect to CNF formulas.

Theorem 5.6 *A CNF $\mathcal{C}$ is unsatisfiable iff there is a polynomial calculus proof of $\mathcal{P}(\mathcal{C})$ in the Boolean ring over $\mathbb{F}$.*

5.2 History and Main Results

The first algebraic-flavor proof system, called the *Nullstellensatz system*, was introduced and analyzed by [Beame *et al.* ('94)]. The polynomial calculus is a simplification and strengthening of this system, and was introduced by [Clegg, *et al.* ('96)], who presented it as a candidate for an automatizable proof system, and also proved an exponential separation between it and its weaker predecessor - the Nullstellensatz system. The first degree lower bounds for the polynomial calculus itself were presented by [Razborov ('96)], for the pigeonhole principle. This lower bound was simplified and improved by [Impagliazzo *et al.* ('97)], but both lower bounds are square root of the number of variables (and match the upper bound for this contradiction). The first *linear* lower bounds on the polynomial calculus degree, were given by [Buss *et al.* ('99)], for Tseitin contradictions based on expander graphs. This latter lower bound used the expansion of the input contradiction, as well as the fact that all constraints are linear equations modulo 2. The following main result of this chapter, is a generalization and simplification of the lower bounds of [Buss *et al.* ('99)], reducing *PC degree* to *resolution width.* This novel reduction (with one extra idea), gives as a bonus lower bounds for random k-CNFs over all fields with characteristic $\neq 2$.

Theorem 5.7 *For $\mathcal{L}$ a set of* linear functions mod 2, *$\mathcal{C}(\mathcal{L})$ its explicit encoding as a CNF, and $\mathcal{P}(\mathcal{C}(\mathcal{L}))$ the representation of this CNF as a set of polynomials over a field $\mathbb{F}$ with $char(\mathbb{F}) \neq 2$,*

$$D(\mathcal{P}(\mathcal{C}(\mathcal{L}))) \geq \frac{1}{4} W(\mathcal{C}(\mathcal{L})) + O(1)$$

Recall our basic method for proving resolution lower bounds: if $\mathcal{F}$ is an *expanding* set of *sensitive* functions, then proving $\mathcal{F}$ requires large *width*, and hence large *size.* For the stronger polynomial calculus, the above result is slightly weaker, and says that if $\mathcal{L}$ is an expanding set of the *parity* functions (which are 1-sensitive), then proving its corresponding set of polynomials in a field with characteristic other than 2 requires large degree, because the resolution proof width is large. Notice that if $char(\mathbb{F}) = 2$ then there is a very low degree proof of the system - we simply add the lines of $\mathcal{L}$ and obtain $1 = 0$ by Gaussian elimination.

As immediate corollaries of the main theorem we get lower bounds for Tseitin contradictions and for generator contradictions whenever the base functions of the generator are parity functions. Another corollary, which requires some additional work, is a lower bound for random k-CNFs. The added idea is that a clause is semantically implied by a parity function (e.g $x \vee y \vee z$ is implied by $x \oplus y \oplus z = 1$). Hence a low degree proof of an expanding CNF can be converted into a low degree proof of an expanding set of linear equations, and so our lower bound applies to expanding CNFs as well. The main drawbacks of our method are that it works only for parity functions and completely fails for fields of characteristic 2. Both these problems have been recently solved by Alekhnovich and Razborov who generalized our results to all fields, and all ℓ-immune polynomials [Alekhnovich and Razborov ('00)]. A polynomial is called ℓ-immune if any non-zero corollary of it has degree $\geq \ell$. Their result states that if $\mathcal{P}$ is an *expanding* set of *ℓ-immune* polynomials, then proving $\mathcal{P}$ requires large *degree* over *any field.* This specifically gives lower bounds for random CNFs over all fields.

Let q be a prime. The simplest way to prove that $\mathcal{L}$, a set of m linear equations mod q, is unsatisfiable is by Gaussian elimination. We start with the lines of $\mathcal{L}$ as axioms, and produce new linear equations by addition of existing lines and scalar multiplication. We prove that $\mathcal{L}$ is unsatisfiable by deriving the unsatisfiable linear equation $1 = 0$. The number of lines needed to prove $\mathcal{L}$ is at most m^2. Let us call such a proof the *Gaussian calculus* proof, or simply, a *Gaussian proof.* Let the *Gaussian width* of proving $\mathcal{L}$, denoted $W_G(\mathcal{L})$, be the maximal number of variables appearing in a line of the proof. This simple proof system was suggested by M. Alekhnovich, during the joint work with Razborov and Wigderson on generator contradictions [Alekhnovich *et al.* A ('00)], and he also proved the following basic claim about it. Recall the definition of an explicit CNF (definition 2.27).

Lemma 5.8 (Alekhnovich 98) *For $\mathcal{L}$ a set of linear equations mod 2, and $\mathcal{C}(\mathcal{L})$ its explicit encoding as a CNF,*

$$W(\mathcal{C}(\mathcal{L})) \leq 2 \cdot W_G(\mathcal{L})$$

With this lemma in hand, our main theorem follows immediately from the following theorem, which we prove in the next section.

Theorem 5.9 *For $\mathcal{L}$ a minimal contradictory set of linear equations mod 2 of width at most k, and $\mathcal{P}$ its formulation in a field $\mathbb{F}$ with characteristic $p \neq 2$,*

$$D(\mathcal{P}) = \max\{k, \frac{1}{2}W_G(\mathcal{L}) + \Theta(1)\}$$

The rest of this chapter is as follows. We start by proving theorem 5.9, in the next section. This is followed by degree lower bounds for Tseitin and generator contradictions (section 5.4), and we end with degree lower bounds for random k-CNFs (section 5.5).

5.3 Gaussian Width and Proof Degree

Let $\mathcal{L} = \{\ell_i\}_{i=1}^m$ be a set of linear equations mod q (a prime) over n variables $x_1, \ldots x_n$. For simplicity let us assume that $\mathcal{L}$ is *minimal* unsatisfiable, i.e. every proper subset of it is satisfiable. For reasons that will become clear later on, we allow variables to appear on both sides of the equality sign, so formally ℓ_i has the following structure:

$$\sum_{S_{i_1}} a_j \cdot x_j = c_i + \sum_{S_{i_2}} b_j \cdot x_j \tag{5.3}$$

where $a_j, b_j, c_i \in \{0, 1 \ldots q-1\}$.

As mentioned above, the most natural way to prove that $\mathcal{L}$ is unsatisfiable, is by Gaussian elimination in the field F_q. Suppose we wish to show that $\mathcal{L}$ is unsatisfiable, but we work within a field $\mathbb{F}$ with characteristic $p \neq q$, which has a primitive q'th root of unity called ω. The multiplicative group generated by ω is isomorphic to the additive group Z_q, and we use this isomorphism to translate $\mathcal{L}$ into a set of polynomials in $\mathbb{F}$:

Definition 5.10 (Linear Equations mod $q \neq p$) *Let $\mathcal{L} = \{\ell_i\}_{i=1}^m$ be a minimal unsatisfiable set of linear equations mod q, where ℓ_i is as described in (5.3). Let $\mathbb{F}$ be a field of characteristic $p \neq q$, having a primitive unity root of order q, denoted ω. The polynomial translation of the equation ℓ_i in $\mathbb{F}$, denoted $P(\ell_i) = P_{\mathbb{F}}(\ell_i)$ is the following polynomial:*

$$\prod_{S_{i_1}} y_j^{a_j} = \omega^{c_i} \cdot \prod_{S_{i_2}} y_j^{b_j} \tag{5.4}$$

The translation of $\mathcal{L}$ in $\mathbb{F}$, denoted $\mathcal{P}_{\mathbb{F}}(\mathcal{L})$ is the union of the following sets of polynomials:

1. $\{y_i^q = 1\}_{i=1}^n$. *This set forces all variables to take on values in* $1, \omega, \omega^2, \ldots, \omega^{q-1}$.
2. $\{P_{\mathbb{F}}(\ell_i)\}_{i=1}^m$

Remarks:

1. $\mathcal{P}_{\mathbb{F}}(\mathcal{L})$ is a set of *binomials*, i.e. polynomials with two monomials. This simple observation will play a crucial role in our investigation.

2. Notice that the transformation described in the previous definition is invertible, i.e. every binomial P which has structure as in (5.4) where $a_j, b_j, c_i \in \{0, 1, \ldots q-1\}$, corresponds to a linear equation ℓ. Moreover, ℓ is satisfied by an assignment $\alpha = \{\alpha_i\}_{i=1}^n \in \{0, \ldots q-1\}^n$ iff $P_{\mathbb{F}}(\ell)$ is satisfied by the assignment $\{\omega^{\alpha_i}\}_{i=1}^n$.

3. Suppose $P_{\mathbb{F}}(\ell_1), P_{\mathbb{F}}(\ell_2)$ are polynomial translations of ℓ_1, ℓ_2, respectively. It is easy to see that $P_{\mathbb{F}}(\ell_1)$ is equivalent to $P_{\mathbb{F}}(\ell_2)$ modulo the ideal generated by $\{y_i^q = 1\}_{i=1}^n$ iff $\ell_1 \equiv \ell_2 \pmod{q}$.

Clearly, $\mathcal{P}_{\mathbb{F}}(\mathcal{L})$ is unsatisfiable in $\mathbb{F}$, but Gaussian elimination is not sufficient for proving this fact (i.e. reaching a contradiction such as $1 = 0$). In order to reach a contradiction, one must allow multiplication of lines by variables, working within the framework of the polynomial calculus.

The following lemma, stated here for $q = 2$, can be naturally generalized to systems of linear equations mod $q \neq 2$:

Lemma 5.11 *Let $P(\ell_1), P(\ell_2)$ be the translations of ℓ_1, ℓ_2, two equivalent linear equations mod 2, into the field $\mathbb{F}$, with characteristic other than 2. There is a PC derivation of $P_{\mathbb{F}}(\ell_2)$ from $P_{\mathbb{F}}(\ell_1) \cup \{y_i^2 = 1\}_{i=1}^n$ with degree at most*

$$1 + \max\{d(P_{\mathbb{F}}(\ell_1)), d(P_{\mathbb{F}}(\ell_2))\}$$

The next theorem can be proven either by means of the Groebner Basis algorithm described by [Clegg, *et al.* ('96)], or by use of the Laurent relation described in [Buss *et al.* ('99)]. A *binomial proof* is a proof in which all polynomials are binomials.

Theorem 5.12 *If $\mathcal{P}$ is a set of binomials, then there is a binomial proof of $\mathcal{P}$ of minimal degree. Moreover, if all coefficients of $\mathcal{P}$ are in $\{1, -1\}$, then there is binomial proof of $\mathcal{P}$ of minimal degree, in which all monomials have coefficients in $\{1, -1\}$.*

Proof: In [Clegg, *et al.* ('96)], it is shown that, if a system of equations has a degree d proof, then such a proof is generated by the Groebner basis algorithm (using any degree-respecting term order). In fact, the proof so generated has the following restriction: each time a line is generated as a linear combination of two previous lines, one of the co-efficients is 1 and the other is chosen to cancel a term. In any such step, if the linear combination is between binomials and cancels monomial m_1, then we can write the two binomials as $m_1 + (-1)^b m_2$ and $m_1 + (-1)^{b'} m_3$, and the result is $(-1)^b m_2 + (-1)^{b'} m_3$. Unless $m_2 = m_3$, this is also a binomial with co-efficients in $\{1, -1\}$. If $m_2 = m_3$, we can immediately divide the result by 2 to get a single monomial with co-efficient 1. Also, if we multiply a binomial by a variable, we obtain a binomial with the same co-efficients. Thus, inductively, the Groebner basis algorithm produces a proof all lines of which are binomials with co-efficients in $\{1, -1\}$. □

We are ready to prove theorem 5.9.

Theorem 5.13 *For $\mathcal{L}$ a minimal contradictory set of linear equations mod 2 of width at most k, and $\mathcal{P}$ its formulation in a field $\mathbb{F}$ with characteristic $p \neq 2$,*

$$D(\mathcal{P}) = \max\{k, \frac{1}{2} W_G(\mathcal{L}) + \Theta(1)\}$$

Proof: To see that $d(\mathcal{P}) \leq \lceil \frac{1}{2} W_G(\mathcal{L}) \rceil$, take any Gaussian proof of width w, and let $d = \lceil w/2 \rceil + 1$. Assume $k < d$. Inductively, for each line in the Gaussian proof, we will show that any equation $\Pi_{S_1} y_i = (-1)^b \Pi_{S_2} y_i$ with S_1, S_2 a disjoint partition of S with each of size at most $\lceil w/2 \rceil$, has a PC proof of degree at most d. Assume the Gaussian proof adds the two lines $\ell_1 := \sum_S x_i = b$ and $\ell_2 := \sum_T x_i = b'$ to obtain $\sum_{S \oplus T} x_i = b \oplus b'$. Let P_1, P_2 be corresponding polynomials, such that P_1 is equivalent to $P_{\mathbb{F}}(\ell_1)$ modulo $\{y_i^2\}_{i=1}^n$, and similarly for P_2. Define $R = S \cap T, r = |R|, s = |S - R|, t = |T - R|$. Note that $s + t = |S \oplus T| \leq w$.

Suppose $r \leq \lceil w/2 \rceil$. Assume wlog $s \leq t$. Let $T' \subseteq T - R$ be an arbitrary set of size $t' = \min\{t, \lceil w/2 \rceil\}$, and let $T'' = (T - R) - T'$ and $t'' = |T''|$. Notice that $t'' + s \leq \lceil w/2 \rceil$, because $s + t' + t'' \leq w$, and $s \leq t$.
Use P_2 to derive

$$P_3 := \prod_{R \cup T''} y_i = (-1)^{b'} \prod_{T'} y_i$$

Use P_1 to derive

$$P_4 := \prod_R y_i = (-1)^b \prod_S y_i$$

and multiply it by $\prod_{T''} y_i$ to derive

$$P_5 := \prod_{R \cup T''} y_i = (-1)^{b'} \prod_S y_i$$

Subtract P_3 from P_5 to derive

$$P_6 := \prod_{(S-R) \cup T''} y_i = (-1)^{b \oplus b'} \prod_{T'} y_i$$

It is easy to verify that the maximal degree of $\{P_1, \ldots P_6\}$ is at most d, and by use of lemma 5.11 this is enough.

Next assume that $r > \lceil w/2 \rceil$. Let $R' \subset R$ be an arbitrary set of size $|R'| = \lceil w/2 \rceil$, and set $R'' = R - R'$. Use P_1 to derive

$$P_3 := \prod_{R'} y_i = (-1)^b \prod_{S - R'} y_i$$

Use P_2 to derive

$$P_4 := \prod_{R'} y_i = (-1)^{b'} \prod_{T-R'} y_i$$

Subtract the two to get the binomial

$$P_5 := \prod_{S-R'} y_i = (-1)^{b \oplus b'} \prod_{T-R'} y_i$$

Notice that $R'' = (S - R') \cap (T - R')$ is exactly the index set of the variables appearing in both monomials of P_5, of which we wish to dispose. For $j \in R''$, using the axiom $y_j^2 = 1$, derive

$$\prod_{(S-R')-\{j\}} y_i = (-1)^{b \oplus b'} \prod_{(T-R')-\{j\}} y_i$$

This derivation will have degree at most d. Continue in this manner till all variables in R'' have been removed, arriving at the polynomial $\prod_{S-R} y_i = (-1)^{b \oplus b'} \prod_{T-R} y_i$. Once again, by lemma 5.11 the derivation has degree at most d. Thus, we have $d(\mathcal{P}) \leq 1/2 W_G(\mathcal{L}) + O(1)$.

In the other direction, assume $\mathcal{L}_p$ has a PC proof of degree d, hence, by theorem 5.12 it has a proof in which each line is a multi-linear binomial and all monomial coefficients are 1 or -1. We may construct inductively a corresponding Gaussian proof with Gaussian degree at most $2d$. To see this, notice that the PC proof has no scalar multiplications by constants other than -1. A multiplication of P by the variable y_j corresponds to changing $\sum_{S_1} x_i = b + \sum_{S_2} x_i$ into $x_j + \sum_{S_1} x_i = b + x_j + \sum_{S_2} x_i$, and for any addition of P_1 and P_2, one of the monomials in P_1, P_2 must be identical (otherwise the output would not be a binomial), and hence adding the corresponding linear equations would yield the proper result .

If we get a single monomial by adding two binomials in the PC proof, then both sides of the above equations involve the same sets S_1, S_2 of variables, but with different constants b. Then adding the corresponding linear equations will give us a non-zero constant equaling 0, and we have obtained a contradiction. Thus, $W_G(\mathcal{L}) \leq 2d(\mathcal{P})$. $\square$

5.4 Tseitin and Generator Contradictions

Our first simple corollaries will be degree lower bounds for Tseitin formulas and definition of functions that are *hard* for the polynomial calculus. We shall give lower bounds for both explicit and implicit encodings of these formulas. Recall the definition of a Tseitin contradiction, definition 2.38. Let $\mathcal{P}(\mathcal{C}[G, \sigma])$ be the formulation of the CNF $\mathcal{C}[G, \sigma]$ as a set of polynomials over a field $\mathbb{F}$. If $char(\mathbb{F}) = 2$ then there is a low degree proof of $\mathcal{P}(\mathcal{C}[G, \sigma])$: we derive $\sum_{v \in e} x_e = \sigma(v)(\text{mod} 2)$ from the set of polynomials $\mathcal{P}(\mathcal{C}(EQ(\bigoplus X_v, \sigma(v))))$, and then use Gaussian elimination to derive the polynomial 1. If on the other hand $char(\mathbb{F}) \neq 2$, then theorem 5.7 reduces the degree to the resolution width. Applying the width lower bound of theorem 2.40 gives the following lower bound, originally proved by [Buss *et al.* ('99)].

Theorem 5.14 *[Buss et al. ('99)] Let G be a graph on n vertices and expansion factor $e(G)$. For $\mathcal{P}(\mathcal{C}[G,\sigma])$ the formulation of an explicit Tseitin formula over a field $\mathbb{F}$ with $char(\mathbb{F}) \neq 2$:*

$$D(\mathcal{P}(\mathcal{C}[G,\sigma])) \geq \frac{e(G) \cdot n}{16}$$

Lemma 5.15 *For $\mathcal{F}(x_1,\ldots,x_n)$ a set of Boolean functions, of maximal support r, $\mathcal{C}(\mathcal{F})$ its explicit encoding, $\mathcal{I}(x_1,\ldots,x_n,y_1,\ldots,y_m) = \mathcal{I}(\mathcal{F})$ an implicit encoding of $\mathcal{F}$, if $\mathcal{P}(\mathcal{I}(\mathcal{F}))$ has a PC proof of degree d then $\mathcal{P}(\mathcal{C}(\mathcal{F}))$ has a PC proof of degree $\leq r \cdot d$.*

Proof: Let $\mathcal{I}(x_1,\ldots,x_r,y_1,\ldots,y_\ell)$ be an implicit encoding of a function $f(x_1,\ldots,x_r)$, where $y_1,\ldots,y_\ell$ are the auxiliary variables. Let $\beta : \{0,1\}^r \rightarrow \{0,1\}^\ell$ be a Boolean function such that

$$f(\alpha) = 1 \Rightarrow \mathcal{I}(\alpha,\beta(\alpha)) = 1$$

Let β_i, $i \in [\ell]$ be the projection of β to the i'th coordinate. Let $P_{\beta_i}(x_1,\ldots,x_r)$ be the (unique) multi-linear polynomial computing β_i. For π a PC proof of $\mathcal{I}(\mathcal{F})$ of degree d, substitute P_{β_i} for any appearance of the auxiliary variable y_i. Notice that the new sequence is a legitimate PC proof of degree $\leq r \cdot d$. For any clause $C \in \mathcal{I}(x_1,\ldots,x_n,y_1,\ldots,y_m)$ let C' be the function resulting by substituting the function β_i for any variable y_i appearing in C. Notice that $\mathcal{C}(f) \models C'$, and hence, by the implicational completeness of the polynomial calculus, there is a derivation of $P(C')$ from $\mathcal{P}(\mathcal{C}(f))$. Since all polynomials are multi-linear over r variables, the degree of this derivation is at most r. Hence, there exists a proof of $\mathcal{P}(\mathcal{C}(\mathcal{F}))$ of the claimed low degree. $\square$

Using the above lemma we can derive lower bounds also on implicit encodings of Tseitin contradictions.

Corollary 5.16 *Let G be a graph on n vertices, of maximal degree Δ and expansion factor $e(G)$. For $\mathcal{P}(\mathcal{I}[G,\sigma])$ the formulation of an explicit Tseitin formula over a field $\mathbb{F}$ with $char(\mathbb{F}) \neq 2$:*

$$D(\mathcal{P}(\mathcal{I}[G,\sigma])) \geq \frac{e(G) \cdot n}{16\Delta}$$

We can similarly show that parity based Boolean functions are *hard* for the polynomial calculus. Recall that a function $F : \{0,1\}^n \rightarrow \{0,1\}^m$ is called *hard* for a proof system if its parametrized formulation is hard (definition 2.42). We will say that F is *hard* for the polynomial calculus if the proof degree of the parametrized formulation of F is large.

Definition 5.17 (Parity based functions) *A multi-output Boolean function $F : \{0,1\}^n \rightarrow \{0,1\}^m$ is said to be* parity based *if for all $i \in [m]$, the projection of F on the i'th bit, is a linear equation mod 2.*

Recalling that parity is 1 sensitive, we apply theorem 2.44 and theorem 5.7 and obtain.

Theorem 5.18 *Let $F : \{0,1\}^n \to \{0,1\}^m$ be an (r,c)-boundary expanding parity based function, of maximal support d. Let $\mathcal{C}(F)$ be an explicit formulation of F, and let $\mathcal{I}(F)$ be an implicit parametrized formulation of F. For any field $\mathbb{F}$ with char$(\mathbb{F}) \neq 2$:*

$$D(\mathcal{P}(\mathcal{C}(F))) \geq \frac{r \cdot \epsilon}{8}$$

$$D(\mathcal{I}(\mathcal{C}(F))) \geq \frac{r \cdot \epsilon}{8d}$$

Applying theorem 2.46 we get.

Corollary 5.19 *For any parameters $m > n > 2$ there exist functions $F : \{0,1\}^n \to \{0,1\}^m$ such that for any explicit formulation $\mathcal{C}(F)$, and implicit formulation $\mathcal{I}(F)$, and any field of characteristic other than 2:*

- $D(\mathcal{P}(\mathcal{C}(F))) = \Omega(n)$.
- $D(\mathcal{P}(\mathcal{I}(F))) = \Omega(n/\log m)$.

5.5 Random k-CNF Lower Bounds

In this section we consider the relative efficiency of PC over Resolution, as a proof system for random k-CNF's. Recall that with high probability a random k-CNF with n variables and Δn clauses, demands exponential size resolution proofs, if $\Delta << \sqrt{n}$ (theorem 2.24). In the previous discussion, we have already noticed that linear equations with high expansion are hard for the polynomial calculus. We shall show a natural reduction from *random CNF's* (definition 2.22) to *random linear equations*, which have high expansion.

Definition 5.20 (Random linear equations) *Let $\mathcal{L} \sim \mathbb{L}_m^{k,n}$ denote that $\mathcal{L}$ is a random set of linear equations mod 2, each having k variables, chosen at random by picking m equation i.i.d from the set of all $2\binom{n}{k}$ equations, with repetitions.*

Given the distribution $\mathbb{L}_{\Delta \cdot n}^{k,n}$, one can easily produce the distribution $\mathbb{C}_{\Delta \cdot n}^{k,n}$, by replacing each $\ell \in \mathcal{L}$ with one of the 2^{k-1} clauses in the explicit encoding of ℓ, picked at random. Let us denote by $\mathcal{C}_{\mathcal{L}}$ any CNF formula derived from $\mathcal{L}$ by this process. Notice that $\mathcal{L} \models \mathcal{C}_{\mathcal{L}}$, but the converse is not necessarily true. In the other direction, for any clause C there is a unique linear equation mod 2 ℓ_C such that $\ell_C \models C$. For $\mathcal{C}$ a CNF formula, let $\mathcal{L}_{\mathcal{C}} = \{\ell_C : C \in \mathcal{C}\}$ be the (unique) *linear closure* of $\mathcal{C}$.

The following definition and lemma are from [Buss *et al.* ('99)].

Definition 5.21 *[Buss et al. ('99)] Let $\mathcal{P}(\overline{x})$, $\mathcal{P}(\overline{y})$ be two sets of polynomials over a field $\mathbb{F}$. Then $\mathcal{P}$ is (d_1, d_2)-reducible to $\mathcal{P}$ if: (1) For every y_i, there is a degree d_1 definition of y_i in terms of the x's. That is, for every i, there exists a degree d_1 polynomial r_i where y_i will be viewed as being defined by $r_i(x_1, \ldots, x_n)$; (2) there exists a degree d_2 PC derivation of the polynomials $\mathcal{P}(\overline{r}(x_1, \ldots, x_n))$ from the polynomials $\mathcal{P}(\overline{x})$.*

Lemma 5.22 *[Buss et al. ('99)] Suppose that $\mathcal{P}(x)$ is (d_1, d_2)-reducible to $\mathcal{P}(y)$. Then if there is a degree d_3 PC proof of $\mathcal{P}(y)$, then there is a degree $max(d_2, d_3 d_1)$ PC proof of $\mathcal{P}(x)$.*

Lemma 5.23 *Let ℓ be a linear equation mod 2, over the variables $x_1, \ldots, x_k$, and let $\mathcal{P} = P_{\mathbb{F}}(\ell) \cup \{y_i^2 = 1\}_{i=1}^k$ be its formulation as a set of polynomials. Let C be a clause over $x_1, \ldots x_k$ such that $\ell \models C$. $\mathcal{P}$ is $(1, k+1)$-reducible to C.*

Proof: Set $x_i = (1 - y_i)/2$. From this definition and $y_i^2 = 1$ one can deduce $x_i^2 - x_i$. PC is implicationally complete, thus, there is a derivation of C from $\mathcal{P}$, using only variables that appear either in $\mathcal{P}$ or in C. This derivation can be assumed to have degree at most $k+1$, because every time a monomial with degree $k+1$ appears in the derivation, some variable x has degree 2, and the monomial can be reduced to degree k by use of the axioms $x^2 - x$. □

The following theorem is the main observation of this section. It says that taking the linear closure of a non-satisfiable CNF which has high expansion, cannot help decrease the proof degree substantially.

Theorem 5.24 *For $\mathcal{C}$ a k-CNF, $\mathcal{P}(\mathcal{C})$ its formulation as a set of polynomials over a field $\mathbb{F}$ of characteristic other than 2, and $\mathcal{P}(\mathcal{L}_\mathcal{C})$ the formulation of the linear closure of $\mathcal{C}$ as a set of polynomials over the same field:*

$$D(\mathcal{P}(\mathcal{C})) \geq \max\{D(\mathcal{P}(\mathcal{L}_\mathcal{C})), k+1\}$$

Proof: Clearly $\mathcal{C} \subseteq \mathcal{C}(\mathcal{L}_\mathcal{C})$, and hence any proof of $\mathcal{P}(\mathcal{C})$ is also a proof of $\mathcal{P}(\mathcal{C}(\mathcal{L}_\mathcal{C}))$. The previous lemma 5.23 implies that $\mathcal{P}(\mathcal{L}_\mathcal{C})$ is $(1, k+1)$-reducible to $\mathcal{P}(\mathcal{C}(\mathcal{L}_\mathcal{C}))$, and the theorem follows. □

Recall that a random CNF is a good expander with high probability (lemma 2.23). Notice that the exact same lemma with the same proof implies the same expansion properties for a random set of linear equations, since the definition of an *expanding CNF* does not depend on the fact that each function is a clause. Thus, applying lemma 2.23 and the previous theorem 5.24 we get (via theorem 5.7):

Corollary 5.25 *For any $k \geq 3$, and any $0 < \epsilon < 1/2$, with high probability $\mathcal{F} \sim \mathbb{C}_{\Delta \cdot n}^{k,n}$ has:*

$$D(\mathcal{P}(\mathcal{C})) = \Omega\left(\frac{n}{\Delta^{1/(k-2)+\epsilon}}\right)$$

over any field of characteristic other than 2.

Chapter 6

Bounded Depth Frege Size Lower Bounds

We close this monograph with a lower bound for the strongest proof system for which a super polynomial lower bound is known. This system is currently at the edge of our knowledge, as far as lower bounds are concerned. In this chapter we settle a ten-year open problem regarding the size of a minimal proof of expander-based Tseitin contradictions, and present an exponential lower bound for them. Our proof is by a non-trivial reduction of the Tseitin contradiction to the pigeonhole principle, for which exponential lower bounds are known.

6.1 Bounded depth Frege

The family of logic-based proof systems, called *Frege* systems is the oldest and most heavily investigated. These proof systems rely on a constant number of logically sound derivation rules (the most important being the *cut* rule), and are classified according to types of formulas which can be used as lines in the proof. At the bottom of the hierarchy lies our familiar *resolution* system, in which formulas are *clauses*, and at the top resides *extended Frege*, in which a line is an arbitrary Boolean circuit. Somewhere between these two extremes lie the *bounded depth Frege* systems that restrict lines to bounded depth formulas.

Following the natural intuition, that circuit lower bounds should yield lower bounds for the corresponding proof systems, and with the discovery of exponential lower bounds for bounded-depth circuits (see e.g. [Hastad ('88)]), the question of lower bounds for bounded depth Frege was raised. Such lower bounds were indeed found for the pigeonhole principle, starting with the seminal work of [Ajtai ('88)] and culminating with the exponential lower bounds of [Pitassi *et al.* ('93)], and [Krajíček *et al.* ('95)] (theorem 6.20). These lower bounds are extremely difficult, and involve nonstandard representations of proofs and delicate use of a switching lemma, specially tailored for use against the pigeonhole principle.

It was only natural to ask whether the weakness of bounded depth Frege is limited to the pigeonhole principle, and what other formulas are hard for this system. In this respect, Tseitin formulas are extremely interesting, given their hardness for other proof systems, and their connection to random CNFs (see next section). In this chapter we solve this open question by proving exponential lower bounds for Tseitin contradictions based on expander graphs. An important step toward resolving this question was given by [Fu and Urquhart ('96)], who proved exponential lower bounds for Tseitin formulas over complete graphs, which are of course excellent expanders. One problem that occurs when using Tseitin formulas over complete graphs, is that their *explicit encoding* has exponential size (see subsection 2.9.1). This problem was circumvented by Fu and Urquhart by using a specific *implicit encoding*, of small size. The lower bound itself is similar to the pigeonhole lower bound, using a slightly different switching lemma. Our current proof can be viewed as a generalization of their result, since the first step of our proof is a reduction to the complete graphs used by Fu and Urquhart, but uses a different implicit encoding. We then claim that the lower bounds apply to any implicit encoding of Tseitin contradictions of complete graphs.

6.1.1 Tseitin Contradictions and Random CNFs

Tseitin contradictions over expanders show a resemblance to random CNF formulas. Namely, both types of formulas are *expanding* according to definition 2.28. As we previously saw, this expansion is the key property used in proving lower bounds for both types of formulas (theorem 2.24, corollary 2.41). The original lower bound of [Chvátal and Szemerédi ('88)] for resolution proof size of random 3-CNFs built on ideas used by Urquhart in proving lower bounds for Tseitin Formulas [Urquhart ('87)]. More recently, the lower bounds of [Grigoriev ('98)] and by [Buss *et al.* ('99)] for Tseitin formulas in the *polynomial calculus*, inspired the random CNFs lower bound of [Ben-Sasson and Impagliazzo ('99)] (corollary 5.25) and of [Alekhnovich and Razborov ('00)]. Lower bounds for Tseitin formulas in bounded depth Frege, may be viewed as a step in the direction of proving lower bounds for random CNFs in this system.

6.2 Results and Proof Outline

In this chapter we prove exponential lower bounds for Tseitin contradictions for bounded depth Frege systems, whenever the graph on which the contradiction is based is an expander. The main theorem of this chapter says that bounded depth Frege is not much better than resolution or the polynomial calculus in proving Tseitin contradictions for expanders. Recall the definition of an explicit Tseitin contradiction (definition 2.38). For a depth d Frege system $\mathbf{F_d}$, let $\mathbf{S_{F_d}}$ denote the minimal size of a proof, measured in the number of symbols.

Theorem 6.1 *For any depth d Frege system $\mathbf{F_d}$, any integer $\Delta > 0$, and any constant $e > 0$, there exists a constant $\delta > 0$ such that for any large enough n, the following holds. For G a graph over n vertices of maximal degree $\leq \Delta$, and expansion constant e, and $\mathcal{C}[G,\sigma]$ an explicit Tseitin contradiction over G:*

$$\mathbf{S_{F_d}}(\mathcal{C}[G,\sigma]) \geq 2^{n^\delta}$$

As mentioned earlier, it is well known that for arbitrarily large n, there exist graphs with maximal degree 3, and expansion constant bounded from 0. For such graphs we get exponential lower bounds for the size of any constant depth Frege proof.

Our proof is a reduction of the Tseitin formula of an expander, to the pigeonhole principle. In other words, we show that if there is a short depth d proof of the Tseitin formula, then there is a short depth $d+c$ proof of the pigeonhole principle, for some small integer c. The exponential lower bound for this latter contradiction gives the lower bound for the Tseitin contradictions.

The reduction is not trivial. The pigeonhole principle discusses *matchings* in *complete bipartite* graphs, whereas a Tseitin formula talks about *sums* of degrees in *bounded degree* graphs. Therefore, our reduction goes through three basic steps:

1. Reduce the Tseitin formula of the expander to an implicit Tseitin formula over a complete graph.

2. Reduce our implicit Tseitin formula over the complete graph to the matching principle, which says that there is no perfect matching in a complete graph of odd size.

3. Reduce the matching principle to the onto-pigeonhole principle, which states that there is no bijection from a set of size $n+1$ to a set of size n.

The first and last steps are simply restrictions, and thus can only decrease the size and depth of the proof (see lemma 6.13). The middle step might increase the size of the proof polynomially, and the depth by a constant. Thus the exponential lower bound for the pigeonhole principle translates to a similar lower bound for the original formula.

The middle step of our reduction deals with removal of auxiliary variables, an interesting problem in its own right. Addition of arbitrary auxiliary variables to *resolution*, the weakest of the Frege systems, makes it as efficient as *extended Frege*, the strongest of the family. An important question, which has not received attention so far is that of the effect of auxiliary variables on length of proofs, and finding cases where such auxiliary variables do not decrease the proof size significantly.

We give a novel and general characterization of cases when one can remove such auxiliary variables safely. This characterization may be generalized further.

6.2.1 Chapter Organization

In the following section we give a general introduction to bounded depth Frege systems. Section 6.4 is devoted to the proof of the main theorem, leaving out the elimination of auxiliary variables, which is dealt with in section 6.5.

6.3 Bounded Depth Frege - Definitions

The lower bounds we prove will be for a concrete bounded depth Frege system, to be defined shortly. If we begin with a bounded-depth Frege proof over a different unbounded fan-in basis, in a different Frege system than the one specified above, then we can convert the proof into a bounded-depth proof in our system by using the ideas in the simulation result of [Cook and Reckhow ('79)]. We would like to point out that although the simulation preserves constant depth, the depth after conversion may increase by a constant factor. Thus, the actual constants in our lower bound, as in the other lower bounds for constant depth Frege, are sensitive to the particular set of connectives and derivation rules one starts with.

For simplicity our logical language will be restricted to constants 0 (representing *false*), and 1 (representing *true*), and connectives $\{\vee, \neg\}$ where $\vee$ is allowed to have unbounded fan-in. We will use the connective $\wedge$ as a shorthand for $\neg \vee \neg$, and $A \to B$ as a shorthand for $\neg A \vee B$.

We will work with the Frege system **H** described by [Bellantoni *et al.* ('92)], which is the standard bounded-depth Frege system, that was used for proving the exponential lower bounds for the pigeonhole principle. Lines of a proof are unbounded fan-in formulas over $\{\neg, \vee\}$. Namely, the allowable formulas are defined inductively by the rules:

1. A variable x is a formula, and so is the literal $\neg x$.

2. If A is a formula, then so is $\neg A$.

3. If Γ is a finite set of formulas, then so is $\vee\Gamma$. We will use the notation $A \vee B$ to mean $\vee\{A, B\}$.

The *depth* of a literal is 0, the *depth* of a formula is the maximal number of alternations of connectives in it and the *size* of the formula is the number of occurrences of connectives. Under this convention, the depth of a clause (disjunction of literals) is 0, the depth of a conjunction of literals is 1, the depth of a DNF is 2, and of a CNF 3. We denote by $d(\phi)$ the depth of the formula ϕ.

We now list the rules of **H**, which form a complete proof system over the basis $\{\vee, \neg\}$. We use the notation $\frac{\phi_1 \ldots \phi_k}{\psi}$ to denote that ψ can be derived from $\{\phi_1 \ldots \phi_k\}$.

1. Excluded Middle axiom: $\frac{}{A \vee \neg A}$

2. Weakening Rule: $\frac{A}{A \vee B}$

3. Merging Rule: $\frac{\bigvee(\{\vee\Gamma\} \cup \Delta)}{\bigvee(\Gamma \cup \Delta)}$

4. Unmerging Rule: $\frac{\bigvee(\Gamma \cup \Delta)}{\bigvee(\{\vee\Gamma\} \cup \Delta)}$

5. Cut Rule: $\frac{(A \vee B), \quad (\neg A \vee C)}{B \vee C}$

A $\mathbf{H_d}$-*derivation* of a formula ϕ from a set of formulas formula Ψ is a sequence of depth d formulas $\pi = \{\phi_1, \ldots \phi_s\}$, the last one being ϕ, and each formula in the sequence is either some formula $\psi \in \Psi$, or an excluded middle axiom, or is derived from previous lines by one of the other rules listed above. A $\mathbf{H_d}$-*proof* of a Ψ is a derivation of the empty clause. The *size* of a proof is the sum of the sizes of the formulas in the proof. The *depth* of the proof is the maximal depth of the formulas in the proof.

For ϕ a formula, the restriction of ϕ by ρ, denoted $\phi|_\rho$, is defined by applying ρ bottom-up to all sub-formulas of ϕ, in the standard way:

1. If ϕ is a variable assigned by ρ, replace it with its value under ρ, otherwise leave it unchanged.

2. If ϕ is $\neg\psi$ then $\phi|_\rho = \neg(\psi|_\rho)$, where $\neg 1 = 0$ and $\neg 0 = 1$.

3. If ϕ is $\vee\{\Gamma\}$ then $\phi|_\rho = \vee\{\Gamma|_\rho\}$ (i.e. apply ρ to all formulas in Γ and take their disjunction), where $1 \vee \{\Gamma'\} = 1$ and $\vee\{0\} = 0$.

Applying a restriction to a proof is equivalent to applying it to all lines in the proof: $\pi|_\rho = \{\phi_1|_\rho, \ldots \phi_s|_\rho\}$.

Lemma 6.2 *If π is a proof of ψ with ℓ lines, depth d and size s, then for any restriction ρ, $\pi|_\rho$ is a proof of $\psi|_\rho$ with at most ℓ lines, depth $\leq d$, and size $\leq s$.*

Sometimes we will treat a CNF as a set of clauses, and sometimes as a single formula, and the following simple lemmas state that we can do this without a great increase in the proof size.

Lemma 6.3 *There is a derivation of ϕ from $\phi \wedge \psi$ of size $O(|\phi| + |\psi|)$ and depth $d(\phi \wedge \psi)$.*

Lemma 6.4 *There is a derivation of $(\phi_1 \wedge \ldots \wedge \phi_k) \vee \psi$ from $\{\phi_1 \vee \psi, \ldots, \phi_k \vee \psi\}$ of size $O(|\phi_1| + \ldots + |\phi_k| + |\psi|)$, and depth $\max\{d(\phi_1), \ldots, d(\phi_k), d(\psi)\}$.*

Lemma 6.5 *If a CNF $\mathcal{C}$ has a resolution proof of size k, then $\neg\mathcal{C}$ has a depth 4 derivation of size $O(k)$.*

6.4 Proof of Theorem 6.1

Recall the definitions of explicit and implicit Tseitin contradictions (definition 2.38), as well as that of graph expansion (definition 2.36), from subsection 2.9.1. Throughout the proof we assume $G =< V, E >$ to be a connected graph on n vertices, with constant degree Δ (say $\Delta = 3$) and expansion factor $e > 0$. We also assume σ to have odd-weight, and thus $\mathcal{C}[G, \sigma]$ is an unsatisfiable Δ-CNF of size $O(n)$.

6.4.1 G has a Large Complete Graph as a Minor

The first step of the proof is to quote the famous theorem of Alon, Seymour and Thomas, stating that every expander "hides" within it a large complete graph, as a minor. Applying this theorem to G gives us the following theorem, which we prove in the rest of this subsection.

Theorem 6.6 *If $G =< V, E >$ is a graph of maximal degree Δ, and expansion constant e, then V can be partitioned into $V_1 \ldots V_h$, such that:*

1. *For $i = 1, \ldots h$ the sub-graph induced by V_i is connected.*

2. *For any $1 \leq i < j \leq h$, there is at least one edge connecting some vertex in V_i to some vertex in V_j.*

3. *$h \geq (\frac{e}{3(\Delta+e)})^{2/3} n^{1/3}$.*

We now give the proof of this theorem. Let $G =< V, E >$ be a graph. H is derived from G by an *edge deletion* if H is isomorphic to $< V, E - e >$ for some edge $e \in E$. Let $e = (u, v)$ be an edge of G. We denote by G/e the graph obtained from G by *contracting* the edge e into a new vertex v_e, which becomes adjacent to all the former neighbors of u and v. Formally, G/e is the graph $< V', E' >$ with vertex set $V' = V - \{u, v\} \cup \{v_e\}$ and edge set $E' = \{(x, y) \in E : (x, y) \cap (u, v) = \emptyset\} \cup \{(v_e, y) : (u, y) \in E - e \text{ or } (v, y) \in E - e\}$.

H is said to be a *minor* of G if it is isomorphic to a graph derived from G by a sequence of edge deletions and edge contractions.

Lemma 6.7 *K_h is a minor of G iff there is a partition of V into $V_1, \ldots V_h$ such that the following hold:*

1. *For $i = 1, \ldots h$ the sub-graph induced by V_i is connected.*
2. *For any $1 \leq i < j \leq h$, there is at least one edge connecting some vertex in V_i to some vertex in V_j.*

A *separation* of a graph $G =< V, E >$ is a partition of V into three sets (A, B, C), such that $|A|, |B| \leq 2/3n$ and no edge of G joins a vertex in A to a vertex in B. Its *order* is $|C|$. A well known theorem of [Alon, Seymor and Thomas ('90)], which extends a theorem of [Lipton and Tarjan ('79)], says:

Theorem 6.8 *[Alon, Seymor and Thomas ('90)] Let $h \geq 1$ be an integer, let G be a graph with n vertices, and no K_h minor. Then there is a separator (A, B, C) of G of order $h^{3/2} n^{1/2}$.*

It is not hard to see that a good expander has large separators:

Lemma 6.9 *Any separation of a graph G of maximal degree Δ and expansion constant e has order $\geq \frac{e}{3(\Delta+e)} \cdot n$.*

Proof: Let (A, B, C) be a separation of G, with $|C| = h$. On the one hand

$$\Delta \cdot h \geq |E(C, \bar{C})| \tag{6.1}$$

On the other hand, assume wlog $|A| \leq |B| \leq n/2$. Since $|B| \leq 2/3n$ we get $|A| \geq n/3 - h$. By definition of separation, all edges leaving A must land in C, and hence

$$|E(C, \bar{C})| \geq |E(A, \bar{A})| \geq e \cdot (n/3 - h) \tag{6.2}$$

Combining equations (6.1), (6.2) completes the proof. □

Proof [Theorem 6.6]: By lemma 6.9, the order of any separator of G is of order at least $\frac{e}{3(\Delta+e)} \cdot n$. By theorem 6.8, G has a minor of size $(\frac{e}{3(\Delta+e)})^{2/3} n^{1/3}$. Lemma 6.7 completes the proof. □

6.4.2 $\mathcal{C}[G,\sigma]$ is an Implicit Encoding of $\mathcal{C}[K_h,\mathbf{1}]$

Theorem 6.6 gives us some hope, as it says that G "hides" within it a large complete sub-graph, and we already know from the work of Fu and Urquhart that some implicit Tseitin contradictions over complete graphs are hard for bounded depth Frege [Fu and Urquhart ('96)]. In this subsection we show that $\mathcal{C}[G,\sigma]$ is actually an implicit encoding of a Tseitin formula over the complete "hidden" sub-graph, where the vertex labeling function is the constant $\mathbf{1}$ function. Thus, a short proof for $\mathcal{C}[G,\sigma]$ will mean the existence of a short proof for some implicit Tseitin contradiction over the complete graph. The main theorem of this subsection is:

Theorem 6.10 *Suppose G has a K_h minor. There exists an implicit Tseitin formula $\mathcal{I}[K_{h'},\mathbf{1}]$, for $h' \geq h-2$, such that*

$$\mathbf{S_{H_d}}(\mathcal{I}[K_{h'},\mathbf{1}]) \leq \mathbf{S_{H_d}}(\mathcal{C}[G,\sigma]).$$

The rest of this subsection gives the proof of this theorem.

Definition 6.11 *For S a subset of V, let $X(S,\bar{S}) \stackrel{\text{def}}{=} \{x_e : e \in E(S,\bar{S})\}$, and $\sigma(S) \stackrel{\text{def}}{=} \bigoplus_{v\in S}\sigma(v)$.*

Lemma 6.12 *For S a connected subset of V, $\bigwedge_{v\in S}\mathcal{C}(EQ(X(v),\sigma(v)))$ is an implicit encoding of*

$$EQ(X(S,\bar{S}),\sigma(S)).$$

Proof: We use the following lemma, stated without proof.

Lemma 6.13 *For $b \in \{0,1\}$, and $v \in e$:*

$$EQ(X(v),\sigma(v))|_{x_e=b} \equiv EQ(X(v)-\{x_e\},\sigma(v)\oplus b)$$

In words: restricting $EQ(X(v),\sigma(v))$ by setting $x_e=b$ is equivalent to removing e from G and changing $\sigma(v)$ to $\sigma(v)\oplus b$.

Define $\mathcal{C}[S,\sigma]=\bigwedge_{v\in S}\mathcal{C}(EQ(X(v),\sigma(v)))$. Look at $\mathcal{C}[S,\sigma]|_\alpha$ for some $\alpha\in\{0,1\}^{X(S,\bar{S})}$. By lemma 6.13, assigning $x_e=b$ for $e\in E(S,\bar{S})$, is equivalent to removing this edge and changing $\sigma(v)$ to $\sigma(v)\oplus b$. Thus $\mathcal{C}'[S,\sigma']=\mathcal{C}[S,\sigma]|_\alpha$ is a Tseitin formula over the connected sub-graph induced by S, with a new vertex label function σ'. By lemma 2.39 $\mathcal{C}'[S,\sigma']$ is satisfiable iff σ' has even weight. Let us calculate the weight of σ':

$$\bigoplus_{v\in S}\sigma'(v) \quad = \quad \bigoplus_{v\in S}\sigma(v)\oplus\bigoplus_{e\in E(S,\bar{S})}\alpha(x_e)$$

Thus σ' has even weight iff

$$\sigma(S)=\bigoplus_{e\in E(S,\bar{S})}\alpha(x_e),$$

This happens iff α satisfies $EQ(X(S,\bar{S}),\sigma(S))$. So $\bigwedge_{v\in S}\mathcal{C}(EQ(X(v),\sigma(v)))$ is an implicit encoding of $EQ(X(S,\bar{S}),\sigma(S))$, according to definition 2.33. $\square$

The previous lemma shows that a sub-formula of a Tseitin formula, corresponding to a connected sub-graph S, is an implicit encoding of the Boolean function $EQ(X(S,\bar{S}),\sigma(S))$. We use the partition of G promised by theorem 6.6 to show that $\mathcal{C}[G,\sigma]$ is an implicit encoding of $\mathcal{I}[K_h,\sigma']$.

Lemma 6.14 *If G has a K_h minor then $\mathcal{C}[G,\sigma] = \mathcal{I}[K_h,\sigma']$, for σ' of the same weight as σ.*

Proof: Partition V into $V_1 \ldots V_h$ according to lemma 6.7, and for reasons soon to become clear, call each V_i a *super-vertex*. Assume wlog that any pair of distinct super-vertices is connected by a single edge (otherwise assign 0 to all other edges connecting the two super-vertices). Partition E into *external* edges, connecting two super-vertices, and *internal* ones, residing within a single super-vertex. By the previous lemma 6.12, $\bigwedge_{v\in V_i} \mathcal{C}(EQ(X(v),\sigma(v)))$ is an implicit encoding of $EQ(X(V_i,\bar{V_i}),\sigma(V_i))$. The auxiliary variables of this encoding are labeled by the internal edges of V_i. Thus, for any pair of distinct super-vertices, their encodings use *distinct* auxiliary variables. It is not hard to see that $\bigoplus_V \sigma(v) = \bigoplus_{i=1}^h \sigma(V_i)$. $\square$

Our next step is to make sure the odd-weight function is the constant $\mathbf{1}$ function over a vertex set of odd size.

Lemma 6.15 *Given $\mathcal{I}[K_h,\sigma]$ with σ of odd weight, there exists a restriction ρ such that $\mathcal{I}[K_h,\sigma]|_\rho = \mathcal{I}[K_{h'},\mathbf{1}]$, $h' \geq h-2$, and h' is odd.*

Proof: Let v be some vertex such that $\sigma(v) = 1$. Such a vertex must exist because σ has odd weight. Let U be some set of vertices labeled 0 of maximal *odd* size. We define the following restriction

$$\rho(x_e) \stackrel{\text{def}}{=} \begin{cases} 1 & e = (v,u), \quad u \in U \\ 0 & e = (v,w), \quad w \notin U \\ \star & \text{otherwise} \end{cases}$$

Under this restriction, we have satisfied $EQ(X(v),\sigma)$ because we have assigned an odd number of edges with 1. By lemma 6.13 all vertices in U are now labeled 1. There might remain one 0 labeled vertex, and we remove this vertex by assigning 0 to all its incident edges. At most 2 vertices were removed, all remaining restricted vertices are labeled 1, and it is not hard to verify that the number of remaining vertices is odd. $\square$

So far we took the expander G, which contains a large minor K_h, partitioned V into h super-vertices, and applied a restriction so that any pair of super-vertices is connected by exactly one edge. Lemmas 6.12, 6.14, and 6.15 show that another restriction transforms $\mathcal{C}[G,\sigma]$ to an implicit encoding of a Tseitin formula over $K_{h'}$ with the labeling function being the constant $\mathbf{1}$ function. Since all transformations were simply restrictions, lemma 6.13 shows that neither the proof size, nor the depth increased, and this completes the proof of theorem 6.10. $\square$

6.4.3 Reducing $\mathcal{I}[K_h,\mathbf{1}]$ to Match(h)

Remark: An alternative proof of our main theorem could proceed along the following lines: [Fu and Urquhart ('96)], proved directly lower bounds for $\mathcal{I}[K_h,\mathbf{1}]$ where each formula $\mathcal{I}(EQ[X,1])$ has a specific structure, and uses well defined auxiliary variables. One can generalize their result to *arbitrary* implicit encodings, and get a direct proof of the main theorem. We wish to show a complete reduction of the Tseitin contradictions to the pigeonhole principle, so proceed differently.

Our next step will be to show that any short proof of $\mathcal{I}[K_h, \mathbf{1}]$ can be transformed into a short proof of the matching principle.

Definition 6.16 (Matching Principle) *The Unique-Or function is the Boolean function that outputs* 1 *iff exactly one of its inputs is* 1*. This function is computed by the CNF formula*

$$\mathbf{UO}[X] \stackrel{\text{def}}{=} (\bigvee_{x \in X} x) \wedge \bigwedge_{x \neq x', x, x' \in X} (\bar{x} \vee \bar{x}')$$

The matching principle of size h is the following formula over the variable set $\{x_e : e \in K_h\}$:

$$\mathbf{Match(h)} \stackrel{\text{def}}{=} \bigwedge_{v \in V(K_h)} \mathbf{UO}[X(v)]$$

$\mathbf{Match(h)}$ is satisfiable iff h is even. Notice that $\mathbf{UO}(x_1 \dots x_n) \models \bigoplus(x_1, \dots x_n)$. Using this observation we wish to reduce a short proof of $\mathcal{I}[K_h, \mathbf{1}]$ to a short proof of $\mathbf{Match(h)}$. The problem is that we must reduce the implicit formula $\mathcal{I}[K_h, \mathbf{1}]$, having many auxiliary variables, to an explicit formula $\mathbf{Match(h)}$, which has none, i.e. we must replace auxiliary variables with explicit formulas, without making the proof size grow too much. In the general case such a removal *will* increase the proof size exponentially. We introduce a general technique, that may be interesting in its own right, for getting rid of auxiliary variables, in certain simple cases. The proof of the following theorem is deferred to section 6.5.

Theorem 6.17 *There exists a constant $c' > 1$ such that*

$$\mathbf{S_{H_{d+10}}}(\mathbf{Match(h)}) \leq (\mathbf{S_{H_d}}(\mathcal{I}[K_h, \mathbf{1}]))^{c'}.$$

6.4.4 Reducing Match(h) to the Pigeonhole Principle

Our proof of theorem 6.1 is almost complete, and the last reduction of the matching principle to the *onto-pigeonhole* principle is once again a simple restriction.

Definition 6.18 (Onto $\mathbf{PHP^m_n}$) *$OPHP^m_n$ is the conjunction of the following set of clauses:*

1. $\bigvee_{1 \leq j \leq n} x_{ij} \qquad 1 \leq i \leq m.$
2. $\bigvee_{1 \leq i \leq m} x_{ij} \qquad 1 \leq j \leq n.$
3. $\bar{x}_{ij} \vee \bar{x}_{i'j} \qquad 1 \leq i < i' \leq m, 1 \leq j \leq n.$
4. $\bar{x}_{ij} \vee \bar{x}_{ij'} \qquad 1 \leq i \leq m, 1 \leq j < j' \leq n.$

Lemma 6.19 *For $h = 2k + 1$,*

$$\mathbf{S_{H_d}}(OPHP^{k+1}_k) \leq \mathbf{S_{H_d}}(\mathbf{Match(h)}).$$

Proof: Let π be a proof of $\mathbf{Match(h)}$. Apply to it the following restriction:

$$\rho(x_{ij}) \stackrel{\text{def}}{=} \begin{cases} 0 & i,j \leq k+1 \text{ or } i,j > k+1 \\ \star & \text{otherwise} \end{cases}$$

It is not hard to verify that

$$\mathbf{Match(h)}|_\rho = OPHP_k^{k+1}$$

By lemma 6.13 the proof size of the restricted proof does not increase. □

6.4.5 Completing the Proof of Theorem 6.1

The following theorem was proved by [Pitassi *et al.* ('93)], and independently by [Krajíček *et al.* ('95)].

Theorem 6.20 *For any integer $d > 0$, there exists a constant $\delta > 0$ such that:*

$$\mathbf{S_{H_d}}(OPHP_n^{n+1}) \geq 2^{n^\delta}.$$

Let $\mathcal{C}[G,\sigma]$ be our explicit Tseitin contradiction where G has maximal degree Δ and constant expansion e, and let $s = \mathbf{S_{H_d}}(\mathcal{C}[G,\sigma])$. By theorem 6.6 G has a minor of size $\Omega(n^{\frac{1}{3}})$. By theorem 6.10 this implies that some implicit Tseitin contradiction $\mathcal{I}[K_h, \mathbf{1}]$ has an $\mathbf{H_d}$ proof of size $\leq s$, where $h = \Omega(n^{\frac{1}{3}})$. By theorem 6.17 the matching principle $\mathbf{Match(h)}$ has an $\mathbf{H_{d+c}}$ proof of size $s^{O(1)}$, for some constant c. By lemma 6.19, this implies an $\mathbf{H_{d+c}}$ proof of $OPHP_k^{k+1}$ of size $s^{O(1)}$, where $k = \frac{h-1}{2}$. Plugging in theorem 6.20 for the Frege system $\mathbf{H_{d+c}}$, we conclude that there is some constant $\delta > 0$ such that $s \geq 2^{n^\delta}$ for sufficiently large n. We can present the proof schematically:

$$\begin{aligned}
\mathbf{S_{H_d}}(\mathcal{C}[G,\sigma]) &\geq \mathbf{S_{H_d}}(\mathcal{I}[K_{n^{1/3}}, \mathbf{1}]) && (6.3)\\
&\geq \left(\mathbf{S_{H_{d+c}}}(\mathbf{Match(n^{1/3})})\right)^\epsilon && (6.4)\\
&\geq \left(\mathbf{S_{H_{d+c}}} OPHP^{\frac{n^{1/3}}{2}}\right)^\epsilon && (6.5)\\
&\geq (2^{n^{\delta/3}})^\epsilon && (6.6)\\
&\geq 2^{n^{\delta/4}} && (6.7)
\end{aligned}$$

The proof of theorem 6.1 is completed. □

6.5 Getting Rid of Auxiliary Variables

In this section we provide a general theorem that allows us to remove auxiliary variables from proofs, without increasing the proof size tremendously (theorem 6.25). This theorem will imply theorem 6.17. The main idea is as follows. Suppose $\mathcal{I}$ is an implicit encoding of some function $f(x_1, \ldots, x_n)$, and let $g(x_1, \ldots, x_n)$ be an arbitrary Boolean function such that $g \models f$, and additionally, $g^{-1}(1)$ is small. Since

$g \models f$, for any satisfying assignment α to g there is some assignment $\beta = \beta_1 \dots \beta_m$ ($\beta_i \in \{0,1\}$) to the auxiliary variables such that $\mathcal{I}(\alpha, \beta) = 1$. Since $g^{-1}(1)$ is small, only a small number of assignments satisfy g, and hence we can represent g by a DNF with a small number of terms. We will replace the auxiliary literal $\neg y_i$ with the DNF that encodes the subset of assignments α that satisfy g and additionally $\beta_i(\alpha) = 1$. The auxiliary literal y_i will be replaced by the dual DNF that encodes the subset of assignments α that satisfy g and $\beta_i(\alpha) = 0$. We will show that this replacement transforms a proof of $\mathcal{I}(f) \wedge \psi$ ($\mathcal{I}(f)$ is some implicit encoding of f) into a proof of $\mathcal{C}(g) \wedge \psi$ ($\mathcal{C}(g)$ is some explicit encoding of g, with no auxiliary variables), and the proof size increases only polynomially.

We start with the necessary definitions, and then state and prove our main theorem, theorem 6.25. Finally, we prove theorem 6.17 as an application of this theorem.

6.5.1 Definitions

Recall that for x_i a variable, $x_i^1 \stackrel{\text{def}}{=} x_i$ and $x_i^0 \stackrel{\text{def}}{=} \bar{x}_i$. For $x = x_1 \dots x_n$ and $\alpha \in \{0,1\}^n$, let $\wedge x^\alpha \stackrel{\text{def}}{=} x_1^{\alpha_1} \wedge \dots \wedge x_n^{\alpha_n}$, and similarly $\vee x^\alpha \stackrel{\text{def}}{=} x_1^{\alpha_1} \vee \dots \vee x_n^{\alpha_n}$. We use the notation $\vee x^{\bar{\alpha}} \stackrel{\text{def}}{=} x_1^{1-\alpha_1} \vee \dots \vee x_n^{1-\alpha_n}$, and notice that by De-Morgan's law $\neg(\wedge x^\alpha) \equiv (\vee x^{\bar{\alpha}})$.

Let $\mathcal{I} = \mathcal{I}(x_1, \dots, x_n, y_1, \dots, y_m)$ be an implicit encoding of $f(x_1, \dots, x_n)$. A function $\beta : \{0,1\}^n \to \{0,1\}^m$ is called a *Skolem function* of $\mathcal{I}$ if for all $\alpha \in \{0,1\}^n$:

$$(f(\alpha) = 1) \quad \text{implies} \quad (\mathcal{I}(\alpha, \beta(\alpha)) = 1) \,.$$

In words: a Skolem function assigns values to the auxiliary variables such that if possible, $\mathcal{I}$ will be satisfied. Notice that there may be many different Skolem functions for any given $\mathcal{I}$. We use the notation β_i to define the projection of β on the i'th bit (i.e. $\beta(\alpha) = \beta_1(\alpha), \dots, \beta_m(\alpha)$).

For $g(x_1, \dots, x_n)$ a Boolean function, let $g^{-1}(1) \subseteq \{0,1\}^n$ be its set of satisfying assignments. The *canonical DNF encoding* of g is the DNF formula:

$$DNF(g) \stackrel{\text{def}}{=} \bigvee_{\alpha \in g^{-1}(1)} (\wedge x^\alpha)$$

The *canonical DNF encoding* of a subset $A \subset \{0,1\}^n$ is the canonical DNF encoding of the characteristic function of A.

For $\mathcal{I} = \mathcal{I}(x_1, \dots, x_n, y_1, \dots, y_m)$ an implicit encoding of $f(x_1, \dots, x_n)$, β a Skolem function of $\mathcal{I}$, $g(x_1, \dots, x_n)$ some function such that $g(x_1, \dots, x_n) \models f(x_1, \dots, x_n)$, and y_i an auxiliary variable in $\mathcal{I}$, define the couple of subsets of $\{0,1\}^n$

$$A_i^0 = A_i^0(\mathcal{I}, f, g, \beta) \stackrel{\text{def}}{=} \{\alpha \in g^{-1}(1) : \beta(\alpha)_i = 0\}$$
$$A_i^1 = A_i^1(\mathcal{I}, f, g, \beta) \stackrel{\text{def}}{=} g^{-1}(1) - A_i^0$$

In words: A_i^1 is the set of satisfying assignments for g such that the corresponding assignment to the auxiliary variable y_i is 1, and A_i^0 is the analogous set for $y_i = 0$. Finally, let Y_i^0 (Y_i^1 respectively) be the canonical DNF encoding of the set A_i^0 (A_i^1 respectively). Notice that Y_i^0, Y_i^1 are depth 4 formulas, of size $\leq n \cdot |g^{-1}(1)|$, that have *no auxiliary variables*. Additionally, notice that the DNF $Y_i^1 \vee Y_i^0$ computes g. We will remove the auxiliary literals by replacing each one by its suitable suitable DNF, as in the following definition.

Definition 6.21 (Replacements) *For $\mathcal{I}(x_1, \ldots, x_n, y_1, \ldots, y_m)$ an implicit encoding of $f(x_1, \ldots, x_n)$, β a Skolem function of $\mathcal{I}$, $g(x_1, \ldots, x_n)$ a function such that $g \models f$, and ϕ a formula over the variables $x_1, \ldots, x_n, y_1, \ldots, y_m$, the $(f, \mathcal{I}, \beta, g)$-*replacement *of ϕ is derived from ϕ by replacing every auxiliary literal $\bar{y}_i$ in ϕ by Y_i^0, and then replacing every remaining auxiliary literal y_i by Y_i^1.*

Applying a replacement to a formula does not increase its size or depth greatly, as the following lemma shows.

Lemma 6.22 *For $\mathcal{I}, f, g, \beta, \phi$ as in definition 6.21, the $(f, \mathcal{I}, \beta, g)$-replacement of ϕ increases its depth by at most 2, and its size by a multiplicative factor of at most $n \cdot |g^{-1}(1)|$. The new formula has no auxiliary literals.*

Proof: We replace every auxiliary literal in $\mathcal{I}$ by a DNF, that increases the depth by at most 2. There are at most $|\mathcal{I}|$ literals in $\mathcal{I}$, and the size of any Y_i is at most $n \cdot |g^{-1}(1)|$. It is easy to see that no auxiliary variables remain after the replacement, because the Y_i's have no such variables. $\square$

The following technical lemma will be used later on. Intuitively, and using the notation of definition 6.21, it says that if $\mathcal{I}'$ is a replacement of $\mathcal{I}$, then for any α that satisfies g there is a simple derivation of $(\wedge x^\alpha) \to \mathcal{I}'$. Later on we shall use this lemma to derive $\mathcal{I}'$ from a CNF that encodes g.

Lemma 6.23 *Let $\mathcal{I}, f, g, \beta$ be as in definition 6.21, and let $\mathcal{I}'$ be the $(f, \mathcal{I}, \beta, g)$-replacement of $\mathcal{I}$. For any $\alpha \in g^{-1}(1)$ there is a depth 6 proof of*

$$(\vee x^{\bar{\alpha}}) \vee \mathcal{I}',$$

and the size of this proof is $O(|\mathcal{I}'|)$.

Notice that the depth of $(\vee x^{\bar{\alpha}}) \vee \mathcal{I}'$ is 6, because the depth of $\mathcal{I}'$ is 5, and we have one additional alternation on top of $\mathcal{I}'$ (the topmost gate of $\mathcal{I}'$ is a negation, whereas the topmost gate of $(\vee x^{\bar{\alpha}}) \vee \mathcal{I}'$ is a $\vee$ gate).

Proof: Let C be a clause of $\mathcal{I}$, and let C' be the DNF formula derived by applying the replacement to C. We prove the existence of a short, depth 6 proof of $(\vee x^{\bar{\alpha}}) \vee C'$. By the definition of a Skolem function, and since $g \models f$, we get $C(\alpha, \beta(\alpha)) = 1$. So there must be some literal $\ell \in C$ that satisfies C under the assignment $(\alpha, \beta(\alpha))$. We split our proof according to whether ℓ is an essential or auxiliary literal.

1. ℓ is an essential literal, i.e. $\ell = x_i^{\alpha_i}$. In this case, $\vee x^{\bar{\alpha}} \vee C'$ is a weakening of the excluded middle axiom $x_i^0 \vee \neg x_i^1$, because $C' = x_i^{\alpha_i} \vee \ldots$ whereas $\vee x^{\bar{\alpha}} = x_i^{1-\alpha_i} \vee \ldots$.

2. ℓ is an auxiliary literal over a variable y_i, i.e. $\ell = y_i^{\beta_i(\alpha)}$. By the definition of the A_i's, $\alpha \in A_i^{\beta_i(\alpha)}$, and hence $Y_i^{\beta_i(\alpha)} = (\wedge x^\alpha) \vee \ldots$. So by the replacement procedure, $C' = (\wedge x^\alpha) \vee \ldots$, whereas $\vee x^{\bar{\alpha}} = \neg(\wedge x^\alpha)$. Once again $(\vee x^{\bar{\alpha}}) \vee C'$ is a weakening of the excluded middle axiom $\neg(\wedge x^\alpha) \vee (\wedge x^\alpha)$.

Notice that in both cases the depth of the derivation is at most 6, since we only used the excluded middle axiom, weakening, and merging/unmerging. The size of the proof is linear in $|C'|$. We conclude that for any clause $C \in \mathcal{I}$ there is a short derivation of $C' \vee (\vee x^{\bar{\alpha}})$. Lemma 6.4 shows we can combine all such clauses into the formula $\mathcal{I}' \vee (\vee x^{\bar{\alpha}})$, and the proof is complete. $\square$

We now define replacements of multiple functions, in a straightforward generalization of definition 6.21.

Definition 6.24 (Replacements - Multiple functions) *Let $\mathcal{F} = \{f_1 \dots f_m\}$ be a set of functions over $x_1, \dots, x_n$, let $\mathcal{I}(\mathcal{F}) = \{\mathcal{I}_1 \dots \mathcal{I}_m\}$ be an implicit encoding of $\mathcal{F}$, according to definition 2.33. Let $\vec{\beta} = \{\beta^1 \dots \beta^m\}$ be a set of Skolem functions for $\{\mathcal{I}_1 \dots \mathcal{I}_m\}$ respectively. Let $\mathcal{G} = \{g_1, \dots g_m\}$ be another set of functions over $x_1, \dots, x_n$, such that $g_i \models f_i \quad i = 1 \dots m$, and let ϕ be a formula over $Vars(\mathcal{I}(\mathcal{F}))$. The $(\mathcal{F}, \mathcal{I}(\mathcal{F}), \vec{\beta}, \mathcal{G})$-replacement of ϕ is derived from ϕ by applying all $(f_i, \mathcal{I}_i, \beta^i, g_i)$-replacements to it.*

Notice that the replacement of is well defined, since by definition 2.33 every $\mathcal{I}_i$ uses distinct auxiliary variables. Thus each auxiliary variable appearing in ϕ appears in exactly one $\mathcal{I}_j$, and we apply the $(f_j, \mathcal{I}_j, \beta_j, g_j)$- replacement to this literal, independently of the other replacements. As in the case of a single function replacement, the multiple function replacement increases the depth by at most 2, and the size by at most a factor of $n \cdot \max\{|g_i^{-1}(1)| : i = 1 \dots m\}$.

6.5.2 Removing Auxiliary Variables - Main Theorem

The main theorem of this section is the following.

Theorem 6.25 *Let $\mathcal{F}, \mathcal{I}(\mathcal{F}), \mathcal{G}$ be as in definition 6.24, and for $i = 1 \dots m$ let $G_1 \dots G_m$ be a set of formulas over $x_1, \dots, x_n$, with the following properties:*

1. *For $i = 1 \dots m$, G_i computes g_i, i.e. $\forall \alpha \in \{0,1\}^n G_i(\alpha) = g_i(\alpha)$.*
2. *For $i = 1 \dots m$, there exists a derivation of depth c and size r of*

$$\neg G_i \vee DNF(g_i)$$

Then

$$\mathbf{S_{H_{d+c+6}}}(\bigwedge_{i=1}^{m} G_i) = O\left(r^2 \cdot \mathbf{S_{H_d}}(\mathcal{I}(\mathcal{F}))\right).$$

Proof: Notice that condition 2 implies $|g_i^{-1}(1)| \le r/n$, and each G_i has depth $\le c$. Let $\pi = \{\phi_1 \dots \phi_k\}$ be a minimal size, depth d proof of $\mathcal{I}(\mathcal{F})$, and denote the size of π by s. For each $i \in [m]$, fix some Skolem function β^i for $\mathcal{I}_i$, and let $\vec{\beta}$ denote the set of these m Skolem functions. Start by forming $\pi' = \{\phi_1' \dots \phi_k'\}$ where each ϕ_i' is the $(\mathcal{F}, \mathcal{I}(\mathcal{F}), \vec{\beta}, \mathcal{G})$-replacement of ϕ_i. Similar to the proof of lemma 6.22, this replacement increases the size of the proof by a factor of r, and the depth by 2. Now, inductively correct the replaced sequence to make it a proof. We split the correction to cases, according to the type of derivation rule used to derive ϕ_j.

1. $\phi_j = \mathcal{I}(\mathcal{F})$. We show how to derive from $\wedge_{i=1}^m G_i$ the $(\mathcal{F}, \mathcal{I}(\mathcal{F}), \vec{\beta}, \mathcal{G})$-replacement of $\mathcal{I}(\mathcal{F})$. For C a clause of $\mathcal{I}_i$, let C' be its replacement. Derive G_i from $\wedge_{i=1}^m G_i$, by lemma 6.3. Derive
$$\neg G_i \vee DNF(g_i), \tag{6.8}$$
For each $\alpha \in g_i^{-1}(1)$, using lemma 6.23, derive
$$(\vee x^{\bar{\alpha}}) \vee C'. \tag{6.9}$$
Finally, apply $|g_i^{-1}(1)| \leq r/n$ cuts to derive C' from (6.8), (6.9). The depth of the derivation is bounded by $c + 6$, and the size is linear in $r \cdot |\mathcal{I}(\mathcal{F})|$.

2. ϕ_j is the result of a Merge of some ϕ_ℓ, for $\ell < j$. Since the replacement operation only affects literals (replacing them with DNFs), ϕ'_j is still the result of a Merge of ϕ'_ℓ. The cases of Unmerge and Weakening inferences are analogous.

3. $\phi_j = A \vee \neg A$. If A is not an auxiliary variable, then similar to the previous case, $\phi'_j = A' \vee \neg A'$ for A' the replacement of A, and ϕ'_j is a legal excluded middle axiom.

   Otherwise $\phi_j = y \vee \neg y$ for some auxiliary variable y appearing uniquely in (say) $\mathcal{I}_i$. In this case the replacement of ϕ_j is $\phi'_j = Y^1 \vee Y^0$, which is simply a partition of $DNF(g_i)$. Use the derivation of $\neg G_i \vee DNF(g_i)$ and cut it with G_i (which can derived from $\wedge_{i=1}^m G_i$ by lemma 6.3). The size of this derivation is $O(r)$ and the depth is at most c.

4. ϕ_j is a result of a cut inference. Specifically, $\phi_i = A \vee B$, where
$$\frac{A \vee C \qquad B \vee \neg C}{A \vee B}$$
If C is not an auxiliary variable, then similar to case 2 above, ϕ'_j is a legitimate cut on $(A \vee C)'$ and $(B \vee \neg C)'$.

   Otherwise $C = y$ for some auxiliary variable y appearing uniquely in (say) $\mathcal{I}_i$. If Y^1 is empty, then $(A \vee y)' = A'$ and hence $A' \vee B'$ can be derived by weakening. Dually, if Y^0 is empty, we can derive $A' \vee B'$ from $(B \vee \neg y)' = B'$. Hence assume both Y^0, Y^1 are nonempty, that is,
$$(A \vee y)' = A' \vee Y^1 = A' \vee (\wedge x^{\alpha^1}) \vee \ldots \vee (\wedge x^{\alpha^p})$$
$$(B \vee \bar{y})' = B' \vee Y^0 = B' \vee (\wedge x^{\gamma^1}) \vee \ldots \vee (\wedge x^{\gamma^q})$$
Where $A^1 = \{\alpha^1 \ldots \alpha^p\}$ is the set of satisfying assignments of Y^1, and $A^0 = \{\gamma^1 \ldots \gamma^q\}$ is the the similar set for Y^0. Derive for all $i = 1 \ldots p, j = 1 \ldots q$ the tautology:
$$Q(i,j) \stackrel{\text{def}}{=} (\neg(\wedge x^{\alpha^i})) \vee (\neg(\wedge x^{\gamma^j})) \tag{6.10}$$
This is a tautology because $\alpha^i \neq \gamma^j$. We leave it as an exercise to verify that any such derivation has size $O(n)$ and depth ≤ 6.

   For any $j \in [q]$, cut $A' \vee Y^1$ with $Q(i,j)$ for all $i \in [p]$. This gives the formula
$$P(j) \stackrel{\text{def}}{=} A' \vee (\neg(\wedge x^{\gamma^j})) \tag{6.11}$$

Finally, cut $B' \vee Y^0$ with $P(j)$ for all $j \in [q]$. This gives us $A' \vee B'$ as desired. The number of cuts is at most $|g_i^{-1}(1)|^2 \leq r^2$, and the depth is at most $6 + d(A' \vee B')$. $\square$

6.5.3 Proof of Theorem 6.17

Applying the previous theorem to our case, we can now provide a proof of theorem 6.17. We start with the following technical lemma.

Lemma 6.26

$$\mathbf{S_{H_4}}\left((\neg\mathbf{UO}[x_1,\ldots,x_n]) \vee (DNF(\mathbf{UO}[x_1,\ldots,x_n]))\right) \leq n^2.$$

Proof: For $i \in [n]$, let e_i^n denote the Boolean vector that is 0 everywhere except the i'th position, where it is 1. Similarly, $\bar{e}_i^n$ denotes the vector that is 1 everywhere except the i'th position, where it is 0. Under this notation,

$$DNF(\mathbf{UO}[x_1,\ldots,x_n]) = \bigvee_{i=1}^{n} (\wedge x^{e_i^n})$$

By lemma 6.5, it is enough to show a short resolution proof of the CNF

$$\mathbf{UO}[x_1,\ldots,x_n] \wedge \neg DNF(\mathbf{UO}[x_1,\ldots,x_n]),$$

where

$$\neg DNF(\mathbf{UO}[x_1,\ldots,x_n]) = \bigwedge_{i=1}^{n} (\vee x^{\bar{e}_i^n}).$$

Denote $D^n = \bigvee_{i=1}^{n} x_i$, and notice that

$$\mathbf{UO}[x_1,\ldots,x_n] = D^n \wedge \bigwedge_{1\leq i<i'\leq n} (\bar{x}_i \vee \bar{x}_{i'}).$$

Resolve $\vee x^{\bar{e}_n}$ with D^n to obtain D^{n-1}. For $1 \leq i \leq n-1$, resolve $\vee x^{\bar{e}_i^n}$ with $\bar{x}_i \vee \bar{x}_n$ to obtain $\vee x^{\bar{e}_i^{n-1}}$. Thus, with n resolution steps we have reduced our problem to the same problem on $n-1$ variables. So the total number of resolution steps needed to derive the empty clause is at most n^2. The proof is complete. $\square$

Proof [Theorem 6.17]: For $\mathcal{I}[K_h, \mathbf{1}]$ an implicit encoding, set

1. $\mathcal{F} = \{EQ(\bigoplus X(v), 1) : v \in V(K_h)\}$.
2. $\mathcal{I}(\mathcal{F}) = \{\mathcal{I}(EQ(\bigoplus X(v), 1)) : v \in V(K_h)\}$.
3. For $v \in V(K_h)$, g_v is the unique-or function over $X(v)$, and $G_v = \mathbf{UO}[X(v)]$.

The conditions of theorem 6.25 hold, and notice that $|g_v^{-1}(1)| = h-1$. By lemma 6.26, $\mathbf{S_{H_4}}(\neg G_v \vee DNF(g_v)) \leq h^2$, and we conclude that $\mathbf{Match(h)}$ has a depth $d+10$ proof of size $O\left(h^4 \cdot \mathbf{S_{H_d}}(\mathcal{I}[K_h, \mathbf{1}])\right)$, and the proof is completed. $\square$

6.6 Open Questions

What is the bounded-depth Frege proof size of random 3-CNFs ?

Appendix A

Proofs

A.1 Proof of Lemma 2.23

Let $\mathcal{F} \sim \mathbb{C}^{k,n}_{\Delta \cdot n}$ a k-CNF and let $\mathbb{G}(\mathcal{F})$ be the bipartite graph associated with $\mathcal{F}$. Let $r = \kappa n \Delta^{-\frac{1+\epsilon}{k-2-\epsilon}}$, where the constant κ will be determined later. Let BAD be the event that $\mathbb{G}(\mathcal{F})$ is not an (r, ϵ)-bipartite expander. We prove that the $\Pr[BAD]$ tends to 0 as n grows. Bound the probability of BAD by the probability that there exists a set $V' \subseteq V$, with $1 \leq |V'| \leq r$, such that $|N(V')| < (1+\epsilon)|V'|$ and then use the union bound to upper bound this probability.

Observe that there are $\binom{\Delta n}{i}$ possible sets $V' \subseteq V$ of size i, and there are $\binom{n}{(1+\epsilon)i}$ possible small sets of neighbors of V'. For a given set V' of size i, and a given set U' of size $(1+\epsilon)i$, the probability that $N(V') \subseteq U'$ is

$$P_i = \left(\frac{\binom{(1+\epsilon)i}{k}}{\binom{n}{k}}\right)^i \leq \left(\frac{(1+\epsilon)i}{n}\right)^{ki}$$

Let us bound the probability of the BAD event:

$$\begin{aligned}
\Pr[BAD] &\leq \sum_{i=1}^{r} \binom{\Delta n}{i} \cdot \binom{n}{(1+\epsilon)i} \cdot P_i \\
&\leq \sum_{i=1}^{r} \left(\frac{e\Delta n}{i}\right)^i \cdot \left(\frac{en}{(1+\epsilon)i}\right)^{(1+\epsilon)i} \cdot \left(\frac{(1+\epsilon)i}{n}\right)^{ki} \\
&\leq \sum_{i=1}^{r} \left[c \cdot \Delta \cdot \left(\frac{i}{n}\right)^{(k-2-\epsilon)}\right]^i \qquad \text{(A.1)}
\end{aligned}$$

The first inequality uses the well-known estimation $\binom{a}{b} \leq (\frac{ea}{b})^b$, and the second is true for the constant $c = c(k, \epsilon) = e^{2+\epsilon} \cdot (1+\epsilon)^{k-1-\epsilon}$ (recall that $0 \leq \epsilon \leq 1/2$).

The proof is split into cases:

Case 1: $\Delta \geq n^{1/10}$

$$\Pr[BAD] \leq \sum_{i=1}^{r} \left[c \cdot \Delta \cdot \left(\frac{i}{n}\right)^{(k-2-\epsilon)}\right]^i$$

$$\leq \sum_{i=1}^{r}[c \cdot \Delta \cdot (\frac{r}{n})^{(k-2-\epsilon)}]^i$$
$$\leq \sum_{i=1}^{r}[\frac{1}{2} \cdot \Delta^{-\epsilon}]^i$$

This geometric sum vanishes as n goes to ∞, because $\Delta > n^{1/10}$.

Case 2: $\Delta < n^{1/10}$

Notice that since $\Delta < n^{1/10}$, $k \geq 3$, and $\epsilon \leq 1/2$, then $r \geq n^{7/10}$. Split the sum of equation (A.1) into two:

$$\begin{aligned}
\Pr[BAD] &\leq \sum_{i=1}^{r}[c \cdot \Delta \cdot (\frac{i}{n})^{(k-2-\epsilon)}]^i && \text{(A.2)}\\
&\leq \sum_{i=1}^{\sqrt{n}}[c \cdot \Delta \cdot (\frac{i}{n})^{(k-2-\epsilon)}]^i && \text{(A.3)}\\
&+ \sum_{i=\sqrt{n}}^{r}[c \cdot \Delta \cdot (\frac{i}{n})^{(k-2-\epsilon)}]^i && \text{(A.4)}
\end{aligned}$$

Bound the first sum by the geometric sum

$$\begin{aligned}
(A.3) &\leq \sum_{i=1}^{\sqrt{n}}[c \cdot n^{1/10} \cdot n^{-\frac{1}{2}(k-2-\epsilon)}]^i\\
&\leq \sum_{i=1}^{\sqrt{n}}[c \cdot n^{1/10-1/4}]^i\\
&\leq \sum_{i=1}^{\sqrt{n}}[c \cdot n^{-1/4}]^i
\end{aligned}$$

As for the second sum, setting $\kappa(k,\epsilon) = \left(\frac{1}{2c(k,\epsilon)}\right)^{\frac{1}{k-2-\epsilon}}$, we get that $(\frac{r}{n})^{k-2-\epsilon} = \frac{1}{2c\Delta^{1+\epsilon}}$. Recalling $\Delta > 1$ (otherwise with high probability $\mathcal{C}$ is satisfiable) we get:

$$\begin{aligned}
(A.4) &\leq \sum_{i=\sqrt{n}}^{r}[c \cdot \Delta \cdot (\frac{r}{n})^{(k-2-\epsilon)}]^{\sqrt{n}}\\
&\leq \sum_{i=\sqrt{n}}^{r}[\frac{c \cdot \Delta}{2c\Delta^{1+\epsilon}}]^{\sqrt{n}} \leq n \cdot 2^{-\sqrt{n}}
\end{aligned}$$

Clearly, both sums vanish as n approaches ∞. lemma 2.23 is proven. □

A.2 Proof of Theorem 2.46

Proof: Let us construct a random bipartite over $V = [m], U = [n]$ as follows. For every $i \in m$, let $N(i) \stackrel{\text{def}}{=} \{j_{i1}, \ldots, j_{id}\}$, where all $j_{i\nu}$ $(i \in [m], \nu \in [d])$ are picked from $[n]$ independently and at random, identifying multiple edges. By lemma 2.29, if G is not an $(r, 3d/4)$-boundary expander, then it is not an $(r, \frac{7d}{8} - 1)$-expander. Thus, we wish to show that

$$\mathbf{Pr}\left[G \text{ is not an } (r, \frac{7d}{8} - 1) \text{ expander}\right] < 1,$$

for some $r = \Omega(n/d)$. Let p_ℓ be the probability that some particular $V' \subset V$, of cardinality ℓ, violates the expansion property. Then, clearly,

$$\mathbf{Pr}\left[G \text{ is not an } (r, \frac{7d}{8} - 1) \text{ expander}\right] \leq \sum_{\ell=1}^{r} m^\ell p_\ell. \tag{A.5}$$

A subset $V' \subset V$ of cardinality ℓ contradicts the expansion property if $|N(V')| < (1 + (\frac{7d}{8} - 1)) \cdot |V'| = \frac{7d\ell}{8}$. Using this definition, let us bound p_ℓ:

$$\begin{aligned} p_\ell &= \binom{n}{\frac{7d\ell}{8}} \cdot \left(\frac{7d\ell}{8n}\right)^{d\ell} && \text{(A.6)} \\ &\leq \left(\frac{8en}{7d\ell}\right)^{\frac{7d\ell}{8}} \cdot \left(\frac{7d\ell}{8n}\right)^{d\ell} && \text{(A.7)} \\ &\leq \left(\frac{\kappa d\ell}{n}\right)^{\frac{d\ell}{8}} \leq \left(\frac{\kappa dr}{n}\right)^{\frac{d\ell}{8}} && \text{(A.8)} \end{aligned}$$

Where κ is the constant $7e^7/8$, and e is the base of the natural logarithm. Plugging this into our previous bound, while setting $d = 16 \log m$ and $r = \frac{n}{2\kappa d}$, we get:

$$\begin{aligned} \mathbf{Pr}\left[G \text{ is not an } (r, \frac{7d}{8} - 1) \text{ expander}\right] &\leq \sum_{\ell=1}^{r} m^\ell p_\ell && \text{(A.9)} \\ &\leq \sum_{\ell=1}^{r} [m \cdot \left(\frac{\kappa dr}{n}\right)^{\frac{d}{8}}]^\ell && \text{(A.10)} \\ &\leq \sum_{\ell=1}^{r} m^{-\ell} && \text{(A.11)} \end{aligned}$$

This geometric sum is substantially smaller than 1 when $m > 2$. □

Bibliography

[Achlioptas ('00)] D. Achlioptas. Setting 2 variables at time yields a new lower bound for random 3-SAT. *In Proceedings of 32th STOC*. pp. 28-37 (2000).

[Aharoni and Linial ('86)] R. Aharoni, N. Linial. Minimal Non-Two-Colorable Hypergraphs and Minimal Unsatisfiable Formulas. In *J. of Combinatorial Theory*, Series A, Vol. 43, No. 2, (1986) pp 196-204.

[Ajtai ('88)] M. Ajtai. The Complexity of the Pigeonhole Principle. *In Proceedings of 29th FOCS*. pp. 346-355 (1988).

[Alekhnovich *et al.* A ('00)] M. Alekhnovich, E. Ben-Sasson, A.A. Razborov, A. Wigderson. Pseudorandom Generators in Propositional Proof Complexity. In *Proceedings of the 41st IEEE FOCS* 2000.

[Alekhnovich *et al.* B ('00)] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, A. Wigderson. Space complexity in propositional calculus. In *Proceedings of the* 32*nd STOC*, pages 358–367, 2000.

[Alekhnovich and Razborov ('00)] M. Alekhnovich and A. Razborov. Lower bounds for the polynomial calculus: non-binomial case. Manuscript, available at http://genesis.mi.ras.ru/˜razborov/misha.ps, 2000.

[Alon, Seymor and Thomas ('90)] N. Alon, P. Seymour, R. Thomas. A Separator Theorem for Graphs with an Excluded Minor and its Applications. In *Proceedings of 22nd STOC*. pp. 293-299 (1990).

[Alon, Spencer and Erdős ('92)] N. Alon, J. H. Spencer, P. Erdős. The Probabilistic Method. Wiley & Sons, 1992.

[Arora *et al.*('98)] S. Arora, C. Lund, R. Motwani, M. Sudan, M. Szegedy. Proof verification and intractability of approximation problems. *J. of the ACM*, vol. 45(3), May 1998, pp. 501-555.

[Arora and Safra ('98)] S. Arora, S. Safra. Probabilistic Checking of Proofs: a New Characterization of **NP**. *J. of the ACM*, vol. 45(1), May 1998, pp. 70-122.

[Beame *et al.* ('98)] P. Beame, R. Karp, T. Pitassi, M. Saks. On the Complexity of Unsatisfiability Proofs for Random k-CNF Formulas. Submitted.

[Beame and Pitassi ('96)] P.Beame, T. Pitassi. Simplified and Improved resolution lower bounds. In *37th Annual Symposium on Foundations of Computer Sciecne*, pp 274-282. Burlington, VT, October 1996. IEEE.

[Beame and Pitassi ('98)] P. Beame and T. Pitassi. Propositional proof complexity: Past, present and future. Technical Report TR98-067, Electronic Colloquium on Computational Complexity, 1998.

[Bellantoni *et al.* ('92)] S. Bellantoni, T. Pitassi, A. Urquhart. Approximation and small-depth Frege proofs. *In SIAM J. of Computing.* vol. 21 (1992), pp. 1161-1179.

[Ben-Sasson and Galesi ('01)] E. Ben-Sasson and N. Galesi. Space Complexity of Random Formulae in Resolution. In *Proceedings of the 16th IEEE Conference on Computational Complexity*, 2001.

[Ben-Sasson and Impagliazzo ('99)] E. Ben-Sasson and R. Impagliazzo. Random CNF"s are Hard for the Polynomial Calculus. In *Proceedings of the 40th IEEE FOCS*, pages 415–421, 1999.

[Ben-Sasson *et al.* ('00)] E. Ben-Sasson, R. Impagliazzo and A. Wigderson. Near-Optimal Separation of General and Tree-Like Resolution. Technical Report TR00-005, Electronic Colloquium on Computational Complexity, 2000.

[Ben-Sasson and Wigderson ('99)] E. Ben-Sasson, A. Wigderson. Short proofs are narrow - resolution made simple. In *Proceedings of the* 31*st ACM STOC*, pages 517–526, 1999.

[Blake ('37)] A. Blake. Canonical Expressions in Boolean Algebra. PhD Thesis, University of Chicago, 1937.

[Bonet *et al.* ('98)] M. L. Bonet, J. L. Esteban, N. Galesi, J. Johannsen. Exponential Separations between Restricted Resolution and Cutting Planes Proof Systems. submitted.

[Bonet and Galesi ('99)] M. L. Bonet, N. Galesi. A Study of Proof Search Algorithms for Resolution and Polynomial Calculus. In *Proceddings of the 40-th Symposium of Foundations of Computer Science*, pp. 422-431. 1999.

[Buss *et al.* ('99)] S. Buss, D. Grigoriev, R. Impagliazzo, and T. Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. In *Proceedings of the* 31*st ACM STOC*, pp 547-556, 1999.

[Beame *et al.* ('94)] S. Beame, R. Impagliazzo, J. Krajíček. , T. Pitassi, and P. Pudlák. Lower Bounds on Hilbert's Nullstellensatz and propositional proofs. In *Proceedings of the* 35*th IEEE FOCS*, pp 794-806, 1994. Journal version to appear in *Proceedings of the London Math. Soc.*.

[Buss and Pitassi ('97)] S. Buss, T. Pitassi. Resolution and the Weak Pigeonhole Principle. Typeset manuscript, to appear in *CSL97*.

[Buss and Turán ('88)] S.R. Buss, G. Turán. Resolution Proof of generalized Pigeonhole principles. In *Theoretical Comp. Sci.*, 62:(1988)311-317.

[Celoni *et al.* ('77)] R. Celoni, W.J. Paul, R.E. Tarjan Space Bounds for a Game on Graphs. In *Math. Systems Theory*, 10 (1977), pp 239-251.

[Chvátal and Szemerédi ('88)] V. Chvátal, E. Szemerédi Many Hard Examples for Resolution In *J. of the ACM*, Vol 35 No. 4, pp. 759-768.

[Clegg, *et al.* ('96)] M. Clegg, J. Edmonds, and R. Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 174–183, Philadelphia, Pennsylvania, 22–24 May 1996.

[Clote and Krajìček ('93)] P. Clote and J. Krajìček Arithmetic, Proof Theory and Computational Complexity, Oxford University Press, 1993.

[Cook ('74)] S. A. Cook. An Observation on Time-Storage Trade-Off. In *J. of Comp. and Sys. Sci.*, Vol 9 (1974), pp. 308-316.

[Cook and Reckhow ('79)] S. A. Cook, R. Reckhow. The relative efficiency of propositional proof systems. In *J. of Symbolic Logic*, Vol. 44 (1979), pp. 36-50.

[Craig ('57)] W. Craig. Linear Reasoning. A new form of the Herbrand-Gentzen theorem. In *Journal of Symbolic Logic.*, Vol 22 (1957), pp. 269–285.

[Davis *et al.* ('62)] M. Davis, G. Logemann, D. Loveland. A Machine program for theorem proving. In *Communications of the ACM*, 5:394-397, 1962.

[Davis and Putnam ('60)] M. Davis, H. Putnam. A computing procedure for quantification theory. In *Journal of the ACM, 7* pp. 201-215, 1960. Reprinted in *Automation of reasoning,* vol. 1, by J. Siekmann and G. Wrightson, ed. Springer-Verlag, Berlin, 1983.

[Dubois, *et al.* ('00)] O. Dubois, Y. Boufkhad, J. Mandler. Typical random 3-SAT formulae and the satisfiability problem. *In 11-th SODA* pp. 126-127 (2000).

[Esteban and Toran ('99)] J. L. Esteban, J. Toran. Space bounds for Resolution. In *Proceedings of the* 16*th* STACS, pages 530–539, 1999.

[Fu and Urquhart ('96)] X. Fu, A. Urquhart. Simplified Lower Bounds for Propositional Proofs. *In Notre Dame Journal of Formal Logic.* vol. 37, Number 4, Fall 1996. pp. 523-544. (1996).

[Goldreich *et al.* ('91)] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems In *JACM*, Vol. 38(1), pp. 691–729, 1991.

[Goldwasser *et al.* ('89)] S. Goldwasser, S. Micali and C. Rackoff. The knowledge compexity of interactive proof systems. In *SIAM Journal on Computing*, Vol. 18 (1), pp. 186-208, 1989.

[Grigoriev ('98)] D. Grigoriev. Tseitin's tautologies and lower bounds for nullstellensatz proofs. In *Proceedings of the* 39*th IEEE FOCS*, pages 648–652, 1998.

[Haken ('85)] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2–3):297–308, Aug. 1985.

[Hastad ('88)] A. Hastad. Almost optimal lower bounds for small depth circuits. In *Advances in Computer Research*, vol 5. *Randomness and Computation*, JAI press, Greenwich, CT, pp. 143-170.

[Hopcroft *et al.* ('77)] J.E. Hopcroft, W. Paul, L. Valiant. On Time vs. Space. In *J. ACM*, 24 (1977), pp. 332-337.

[Impagliazzo *et al.* ('97)] R. Impagliazzo, P Pudlák, J. Sgall. Lower Bounds for the Polynomial-Calculus and the Groebnes Basis Algorithm. Found at *Electronic Colloqium on Computational Complexity*, Reports Series 1997, Available at http://www.eccc.uni-trier.de/eccc/. Technical Report TR97-042.

[Janson *et al.* ('00)] S. Janson, Y.C. Stamatiou, M. Vamvakari. Bounding the unsatisfiability threshold for 3-SAT. *Random Structure and Algorithms* (17) 2 pp.118-116 (2000).

[Krajíček ('95)] J. Krajíček. Bounded Arithmetic, Propositional Logic, and Complexity Theory. *Encyclopedia of Mathematics and its Applications,* vol. 60. Ed. G.-C. Rota. Cambridge University Press, 1995.

[Krajíček ('97)] J. Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. of Symbolic Logic,* vol. 62(2), pp. 457-486 (1997).

[Krajíček ('99)] J. Krajíček. On the weak pigeonhole principle. Manuscript, 1999.

[Krajíček and Pudlák ('89)] J. Krajíček. P. Pudlák, Propositional Proof systems, the consistency of first order theories, and the complexity of computations. In *J. symbolic Logic*, 54(3), pp. 1063–79 (1989).

[Krajíček *et al.* ('95)] J. Krajíček, P. Pudlák, A. Woods. Exponential Lower Bounds to the size of Bounded Depth Frege Proofs of the Pigeonhole Principle. *In Random Structures and Algorithms.* vol. 7 pp. 15-39 (1995).

[Lipton and Tarjan ('79)] R.J. Lipton, R.E. Tarjan. A Separator Theorem for planar graphs. *SIAM J. Appl. Math. 36 (1979)*, pp. 177-189.

[Lund *et al.* ('92)] C. Lund, L. Fortnow, H. Karloff, N. Nisan. Algebraic methods for Interactive Proof Systems. In *J. of the ACM*, Vol 39 No. 2, pp. 859-868 (1992).

[Nisan and Wigderson ('94)] N. Nisan and A. Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49(2):149–167, October 1994.

[Pippenger] N. Pippenger Pebbling. *Technical Report*, IBM Watson Research Center.

[Pitassi ('96)] T. Pitassi. Algebraic propositional proof systems. In *Descriptive Complexity and Finite Models.*

[Pitassi *et al.* ('93)] T. Pitassi, P. Beame, R. Impagliazzo. Exponential Lower Bounds for the Pigeonhole Principle. *In Computational Complexity.* vol. 3 (1993) pp. 97-140.

[Pitassi and Raz ('01)] T. Pitassi, R. Raz. Regular Resolution lower bounds for the Weak Pigeonhole Principle. In *Proceedings of the 33rd ACM STOC*, 2001.

[Pitassi and Urquhart ('92)] T. Pitassi, A. Urquhart. The Complexity of the Hajós Calculus. In *Proceedings of the 33rd IEEE FOCS*, pp. 187-196, 1993,.

[Pudlák and Impagliazzo ('00)] P. Pudlák and R. Impagliazzo, Lower bounds for DLL algorithms for k-SAT, to appear in SODA '00.

[Raz ('01)] R. Raz. Resolution Lower Bounds for the Weak Pigeonhole Principle. Found at *Electronic Colloqium on Computational Complexity*, Reports Series 2001, Available at http://www.eccc.uni-trier.de/eccc/. Technical Report TR97-021.

[Raz and McKenzie ('98)] R. Raz, P. McKenzie. Separation of the Monotone NC Hierarchy. submitted.

[Razborov ('85)] A. A. Razborov. Lower Bounds on the monotone complexity of some Boolean Functions. Doklady Akad. Nauk SSSR 282, 1033-1037 (1985).

[Razborov A ('95)] A. A. Razborov. Bounded Arithmetic and lower bounds in Boolean complexity. In P. Clote and J. Remmel, editors, *Feasible Mathematics II. Progress in Computer Science and Applied Logic, vol.* 13, pages 344–386. Birkhaüser, 1995.

[Razborov B ('95)] A. A. Razborov Unprovability of Lower Bounds on Circuit Size in Certain Fragments of Bounded Arithmetic. *Izvestia of the RAN*, 59 (1), pages 201–222, 1995.

[Razborov ('96)] A. A. Razborov Lower Bounds for the Polynomial Calculus. Typeset manuscript.

[Razborov ('01)] A. A. Razborov Improved Resolution Lower Bounds for the Weak Pigeonhole Principle. Typeset manuscript.

[Razborov and Rudich ('94)] A. A. Razborov, S. Rudich. Natural Proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997. Preliminary version in Proc. of the 26th STOC, 1994, pp. 204-213.

[Razborov *et al.* ('97)] A. A. Razborov, A. Wigderson, A. Yao. Read-Once Branching programs, rectangular proofs of the pigeonhole principle and the transversal calculus. In Proc. of the 29th STOC, 1997, pp. 739-748.

[Robinson ('65)] J. A. Robinson A machine-oriented logic based on the resolution principle. In *Journal of the ACM*, vol. 12, pp. 23-41. Reprinted in *Automation of reasoning,* vol. 1, by J. Siekmann and G. Wrightson, ed. Springer-Verlag, Berlin, 1983.

[Selman ('95)] B. Selman, personal communication, 1995.

[Shamir ('92)] A. Shamir. IP=PSPACE. In *Journal of the ACM*, vol. 39(2), pp. 869-877 (1992).

[Torán ('99)] J. Torán. Lower Bounds for Space in Resolution. In *Proceedings of CSL 1999*, pages 362–373. 1999.

[Tseitin ('68)] G. S. Tseitin, On the Complexity of Derivation in Propositional Calculus. *In Studies in Constructive Mathematics and Mathematical Logic, Part 2*, pp. 115-125, edited by A. O. Slisenko, 1970 (reprinted in *Automation of Reasoning, vol. 2*, edited by J. Siekmann and G. Wrightson, Springer-Verlag, New-York, 1983, pp. 466-483).

[Urquhart ('87)] A. Urquhart. Hard Examples for Resolution. In *J. of the ACM*, Vol 34 No. 1, pp. 209-219, Jan. 1987.

[Urquhart ('95)] A. Urquhart. The Complexity of Propositional Proofs In *The Bulletin of Symbolic Logic*, Vol 1 No. 4 (1995), pp. 425-467.

[Yao ('82)] A. Yao. Theory and applications of trapdoor functions. In *Proceedings of the* 23*rd IEEE FOCS*, pages 92–99, 1982.