TIME AND SPACE IN PROOF COMPLEXITY

CHRISTOPHER BECK

A DISSERTATION

PRESENTED TO THE FACULTY

OF PRINCETON UNIVERSITY

IN CANDIDACY FOR THE DEGREE

OF DOCTOR OF PHILOSOPHY

RECOMMENDED FOR ACCEPTANCE

BY THE DEPARTMENT OF

COMPUTER SCIENCE

ADVISER: SANJEEV ARORA

JANUARY, 2017

1 Abstract

In this thesis we explore the limitations of efficient computation by way of the complexity of proofs.

One of the goals of theoretical computer science is to understand algorithms broadly – to identify meta-algorithms that may be useful in a variety of cases, understand how they work, when they work well, and when they don't.

For several important groups of algorithms, there is a relatively simple mathematical proof that the algorithm is correct for every input. Moreover, when the algorithm is run on any particular input, this proof specializes to produce a very simple combinatorial proof of the correct answer for that input. A simple example is a linear program, which may find an optimal point and give a dual solution proving the optimality, or a graph search algorithm which may fail to find a path between two points and yield a cut separating them instead.

Often times, self-contained proofs that the answer to some query is yes or no are just as important as the answer itself. When the algorithm is meant to identify problems in some system, or opportunities, the proof may actually represent actionable information in some domain. A good example is when a SAT solver is used to validate a system that is being designed. When the validation fails, the proof indicates where the problem is with the current design.

Proof complexity attempts to place limits on how efficient such algorithms can be by showing that these proofs must sometimes be very complex, growing rapidly as the problem scales up. More broadly, proof complexity attempts to understand the proving power of traditional formal systems for logic, and to corroborate hypotheses like P != NP.

We develop novel combinatorial techniques for studying proof complexity in several well-studied proof systems. We give substantial quantitative improvements on existing results, and also, develop a new method that allows us to study the time and space needed for a proof simultaneously and prove "tradeoff" lower bounds. We show that small reductions in space can sometimes lead to large increases in time, even in situa-

tions where the algorithm has large amounts of space to work with. Previously no such results were known in that situation.

Acknowledgements

I have to thank first and foremost my advisor Sanjeev Arora. Sanjeev always gave me thoughtful guidance. He always believed in me and offered unflagging support throughout my career. I have to thank Russell Impagliazzo, and I have also to mention what a joy it was to work with him on the problems that we thought about. Russell has incredible intuition and seemingly a special talent for making things as simple as possible. I can only hope that a fraction of that may have rubbed off on me, since I have no idea how one can learn or teach such things. I have to thank several others who I learned from and who helped me generously – Paul Beame, Jakob Nordstrom, Shachar Lovett, Toniann Pitassi, Sam Buss, Mohan Paturi, Alexander Razborov, Zeev Dvir, Avi Wigderson.

Finally, I have to thank my parents.

I would like to extend a special thanks to Tom and Jackie for their hospitality in the month of August, when I did much of the work of actually compiling this thesis.

2 Introduction

We assume that the reader has some mathematical background and so has some familiarity with the idea of proofs. However, the proofs which we study in proof complexity are not exactly the same as the proofs that students learn and that mathematicians produce, and we would like to clarify the difference right away.

For our purposes, a *proof* is a rigorous argument establishing a fact which is based in logic and independent of physical reality. (We'd like to avoid epistemological issues in this discussion.) Proofs are the currency of modern mathematics. *Rigor* means that the proof leaves no room for doubt and that the argument is ultimately rooted in the commonly accepted foundational principles of mathematics.

Proofs are often informally stated, and even in a professional context, proofs are often written in a way that places some of the burden of verification upon the reader. A rigorous proof is one which could in principle be written such that verification is made purely *syntactic*. This means that the claims of the proof could be encoded using mathematical symbols in the language of a *formal proof system*, and each statement of the proof could then be deduced from earlier statements or axioms of the proof system. In a formal proof, any deduction is required to be exact – rather than leaving room for the ambiguity of human language, each implication and condition must match exactly at the level of the symbols in order for the conclusion to be drawn. Then, each step of the proof could be checked by inspection without thinking of the meaning, or even mechanically. When a statement has been proved and verified in this way, we are absolutely certain that that the statement is true – this is the most compelling form of scientific evidence that we as humans can establish.

Besides this epistemic role, formal proofs and proof systems also play a role within mathematics itself. In logic and proof theory, proofs themselves are the object of mathematical study, with a goal of understanding the power and limits of different proof systems. An important chapter in the historical development of mathematics was the discovery of "Gödel's first incompleteness theorem", which shows that in any (finitely generated) proof system sufficiently powerful to reason about the integers, there are

statements such that neither the statement nor its negation can be proved. The system can't resolve the truth of these statements – they are "independent" of the axioms underlying the system. Subsequently, the technique of "forcing" was used to resolve certain questions about infinite sets in this way – within the commonly accepted foundations of mathematics, there are some natural questions like the Continuum Hypothesis which simply cannot be resolved, and moreover, this phenomenon is inevitable and cannot be "fixed" by adding more axioms or using a different system. This came as a great shock to many mathematicians of the day, who expected that anything that is true can be proven [50, 51].

Let us give a concrete example of a formal proof system.

Peano Arithmetic is a proof system for properties of the natural numbers [73]. We won't define it in detail, but we'll give a brief description of some of the syntax that it uses, to give the idea. In Peano Arithmetic, a number referred to in some step of the proof may be named by a *variable* such as x or y. A related number can be represented by a *numeric expression* using arithmetic operators, like $x \times y$ or $x + y$. When numbers are substituted for each variable, a numeric expression *evaluates* to a number.

Two numeric expressions can be compared to each other using *relational operators* like $=, <$, which represent *equals to* and *less than*. The combined expression is a *boolean expression*, which resolves to a *truth value*, either true or false. For example $x \times y < x + y$ might be true or might be false, depending on the values of x and y. A complex boolean expression can be formed from smaller boolean expressions using boolean connectives, such as $\vee$, $\wedge$, which represent logical-or and logical-and, or $\rightarrow$ which represents logical impliction. For example the expression $(x > y) \vee (y > x)$ is true when x and y are not equal. Boolean expressions can be negated with the unary operator $\neg$. For example the expression $\neg(x = y)$ has the same meaning.

Finally, variables in a boolean expression can be *quantified*. The two quantifiers are $\exists$ and $\forall$, which may be explained by example. The expression $x + 5 = y \wedge 10 \times y = z$ may be true or false depending on the values of x, y and z. When y is quantified, as in the expression $\exists y, x + 5 = y \wedge 10 \times y = z$, this expression is true or false depending

only on the values of x and z – it is true if there is *some* value for y which makes the condition true, and false if not. The expression $\forall x, x = y \vee x > 1$ is true if *every* number x is equal to y or greater than one.

A boolean expression is a *sentence* if all of its variables are bound to quantifiers. In this case, its truth value doesn't depend on anything else – it either represents a true property of the natural numbers or it doesn't.

A *proof* in Peano Arithmetic is a sequence of statements, each of which is either an axiom, or a deduction using two previous statements of the proof. (We won't describe here the axioms of Peano arithmetic, which are numerous.)

A *sound* proof system is one in which only true statements can be derived in this manner.

A *consistent* proof system is one which does not derive a contradiction. That is, for each sentence ϕ, at most one of $\phi, \neg\phi$ is derived.

A *complete* proof system is one in which for *every* sentence ϕ, exactly one of $\phi, \neg\phi$ is derived. [1]

An example of a famous mathematical conjecture which can easily be stated in the language of Peano Arithmetic is the *twin primes conjecture*. A twin prime is a number p such that both p and $p+2$ are prime. The conjecture is that there are infinitely many such primes. First, the property of a number n being prime can be coded as an expression

$$\forall x, \forall y, x \times y = n \rightarrow (x = 1 \vee x = n) \ .$$

In english, this can be read literally "for all x and all y, if x times y is n, then either x is 1 or x is n." Or more simply, "for all x which divides n, x is 1 or x is n.". We could

[1] From a model-theoretic point of view, the notion of *truth* is relative and *soundness* is not conceptually a property of a proof system – statements are only true or false relative to a *model*, and a proof system or *theory* might apply to many different models. In computer science, or in finite settings, this is less relevant and there may not be any interesting alternative models. It's common to see *soundness* discussed in the context of propositional proof systems, and also probabilistically-checkable proofs, protocols, and other settings. In logic, usually it's better to consider *consistency*.

Soundness aside, it is widely believed that Peano Arithmetic is *consistent*, and that the natural numbers themselves are a model of it. (Of course that is begging the question.) Gentzen gave several proofs of PA's consistency using weaker theories of arithmetic plus the principle of transfinite induction, but it's ultimately a foundational issue. Gödel famously proved that Peano Arithmetic is *not* complete.

then state the conjecture using the form, "for any n, there is a twin prime which is larger than n".

$$\forall n \exists p, p > n \wedge (\forall x \forall y, x \times y = p \rightarrow (x = 1 \vee x = p)) \wedge (\forall x \forall y, x \times y = p+2 \rightarrow (x = 1 \vee x = p+2))$$

This is a complete sentence – every variable which appears is quantified either with $\forall$ or $\exists$, and as a result it is either a true statement about the natural numbers or it isn't. Either there is a largest twin prime, or there isn't. However, no one knows scientifically (at time of writing) whether this statement is true or false, and no one knows any finite mechanical procedure which could determine this. One could imagine systematically trying different integers, but you can't try all the integers, and there's no way to know whether you have tried "enough". One could imagine searching mechanically for valid proofs of the sentence within Peano Arithmetic – if there is a proof, then it is true, but there is no bound on how large that proof could be. And even if it is true, there may be no proof. The only known way it could be resolved, is if a mathematician has a new insight into the prime numbers that allows a proof one way or the other.

Moreover, it follows from Gödel's work that there is no general algorithm that can resolve the truth of *any* question formed in this fashion. Any program, no matter how clever, must fail to compute the correct answer for some of the sentences, even if it is allowed unlimited time and memory. This is closely related to the undecidability of the Halting Problem, a foundational result in Computability Theory. In general, we know for quite fundamental reasons that the automated proving of theorems is hard in most circumstances.

There is a major distinction to be made, however, between the proving of theorems about all of the integers, statements about an infinite number of objects, and statements about a finite number of objects. If we ask instead a question like "is there a twin prime between 1,000,000 and 2,000,000", then of course there is a finite mechanical procedure to resolve it.

With few exceptions, mathematicians generally are focused day-to-day on proving

the former kind of statements within whatever field they work. But we are nevertheless very interested in efficient algorithms for the latter kind, and indeed, it is a question of fundamental importance how efficient they can be.

The *boolean satisfiability* problem is a computational problem which is representative of this latter kind. It is much simpler to define than Peano Arithmetic. An instance of the *boolean satisfiability* problem is an expression containing only variables and boolean connectives – the variables represent *boolean* values rather than integers here. The expression is *satisfiable* if there is *some* setting of the variables that makes it true. If you like, you can think that in this problem, every variable is implicitly quantified using $\exists$, and we want to know if the resulting sentence is true.

For example, the expression

$$(x \vee y) \wedge (\neg x)$$

is satisfiable, since we can take x to be false and y to be true.

The boolean satisfiability problem, often abbreviated SAT, is one of the fundamental NP-complete problems, and has been intensely studied for decades. Usually we are most interested in the scenario where there are n different boolean variables in the expression, and the overall expression has size similar to n – say linear, or polynomial in n. This problem can be solved by brute force search in time $2^n \cdot poly(n)$, but as n gets large this is infeasible. The P vs NP question is equivalent to, whether or not this problem has a solution which uses time only *polynomial* in n. Many researchers conjecture that P does not equal NP, and that this problem is in fact hard in the worst case. Impagliazzo and Paturi make a stronger conjecture – the *exponential time hypothesis* is that SAT requires time $2^{cn)}$ for some $c > 0$ [70].

The problem also has significant practical significance. Like all NP-complete problems, it is polynomially equivalent to many search and optimization problems which are important in practice, but it is also directly important for certain problems like validating a design of a complex digital circuit, and for "model checking" of software systems [44]. Automated theorem provers based on modern SAT solver technologies

are used in a variety of application areas, including planning and AI [75, 77, 115, 116]. There are SAT conferences, and a community of researchers who develop and maintain highly optimized SAT solvers and compete on an annual basis against a large selection of benchmarks from various application areas [104]. These "International SAT Solver Competitions" have been active since at least 1992.

Despite the importance of P vs NP, it is generally believed that we lack the mathematical tools to resolve the question right now. The known techniques which can show that a computational problem requires superpolynomial time in the size of the input generally are highly specific and can't be applied in the case of SAT.

To understand how proof complexity factors into this, we'd like to introduce the concept of a *witness*. Some computational problems have the property that, given an instance of the problem, x, there is another string y that convinces you of what the answer is. For instance, in the case of SAT, when I have a formula x which is satisfiable, the witness y is simply the satisfying assignment to the variables. Moreover, any supposed witness to a formula can be verified very quickly and deterministically, and any satisfiable formula has such a witness, while unsatisfiable formulas never have a witness. The existence of such witnesses for the "affirmative" case of the problem is in fact one way to *define* NP – any problem in NP has an efficient witness scheme like this [86].

It's natural to ask, are there also witnesses for the negative case of the problem? Is there always a short string that convinces you that a formula is *unsatisfiable*? This is called the NP vs. coNP problem. It is conjectured that there are no such short witnesses for unsatisfiable formulas – that in the worst case, the only "proofs" are essentially transcripts of the brute-force algorithm.

However, an obvious candidate for such witnesses is the set of strings corresponding to *formal proofs* in some deductive proof system. When such a proof system is sound and complete, then its proofs can serve as negative witnesses. In the case of boolean statements, where each sentence is inherently finite and thus checkable, it is easy to construct sound and complete proof systems. However, no such system is known where every true sentence has a *short* proof. If such a system exists, then NP = coNP, con-

tradicting our assumptions. This observation was originally made by Cook and Reckhow [47].

Historically, this has been one way that researchers have motivated proof complexity, by way of its connection to NP vs coNP. However, it seems fair to say that proof complexity is not realistically an attempt to solve this problem, at best it would build intuition for this, or disprove our conjectures. The reason is that SAT may have an efficient witness system for negative instances which has no connection to deductive proof systems. Thus, even if we made a major breakthrough and proved strong exponential lower bounds for all known propositional proof systems, it would not imply directly that NP != coNP, nor would it lead in any obvious way necessarily to a proof of such. Proving NP != coNP would imply P != NP, and any potential "avenue" towards a proof is speculative at best.

A different way to motivate the study of proof complexity lower bounds from the point of view of Computer Science, is simply to think of them as lower bounds for "classes" of algorithms rather than general lower bounds against computation. In many cases, algorithms quite naturally yield proofs, stemming directly from their operation.

- A linear programming algorithm may produce a solution to a linear program, and a dual solution showing that it is optimal.
- An algorithm for st-connectivity in graphs may fail to find a path, and produce an explicit cut separating two vertices instead.
- An algorithm for solving systems of polynomial equations may prove that there is no solution, by deducing an explicit algebraic contradiction, e.g. $1 = 0$, from the system.
- A pure backtracking algorithm for SAT may prove that a formula is not satisfiable by producing a decision tree, such that if any path is followed from the root to a leaf and the variables of the formula assigned accordingly, it simplifies immediately to "false". (This can be confirmed in time similar to the size of the formula and the size of the decision tree.)

This phenomenon is somewhat broader than SAT and NP. One way to explain why we might expect an algorithm to yield a proof like this is that, we usually only use algorithms that we know will work and give us the right answer. Generally, this means that we actually have a mathematical proof of the algorithm's correctness, for all n, which might be formalizable in some system like Peano Arithmetic. Oftentimes, the proof may not actually need the full power of Peano Arithmetic – it may be able to use a small subset of that power. It may be possible that it can be formalized in one of a series of special sub-systems called *Bounded Arithmetic* [37, 73]. Bounded Arithmetic is one way of restricting Peano Arithmetic that forces it to create "effective" or "constructive" proofs. In cases where the correctness of an algorithm can be proved in such a system, generally it means that for each n, a *formula* corresponding to the correctness of the circuit corresponding to the algorithm can be proved to be true for all inputs of size n using a short *propositional proof*, e.g. polynomial in n.

The bounded arithmetic proof (which is finite) essentially serves as a template for each member of this (infinite) series of *propositional* proofs. Exactly which proof system is used for them, depends on which Bounded Arithmetic the original proof used. Thus, just as Turing Machines may be thought of as a way of specifying *uniform* sequences of circuits, built from a common template, Bounded Arithmetic is a way of specifying uniform sequences of propositional proofs. To the extent that Bounded Arithmetic can formalize many arguments in combinatorics and algebra, we can expect that many common algorithms will "yield proofs" in some sense.

This is all that we will say about proof theory, arithmetic, and general motivations of proof complexity – proof theory and arithmetic will not be directly relevant to our results or subsequent discussion. Instead, in this thesis we are generally concerned with *combinatorial* and *algebraic* techniques which have applications in proof complexity. In the next section we will formally define two well-studied proof systems directly relevant to the thesis, *resolution* and *polynomial calculus*.

2.1 Resolution

In the effort to study SAT, researchers have often considered restricted forms of SAT and attempted to resolve whether they are as hard as the general problem, or easier. Generally, this means that instead of considering all possible boolean expressions, we consider only those with a special form. One such form is called *conjunctive normal form* or "CNF".

Given a set of boolean variables $x_1, \ldots, x_n$, we say that a *literal* is an expression which is either a variable or its negation. A *clause* is a disjunction of literals, such as, $x_1 \vee \neg x_2 \vee x_7$. A *CNF* is a conjunction of clauses. (An "and of or's".)

It is well known that the general case of SAT can be reduced in polynomial time to the special case of CNF satisfiability ("CNF SAT"), that is, any boolean expression can be converted efficiently to a related expression in the conjunctive normal form. In fact, in this reduction, each clause of the CNF may be assumed to contain at most three variables. SAT in this form is often called "3SAT", and has been intensely studied [86].

Given a formula in CNF, a natural point of view to take is that it is a collection of "assumptions", which are each small clauses, and the satisfiability problem then asks if they are logically consistent. *Resolution* is a very simple proof system which operates *only* with clauses like this. It is usually said to have originated with work of Davis and Putnam [53, 52]. Given a collection of "axiom" clauses, the *resolution rule* permits to derive a new clause from two clauses fitting the following schema:

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

We say that the variable x is *resolved* in this instance of the resolution rule.

A *resolution proof* consists of a sequence of derivations of this form, which each statement is either an axiom or derived from earlier statements.

A *resolution refutation* is a derivation of a contradiction from some set of axioms. The contradiction is usually represented by the "empty clause" or by the symbol $\perp$, and may be derived from any pair of unit clauses corresponding to a variable and its

negation:

$$\frac{x \qquad \overline{x}}{\bot}.$$

It is straightforward to see that the resolution rule is logically *sound*, that is, if a truth assignment satisfies $A \vee x$ and it satisfies $B \vee \overline{x}$, then it must also satisfy $A \vee B$. By induction, if the contradiction is ever derived, then the original axioms clauses cannot be simultaneously satisfied.

It is also straightforward to see that resolution is *complete*, that is, the contradiction can always be derived whenever a collection of unsatisfiable axioms is given. A *trivial* refutation may always be constructed with size $\approx 2^n$. This may be proved for instance by induction on the number of variables.

2.2 Polynomial Calculus

Another way of approaching proofs of unsatisfiability is via algebraic techniques.

One of the oldest topics in mathematics is the solutions to systems of polynomial equations. A fundamental result in the area is *Hilbert's Nullstellensatz*. This theorem provides a basic connection between the collection of solutions to a set of polynomial equations (in multiple variables), and the set of polynomials which can be derived from those polynomials by adding and multiplying them (the *algebraic ideal* generated by the polynomials).

It's not too difficult to see that 3SAT instances can be transformed into collections of polynomial equations over a field. Intuitively, we can convert our boolean variables into variables over the field, by associating true and false with one and zero respectively. We can add an equation $x_i^2 - x_i = 0$ for each variables, which prohibits it from taking other values. Then we can translate each initial clause to a small polynomial. For instance given the clause $x_1 \vee x_2 \vee \overline{x_3}$, we could write $(1-x_1)\cdot(1-x_2)\cdot x_3 = 0$. Since each factor is zero-one valued (because of our earlier equations), the product is zero if and only if at least one of them is actually zero. Then, the collection of polynomials is satisfiable

over the field if and only if the original clauses were.

The *polynomial calculus* proof system, originally introduced by Clegg, Edmonds and Impagliazzo [45], manipulates expressions of the form $p = 0$ for p a polynomial over a field. The allowed derivations are adding equations, and multiplying equations by a variable or constant. (Since, these are the closure properties of an ideal.)

$$\frac{p = 0 \qquad q = 0}{p + q = 0} \ .$$

$$\frac{p = 0}{x_i \cdot p = 0} \ .$$

Clearly, if an assignment satisfies the assumptions of either of these rules, it satisfies the conclusion, so the proof system is sound. There are several ways to see that polynomial calculus is *complete* – one might appeal to the Nullstellensatz, or simply observe that a resolution refutation can be simulated via polynomials [17].

For purposes of establishing the completion of polynomial calculus, it doesn't matter if this simulation is efficient or not. However, it easy to see that in general it isn't, because clauses must map to products, and literals may sometimes map to binomials $(1 - x_i)$. In polynomial calculus, all polynomials are represented in a fully expanded form, so there is (naively) an exponential blow-up in the representation of the clause as a polynomial.

For this reason, researchers tend to study instead a common extension of Resolution and Polynomial Calculus called *Polynomial Calculus Resolution* or *PCR*. In this version of polynomial calculus, we introduce for each boolean variable x_i also a variable $\overline{x_i}$, and a constraint that $1 - x_i = \overline{x_i}$. In this system, each clause of a resolution proof may be mapped to a *monomial* in polynomial calculus, avoiding the exponential blow-up. If polynomial has few terms, then for any i we can exchange the x_i for $\overline{x_i}$ when we like using the axiom, incurring small blow-up. If a polynomial has many terms, it implies that the proof was already large to begin with, and hence not terribly efficient anyways. It's easy to see that the new system *efficiently* simulates both Resolution and naive

Polynomial Calculus, so the added robustness makes it an attractive target for study.

A basic question is, does this reformulation as polynomials over a field give any advantage for SAT, and when? Naively, it appears much more expressive than resolution, since for instance, if you work over a field $\mathbb{F}_2$, you can succinctly represent parity as a short proof line, or if you work over another prime, you can efficiently represent MOD_p constraints. More generally, inconsistent systems of linear equations can be refuted efficiently in the same system as the prime that they work over, since one can essentially mimic the short Gaussian elimination proof of this fact.

Despite their differences, the theories of Resolution and Polynomial Calculus are historically linked. In a well-known paper of Ben-Sasson and Wigderson [24], it was shown that the "Size-Degree tradeoff" technique developed for Polynomial Calculus also translates naturally to Resolution, in the form of a "Size-Width tradeoff". In our modern understanding, we now think of these systems as somehow closely related.

2.3 Complexity of Proofs

Now we'd like to talk about, how precisely do we define when a proof is complex, in a way that corresponds usefully to algorithms and the mathematics that we wish to study. Particularly, we want to define analogs of time and space complexity associated to a propositional proof. This line of research was essentially initiated by Cook and Reckhow [47, 48].

There are a few different ways that this could be formalized, and the definitions that we select are quite important. In this section we explain what we study in this thesis and give a rationale.

The issue is that there are basically two ways to look at what a proof complexity lower bound is doing.

- It is giving a lower bound for the resource usage of a "class" of deterministic algorithms.
- It is giving a lower bound for the resource usage of a *specific* nondeterministic

algorithm.

Historically, researchers tend to be motivated most by the first in forming a definition, but post hoc, it seems that the definitions that get traction tend to have simple explanations in terms of both of these things.

Generally speaking, we adopt measures of proof complexity that will correspond closely to the Time or Memory used by the *verifier* algorithm, given the proof. We don't require the verifier to *find* the proof – showing that it is hard to *find* the proofs is studying a different aspect of algorithms and complexity.

For this reason, we generally associate the *Size* of the description of the proof with the *Time*. In resolution, even further, we tend to focus just on the *number of clauses*. A clause can only contain at most n variables, so this can't be much different from the representation size of the entire proof. Resolution is simple in the sense that Size thus corresponds roughly to "number of lines" in the proof. In Polynomial Calculus, this is not true in a satisfactory sense – a single polynomial may have an exponential number of monomials. Thus, we tend to count the total number of monomials in the proof instead. Since for each variable we have an axiom $x_i^2 = x_i$, a proof never needs to contain large powers of any variable, and if it did, it could always be restructured so that they are eliminated. So, an individual monomial, like a clause, is never very large anyways.

It's clear that this definition also provides lower bounds against the time of an algorithm *searching* for a proof in the proof system, since it is actually generating this proof. In practice, surely it would have to be larger than this, since the search will backtrack inevitably many times. What is most interesting for us is that often, we can prove proof size lower bounds which are comparable to the running time of a trivial algorithm for SAT, so in some sense, we can't actually be losing too much by focusing on proof size.

Defining proof space is a little more difficult. Naturally, we'd like to adopt a definition that will allow us to bound the memory usage of a proof search algorithm. If a formula is hard to prove, we might hope to show that not only does the search algorithm have to consider a large number of clauses / proof lines, but also that it must hold a large

number of them in memory while it executes.

However, if the point of view we are taking is that the space should correspond to the memory usage of the *verifier* in a nondterministic algorithm, then this seems problematic. Since, for an NP language, the verifier can always be assumed to be in logarithmic space. And for e.g. verifying a resolution proof given as input, the verification should also be doable in logarithmic space if one has random access to the proof, since the verifier only needs a counter to remember what proof line they are currently at, and some space for pointers so that it can find the referenced clauses and confirm the derivation of that line.

In Resolution, the fundamental measure of proof space is *clause space*, introduced by Esteban and Toran [56].

Definition 2.1. The *proof DAG* associated to a proof is the directed acyclic graph whose vertices are the lines of the proof, such that each proof line has an arc leading to it from all of the proof lines it directly relies upon.

Definition 2.2. A *topological sort* of a DAG is a linear ordering of the DAG consistent with each arc.

Definition 2.3. Given a topological sort of the proof DAG of a resolution derivation, say that the *clause space* is the maximum over all *time points* (positions in the linear ordering) of the number of arcs crossing the cut corresponding to "vertices before this time" and "vertices after this time".

The clause space of a proof is the minimum over all toplogical sorts of this value.

This definition makes a lot of sense from the point of view of a *deterministic* search algorithm. If a search algorithm is generating a proof, and not keeping the whole proof in memory, that means that some of the clauses get used at the beginning, but discarded later. If a proof is being generated in some order, the clause space gives a lower bound on how many clauses must be held in memory at once, since if there is an arc crossing this cut, it means that a clause was derived before some point in time, but is still needed after that point in time, so it must still be in memory that point, if this proof is ultimately generated.

Note that the definition allows that a clause may be evicted from memory, and then an identical clause later rederived. This is essential especially in work that considers sublinear proof space, i.e. less than enough to hold the entire set of axioms at once. In such a scenario, useful proofs require "random access" to the axiom set at various points in the derivation. When considering superlinear proof space, i.e. enough to hold the entire set of axioms at once, it's not as essential, but if a clause could only be derived once in the entire proof, that would unreasonably cripple the model.

From the point of view of the *nondeterministic verifier*, one way to look at this is that the verifier has only *one-way access* to the proof string when it is verifying the proof. In that case, the proof cannot be verified in log-space anymore using pointers, since it can't go back and look at clauses that it already processed long ago. Instead, while verifying the proof, parts of the proof string that are needed for later need to be copied into the work tape of the verifier, and they can only be removed when they aren't needed anymore. One can assume that, the information telling the verifier when a clause is no longer needed is provided as an *annotation* to the lines of the proof. This does not meaningfully increase the size of the proof string, and it allows the naive verifier to achieve optimal memory usage, essentially corresponding to the clause space.

While there's no obvious reason to assume that the nondeterministic verifier has only one-way access to the proof tape – that is usually only assumed for space-bounded computation, rather than in the context of NP – this does seem to produce the most interesting and useful proof complexity measures. And it does make sense when the real focus is a deterministic search algorithm – in that case, the proof tape isn't really there anyways, and the nondeterminism is standing in for decisions that in practice are being made by complex heuristics [15, 75, 77, 115, 116].

A final way to motivate clause space is that ostensibly, the definition appears similar to the definitions of other important graph-theoretic complexity measures, such as *bandwidth* or *pathwidth*. Intuitively, a proof requires large clause space relative to its size, when the *proof DAG* is highly connected, in the sense of it being difficult to give a hierarchical decomposition of the graph without ever making large cuts. So, space com-

plexity in proofs can also be understood as a measure of the internal complexity of the proof's *graph-theoretic structure*, without necessarily giving reference to a machine[2].

Note that just as with proof size, we are potentially giving up a factor n by treating each clause as of size one, rather than of size corresponding to the number of variables. In our work, we generally don't care too much about this, since we will usually consider space much larger than n. It has only been recently that lower bounds on any of these space measures could really be proved – somehow, clause space is the natural place to start. Later work has considered a measure called *total space*, which really focuses on the automata-theoretic view of a proof, and considers the space to be the maximum size of the representation of a configuration of the verifier during the course of verifying the proof – that is, for each clause in memory at each time step, you have to pay one for each variable in that clause, rather than just one for the whole clause. We only mention this in passing – we won't actually study that measure in this thesis.

In Polynomial Calculus, the measure of *size* is the sum over all polynomials in the proof of the number of monomials. Similar to clause space, we study *monomial space*, that is, the same as clause space, except that for each polynomial in memory it counts for an amount of memory equal to the number of monomials.

[2]This may sound like a superficial observation, but in my opinion it is actually a useful technical insight, at least at a high level. Intuitively, a time-space lower bound is supposed to show that short proofs of some sentence require large space. By this analogy, what one should actually attempt to show formally is that short proofs require *large cuts* in some sense. Since there is a well-understood connection between cuts and flows in graph theory, this leads to the idea that we should try to construct a useful *flow* in the proof DAG of a short proof, in order to prove a time-space lower bound. This is more closely related to the bottleneck-counting argument of Haken [62, 63, 108], and much subsequent research. See section 3.6 for an illustration.

2.4 Contents of this thesis

In the next two chapters, we show new *time-space tradeoff results* in Resolution and in Polynomial Calculus, respectively.

A *time-space tradeoff* result shows, informally, that a computational model exhibits a *true* time-space tradeoff in regards to solutions of a given problem. There are two components – an *upper bound*, showing that an efficient solution exists which uses a lot of memory, and a *lower bound* showing that if the memory is reduced, then any solution must require significantly more time. These results are interesting because in general models of computation, it's a major open quesiton whether this ever happens. This is in some sense the essence of the L vs. P problem for instance. [3]

The results in these chapters represent significant qualitative improvements over the previous state of the art. Prior to our work, there were time-space tradeoff results known in resolution, but only when the model is restricted to sublinear amounts of space, which is somewhat artificial in the context of proof complexity. Moreover, those lower bounds relied on very different techniques from the other known lower bound techniques in proof complexity. Our results yield tradeoffs for all space bounds from linear all the way up to exponential, and produce a super-polynomial blowup in time when the space is reduced. Our techniques our novel, but they build naturally on existing lower bound arguments from proof size, so we end up with a more unified theoretical understanding of intractibility in proof complexity than we had previously.

In the final chapter, we consider the worst-case resolution proof size, in connection to the Strong Exponential Time Hypothesis. We prove a version of this hypothesis for regular resolution, and get a substantially improved bound for general resolution than was previously known.

[3] In some cases, researchers are able to show nontrivial time-space lower-bounds in which the upper bound is not known or even suspected to be true. These results may still be very interesting, but our results have the stronger character of proving formally that time and space cannot always be simultaneously optimized in well-established models of computation.

3 Time Space Tradeoffs in Resolution

We give the first time-space tradeoff lower bounds for Resolution proofs that apply to superlinear space. In particular, we show that there are formulas of size N that have Resolution refutations of space and size each roughly $N^{\log_2 N}$ (and like all formulas have Resolution refutations of space N) for which any Resolution refutation using space S and length T requires $T \geq (N^{0.58 \log_2 N}/S)^{\Omega(\log\log N/\log\log\log N)}$. By downward translation, a similar tradeoff applies to all smaller space bounds.

We also show somewhat stronger time-space tradeoff lower bounds for Regular Resolution, which are also the first to apply to superlinear space. Namely, for any space bound S at most $2^{o(N^{1/4})}$ there are formulas of size N, having clauses of width 4, that have Regular Resolution proofs of space S and slightly larger size $T = O(NS)$, but for which any Regular Resolution proof of space $S^{1-\epsilon}$ requires length $T^{\Omega(\log\log N/\log\log\log N)}$.

3.1 Introduction

For many modern SAT solvers, memory use is as much a bottleneck as time. Earlier DPLL-based SAT-solvers used very little memory, just needing to keep track of the stack in recursion. However, a major reason for the improved performance of recent SAT solvers is the inclusion of clause learning [75], which adds a large number of derived clauses to the input clauses. Clause learning uses considerable extra space corresponding to the number (and size) of derived clauses that are active at any one time. This creates opportunities for finding smaller refutations of unsatisfiable formulas, but at the cost of potentially requiring huge amounts of memory. Balancing these two factors is an art that determines the success of the SAT-solver. (See e.g, [116] for a discussion of how one successful SAT-solver handles this.)

This raises the question of whether such tradeoffs between total time and memory is an inherrent limitation, or just an artifact of the known algorithms. To make this question precise while also being general enough to handle the wide variety of SAT-solving algorithms, we can use the well-known correspondence between these algorithms and the Resolution proof system. All of the DPLL-based SAT solvers, when run on unsatisfiable formulas, implicitly or explicitly find resolution refutations of the input formulas. Thus, the minimum size of such a refutation is a lower bound on the time taken by any such algorithm. This observation has been part of the motivation for proof complexity studies of resolution for many years. More recently, an analogous measure of the *space* of a resolution proof has been introduced, which lower bound the number of derived clauses that must be remembered throughout the algorithm, and hence bounds the memory requirements of resolution-based SAT solvers.

If we use a pure backtracking approach to SAT-solving, like many of the earlier generation of SAT-solvers did, the proofs generated have at most linear space. Thus, we cannot hope to prove a super-linear space lower bound in isolation. Our goal is instead to show that the small space of these proofs come at the expense of being substantially larger than proofs using more space, i.e., to give time-space tradeoffs for proof complexity. There has been substantial work devoted to understanding space in proof

complexity, proving almost linear lower bounds on the space required, and giving sharp time-space tradeoffs for refutations. However, previous work always hit a barrier at linear space. Until this current paper, no known time-space trade-off lower bound for proof complexity was meaningful when the space allowed was greater than that of the input formula. Since SAT solvers have ample memory to hold the input formulas, this meant that such work was not of direct benefit to understanding time-memory trade-offs for SAT-solvers. Ben-Sasson (see [79, 81]) posed the question of whether such trade-offs existed as follows: do all CNF formulas have resolution proofs of linear space that are at most polynomially larger than the resolution proof length possible when space is unrestricted?

We give a strong negative answer to this question. We give explicit CNF tautologies of size N that have proofs of size $T_{UB} \leq N^{O(\log_2 N)}$ (and hence space at most T_{UB}), but when space is restricted to $S_{LB} \leq T_{UB}^{1/2}$, any resolution proof requires size $T_{LB} \geq (T_{UB})^{\Omega(\log\log N/\log\log\log N)}$. In other words, restricting memory to any polynomial in the input formula size has a super-polynomial cost in terms of time the SAT-solver will take. Our formulas are Tseitin graph tautologies for complete bipartite graphs connected along a path and so are totally explicit, as are the proofs in the upper bound on T_{UB}. In fact, [3] has shown that proofs of such tautologies are derivable by standard SAT-solving techniques in time proportional to T_{UB}, so the trade-off applies to actual SAT-solvers, not just the theoretical optimal SAT-solver.

We obtain an even stronger tradeoff for Regular Resolution, a subclass of Resolution proofs that includes most natural Resolution proofs. Using Tseitin formulas on a family of grid graphs, we show that for any integer m, there is a size $N = O(m^4)$ formula with unbounded space regular resolution proof size at most $T_{UB} = poly(m)2^m$ so that any space $S_{LB} \leq 2^{m(1-\Omega(1))}$ proof requires size $T_{LB} \geq (T_{UB})^{\Omega(\log\log m/\log\log\log m)}$. This is an improvement over the previous result in that space restrictions provably cost in terms of size even for (weak) exponential amounts of space. Furthermore, the lower bound holds for space almost equal to the upper bound on total size. Finally, the tautologies where we prove this lower bound are constant width.

These results show that restricting space can increase size by more than a polynomial amount. It seems possible that space restrictions could have an exponential cost; however, that would require a different kind of tautology than the ones we consider. The quasipolynomial time-space tradeoff lower bound we achieve for Resolution proofs of these Tseitin tautologies is qualitatively not far from optimal. In particular, the Tseitin tautologies to which our bounds apply have Regular Resolution refutations of space $O(\log T_{UB} \log n)$ and size $T_{UB}^{O(\log n)}$. A recent result [8] shows a general purpose resolution based SAT algorithm which finds this proof as well, answering a question of [3]; see also [11]. [8] also gives an algorithm for high space which finds a refutation matching our high space refutation – this is also achieved by the algorithm of [3].

3.1.1 History and Related Work

Much initial work on proof space concentrated on the space required by Resolution proofs as a parameter on its own. Early work [56] showed that every unsatisfiable formula has a Resolution refutation in which the total space required is at most the number of variables (plus one), and the focus moved to finding matching space lower bounds. Atserias and Dalmau [9] showed that Resolution space is at least Resolution *width*, the length of the longest clause in any Resolution proof, for which $\Omega(n)$ lower bounds were already known for many formulas because width lower bounds are the most widely used method for proving Resolution proof size lower bounds [24]. (Despite its utility, Resolution width alone can be far from characterizing Resolution space [78].)

Lately, there has been great theoretical interest in understanding the interplay between the resources of Resolution proof length (time) and space [79, 21], and in showing tradeoffs between them [22]. For example, Nordström [80] shows that there are formulas with linear size proofs but for which reducing the total space used by a constant increases the length required to exponential[4]. At higher space levels, Ben-Sasson and Nordström [22] recently showed the existence of families of formulae that have linear size (and hence linear space) proofs, but which require exponentially large proofs

[4]This was claimed in [66] which used a more complicated approach, but this has been retracted as has the claim that determining the minimum space required for Resolution refutations is PSPACE-hard.

when the space is constrained to be $O(n/\log n)$.

3.1.2 Overview of our techniques

Many of the earlier papers on bounds for sub-linear space proofs modified arguments taken from time-space tradeoffs for pebbling games. We also use arguments based on time-space tradeoffs for straight-line programs such as [109, 89]. In such a bound, one typically defines a notion of "progress", and divides up the program into intervals called "epochs". Then one shows that, starting from any given point in the program, on a random input, one is unlikely to make significant progress in a short amount of time. The conclusion is that either there are many values that are in memory at those points, or the length of epochs is large, giving a time-space trade-off.

We follow a similar strategy, combined with a "bottleneck counting" argument as introduced in [62]. We consider the time steps to be the derived clauses in order, and divide these into equal sized intervals, or *epochs*. We consider the sets of clauses in memory at the start and end of epochs. We use a simple measure of the complexity of clauses, akin to those in [62, 24], the number of vertices of the graph G associated with the input clauses of the Tseitin formula that are required to derive them. The input clauses have complexity 1, the final contradiction has complexity $|V(G)|$, and each resolution step can at most double the complexity. Thus, clauses of each approximate (to within factors of 2) complexity between 1 and $|V(G)|$ must occur in the proof. The current complexities of clauses in memory is our measure of progress.

We show that clauses of intermediate complexities involve many variables and are hence intuitively are "unlikely" to be false, and we show that the likelihoods of very different clauses are in some sense independent. Because of the small space, the clauses in memory at the frontiers of epochs are "bottlenecks" ; it is unlikely that clauses of many distinct complexities are in memory at these times. If this does not happen, we must make a great deal of progress during some epoch, in that the largest complexity of a clause in memory has increased substantially. We then apply the argument recursively to that epoch.

While this intuitive picture is the same for the two lower bounds, the techniques used to formalize them are quite different. For general resolution, we formalize "likelihood" by looking at the proof after a random restriction. We show that if we apply a restriction to a Tseitin formula on a suitable graph G, then the probability that a clause has medium complexity after the restriction is quite small, and the probability that k clauses simultaneously have different intermediate complexities is a $\Theta(k)$-th power of this small probability. We use this to show that, with high probability, there are not many distinct complexities in memory at the frontiers of epochs after the restriction. Thus, after the restriction, there is one epoch that contains clauses of many approximate complexities within a large interval. We then use a recursive version of the same argument to show that this is also unlikely.

In the case of regular resolution, we define a probabilistic process that works its way backwards from the contradiction to one of the input clauses. It always visits clauses of each approximate complexity. "Likelihood" is then the probability of being visited in this process. We show that this probability is small for intermediate complexity clauses, and, with a few exceptions, uncorrelated for clauses of very different complexities. A bottleneck counting argument then shows that it is unlikely that this process visits many different clauses of distinct complexities that are each in memory at the frontiers of epochs, and thus must almost always visit a large interval of complexities within some epoch. The argument is repeated recursively within the chosen epoch.

3.2 Basic Definitions and Notation

We consider Boolean formulas over a set of variables $X = \{x_1, ..x_n\}$. As usual, a literal is a Boolean variable or its negation, a clause is a disjunction of literals, and a CNF is a conjunction of clauses. We think of clauses as being specified by their sets of literals, and CNFs as specified by their sets of clauses. For a clause C, we write $Vars(C)$ to be the set of variables appearing in C. The *width* $w(C)$ of a clause C is $|\,Vars(C)|$ and the width of a set or sequence of clauses F, is the maximum width of clauses in F. The *size* of a CNF formula F is the total number of literal occurrences in the formula, i.e.,

$\sum_{C \in F} w(C)$.

One of the simplest and most widely studied propositional proof systems is resolution which operates with clauses and has one rule of inference, the resolution rule: $\frac{A \vee x \qquad B \vee \overline{x}}{A \vee B}$. We say that the variable x is *resolved* in this instance of the resolution rule. A *resolution refutation* of a CNF formula (a set of clauses) is a sequence of clauses ending in the unsatisfiable empty clause $\perp$, each of which is either a clause of from the formula (an "axiom") or follows from two previous clauses via the resolution rule. (The term *resolution proof* is used more generally to refer to any inference of this sort that may not necessarily result in $\perp$.) Every resolution proof naturally corresponds to a directed acyclic graph (DAG), known as the proof DAG, in which every clause derived via the resolution inference rule has a directed edge "backwards" from a derived clause to each of its antecedents. (Note that, formally, a resolution proof corresponds to one of possibly many topological sorts of its proof DAG.)

A resolution proof is *regular* if along each path in the proof DAG, each variable is resolved at most once. (Regular) resolution is *sound and complete* in that every CNF formula is unsatisfiable if and only if it has a regular resolution refutation.

The *size* or *length* of a resolution proof is the total number of clauses in the proof. The usual definition of the *space* (*clause space*) used by a resolution proof is the maximum number of clauses which need to be held in memory at any one time when carrying out the proof. Say that a clause is active at time step t if it has been derived before t but is still needed for an inference step to be made at time t or later. Then the clause space is the maximum number of clauses active at any time step. In the most straightforward model, the initial clauses are made available at $t = 0$. To consider sublinear space, this model is modified, and the input clauses become available only as needed, accessed by "axiom download" steps, and may be cleared from local memory and derived again later, in analogy with the usual off-line definition of Turing machine space. In our results, we will be considering superlinear space and hence it is unnecessary to treat the input clauses this way. Doing so does not increase the space bound by even a constant factor. We allow clauses to be derived multiple times in a proof, since that may allow

fewer clauses to be active at any one time.

A *restriction* is a mapping $\rho : X \to \{0, 1, \star\}$. Restrictions on X can be identified with partial assignments on X by viewing unassigned inputs as being mapped to $\star$ and vice versa. We will use the two terms interchangeably, depending on the context. Given two partial assignment π and ρ that are *compatible*, i.e., they agree on $\mathrm{dom}(\pi) \cap \mathrm{dom}(\rho)$, we use $\rho \cup \pi$ to denote the partial assignment that applies both assignments. The restriction of a clause C by ρ, denoted by $C|_\rho$ is the clause obtained from C by setting the value of each $x \in \rho^{-1}(\{0, 1\})$ to $\rho(x)$, and leaving each $x \in \rho^{-1}(\star)$ as a variable. The restriction of a set of clauses is defined by restricting each one.

3.2.1 Tseitin Tautologies

Following [110] we use the following formula to represent the principle that every undirected graph has even total degree.

Definition 3.1. Let $G = (V, E)$ be an undirected graph, and $\chi : V \to \{0, 1\}$ an *odd charge function*, i.e. one such that $\bigoplus_{v \in V} \chi(v)$ is odd. For each edge $e \in E$, we have a variable x_e. For $v \in V, \epsilon \in \{0, 1\}$, let

$$PARITY_{v,\epsilon} := \bigwedge \{ \bigvee_{e \sim v} x_e^{a(e)} \mid \bigoplus_{e} (a(e) \oplus 1) \not\equiv \epsilon \}$$

be the CNF representation of the parity constraint $\bigoplus_{e \sim v} x_e \equiv \epsilon$. The *Tseitin tautology* on (G, χ) is defined by the conjunction

$$\tau(G, \chi) := \bigwedge_{v} PARITY_{v,\chi(v)} \ .$$

Also, independent of whether or not G is connected and χ has odd charge, the resulting formula is known as a Tseitin formula. Frequently we will suppress reference to χ, since when G is connected, any two odd charge functions χ give essentially equivalent formulae.

By a simple counting argument, if charge function χ is odd, then the parity con-

straints cannot all be satisfied – this would correspond to an undirected graph such that the sum of its degrees is odd.

Observation 3.2. *When χ is odd, $\tau(G, \chi)$ is unsatisfiable.*

When the degree of the graph is d, each constraint can be written as a CNF formula of $O(2^d)$ clauses of width d. It is easy to see that the formula $\tau(G)$ has size $O(2^d|V|)$.

Definition 3.3. Let $G = (V, E)$ be a graph. For $E' \subset E$ let $G|E'$ denote the graph $G = (V, E - E')$. When π is a partial assignment to E, we overload the notation $G|\pi$ to mean $G|\operatorname{dom}(\pi)$. (Alternatively, if ρ is a restriction then $G|\rho$ denotes $G|\rho^{-1}(\{0, 1\})$.)

Definition 3.4. If χ is a charge function on a graph $G = (V, E)$ and π is a partial assignment to E then let $\chi \oplus \pi$ denote the new charge function χ' on V given by $\chi'(v) = \chi(v) \oplus \bigoplus_{v \in e \in \operatorname{dom}(\pi)} \pi(e)$.

The following formalizes how restrictions affect Tseitin formulas.

Proposition 3.5. *If $\tau(G, \chi)$ is a Tseitin formula and ρ is a restriction on the edges of G, then $\tau(G, \chi)|_\rho$ is $\tau(G|\rho, \chi \oplus \rho)$, and the parity of χ and $\chi \oplus \rho$ are the same.*

3.3 Resolution refutations of Tseitin formulas

For the graphs we will consider, some generic resolution refutations of odd charged Tseitin formulas follow from bounds on their *cut width*. We give two such refutations. One uses large space and implies that our lower bound results yield tradeoffs of the form that restricting the space does increase the minimum proof length required. The other uses a very small amount of space (indeed, exponentially less) and shows that there is a quasi-polynomial upper limit on the kind of tradeoff one can prove for the formulas we consider.

Definition 3.6. The *cut width* of a graph G is the smallest W such that there is a linear ordering of the vertices $v_1, \ldots, v_n$ such that, for every $1 \leq t \leq n$, there are no more than W edges crossing the cut $(\{v_1, \ldots, v_t\}, \{v_{t+1}, \ldots, v_n\})$.

Lemma 3.7. *Let G be a graph with n vertices, maximum degree d, and cut width W. Then there is a regular resolution refutation of a Tseitin tautology on G using $\leq n \cdot 2^d \cdot 2^{(W-1)}$ resolution steps, and space $\leq 2 \cdot 2^{(W-1)} + d$, plus space for the initial axioms.*

Note that the number of resolution steps is within a constant factor of the proof size.

This refutation described by this lemma works by maintaining clauses corresponding to parities of the edges in the cuts and slowly moving those cuts from one end to the other end of the graph. It mirrors a refutation for similar formulas given in [38]. A detailed proof is given in the appendix.

The following lemma shows that for the graphs we consider here, even radically restricted space can only increase the size required for Tseitin tautologies with small cut width by an $O(\log n)$ power.

Lemma 3.8. *Under the same conditions as Lemma 3.7, the Tseitin tautology on $\tau(G)$ has a tree-like Resolution refutation using space $W\lceil \log n \rceil$ and $2^{W\lceil \log n \rceil}$ resolution steps.*

The refutation in this case involves simulating a binary search for a charge violation over the vertex ordering that achieves the cut width. At each cut in the search, the proof maintains all the clauses in the parities of the edges crossing that cut. A detailed proof is given in the appendix.

Though the graphs we consider all have small cutwidth, there are versions of both these upper bounds that can be expressed in terms of a smaller and more precise parameter, *carving width*. Moreover, some of these refutations can also be found algorithmically with similar complexity. Details are in the appendix.

3.3.1 A measure of complexity of clauses

First we need some preliminary definitions.

Definition 3.9. Consider assignments to the variables of a Tseitin formula τ. A *critical assignment* is a total assignment that violates exactly one clause of τ. The vertex associated with that constraint is said to be its *bad vertex*. For any partial assignment π, we denote by $Crit(\pi)$ the set of critical assignments extending π.

Definition 3.10. Given a partial assignment π to a Tseitin formula τ over graph G, we say that a vertex v is a *critical vertex for* π iff v is bad for some critical assignment to the edges of G consistent with π. We define $crit_\tau(\pi)$ to be the set of critical vertices for π. We drop the subscript τ when it is understood from the context. We also define critical vertices for clauses by letting $crit(C)$ be the set of critical vertices of the partial assignment associated with $\neg C$.

Fortunately, for Tseitin tautologies, the set $crit(\pi)$ has a nice, well-known characterization:

Proposition 3.11. *$crit(\pi)$ is non-empty if and only if $G|\pi$ has exactly one component of odd charge, in which case $crit(\pi)$ is equal to that component.*

Let S be a subset of nodes. We write $\delta(S)$ for the edges on the boundary of set S.

Corollary 3.12. *If π is a partial assignment with $crit(\pi) = S$, then every edge on the boundary of S is assigned by π.*

Finally, we note how partial assignments impact critical sets.

Proposition 3.13. *Let $\tau(G, \chi)$ be a Tseitin formula and ρ be a restriction (partial assignment) on the edges of G. Then for any partial assignment π compatible with ρ, $crit_{\tau(G,\chi)^\rho}(\pi) = crit_{\tau(G,\chi)}(\pi \cup \rho)$.*

Proof. This follows immediately from the fact that the set of critical assignments for $\tau(G, \chi)$ that are consistent with ρ is precisely the set of critical assignments for $\tau(G, \chi)^\rho$ with assignment ρ appended. □

We use $|crit(C)|$ as a measure of the complexity of clause C.

Proposition 3.14. *Fix any odd-charged Tseitin formula τ over a connected graph G.*

(a) $|crit_\tau(\bot)| = |V(G)|$ where $\bot$ is the empty clause.

(b) For any clause C of τ, $|crit_\tau(C)| = 1$.

(c) *If clause C follows from clauses C_1, C_2 by resolution then* $|crit_\tau(C)| \leq |crit_\tau(C_1)| + |crit_\tau(C_2)|$.

Proof. Part (a) and (b) follow immediately from Proposition 3.11. Part (c) follows from the soundness of resolution since any critical assignment that falsifies C must falsify either C_1 or C_2, hence $Crit(C) \subseteq Crit(C_1) \cup Crit(C_2)$. □

Subadditivity of $|crit(C)|$ from Proposition 3.14(c), immediately implies the following.

Corollary 3.15. *For any t, any resolution derivation of a clause D with $|crit(D)| > 2t$ from a collection of clauses C having $|crit(C)| \leq t$, must contain a clause C' such that $t < |crit(C')| \leq 2t$.*

Proof. If not, then at the first point where such a D is derived, it must be derived from two clauses with complexity $\leq t$, contradicting subadditivity. □

3.4 Time-Space Tradeoff for General Resolution

To show lower bounds in General Resolution, we consider Tseitin tautologies on a graph $G = (V, E)$ that is a path of length ℓ of complete bipartite graphs $K_{n,n}$, where the left side of one is the right side of the next. Equivalently, this is the tensor product of the complete graph K_n with the path graph P_ℓ. For $c > 0$ an absolute constant, our results hold for any $2^{cn} > \ell > n^4$, but are most impressive for larger ℓ. We picture the vertices of this graph as lying in an $n \times \ell$ grid, with each column corresponding to a vertex of P_ℓ. We let V_i denote the set of vertices in the i-th column, and E_i denote the edges between the i-th column and the $(i+1)$-st column.

Proposition 3.16. *Any Tseitin tautology over $G = K_n \otimes P_\ell$ has (regular) resolution size at most $O(2^{n^2+2n} n\ell)$.*

Proof. It is easy to see that G has cut width n^2 so this is immediate from Lemma 3.7. □

The main result of this section is the following time-space tradeoff lower bound for general resolution.

Theorem 3.17. *If n is sufficiently large and $2n^4 \leq \ell < (4n)^{-1}\left(\frac{9}{8}\right)^n$ for any resolution refutation of a Tseitin tautology on the graph $G = K_n \otimes P_\ell$ of size T and space S*

$$T \geq \left(\frac{2^{0.58n^2}}{S}\right)^{\frac{\log_2 L}{\log_2 \log_2 L}}$$

where $L = \log_2(\ell/(2n^3))$.

Corollary 3.18. *There is a $c > 0$ such that for sufficiently large N and any S satisfying $N \leq S \leq N^{\frac{1}{9}\log_2 N}$, there is a CNF formula of size N that has a resolution refutation of size at most $T_{UB} \leq N \cdot S$ and space at most S such that any resolution refutation using space $S^{1/2}$ requires size $T_{LB} \geq T_{UB}^{c \log_2 \log_2 N / \log_2 \log_2 \log_2 N}$.*

Proof. Choose the largest n such that $2^{n^2} \leq S$. Since $N \leq S \leq N^{\frac{1}{9}\log_2 N}$, we must have $2^{3n} \leq N \leq 2^{n^2}$ and hence $log_2 \log_2 N$ is $\Theta(\log n)$. Set $\ell = 2^{n/8} < (9/8)^n/(4n)$. With this value of ℓ, L is at least $n/9$ for N sufficiently large and hence $\log_2 L/\log_2 \log_2 L$ is $\Theta(\log \log N/\log \log \log N)$. Since the Tseitin formula for $K_n \otimes P_\ell$ has size at most $\ell n 2^{2n} < N$. We can pad this formula by dummy clauses until it has N clauses. Applying the above theorem to this formula with space $S^{1/2}$ instead of S yields the claimed lower bound. □

We use the "progress argument" sketched in the introduction, together with a bottleneck counting based on random restrictions. One subtle point is that, although our argument is applied recursively, we only apply the random restriction once, not after each step of the recursion.

We use the complexity measure on clauses defined in the previous section, which is 1 for the input clauses, is $|V(G)|$ for the final contradiction $\perp$, and grows slowly throughout a resolution proof (possibly not monotonically). We define many different medium complexity levels of clauses.

Definition 3.19. Let $t_0 = n^4$. Define $\mathcal{L}_i = \{C \in \Pi : 2^i t_0 < |crit(C)| \leq 2^{i+1} t_0\}$ for $0 \leq i < \log_2(\ell n/t_0) - 1$. Each $\mathcal{L}_i$ represents a different *complexity level*. We say that a clause of Π has *medium complexity* if it is in one of the $\mathcal{L}_i$.

We will use the fact that the restriction of a proof of a Tseitin tautology is itself a proof of a Tseitin tautology on a subgraph of the original graph; hence the proof for a graph G contains proofs of Tseitin tautologies for all of the subgraphs of G. We choose such a restriction randomly from a suitable distribution that is overwhelmingly likely to keep the complexity of the final contradiction high.

Following standard notation, let $\mathbb{R}_{1/3}$ be the probability distribution on restrictions $\rho : E \to \{0, 1, *\}$ such that for each e independently, $\Pr_{\rho \sim R_{1/3}}[\rho(e) = 0] = \Pr_{\rho \sim R_{1/3}}[\rho(e) = 1] = \Pr_{\rho \sim R_{1/3}}[\rho(e) = *] = 1/3$.

After we apply a random $\rho \sim R_{1/3}$ to a refutation of $\tau(G, \chi)$, we get a refutation of the Tseitin formula on the smaller graph $G|\rho$ with properties as in Proposition 3.5. We will measure the progress of the proof using the medium complexity levels of its clauses. We therefore will compute bounds on the probability that a clause becomes a medium complexity clause for the Tseitin tautology on the graph $G|\rho$ after such a restriction; we will do the same for the probability that a set of clauses have different medium complexity levels after restriction.

The graph $G|\rho$ and the charge $\chi \oplus \rho$ that results from applying the restriction ρ to $\tau(G, \chi)$ is not fixed. Therefore, it is easier to analyze the critical sets and complexity of the restricted clauses using Proposition 3.13 which says that for any partial assignment π compatible with ρ, $crit_{\tau(G,\chi)^\rho}(\pi) = crit_{\tau(G,\chi)}(\pi \cup \rho)$. For any clause C over G, this means that

$$crit_{\tau(G,\chi)^\rho}(C|_\rho) = crit_{\tau(G,\chi)}(C \vee \overline{\rho})$$

where $\overline{\rho}$ is the clause that is falsified precisely on those assignments consistent with ρ. This allows us simply to work with clauses $C \vee \overline{\rho}$ defined over the original graph G, so for the remainder of this section we write $crit$ for $crit_{\tau(G,\chi)}$.

Now, we view a refutation of total size T and space S as ordered into epochs and sub-epochs as follows: Let L be the number of different medium complexity levels,

$L = \log_2((n\ell)/(2t_0)) \geq \log_2 n$. Let $k = \lfloor \log_2 L \rfloor - 1$ and $h = \lfloor \log_2 L / \log_2 \log_2 L \rfloor < (k+2)/\log_2 k$. With these values we have $(k+1)k^{h-1} < L$. Now choose

$$m \leq \left\lfloor \frac{(3/2)^{n^2} \ell^{-1} L^{-1} 2^{-4n}}{S} \right\rfloor^{1-3/\log_2 k}.$$

One can easily verify that $m \geq \max(8, 2^{0.58n^2}/S)$ for sufficiently large n.

The theorem follows immediately if $T \geq m^h$ so assume, by contradiction, that $T < m^h$. An *epoch at the leaf level* is an interval of m consecutive clauses in the proof, beginning at a time step that is a multiple of m. We consider all clauses in the leaf level epoch to be *frontier* clauses. An *epoch at level* $2 \leq i \leq k$ is an interval of m^i consecutive clauses in the proof, beginning at a time step that is a multiple of m^i. Its sub-epochs are the epochs of level $i-1$ within the epoch, and the *frontier* clauses for the epoch are the ones that are active at the end of any of its sub-epochs. Thus, leaf level epochs have m frontier clauses and higher level epochs have at most mS frontier clauses, since at most S clauses are active at each of the m times when a sub-epoch ends.

Lemma 3.20. *For any refutation of a Tseitin tautology on a connected graph with $n\ell$ nodes, at least one epoch has k frontier clauses of distinct medium complexity levels.*

Proof. Assume that no such epoch exists, at any level of epochs. We will inductively find an epoch E_j at level $k-j$ and medium complexity levels i_{min} and $i_{max} \geq i_{min} + L/k^j$ so that all active clauses at the start of E have complexity at most that of medium complexity level i_{min} and some clause at the end of E has complexity at least that of complexity level i_{max}.

At the start, we can choose E_0 to be the entire proof, since at the beginning, we have only clauses of complexity 1 (input clauses) and at the end, we have only the empty clause, $\perp$, of complexity $n\ell$.

Assume that we have E_j, i_{min} and i_{max} as above. Consider the partition of E_j into m sub-epochs at level $k-j-1$. Because there are at most $k-1$ distinct complexities among E_j's frontier clauses, there must be a sub-interval i'_{min} to i'_{max} of length at least

$(i_{max} - i_{min})/k \geq L/k^{j+1}$ where no frontier clause is of medium complexity level between i'_{min} and i'_{max}. Consider the first sub-epoch of E_j that ends with an active clause of medium complexity level at least i'_{max}; since E_j ends with a clause of medium complexity level $i_{max} \geq i'_{max}$, such a sub-epoch must exist. Since no clause of medium complexity level at least i'_{max} was active at the start of this sub-epoch, and no clause of medium complexity level between i'_{min} and i'_{max} is active at the start of any sub-epoch, we can choose this first sub-epoch as E_{j+1}.

Inductively, we get a leaf-level epoch E_{h-1} and an interval i_{min} to i_{max} of length at least $L/k^{h-1} \geq k+1$ so that all active clauses at its start have complexity at most that of medium complexity level i_{min} and at least one clause has medium complexity level i_{max} or greater at the end. Then clauses of all k medium complexity levels between i_{min} and i_{max} appear in the leaf-level epoch. Since all lines in a leaf-level epoch are frontier clauses, this is a contradiction. □

To use the above to get a contradiction, we show that, for $G = K_n \otimes P_\ell$s, the graph is very likely to stay connected after a random restriction, and any small set of clauses is very unlikely to contain many different medium complexity levels of clauses after the restriction. We then apply the above lemma to the restricted proof.

The following lemmas, proved in the next section, summarize our bounds on these probabilities.

Definition 3.21. Let $CONN(\rho)$ denote the event that each bipartite graph $(V_i, V_{i+1}, E_i \setminus \mathrm{dom}(\rho))$ is connected.

Proposition 3.22. $\Pr_\rho[\neg CONN(\rho)] \leq \ell \cdot 2(n-1)(8/9)^n$.

Lemma 3.23. *For $k \leq 1/3L$, any k clauses $C_1 \ldots C_k$, and any k medium complexity levels $\ell_1, \ldots, \ell_k$ with $\ell_{i+1} \geq \ell_i + 3$ for $1 \leq i \leq k-1$,*

$$\Pr_{\rho \sim \mathbb{R}_{1/3}}[CONN(\rho) \wedge \forall i, (C_i \vee \overline{\rho}) \in \mathcal{L}_{\ell_i}] \leq \left(\ell \cdot 2^{4n} \cdot (2/3)^{n^2}\right)^k$$

Corollary 3.24. *Let $\mathcal{C}$ be a collection of $\leq M$ clauses. The probability that $\mathcal{C}|_\rho$ contains k clauses of distinct medium complexity levels is at most $\left(ML(2/3)^{n^2}\ell 2^{4n}\right)^{\lceil k/3 \rceil}$.*

Proof. If there are k distinct medium complexity levels, a subset of $\lceil k/3 \rceil$ of them are mutually separated by 3. There are at most $L^{\lceil k/3 \rceil}$ choices for this subset of medimum complexity levels $\ell_1, .., \ell_{\lceil k/3 \rceil}$, and for each element of the subset, at most M choices for a clause $C_i \in \mathcal{C}$ to become this complexity. Since the complexity of $C_i|_\rho$ is the same as that of $C_i \vee \overline{\rho}$ we can apply the previous lemma with a union bound to get the probability bound in the corollary. □

of Theorem 3.17. Assume that m, k, and h are defined as above and that there is a refutation of the Tseitin tautology on $K_n \otimes P_\ell$ of space S and size $T < m^h$. Under any restriction ρ, either $G|\rho$ is disconnected, or after the restriction ρ some epoch in the proof contains frontier clauses of k distinct medium complexity levels. By proposition 3.22, the first probability is $< 1/2$. There are fewer than $2m^{h-1}$ epochs in all, and each has a frontier set of size at most $M = mS$. Thus, from the above corollary, the probability that there is an epoch that has k distinct medium complexity levels among the restriction of its frontier clauses is at most

$$
\begin{aligned}
& 2m^{h-1} \left(mSL(2/3)^{n^2} \ell 2^{4n} \right)^{\lceil k/3 \rceil} \\
& < \frac{1}{2} \left(m^{1+3/\log_2 k} SL(2/3)^{n^2} \ell 2^{4n} \right)^{\lceil k/3 \rceil} \\
& \qquad\qquad \text{since } m \geq 8 \text{ and } h < (k+2)/\log_2 k \\
& \leq \frac{1}{2} \left[\left(\frac{(3/2)^{n^2} \ell^{-1} L^{-1} 2^{-4n}}{S} \right)^{1-9/\log_2^2 k} SL(2/3)^{n^2} \ell 2^{4n} \right]^{\lceil k/3 \rceil} \\
& \leq \frac{1}{2}.
\end{aligned}
$$

This is a contradiction to Lemma 3.20 and the theorem follows. □

3.4.1 Medium complexity clauses and random restrictions

In this section, we prove the lemmas about the effect of random restrictions on Tseitin tautology clauses. We begin with the proof of Proposition 3.22 that the graph remains connected after a random restriction is applied.

of Proposition 3.22. We apply a union bound over the columns and compute an explicit bound for the well-known connectivity properties of random bipartite graphs $G(n, n, \frac{1}{3})$.

Consider the left side of $G(n, n, 1/3)$. For any two vertices here, the probability that they are not connected with a two-step path in $G(n, n, 1/3)$ is at most $(8/9)^n$, since there are n possible disjoint two-step paths. By a union bound, the left side and right side are both connected to themselves except with probability $2(n-1)(8/9)^n$. Since these imply that an edge exists, the $G(n, n, 1/3)$ is connected except with this probability. □

Note that the condition $\ell < (4n)^{-1}(9/8)^n$ implies that $\Pr_\rho[\neg CONN(\rho)] < 1/2$.

Lemma 3.25. *For every clause C and every fixed set of edges $E' \subseteq E$, the probability over $\rho \sim \mathbb{R}_{1/3}$ that $C \vee \overline{\rho}$ is non-trivial and $E' \subseteq Vars(C \vee \overline{\rho})$ is at most $(2/3)^{|E'|}$.*

Proof. For $C \vee \overline{\rho}$ to be non-trivial, ρ must not set any edge to falsify C. In particular, this applies to each edge in E' that appears in $Vars(C)$. In order for $E' \subseteq Vars(C \vee \overline{\rho})$, each edge in E' not in $Vars(C)$ must be set by ρ. For each edge in E', these successes happen with probability $2/3$ and the probabilities are independent, so the total probability that $E' \subseteq \delta(crit(C \vee \overline{\rho}))$ is at most $(2/3)^{|E'|}$. □

To prove a non-trivial tradeoff lower bound using our outline above, we need a stronger lower bound than the upper bound for unrestricted space of roughly 2^{n^2} given in Proposition 3.16. Therefore, we need to be able to argue that after such a restriction the probability that a clause will become a medium complexity clause is exponentially small in n^2.

Intuitively, a set of vertices S of size at least n^2 and less than $\ell n/2$ will have a boundary of size at least n^2 for the following reasons. If it has as many as n "partial columns" in which it contains some, but not all, of the vertices, then each such column will contribute at least n boundary edges. If it has fewer than n partial columns, it must contain a full column and an empty column, and its not hard to see that deleting the elements in the partial columns cannot decrease the size of the boundary, so this configuration has a boundary of at least n^2 as well.

By Corollary 3.12, for every clause C, $crit(C \vee \overline{\rho}) = S \subset V$ only if $\delta(S) \subseteq Vars(C \vee \overline{\rho})$. Therefore, by Lemma 3.25, $\Pr_{\rho \sim \mathbb{R}_{1/3}}[crit(C \vee \overline{\rho}) = S] \leq (2/3)^{n^2}$ for any C and any S of size between n^2 and $\ell n/2$. Though the bound for a single S is of the form we want, the number of such S is huge, so a simple union bound over all choices of S is insufficient to derive the bound we need. Nonetheless, we can show a bound very close to this one for the probability that $crit(C \vee \overline{\rho})$ is *any* medium sized set. We also generalize it to show bounds on the probability that, after applying ρ, k clauses appear from k different $\mathcal{L}_i$

Definition 3.26. Let S denote a connected subset of the vertices V. Define $S_i := S \cap V_i$ and $s_i := |S_i|/|V_i|$. We say that

- i is a *full column* of S if $s_i = 1$, an *empty column* if $s_i = 0$, and a *partial column* otherwise;
- i is a *transition point of* S if $i-1, i, i+1, i+2 \in \{1, \ldots, \ell\}$ are columns, and either both $s_i \geq 1/2$ and $s_{i+1} \leq 1/2$, or both $s_i \leq 1/2$ and $s_{i+1} \geq 1/2$;
- S has a *transition point with signature* (i, A_0, A_1, A_2, A_3) if i is a transition point of S, $S_{i-1} = A_0$, $S_i = A_1$, $S_{i+1} = A_2$, and $S_{i+2} = A_3$.

The transition point concept is useful because of the following lemma which says that the existence of a transition point for a set S implies that S has a large boundary.

Lemma 3.27 (Transition Point Lemma). *If i is a transition point of S, then $|\delta(S)| \geq n^2$. Moreover, for any signature (i, A_0, A_1, A_2, A_3), there exists a set of at least n^2 edges, $E^* \subseteq E_{i-1} \cup E_i \cup E_{i+1}$, depending only on (i, A_0, A_1, A_2, A_3), such that if S has a transition point with signature (i, A_0, A_1, A_2, A_3) then $E^* \subseteq \delta(S)$.*

There are only $\ell \cdot 2^{O(n)}$ possible signatures, but, by this lemma, the probability that $crit(C \vee \overline{\rho})$ has a transition point with a specific signature is at most $2^{-\Omega(n^2)}$ for any particular C, so this will allow us to obtain a very strong upper bound on the probability that C has any transition point at all.

of Transition Point Lemma. Let S be a connected set of vertices and have a transition point with signature (i, A, B, C, D). We would like to find, from this information only, a set of E^* edges on the boundary of S. We write $a = |A|/n$, $b = |B|/n$, $c = |C|/n$ and $d = |D|/n$. Without loss of generality we can assume that $b \geq 1/2, c \leq 1/2$.

We will find a set of n^2 edges in the columns E^{i-1}, E^i, E^{i+1} on the boundary of any S with this signature. Since the signature determines exactly which edges in these columns are boundary edges, and the graph between each pair of layers is the complete bipartite graph, only the numbers (a, b, c, d) matter. The number of boundary edges may be computed exactly as

$$n^2(a(1-b) + b(1-a) + b(1-c) + c(1-b) + c(1-d) + d(1-c)) \, .$$

However,

$$\begin{aligned} a(1-b) + b(1-a) &= a(1-2b) + b \\ &= 1 - b + (2b-1)(1-a) \\ &\geq 1 - b \qquad \text{since } b \geq 1/2 \end{aligned}$$

and

$$\begin{aligned} c(1-d) + d(1-c) &= c + d(1-2c) \\ &\geq c \qquad \text{since } c \leq 1/2. \end{aligned}$$

So, the expression above is at least

$$\begin{aligned} n^2(1 - b + b(1-c) + c(1-b) + c) &= n^2(1 + 2c - 2bc) \\ &= n^2 + n^2 2c(1-b) \\ &\geq n^2 \end{aligned}$$

as claimed. □

In the informal argument that medium sized sets of vertices have at least n^2 edges on their boundaries, we identified two cases to handle – those with many partial columns and those with few. If the set is in the right size range and has few partial columns, it can be shown to have both a full column and an empty column, and it is easy to see that such sets must have a transition point somewhere between the full and empty column. We handle the other case, with many partial columns, by showing that it holds conditioned on the graph being connected.

Lemma 3.28. *For any clause C and restriction ρ, if $CONN(\rho)$ holds then $|C|$ is at least the number of partial columns in $crit(C \vee \overline{\rho})$.*

Proof. Let $S = crit(C \vee \overline{\rho})$. If i is a partial column of S, then both $S_i \cup S_{i+1}$ and its complement in $V_i \cup V_{i+1}$ are nonempty. If $CONN(\rho)$ holds, then these two sets of vertices are adjacent, and in fact there is an edge between them that is not assigned by ρ. By Corollary 3.12, all edges on the boundary of the critical set must be assigned, so C must assign some edge in E_i. Thus C contains at least one edge for each partial column of S. □

Now we will show our bound for the probability that multiple clauses come to occupy many separated medium complexity levels. We will use the separation in complexity levels to argue that there must be many transition points for these clauses rather than just one. This will be based on bounding the number of distinct endpoints of sets of intervals of increasing size on a line or circle for which we will use the following lemma proved in Section 3.5.

Lemma 3.29. *Suppose that we have k points of the n point circle $a_i \in Z_n$, and k natural numbers $d_i \leq n/2$, determining k intervals on the circle $(a_i, b_i = a_i + d_i)$ each of length d_i. Let U denote the set of endpoints of these intervals; i.e., $U := \bigcup_i \{a_i, b_i\}$. It follows that*

1. *if $\forall i$, $d_{i+1} \geq 2d_i$, and $1 \leq d_1, d_k \leq n/2$ then $|U| \geq k + 1$.*

2. *Further, if for some a, $k \cdot a < d_1$, then there exists a subset U' of U such that*

$|U'| \geq k+1$ *and any two points of* U' *are at distance greater than* a *from each other.*

The same argument we give can also be used to prove a more general statement, but we limit ourselves to the following for simplicity.

of Lemma 3.23. If any C_i is as large as kn^2, it has probability at most $(2/3)^{kn^2}$ to survive the restriction and we are done, so suppose that this is not the case. Using Lemma 3.28, we can restrict attention to candidate critical sets S^i that have fewer than $kn^2 \leq n^3/3$ partial columns, containing fewer than $n^4/3$ vertices. Because $t_0 = n^4$, for each S^i there must be many full columns. There are also many empty columns since the largest critical sets in medium complexity levels have size at most $\ell n/2$.

So, we can already see that each S^i has a transition point. What we would like to see is that there must be at least k transition points among all of them, and that these transition points are all far apart from each other. We use Lemma 3.29 for this.

For each S^i, let a_i denote the first column from the left so that $|S^i_{a_i}| \geq n/2$, and let b_i similarly denote the first such column from the right. So long as b_i is not one of $\{1, 2, \ell - 1, \ell\}$, it is a transition point, and the same is true for $a_i - 1$. So, it suffices to prove that $|\bigcup_i \{a_i - 1, b_i\} \setminus \{1, 2, \ell - 1, \ell\}| \geq k$. In fact it suffices to meet the requirements for the strong form of Lemma 3.29 with $a \geq 3$, since then at most one point of U' is in the set $\{1, 2, \ell - 1, \ell\}$.

Let d_i denote the difference $b_i - (a_i - 1)$. Since C_i has medium complexity level ℓ_i, d_i is at least the number of full columns, i.e., $d_i \geq 2^{\ell_i} t_0/n - n^3/3$. We can also upper bound the total number of full and partial columns as $2^{\ell_i+1} t_0/n + n^3/3$, which is an upper bound on d_i, since S^i is connected. Since $\ell_{i+1} \geq \ell + 3$ and $t_0 \geq n^3$, we have $2^{\ell_{i+1}} \geq 4 \cdot 2^{\ell_i+1}$ and

$$\frac{d_{i+1}}{d_i} \geq \frac{4 - \frac{1}{3}}{1 + \frac{1}{3}} > 2\,.$$

Further, $d_k \leq \ell n/4 + n^3/3 < \ell n/2$, as required, and $d_1 \geq \frac{2}{3} n^3$, while $k < n/3$ implies $3k + 1 < n + 1 << d_1$. Therefore, Lemma 3.29 may be applied to say that there are k distinct transition points that are far apart.

If we fix a particular sequence of signatures for these transition points then the Transition Point Lemma implies that there is a fixed set E' of kn^2 edges, which depends only on the signatures, contained in $\bigcup_{i=1}^{k} \delta(C_i \vee \overline{\rho})$, consisting of the union of the k disjoint sets of edges for each transition point. As in the proof of Lemma 3.25, any such edge must either be set by ρ, or appear in some C_i and hence cannot be set by ρ to violate that C_i. Each such event occurs with probability at most $2/3$ and therefore, by the independence over edges, the total probability is at most $(2/3)^{kn^2}$. The number of sequences of k signatures is at most $(\ell \cdot 2^{4n})^k$ and the bound follows. □

3.5 Endpoints of Intervals

This section is devoted to the proof of Lemma 3.29.

Suppose that we have a collection of k intervals on the line. How many distinct endpoints are there? If nothing else is known about the intervals, there can be as few as $\mathbf{O}(\sqrt{k})$ endpoints. However, if each interval is at least twice as long as the interval before it, then intuitively the intervals cannot be packed together so nicely; we will see that under this assumption there are at least $k+1$ distinct endpoints.

Although we only care about the line, in Lemma 3.29 it is more convenient to analyze the case of the circle.

of Lemma 3.29. Suppose that we have k points of the n point circle $a_i \in Z_n$, and k natural numbers $d_i \leq n/2$, determining k intervals on the circle $(a_i, b_i = a_i + d_i)$ each of length d_i. Let U denote the set of endpoints

$$U := \bigcup_i \{a_i, b_i\} .$$

Suppose that $d_{i+1} \geq 2d_i$ for all i and $1 \leq d_1, d_k \leq n/2$. Let us define a graph G on vertex set is U, and for each i, an edge from a_i to b_i. Thus there are k total edges. For the first claim, the strategy is to show that this graph is a forest, and hence has at least $k+1$ vertices. To show the second claim, we will show a process which constructs an acceptable set of $k+1$ vertices.

We will still use the terms "distance" and "displacement" to refer to distances in the circle.

Consider any walk in G which does not use any edges more than once. (Note that this is not the same as a path, as it may include cycles.) The total displacement as we move along from vertex to vertex can be written

$$\sum_{i \in S} \pm d_i$$

where S is the set of indices corresponding to edges appearing on the walk.

By induction on $|S|$, we see that for any nonempty S

$$\sum_{i \in S, i < \max S} d_i \leq d_{\max S} - d_1.$$

The base case is trivial, and if the claim holds for S, we may add $d_{\max S}$ to both sides, and we can apply the assumption $2d_{\max S} \leq d_{\max S+1} \leq d_{s'}$ for any $s' > \max S$, and so deduce the claim for $S \cup \{s'\}$, thus the inductive hypothesis also holds.

Suppose S is nonempty. By the triangle inequality,

$$d_1 \leq \left| \sum_{i \in S} \pm d_i \right| \leq 2d_{\max S} - d_1 \leq n - d_1 \ ,$$

using the $d_k \leq n/2$ assumption.

Now we can deduce the first claim. Suppose the graph contains a cycle. Then there is a walk in G as before which starts and ends at the same vertex, thus the total displacement is $0 \mod n$. But since the $1 \leq d_1$, the above inequality contradicts this. Thus G contains no cycles and hence is a forest with k edges – this implies there are at least $k + 1$ vertices, or $|U| \geq k + 1$ as desired.

To see the second claim, we will show a greedy algorithm to find U'.

1. Begin with $U' := \emptyset$. We will process the components of the graph in some order. The first one we can just add entirely to U'.

2. While a component we have not yet processed contains some vertex v within a of

anything in U', choose such a v and add every other vertex in that component C_v to U'.

3. If all of the remaining components have every vertex at distance greater than a from U', call the algorithm recursively on the remaining unprocessed components.

Our first observation is that the algorithm omits at most one vertex from any component from U', and for at least one component omits none. Since a forest on k edges and c components has at least $k + c$ vertices, U' will always contain at least $k + c - (c - 1) = k + 1$ vertices at the end.

It suffices to show the algorithm never adds any two vertices at distance less than a to U'. Call such a pair of vertices an "unsafe pair". The idea is that when $d_1 > k \cdot a$, the argument before that G is cycle-free will hold even if we add as many as k unsafe pairs as edges to the graph G, since these new edges can only change the sum above by at most $k \cdot a$, and d_1 is large enough that we can tolerate that. Our strategy is to view the $(k - 1)$ unsafe pairs identified in step 2 as edges which we add to G. For any vertices of U' which become connected by this, we will argue that the path connecting them contains at least one original edge of G and so the previous calculation shows their distance is at least a. For any vertices which were not connected by this, we will see that they could not have formed an unsafe pair – this can only happen when step 3 occurs, and we will handle this scenario as follows.

It suffices to assume that step (3) never happens. When step 3 happens, there are no unsafe pairs between the remaining components and the U' obtained from the components processed already. If the algorithm didn't add any unsafe pairs when run just on the first group of components, and also doesn't add any unsafe pairs when just run on the second group of components, it will succeed when run on all of the components, so we may break up the current instance of the problem into two subinstances and restrict attention to these subinstances. Appealing to this argument sufficiently many times, it suffices to show correctness in cases where step (3) never occurs. (To make this completely precise, one could remove the arbitrary choice in the definition of the algorithm

by assigning a priority to each component, and specify that the algorithm deterministically chooses to process the highest priority component with an unsafe pair to U'. Then, when we split an instance on which step (3) occurs into two subinstances, the subinstances inherit a priority ordering, and the U' returned by the algorithm on the initial instance will be exactly the union of the U''s returned by the algorithm on the two subinstances, so correctness indeed follows from correctness of the runs on the two subinstances.)

Therefore without loss, step 1 occurs once and then step 2 occurs $k-1$ times. When step 2 occurs, by definition there is some vertex $u \in U'$ and some vertex v which is chosen so that uv is an unsafe pair, and component C_v is the next processed component. Let G' be the graph G, plus an edge for each such uv. The edge uv connects the component C_v to the previously processed components. When we start, G is a forest, and at the end, every component has been processed, so G' is a tree.

Now consider a walk in G' which does not use any edge more than once. As long as it contains at least one edge of G, by the triangle inequality the total displacement is between $d_1 - (k-1) \cdot a$ and $n - (d_1 - (k-1) \cdot a)$, so since $d_1 > k \cdot a$, the start and end point of such a walk do not form an unsafe pair.

The tree G' contains a path connecting every $u, u' \in U'$, so it suffices to see that this path contains at least one edge of G. What vertices are connected using only the edges of G' that don't appear in G? By construction this set of edges forms a collection of stars with vertices of U' at the centers, since for each unsafe pair found in step 2, the vertex v corresponds to a distinct component, and v may not be chosen as u at some later step since it is not added to U'. Thus for no two distinct points of U' does the unique path in G' consist only of unsafe pair edges – each such pair has a path in G' with at least one edge of G, so they do not form an unsafe pair. □

3.6 Time-Space Tradeoff for Regular Resolution

For regular resolution, it is possible to use a fairly different argument to get super-polynomial size lower bounds for space that is polynomially large for Tseitin formulas

on a family of constant-degree graphs (and hence constant clause size) in contrast to the large degree of the graphs we used in the general case. Though not qualitatively different, the tradeoffs themselves are slightly sharper than in the general case and also apply in the case of substantially larger space bounds (up to sub-exponential space rather than up to quasipolynomial space).

To achieve this, the random restriction argument from the case of general resolution is replaced with a random adversary argument and the analysis now takes a top-down flavor. This gives much more freedom in the choice of hard graphs since we no require connectivity under random restrictions that we needed in the general resolution case. We still must use graphs that are "long and skinny" as before. For simplicity and for applicability to a wider range of space bounds than for the graphs we used previously, we will choose the $n \times \ell$ grid graph, $G_{n,\ell}$ of n rows and ℓ columns.

We first note that the Tseitin tautologies over grid graphs have short regular resolution refutations.

Proposition 3.30. *Any Tseitin tautology over $G_{n,\ell}$ has a regular resolution refutation size at most $32\ell n \cdot 2^n$ and space at most $2^{n+1}+5+|\{axioms\}|$, and $|\{axioms\}| \leq 8\ell n$.*

Proof. $G_{n,\ell}$ has cut width $n+1$ and maximum degree 4 so this follows from Lemma 3.7. The number of axioms is easily counted as $\leq 2^{d-1}|V|$. □

By contrast, the main result that we will prove in this section is that if the space allowed is reduced significantly below 2^n, and ℓ is not too large then such a formula requires regular resolution refutations of superpolynomial size.

Theorem 3.31. *For any ℓ such that $n^3 \leq \ell < 2^n$, any size T and space S regular resolution refutation of the Tseitin tautology on a n by ℓ grid must satisfy*

$$T \geq \left(\frac{2^{(1-o(1))n}}{S}\right)^{\frac{\log_2 L}{2\log_2\log_2 L}}$$

where $L = \log_2(\ell/(4n^2))$.

To prove Theorem 3.31, we will begin with a regular resolution refutation Π, and repeatedly subdivide Π into polynomially many epochs, at each point choosing an epoch in which enough "progress" happens that we may continue subdividing the epoch. The process peters out after $\mathbf{O}\left(\frac{\log L}{\log \log L}\right)$ subdivisions.

In the next three subsections we develop the main technical tools that allow us to derive this lower bound. In section 3.6.1 we modify the complexity measure for clauses from general resolution to a complexity measure specific to regular resolution. This measure has the advantage of growing monotonically with the inferences in the proof, which makes it a much more precise tool. We use this measure to define the complexity levels that are the indicators for progress through the proof. In section 3.6.2 we describe a probabilistic adversary that follows the inferences in the proof. One can think of this adversary as observing the progress of the proof. We show that the probability that adversary reaches a clause (or set of clauses) can be analyzed in terms of a potential function Φ applied to a set of edges associated with that clause (or set of clauses). Φ turns out to be the rank function of a certain matroid and we characterize several properties of Φ that we need in the overall argument. In section 3.6.3, we give fairly straightforward proofs that the potential Φ will be large for sets that correspond to clauses of many different medium complexity levels.

In section 3.6.4, we put the pieces together and give the inductive argument that yields Theorem 3.31. The base case shows simply that any sub-epoch in the proof in which the adversary has a reasonable probability of visiting clauses of many different complexity levels must have length exponential in the width n of the grid graph. The inductive step shows that if one divides an epoch of the proof through which the adversary passes into sub-epochs then either the adversary visits clauses of many different complexity levels at the boundaries between sub-epochs or the adversary sees a large fraction of the overall progress in complexity levels within a single sub-epoch. If the space is small then the first case is very unlikely; if the second case holds then we can subdivide that sub-epoch and apply the argument recursively.

3.6.1 Read-once branching programs and common information

It is well known (cf. [73]) that a resolution refutation yields a branching program for the "clause search problem" and this branching program is read-once if and only if the resolution refutation is regular. In this construction, we start with the proof DAG and labeled the edges by the following rule: If $C \vee D$ is derived by resolving $C \vee x$ with $D \vee \neg x$, then the node for clause $C \vee D$ is also labeled by a query to x, the edge directed from $C \vee D$ to $C \vee x$ is labeled $x = 0$, and the edge from $C \vee D$ to $D \vee \neg x$ is labeled $x = 1$. Like the proof DAG, the single-source, or root, of the branching program is the contradiction $\perp$ and the sinks, or leaves, are the input clauses or axioms. Note that paths p in this branching program correspond to partial assignments. We identify a regular resolution proof with both its DAG and its associated branching program. The following is immediate from the definition.

Proposition 3.32. *Every node reached by an assignment starting at the root of the branching program for any regular resolution proof* Π *is labeled by a clause that is falsified by that assignment.*

For purposes of a top-down analysis, more relevant than the literals appearing in a clause (node) in a proof is a closely related concept we call its *common information*. A similar definition has appeared previously, e.g. in [91].

Definition 3.33. For C a clause in a regular resolution refutation Π, define the *(common) information at* C *in* Π to be a partial assignment $I_{C,\Pi}$ in terms of the proof Π as follows:

If every directed path in Π from the contradiction $\perp$ to C contains the label $x = 1$, then $I_{C,\Pi}$ assigns $x = 1$; if every such path contains the label $x = 0$, then $I_{C,\Pi}$ assigns $x = 0$; Otherwise $I_{C,\Pi}$ does not assign x.

Note that, by definition, $I_{C,\Pi}$ assigns C to false since every literal in C must be made false on any path between C and $\perp$. The regularity assumption permits us to prove the following crucial consistency lemma.

Lemma 3.34. *Let τ be a Tseitin tautology on graph G with regular resolution refutation Π. If a clause C is reachable from the root $\perp$ by a path p in Π consistent with a critical assignment, then $crit_\tau(I_{C,\Pi}) = crit_\tau(p)$, and p and $I_{C,\Pi}$ assign all edges incident to this critical set identically.*

Proof. Let p be such a path from $\perp$ to C, consistent with a critical assignment σ with bad vertex b. Let e be any edge incident to b, and p' be any other path in Π from $\perp$ to C. Then we claim that p, p' assign e identically.

The assignment σ corresponds to a root-to-leaf path in Π extending p. Let q denote the portion of this path which occurs after it reaches C, so that pq is the entire root-to-leaf path. By the regularity assumption, $p'q$ is also consistent with some assignment, and by correctness of the proof, $p'q$ violates the axiom labeling the leaf that $p'q$ and pq both reach, which is also violated by pq. e is a variable of this axiom, so $p'q(e) = pq(e)$, and both do assign e. If p assigns e, then q does not assign e, so p' must assign e, and in the same way as p does. If p does not assign e, then q must, so p' cannot, by the regularity assumption.

By definition of $I_{C,\Pi}$, we conclude that $I_{C,\Pi}$ and p assign all edges incident to $crit_\tau(p)$ identically, and so $crit_\tau(I_{C,\Pi}) \subseteq crit_\tau(p)$ since $I_{C,\Pi}$ and p assign the boundary of $crit_\tau(p)$ in the same way and with the same parity. But since p also extends $I_{C,\Pi}$ then $crit_\tau(p) \subseteq crit_\tau(I_{C,\Pi})$, and so they are equal. □

For the rest of this section we will drop the subscript τ because there will be one fixed τ for our arguments. For regular resolution, our measure of complexity for a clause C will be $|crit(I_{\Pi,C})|$.

Definition 3.35. Let $\mathcal{L}_i^* = \{C \in \Pi : 2^i n^3 < |crit(I_{\Pi,C})| \leq 2^{i+1} n^3\}$ for $0 \leq i \leq \log_2(\ell/(4n^2)) - 1$. Each $\mathcal{L}_i^*$ represents a different complexity level. We say that a clause of Π has *medium complexity* if it is in one of the $\mathcal{L}_i^*$.

The following is a key property that makes regular resolution refutations much each easier for us to analyze than general resolution ones, namely that the complexity of

clauses decreases monotonically throughout the proof, insofar as the critical assignments are concerned.

Lemma 3.36. *Let Π be a regular resolution refutation of a Tseitin tautology. If there is a path consistent with a critical assignment from the root to clause C to D in Π then $crit(I_{D,\Pi}) \subseteq crit(I_{C,\Pi})$*

Proof. By Lemma 3.34, $crit(I_{C,\Pi}) = crit(p)$ for any path p from the root to C consistent with a critical assignment. Let q be any path from C to D in Π. Again, $crit(I_{D,\Pi}) = crit(pq)$, and the result follows immediately from the fact that by definition $crit$ is monotonically decreasing over partial assignments and that pq extends p. □

3.6.2 A probabilistic adversary

For this section we fix a connected graph $G = (V, E)$, a Tseitin tautology τ over G, and a regular resolution refutation Π of τ. Rather than fix a distribution over partial assignments or total assignments depending only on G as we did for general resolution, for regular resolution we can use a probabilistic adversary which navigates Π to define a distribution on assignments that depends on Π itself.

Definition 3.37. The *probabilistic adversary* A_Π responds to the queries in the proof Π in the following way. Let σ be the assignment that the adversary has given to the queries already asked, and suppose that Π queries edge e at the proof node reached by following the path labeled σ,

- If e is not a cut edge of $G|\sigma$, then the adversary responds randomly.
- If e is a cut edge of $G|\sigma$, then the adversary responds so as to maximize the size of the critical vertex set, breaking ties arbitrarily. More precisely, letting σ_0 and σ_1 denote the extensions of σ that assign 0 or 1 to e, respectively, the adversary answers according to the larger of $|crit(\sigma_0)|$ and $|crit(\sigma_1)|$.

It is notable that our adversary strategy is very similar to one used in studying Prover-Delayer games and lower bounds for tree-like resolution as in [20, 94, 93].

The sizes of the critical sets associated with the clauses visited by the adversary satisfy some basic properties analogous to ones shown in Proposition 3.14 and Lemma 3.15 for general resolution, as well as an extra monotonicity property.

Proposition 3.38. *Let p be any root-leaf path in Π in the support of the distribution induced by A_Π and let $C_0 = \bot, C_1, \ldots, C_r$ be the clauses labeling the nodes of p in order.*

1. $crit(I_{\bot,\Pi}) = V$.
2. $|crit(I_{C_r,\Pi})| = 1$.
3. $|crit(I_{C_{i+1},\Pi})| \leq |crit(I_{C_i,\Pi})|$ *for every* i *with* $0 \leq i < r$.
4. $|crit(I_{C_{i+1},\Pi})| \geq |crit(I_{C_i,\Pi})|/2 > 0$ *for every* i *with* $0 \leq i < r$.

Proof. Let p_i be the partial assignment given by the length i prefix of path p. Observe that $crit(I_{C_i,\Pi}) = crit(p_i)$. The first two are immediate, the third follows immediately because by definition $crit$ is monotone decreasing over partial assignments and the fourth follows from sub-additivity of $crit$ and the maximizing choice of the adversary. □

Corollary 3.39. *For any t with $1 \leq t \leq |V|/2$, the path taken by adversary A_π must pass through a clause C with $t < |crit(I_{C,\Pi})| \leq 2t$. In particular, the path taken by A_π must pass through a clause from every $\mathcal{L}_i^*$.*

To make use of the probability distribution given by the adversary, we will need to understand the probability that the adversary reaches a given in Π, as well as the conditional probability of reaching a clause given that the adversary has already followed some partial path Π. The following measure will allow us to bound these probabilities.

Definition 3.40. Let $\#(H)$ denote the number of connected components of a subgraph H of G. Define a function Φ on (pairs of) subsets of the edges E of G by

- $\Phi(A|B) = |A \backslash B| - \#(G|(A \cup B)) + \#(G|B)$, for $A, B \subseteq E$, and

- $\Phi(A) = \Phi(A|\emptyset)$ for $A \subseteq E$.

For ρ, σ, both partial assignments to the edges of G, we write $\Phi(\rho|\sigma)$ for $\Phi(\mathrm{dom}(\rho)|\,\mathrm{dom}(\sigma))$.

The following gives some simple intuition about Φ.

Proposition 3.41. *Let $A \subseteq E$. For any edge set A and edge $e \notin A$,*

$$\Phi(A \cup \{e\}) - \Phi(A) = \begin{cases} 0 & \textit{if } e \textit{ cuts } G|A \\ 1 & \textit{if not.} \end{cases}$$

Proof. By definition,

$$\begin{aligned} &\Phi(A \cup \{e\}) - \Phi(A) \\ &= |A \cup \{e\}| - |A| - \#(G|A \cup \{e\}) + \#(G|A) \\ &= 1 - \#(G|A \cup \{e\}) + \#(G|A). \end{aligned}$$

and the result follows immediately. □

Definition 3.42. We call a set of edges in G *independent* if and only if their removal from G does not increase the number of connected components of G.

Corollary 3.43. *$\Phi(A)$ is the maximum size of an independent set $A' \subseteq A$.*

Proof. Apply the Proposition 3.41 inductively starting with $\Phi(\emptyset)$ and adding the edges in A one at a time beginning with the edges in A'. □

Remark 3.44. $\Phi(A)$ is the rank of edge set A in the *cut matroid* (also known as on the *bond matroid*) of the graph G. This is the dual of the more well-known cycle matroid of G. The independent sets of the cut matroid are those sets of edges that do not separate any component of the graph, as in our definition. The rank of a set of edges A in the cut matroid is the the cardinality of the largest independent $A' \subseteq A$. In the literature, $\Phi(\cdot|\cdot)$ is called the *conditional rank* of the cut matroid.

Our reason for using this definition is the following lemma, which is the key to understanding the distribution on assignments given by our probabilistic adversary arguments. Though the formulation looks fairly different from the arguments based on the sizes of boundaries of critical sets that we used in the case of general resolution, it deals with a similar property. In particular, in the case of a connected set of vertices S (such as a critical set) whose complement also happens to be connected, it is easy to check that $\Phi(\delta(S)) = |\delta(S)| - 1$.

Lemma 3.45. *[Main Adversary Lemma] Let Π be a regular resolution refutation of a Tseitin tautology. For any clause C in Π of with information $I_{C,\Pi} = \rho$, the probability that adversary A_Π reaches C conditioned on following a path labeled σ is at most $2^{-\Phi(\rho|\sigma)}$. In particular, the probability that the adversary A_Π reaches a clause C with information $I_{C,\Pi} = \rho$ is at most $2^{-\Phi(\rho)}$.*

Proof. For a fixed ρ, we prove the bound by induction on the length of σ, with large σ as the base case.

If the assignment given by σ contains ρ, the result holds trivially, since the bounding expression is 1. Similarly, for σ not consistent with ρ the probability of the event is 0, so the bound holds.

Suppose inductively that the result is true for every assignment strictly larger than σ. If the adversary has followed the path labeled σ in Π, at which point edge e is resolved on, then, by the induction hypothesis, the bound holds for $\sigma_0 = \sigma \cup \{e = 0\}$ and $\sigma_1 = \sigma \cup \{e = 1\}$. We always have $\Phi(\rho|\sigma_0) = \Phi(\rho|\sigma_1)$ since σ_0 and σ_1 make assignments to the same set of edges.

Suppose that $\Phi(\rho|\sigma) \leq \Phi(\rho|\sigma_0)$. Since σ is not as large as ρ, the probability that the adversary reaches C after following σ is a weighted average of the probabilities that the adversary reaches C after following σ_0 and σ_1, respectively. Therefore, by induction, the conditional probability of reaching C after following σ is at most $2^{-\Phi(\rho|\sigma_0)} \leq 2^{-\Phi(\rho|\sigma)}$ as required.

Suppose now that $\Phi(\rho|\sigma_0) < \Phi(\rho|\sigma)$. There are two subcases, depending on whether e is assigned by ρ. Consider first the subcase that e is assigned by ρ. Then

$\mathrm{dom}(\rho) \cup \mathrm{dom}(\sigma) = \mathrm{dom}(\rho) \cup \mathrm{dom}(\sigma_0)$. Therefore, since $\Phi(\rho|\sigma_0) < \Phi(\rho|\sigma)$, we have

$$\begin{aligned} 0 &< \Phi(\rho|\sigma) - \Phi(\rho|\sigma_0) \\ &= |\mathrm{dom}(\rho) \setminus \mathrm{dom}(\sigma)| - |\mathrm{dom}(\rho) \setminus \mathrm{dom}(\sigma_0)| \\ &\quad -[\#(G|(\rho \cup \sigma)) - \#(G|(\rho \cup \sigma_0))] \\ &\quad +\#(G|\sigma) - \#(G|\sigma_0) \\ &= 1 + \#(G|\sigma) - \#(G|\sigma_0). \end{aligned} \tag{3.1}$$

Hence $\#(G|\sigma) - \#(G|\sigma_0) > -1$. Therefore, since $\#(G|\sigma) \leq \#(G|\sigma_0)$ we must have $\#(G|\sigma) = \#(G|\sigma_0)$, which means that e is not a cut edge of $G|\sigma$. Moreover, (3.1) implies that $\Phi(\rho|\sigma_0) = \Phi(\rho|\sigma) - 1$. By the definition of the adversary, the value the adversary assigns to e is chosen randomly in this case. However, one of σ_0 and σ_1 is inconsistent with ρ because ρ assigns some particular value to e. So, without loss of generality,

$$\Pr_{adv}[\rho|\sigma] = \frac{1}{2} \Pr_{adv}[\rho|\sigma_0] \leq 2^{-\Phi(\rho|\sigma)},$$

as desired. Now consider the subcase that e is not assigned by ρ. In this case we follow the definitions of $\Phi(\rho|\sigma_0)$ and $\Phi(\rho|\sigma)$ and observe that $|\mathrm{dom}(\rho) \setminus \mathrm{dom}(\sigma_0)| = |\mathrm{dom}(\rho) \setminus \mathrm{dom}(\sigma)|$. Since $\Phi(\rho|\sigma_0) < \Phi(\rho|\sigma)$ we must have

$$\begin{aligned} &0 < \Phi(\rho|\sigma) - \Phi(\rho|\sigma_0) \\ &= -[\#(G|(\rho \cup \sigma)) - \#(G|(\rho \cup \sigma_0))] + \#(G|\sigma) - \#(G|\sigma_0) \end{aligned} \tag{3.2}$$

and hence

$$\#(G|(\rho \cup \sigma_0)) - \#(G|(\rho \cup \sigma)) > \#(G|\sigma_0) - \#(G|\sigma).$$

The left hand side is equal to 1 or 0 indicating whether e cuts $G|(\rho \cup \sigma)$, and the right hand side indicates whether e cuts $G|\sigma$. The inequality implies the former is one and the latter is zero, hence e is not a cut edge of $G|\sigma$ but it is a cut edge of $G|(\rho \cup \sigma)$. Plugging these facts back in (3.2), we see that the potential drop is exactly one; that is,

$\Phi(\rho|\sigma_0) = \Phi(\rho|\sigma) - 1$. Again, by the definition of the adversary, e is assigned randomly here.

Claim 3.46. Only one of σ_0, σ_1 can permit the adversary to reach ρ.

Suppose to the contrary that π_0 and π_1 extend σ_0 and σ_1 respectively, each corresponding to paths that can be travelled by the adversary and that meet at the target clause C with information ρ. Then, $crit(\pi_0) = crit(\pi_1) \neq \emptyset$, by Lemma 3.34 and Proposition 3.38 (4). But, by the definition of common information, π_0 and π_1 extend $\sigma_0 \cup \rho$ and $\sigma_1 \cup \rho$ respectively, and by Proposition 3.11 these two assignments have disjoint critical sets since e cuts $G|(\rho \cup \sigma)$. Since by definition $crit$ is monotone decreasing over partial assignments, $crit(\pi_0)$ and $crit(\pi_1)$ are also disjoint, a contradiction.

The claim now implies the bound we need exactly as before, and thus completes the proof. □

If C is a clause and p is a path from the root of Π, it is most convenient to abbreviate the conclusion of the lemma as

$$\Pr_{adv}[C|p] \leq 2^{-\Phi(I_{C,\Pi}|p)} .$$

For our superpolynomial bounds we need to use the following properties of Φ, all of which are consequences of Φ being a matroid rank function. While these can be proved easily using matroid theory, we sketch simple proofs for our specific case without relying on matroids.

Proposition 3.47. *For all non-empty sets of edges A, B, and C in G,*

(a) $\Phi(\cdot)$ *is non-negative and monotone increasing.*

(b) $\Phi(A|B) = \Phi(A \cup B) - \Phi(B)$.

(c) (Non-negativity and Monotonicity) $\Phi(\cdot|\cdot)$ *is non-negative, increasing in its first argument, and decreasing in its second argument.*

(d) (Chain Rule) $\Phi(A) \leq \Phi(A|B) + \Phi(B)$. *More generally,* $\Phi(A|C) \leq \Phi(A|B) + \Phi(B|C)$ *when* $B \supseteq C$.

(e) (Subadditivity) $\Phi(A \cup B|C) \leq \Phi(A|C) + \Phi(B|C)$.

Proof. Part (a) is immediate from Corollary 3.43. By definition,

$$\begin{aligned}
&\Phi(A \cup B) - \Phi(B) \\
&= |A \cup B| - |B| - \#(G|(A \cup B)) + \#(G|B) \\
&= |A \backslash B| - \#(G|(A \cup B)) + \#(G|B)
\end{aligned}$$

so part (b) follows. The nonnegativity and monotonicity in the first argument given in (c) follow from (b) and the monotonicity from (a). The monotonicity in the second argument given in (c) follows from (b), Proposition 3.41, and the fact that if e is not a cut edge with respect to $B \supseteq B'$ then e is not a cut edge with respect to B' either. For part (d), since $C \subseteq B$, we have $B \cup C = B$ and $\Phi(A \cup C) \leq \Phi(A \cup B)$ by the monotonicity of Φ. Therefore by (b),

$$\begin{aligned}
\Phi(A|C) &= \Phi(A \cup C) - \Phi(C) \\
&\leq \Phi(A \cup B) - \Phi(B) + \Phi(B \cup C) - \Phi(C) \\
&= \Phi(A|B) + \Phi(B|C)
\end{aligned}$$

which yields the chain rule. By the chain rule,

$$\Phi(A \cup B|C) \leq \Phi(A \cup B|B \cup C) + \Phi(B \cup C|C) \,,$$

but by definition $\Phi(B \cup C|C) = \Phi(B|C)$, and also $\Phi(A \cup B|B \cup C) = \Phi(A|B \cup C)$ which, by monotonicity, is bounded above by $\Phi(A|C)$. Hence subadditivity (e) follows. □

We will also need this fact, specific to cut matroids.

Lemma 3.48. *[Matroid Cut Argument] Suppose that S is a set of vertices of G, and that $\delta(S) \subseteq B$. Then for any A,*

$$\Phi(A|B) = \Phi(A \sqcap S|B \sqcap S) + \Phi(A \sqcap \overline{S}|B \sqcap \overline{S}) \,,$$

where $E' \sqcap V' := \{e \in E' : e \text{ is incident to } V'\}$, for E' a set of edges and V' a set of vertices.

Proof. We can easily prove this by induction on $|A|$. Suppose that $e \notin A$. We would like to show that when e is added to A, the equality still holds. If $e \in B$, then none of $\Phi(A|B)$, $\Phi(A \sqcap S|B \sqcap S)$, or $\Phi(A \sqcap \overline{S}|B \sqcap \overline{S})$ can change; so, without loss of generality $e \notin B$. This implies that $e \notin \delta(S)$ as well. Without loss of generality, e is adjacent to S but not $\overline{S}$, so the only terms that can change are $\Phi(A|B)$ and $\Phi(A \sqcap S|B \sqcap S)$. It is enough to show that these terms must change in the same way when e is added; so, in fact, it is enough to see that $\Phi(A \cup B)$ changes the same way as $\Phi((A \cup B) \sqcap S)$ does when e is added. In light of Proposition 3.41, this holds because an edge e is a cut edge in a graph G if and only if it is a cut edge in the component containing it. □

3.6.3 Isoperimetric inequality for grid graphs

In light of Lemma 3.45, for regular resolution we replace the "isoperimetric inequality" used for general resolution which showed that medium sized sets S have have many edges on their boundaries (large $\delta(S)$) by one showing that such sets have large $\Phi(\delta(S))$, i.e. large independent subsets in $\delta(S)$.

For every medium size set of vertices S of $G_{n,\ell}$, we will show that $\delta(S)$ has large rank ($\Phi(\delta(S))$ is large) rather than large cardinality. Moreover, we show that for several medium-sized sets of vertices $S_1 \ldots S_k$ of sufficiently separated cardinalities, the rank of $\bigcup_i \delta(S_i)$ grows linearly with k. We will be able to prove a tight lower bound on these ranks without much more difficulty, and without giving up much compared to cardinality.

The following is an analog of the properties we proved in the case of general reso-

lution using transition points and Lemma 3.5 about endpoints of intervals of increasing sizes. The proof here uses the same lemma on intervals but directly uses common information rather than transition points.

Lemma 3.49. *Let $S_1 \dots S_k$ be sets of vertices in $G_{n,\ell}$. If $2kn^2 \leq |S_1|$, for each $i \in [k-1]$, $4|S_i| \leq |S_{i+1}|$, and $|S_k| \leq n\ell/4$, then $\Phi(\bigcup_i \delta(S_i)) \geq k(n-1)$.*

Proof. If any S_i contains $k(n-1)$ partial columns, then each such column contributes a vertical edge to $\bigcup_i \delta(S_i)$, and these vertical edges together form an independent set of the necessary size, so we are done.

If no S_i contains $k(n-1)$ partial columns, then by the size bounds on S_i, each contains a full column, and an empty column. In particular, each S_i contains a horizontal edge in every row. We will show that each row contains at least k edges of $\bigcup_i \delta(S_i)$. Choosing any $n-1$ rows and the k edges from each such row give an independent set of size $k(n-1)$, so this will finish the proof.

Fix a row r. Let a_i be the column number of the leftmost vertex in S_i and in the r-th row, let b_i be the column number of the rightmost such vertex. Then S_i has boundary edges with left endpoints in columns $a_i - 1$ and b_i, provided that these are not 0 or ℓ. Thus all we need to show is that

$$\left| \bigcup_i \{a_i - 1, b_i\} \setminus \{0, \ell\} \right| \geq k \, .$$

Since 0 and ℓ are the same modulo ℓ, it suffices to show that these intervals meet the conditions for the first part of Lemma 3.29.

Since S_i has fewer then $k(n-1)$ partial columns, $b_i - (a_i - 1) > |S_i|/n - k(n-1)$. On the other hand, the number of full columns in S_i is at most $|S_i|/n$, so $b_i - (a_i - 1) < |S_i|/n + k(n-1)$. Thus the ratio of successive differences is at least

$$\begin{aligned} \frac{b_{i+1} - (a_{i+1} - 1)}{b_i - (a_i - 1)} &> \frac{|S_{i+1}| - kn(n-1)}{|S_i| + kn(n-1)} \\ &\geq \frac{4|S_i| - kn(n-1)}{|S_i| + kn(n-1)} \, . \end{aligned}$$

Since $|S_i| \geq |S_1| \geq 2kn^2$, the ratio is at least $\frac{7}{3} > 2$ and we can apply Lemma 3.29 as desired. □

It is convenient to have a lower bound on $|S_1|$ that does not depend on k. To do this we need to upper bound ℓ.

Corollary 3.50. *Suppose that $\ell \leq 2^n$. Let $S_1 \dots S_k$ be sets of vertices in $G_{n,\ell}$. If $n^3 \leq |S_1|$, $4|S_i| \leq |S_{i+1}|$ for each i, and $|S_k| \leq n\ell/4$, then $\Phi(\bigcup_i \delta(S_i)) \geq k(n-1)$.*

Proof. Since $\ell \leq 2^n$, and $|S_1| \geq n^3$, we have $k \leq 1 + \log_4(|S_k|/|S_1|) \leq \log_2 n/2 \leq n/2$ and Lemma 3.49 applies. □

3.6.4 Regular resolution time-space tradeoff for grid graphs

To prove Theorem 3.31, we will begin with a regular resolution refutation Π, and repeatedly subdivide Π into polynomially many epochs, at each point choosing an epoch in which enough progress happens that we may continue subdividing it. The process peters out after $\mathbf{O}\left(\frac{\log L}{\log\log L}\right)$ steps.

By Corollary 3.39, the adversary path will pass through clauses of every medium complexity level. In this analysis, an epoch in the refutation is viewed as representing a lot of progress if there is a path p to a clause appearing in that epoch such that, conditioned on having followed p, the adversary *probably* reaches a *much smaller* complexity clause by the end of that epoch than it began with. The major technical step is to show how to divide an epoch that represents a significant amount of progress into much smaller epochs, one of which does a comparable amount of work.

Lemma 3.51. *Let $0 < \epsilon < 1$ and Π be a regular resolution refutation of a Tseitin tautology on the grid graph $G_{n,\ell}$ for $\ell \leq 2^n$. Suppose that p is a path in Π from the root to some clause C. Let $0 \leq \ell_1 < \dots < \ell_k$ satisfy $\ell_{i+1} \geq \ell_i + 3$ for each $i \in [k]$. If for each $i \in [k]$ there is some $C_i \in \mathcal{L}^*_{\ell_j}$ appearing in Π with $\Pr_{adv}[C_i|p] \geq 2^{-(1-\epsilon)(n-1)}$ then $\Phi(I_{C,\Pi}) \geq k\epsilon(n-1)$.*

Proof. Fix the choices of ℓ_i and the associated clauses C_i. In order to make it relatively likely to reach each of the C_i from C as in the hypothesis, we show that the extra

information at C_i over that at C cannot be too large. On the other hand we can show, using the isoperimetric properties of the grid graph, that the clauses C_i in total have a large amount of information and hence C must also. We start with the latter.

Let $S_i = crit(I_{C_i,\Pi})$ for $i = 1, \ldots, k$ and $S = crit(p)$. Since $\ell_{i+1} \geq \ell_i + 3$ we have $|crit(I_{C_{i+1},\Pi})| \geq 4|crit(I_{C_i,\Pi})|$ and Corollary 3.50 implies that

$$\Phi(\bigcup_i \delta(S_i)) \geq k(n-1) \tag{3.3}$$

On the other hand, by Lemma 3.45, the hypothesis that $\Pr_{adv}[C_i|p] \geq 2^{-(1-\epsilon)(n-1)}$ implies that $\Phi(I_{C_i,\Pi}|p) \leq (1-\epsilon)(n-1)$ where, as usual, we have identified p with the partial assignment labeling it. Moreover, since $\Pr_{adv}[C_i|p] > 0$, there is some path q_i from C to C_i in Π. Therefore,

$$S_i = crit(I_{C_i,\Pi}) = crit(pq_i) \subseteq crit(p) = S,$$

where the third equality follows from Lemma 3.34 and the containment follows since by definition $crit$ is monotone decreasing over partial assignments. Note that Lemma 3.34 also implies that $p \sqcap S = I_{C,\Pi} \sqcap S$ and Corollary 3.12 implies that $\delta(S) \subseteq \mathrm{dom}(p)$. Therefore, we may apply Proposition 3.48 with $B = \mathrm{dom}(p)$ to obtain

$$\begin{aligned}
\Phi(I_{C_i,\Pi}|p) &\geq \Phi(\delta(S_i)|p) && \text{by monotonicity} \\
&= \Phi(\delta(S_i) \sqcap S|p \sqcap S) + \Phi(\delta(S_i) \sqcap \overline{S}|p \sqcap \overline{S}) \\
&&& \text{by Lemma 3.48} \\
&\geq \Phi(\delta(S_i) \sqcap S|p \sqcap S) && \text{by nonnegativity} \\
&= \Phi(\delta(S_i)|p \sqcap S) && \text{since } S_i \subseteq S \\
&= \Phi(\delta(S_i)|I_{C,\Pi} \sqcap S) && \text{by Lemma 3.34} \\
&\geq \Phi(\delta(S_i)|I_{C,\Pi}) && \text{by monotonicity.}
\end{aligned}$$

It follows that $\Phi(\delta(S_i)|I_{C,\Pi}) \leq (1-\epsilon)(n-1)$ for each $i \in [k]$. Hence, by the subaddi-

tivity of Φ given in Proposition 3.47 we have

$$\Phi(\bigcup_i \delta(S_i)|I_{C,\Pi}) \leq (1-\epsilon)k(n-1).$$

On the other hand, by the chain rule of Proposition 3.47 and (3.3) above, we have

$$\begin{aligned}\Phi(I_{C,\Pi}) &\geq \Phi(\bigcup_i \delta(S_i)) - \Phi(\bigcup_i \delta(S_i)|I_{C,\Pi}) \\ &\geq k(n-1) - (1-\epsilon)k(n-1) = \epsilon k(n-1)\end{aligned}$$

which is what we needed to prove. □

The real value of Lemma 3.51 is in its contrapositive. If the proof is small, then by Lemma 3.45, the vast majority of the time the adversary path will only reach clauses C with $\Phi(I_{C,\Pi})$ small. Moreover, there can only be a few medium complexity levels of clauses that the adversary is now relatively likely to visit. For most of the levels, the adversary on visiting C has essentially no greater chance to reach any clauses of that complexity level than the chance when the adversary began at the $\perp$ clause.

With this lemma in hand we can now try to implement the overall plan for the proof of Theorem 3.31. Based on some parameters that we will set later, we first specify a property of clauses, which, by Lemma 3.45 and a union bound, are rarely encountered by an adversary in any refutation that is not too large.

Definition 3.52. Let $0 < \epsilon < 1$ and m be parameters, which we will fix later. We say that a clause C has *high potential* if $\Phi(I_C) \geq \epsilon m(n-1)$.

Recall that a refutation is a *sequence* of clauses ending in $\perp$, and that the adversary walks down the proof DAG starting at $\perp$, which involves moving backwards through this sequence, possibly jumping over intermediate clauses. The ordering on the sequence of clauses gives a sequence of *time steps*. Write $time(C)$ for the time step in the proof sequence in which clause C appears. Recall that every arc in the proof DAG that crosses from one time step to an earlier one corresponds to a clause that must be in memory during all intervening time steps. Moreover, by Corollary 3.39, the adversary

path must pass through clauses in every $\mathcal{L}_i^*$ starting from large values of i and ending with $i = 0$. For a medium complexity clause C we write $level(C)$ to denote the i such that $C \in \mathcal{L}_i^*$. These properties motivate us to define a way to measure the progress of a portion of the proof.

Definition 3.53. Let C be a medium complexity clause in Π. Say that a path p in Π from the root to clause C is a (T, gap, δ)-*path in* Π if, conditioned on following p, with probability at least $1 - \delta$, the adversary A_Π

- does not reach any high potential clauses, and
- reaches a clause C' with $level(C') \leq level(C) - gap$ such that $time(C') \geq time(C) - T$.

Lemma 3.54 (Inductive Step). *Let Π be a space S regular resolution refutation of a Tseitin tautology on $G_{n,\ell}$ for $n^3 \leq \ell \leq 2^n$. Suppose that p is a (T, gap, δ)-path in Π, $gap \geq 1$, and B is any natural number. Then there exists a (T', gap', δ')-path p' in Π extending p such that*

- $T' = \lceil T/B \rceil$,
- $gap' = \lfloor \frac{gap}{3m} \rfloor$, *and*
- $\delta' = \delta + B \cdot S \cdot 2^{-(1-\epsilon)(n-1)}$.

Proof. If $\delta \geq 1$ then the claim is vacuous, so assume that $\delta < 1$. Let C be the clause reached by p. By hypothesis, $level(C) \geq gap$ and C must not have high potential.

Therefore, by the contrapositive of Lemma 3.51 with $k = m$, there do not exist m complexity levels $\ell_1 < \ldots < \ell_m$ with $\ell_{i+1} \geq \ell_i + 3$ such that there is a clause $D_i \in \mathcal{L}_{\ell_i}^*$ with $\Pr_{adv}[C_i|p] \geq 2^{-(1-\epsilon)(n-1)}$. In particular, this means that there do not exist $3m$ distinct complexity levels of clauses $\mathcal{L}_j^*$ such that there is a clause $D \in \mathcal{L}_j^*$ with $\Pr_{adv}[D|p] \geq 2^{-(1-\epsilon)(n-1)}$.

It follows that there exists a sequence of at least $gap' = \lfloor \frac{gap}{3m} \rfloor$ *consecutive* complexity levels between $level(C)$ and $level(C) - gap$, such that for any D in one of

these levels, $\Pr[D|p] \leq 2^{-(1-\epsilon)(n-1)}$. Let i' be the largest level in this sequence, and $i' - gap' + 1$ be the smallest.

Divide the T time steps between $time(C)$ and $time(C) - T$, into B epochs of length at most $T' = \lceil T/B \rceil$, and let $\mathcal{M}$ denote the union over the B breakpoints between these epochs (including the start of the first epoch), of the sets of clauses in memory that are of complexity level between i' and $i' - gap' + 1$. By a union bound,

$$\Pr[\text{adversary reaches some } C'' \in \mathcal{M}] \leq B \cdot S \cdot 2^{-(1-\epsilon)(n-1)}.$$

Thus, except with probability at most δ', the adversary A_Π, conditioned on following p,

1. does not hit any high potential clauses,
2. reaches a clause C' with $level(C') \leq level(C) - gap$ and $time(C') \geq time(C) - T$, and
3. does not hit any clause in $\mathcal{M}$.

By averaging, there must exist a path p' extending p and reaching a clause $C' \in \mathcal{L}^*_{i'}$ such that this also holds, except with probability at most δ'.

Then, we claim that p' is a (T', gap', δ')-path. Let C' denote the clause reached by p'. C' falls in some epoch of length at most T'. If no clause in $\mathcal{M}$ is reached and the adversary has followed p', then by the beginning of the epoch containing C', the complexity level of the clause reached by the adversary must be less than or equal to $i' - gap'$. Thus, by construction, p' is indeed a (T', gap', Δ')-path in Π. □

Additionally, whenever we have a (T, gap, δ)-path p in Π with nontrivial gap and δ parameters, we can show that T is nontrivial.

Lemma 3.55 (Base Case). *Let Π be a regular resolution refutation of a Tseitin tautology on $G_{n,\ell}$ for $n^3 \leq \ell \leq 2^n$. If p is a (T, gap, δ)-path in Π with $gap > 3m$, then $T \geq (1-\delta)2^{(1-\epsilon)(n-1)}$.*

Proof. If $\delta \geq 1$, the claim is trivial. Suppose that $\delta < 1$. Let C be the clause reached by p and consider the clauses in the epoch between time steps $time(C) - T$ and $time(C)$ in Π.

By Lemma 3.51, there exists some complexity level $\mathcal{L}_i^*$ with $i > level(C) - gap$ such that, conditioned on following p, the adversary A_Π

- reaches a clause $C' \in \mathcal{L}_i^*$, with $time(C') \geq step(C) - T$ with probability at least $(1 - \delta)$, and
- does not reach any fixed clause D in $\mathcal{L}_i^*$, except with probability at most $2^{-(1-\epsilon)(n-1)}$.

By a union bound over the clauses in this epoch, we conclude that $T \geq (1-\delta)2^{(1-\epsilon)(n-1)}$.

□

We now have the ingredients needed to complete the proof of our time-space trade-off for regular resolution.

of Theorem 3.31. We begin with a regular resolution proof Π of length T that uses space S and refutes a Tseitin tautology on $G_{n,\ell}$. There are $L = \log_2(\ell/(4n^2))$ distinct medium complexity levels of clauses with respect to T. By Corollary 3.39, the path followed by adversary A_Π must pass through clauses with each of these complexity levels.

The outline of the remainder of the proof is as follows:

- First we show, via a union bound over the number of steps in the proof, that there is a (T, L, δ_0)-path in Π for some suitably small δ_0. This step only involves calculating the probability that the adversary passes through some high potential clause. This calculation will require that $\epsilon \cdot m$ not be small compared to $\frac{\log_2 T}{\log_2 n}$.
- Next we choose an appropriate value for B and continually apply the inductive step from Lemma 3.54, until either the accumulated error δ becomes too large, or the gap becomes too small. (We use the same value of B at each step.)
- Finally, some number r of rounds, just before gap has become too small after r rounds, we apply the base case Lemma 3.55 to deduce that the T' at the last step is reasonably large, and that $T \geq B^r \cdot T'$.

Let r denote the number of rounds that we will apply the inductive step and let gap_i denote the value of gap after each round and δ_i denote the value of δ after each round. We will set $gap_0 = L = \log_2(\ell/(4n^2))$. By Lemma 3.45, the probability that A_Π reaches a high potential clause is at most $2^{-\epsilon m(n-1)}$. Therefore if

$$T \cdot 2^{-\epsilon m(n-1)} \leq \delta_0 \tag{3.4}$$

the adversary must reach some clause of complexity L. Fix any path to such a clause. We obtain that this path is a (T, L, δ_0)-path,

We will choose B so that gap, rather than the error δ, is the limiting resource, so we take

$$r = \frac{\log_2 L}{\log_2(3m)} - 1 \tag{3.5}$$

Observe that by Lemma 3.54, $\delta_r = \delta_0 + r \cdot B \cdot S \cdot 2^{-(1-\epsilon)(n-1)}$. and applying Lemma 3.55 after the r-th round point yields $T' \geq (1 - \delta_r)2^{(1-\epsilon)(n-1)}$. By choosing

$$\delta_0 = \frac{1}{3} \text{ and } B = \frac{1}{3} \cdot \frac{2^{(1-\epsilon)(n-1)}}{S \cdot r},$$

we obtain that $\delta_r = 2/3$ and hence

$$T \geq \frac{1}{3} 2^{(1-\epsilon)(n-1)} \cdot \left(\frac{2^{(1-\epsilon)(n-1)}}{3 \cdot S \cdot r} \right)^r. \tag{3.6}$$

Together with the $n^3 \leq \ell \leq 2^n$ and the values of δ_0, L, and r, inequalities (3.4) and (3.6) provide the only constraints on our parameters. It remains to choose m and ϵ to optimize them and derive a convenient tradeoff lower bound. It is convenient to choose $m = \log_2 L$ so that $r = \log_2 L / \log_2(3m) - 1$ is between $\frac{1}{2} \log_2 L / \log_2 \log_2 L$ and $\log_2 L / \log_2 \log_2 L$. For convenience we also choose ϵ to be $2/\log_2 m = 2/\log_2 \log_2 L$. With these choices, the constraint (3.4) is satisfied whenever T is at most $\frac{1}{3} 2^{\frac{2 \log_2 L}{\log_2 \log_2 L}(n-1)}$ and this upper bound is strictly larger than the lower bound from (3.6). On the other hand, since $r \leq \log_2 L / \log_2 \log_2 L$ and $L \leq n$, the $3^{r+1} r^r$ in the denominator of the

expression in (3.6) is at most $2^{(1-\epsilon)(n-1)}$. Hence

$$T \geq \left(\frac{2^{(1-\frac{2}{\log_2 \log_2 L})(n-1)}}{S} \right)^{\frac{\log_2 L}{2 \log_2 \log_2 L}}.$$

Since L is grows at least logarithmically with n, the statement of the theorem follows.

□

3.7 Conclusion

We have shown superpolynomial size-space tradeoff lower bounds for resolution proofs which are the first to apply for superlinear space. The two different methods for deriving these bounds are based on a similar recursive decomposition of proofs into epochs. Our results suggest a number of open questions: Can this decomposition framework be applied to show size-space tradeoffs for stronger proof systems? With very small space, resolution size upper bounds for the Tseitin formulas we consider are $\log_2 n$ powers of their size for unlimited space. Is this tight? Can we increase the exponent in our lower bound from $\Theta(\log \log n / \log \log \log n)$ to $\Theta(\log n)$? More generally, is it true that for every k and every formula of size n with a proof of size n^k, there exists a proof in space $O(n)$ with size $n^{O(\log n)}$? with size 2^{n^ϵ} for any $\epsilon > 0$? Finally, our tradeoff lower bound can be viewed as a separation between two search paradigms: dynamic programming vs. divide and conquer. Can we find other settings in which these paradigms may be separated?

3.8 Some Upper Bounds

For the graphs we will consider, some generic upper bounds follow from bounds on their *cut width*. The first will form the high space upper bound in our time space tradeoff result, and the second is a low space upper bound we use to argue about the tightness of the lower bound we get for low space.

Definition 3.56. The *cut width* of a graph G is the smallest W such that there is a linear ordering of the vertices $v_1, \ldots, v_n$ such that, for every $1 \leq t \leq n$, there are no more

than W edges crossing the cut $(\{v_1, \ldots, v_t\}, \{v_{t+1}, \ldots, v_n\})$.

Throughout this section we will use the following well-known fact which we prove for completeness:

The *rank* of a resolution proof is the height of its proof DAG.

Observation 3.57. *Any tree-like proof of rank r has size at most 2^{r+1} and clause space at most $r + 1$.*

Proof. The first is an elementary fact about binary trees of height r. The second, we prove by induction. In the base case, suppose that a proof has rank 1. Then the proof consists of one step and so we need 2 units of space to hold both resolvents. Now suppose the inductive hypothesis holds for rank r, and that we have a proof of rank $r + 1$. Suppose C is the last clause of the proof, and A, B are its children. Then the derivations leading up to A, B are rank r. We then derive C as follows – run the derivation of A using r clause space, then A is the only remaining active clause by the tree-like assumption, so clear all of this space and store only A. Then, derive B using r clause space. When we have B, resolve it with A to finish the derivation of C. We only needed $r + 1$ space in total, as desired. □

Lemma 3.58 (Lemma 3.7). *Let G be a graph with n vertices, maximum degree d, and cut width W. Then there is a resolution refutation of a Tseitin tautology on G using $\leq n \cdot 2^{d+W}$ resolution steps, and space $\leq \cdot 2^W + d + 1$, plus space for the initial axioms.*

Proof. We will specify a sequence of $n + 1$ collections $\mathcal{C}_i$ of clauses, such that

1. $\mathcal{C}_1$ is a subset of the axioms .
2. $|\mathcal{C}_i| \leq 2^{W-1}$ for all i .
3. $\mathcal{C}_n = \{\bot\}$.
4. Given a configuration with $\mathcal{C}_i$ and the axioms in memory, we derive any clause in $\mathcal{C}_{i+1}$ using at most 2^{d+1} proof steps and $d + 1$ extra work cells.

This construction mirrors a construction in [38]. For each $1 \leq i < n$, let E_i be the collection of at most W edges crossing the cut $(\{v_1 \dots v_i\}, \{v_{i+1}, \dots, v_n\})$ as described in the assumption. Let $\mathcal{C}_i$ be the collection of $2^{|E_i|-1}$ clauses whose conjunction is semantically equivalent to the parity constraint $\bigoplus_{e \in E_i} x_e \equiv \bigoplus_{1 \leq j \leq i} \chi(j)$.

We've defined everything so that (1), (2) and (3) hold immediately. Now we will show that item (4) holds for $i < n - 1$.

The symmetric difference of E_i and E_{i+1} is, by definition, the edges incident tohas v_{i+1}. Let A_{i+1} denote the axioms associated with vertex v_{i+1}. By definition A_{i+1} is logically equivalent to the constraint $\bigoplus_{e \sim v_{i+1}} x_e \equiv \chi(i+1)$. If we think of parity constraints as $\mathcal{GF}(2)$-linear equations, and we add the two equations associated with A_{i+1} and $\mathcal{C}_i$, we obtain the equation associated with $\mathcal{C}_{i+1}$, so we conclude that

$$A_{i+1}, \mathcal{C}_i \models \mathcal{C}_{i+1} \, .$$

This move from $\mathcal{C}_i$ to $\mathcal{C}_{i+1}$ thus corresponds to adding two linear equations together and deleting one of them from memory.

Since resolution is implicationally complete, it is clear that it is possible to complete this move. We would like to now say that with at most 2^{d+1} size and $d+1$ space, we can derive any particular clause of $\mathcal{C}_{i+1}$, so that we can complete the move using $|\mathcal{C}_{i+1}| \cdot 2^{d+1}$ size and $d+1$ work space in addition to the space needed just to hold $\mathcal{C}_i, \mathcal{C}_{i+1}$ and the axioms.

Let C be an arbitrary clause in $\mathcal{C}_{i+1}$. It is well known that we may convert any derivation

$$A_{i+1}|_{\neg C} \, , \; \mathcal{C}_i|_{\neg C} \vdash \bot$$

into a derivation

$$A_{i+1}, \mathcal{C}_i \vdash C \, ,$$

using exactly the same proof DAG and possibly with some weakening steps added. It should be clear that weakening steps can be eliminated at the end of our construction without increasing the space or size used.

Since $\mathcal{C}_{i+1}$ is logically equivalent to a parity constraint, every $C \in \mathcal{C}_{i+1}$ assigns every variable E_{i+1}. Thus the only variables remaining in any of the $\mathcal{C}_i$ after the restriction $\neg C$ is applied correspond to edges incident to v_{i+1}. Thus there are at most d variables in any such derivation, and this derivation may be carried out trivially in a tree-like fashion using only 2^{d+1} size and $d+1$ space.

So, by completing a sequence of 2^{W-1} derivations each of length 2^{d+1}, reusing the $d+1$ space for each, we may fill 2^{W-1} new cells with the clauses in $\mathcal{C}_{i+1}$. We may then safely delete the cells used to hold $\mathcal{C}_i$, so that we never need more than $2 \cdot 2^{W-1} + d + 1$ space, plus the space for the axioms.

Finally we will see (4) holds for $i = n$, that is, from $\mathcal{C}_{n-1}$ and the axioms, we may derive a contradiction using 2^{d+1} size and $d+1$ space.

Since E_{n-1} is just the set of neighbors of v_n, $\mathcal{C}_{n-1}$ is logically equivalent to $\bigoplus_{e \sim v_n} x_e \equiv \bigoplus_{1 \leq j \leq n-1} \chi(j)$. But A_n is logically equivalent to $\bigoplus_{e \sim v_n} x_e \equiv \chi(n)$, which by assumption that χ is odd-charged, is of different parity. So $\mathcal{C}_{n-1}$ and A_n are contradictory sets of clauses, and on at most d variables. Thus a contradiction can be derived in tree-like fashion as claimed.

In total we carried out n phases, each with $2^{W-1}2^{d+1} = 2^{d+W}$ steps, and each using $2 \cdot 2^{W-1} + d + 1$ workspace in addition to the axioms. If we include the axioms in the space budget we will need additional space for $n2^{d-1}$ clauses. □

A little thought shows that this proof may be carried out in regular resolution, since within each phase, the derivation is tree-like, and the phases can be seen to operate on disjoint sets of variables.

We will appeal to this lemma to establish upper bounds for large space, against which we will prove size-space trade-offs. The following lemma shows that for Tseitin graphs satisfying the conditions of Lemma 3.7, which include the ones that we consider here, even radically restricted space can only increase the size required by a $O(\log n)$ power.

Lemma 3.59 (Lemma 3.8). *Under the same conditions as Lemma 3.7, the Tseitin tautology on $\tau(G)$ has a tree-like Resolution refutation using space $W\lceil \log n \rceil + 1$ and*

$\leq 2^{W\lceil \log n \rceil + 1}$ *resolution steps.*

Proof. We prove that when n is a power of two, $\tau(G)$ has a tree-like refutation of rank $W \log n$, which implies the claim. Suppose that G is such a graph of size n. Suppose inductively that the claim holds on graphs of size $n/2$. As before let $E_{n/2}$ denote the edges crossing the cut

$$(\{v_1, \ldots, v_{n/2}\}, \{v_{n/2+1}, \ldots, v_n\}) .$$

Let C be any clause containing every variable in $E_{n/2}$. Then $\tau(G)|_{\neg C}$ can be written as a pair of disjoint Tseitin formulae, one from each side of the cut, one of which is odd and therefore unsatisfiable. The induced subgraphs on either side of have cut width at most W, therefore $\tau(G)|_{\neg C}$ has a tree-like resolution refutation of rank at most $W(\log n - 1)$, which can be lifted to a tree-like derivation of C from the axioms $\tau(G)$ in the same rank.

It is easy to see that there is a tree-like refutation of rank $|E_{n/2}| \leq W$ using the set of all such C as axioms. By replacing the appearances of these axioms in that refutation with their $W(\log n - 1)$ rank tree-like refutations from the $\tau(G)$ axioms, we obtain a $W \log n$ rank tree-like refutation of $\tau(G)$. □

Generalizations

These ideas can be generalized to the stronger notion of carving width, yielding both a high space version and a low space version for any graph. For the high space version, we essentially just restate part of the main result of Aleknovich and Razborov [3]:

Definition 3.60. A *carving* of a graph G is a rooted binary tree $\mathcal{T}$ whose leaves are in bijection with the vertices of G. For t a node of $\mathcal{T}$, we let $v(t)$ denote the vertices corresponding to leaves which are under t, and $Cut(t)$ denote the edges on the boundary of $v(t)$. The size of the largest $Cut(t)$ is the *width* of a particular carving, and the *carving width* of a graph G is the width of the narrowest carving of G.

Observation 3.61. *The carving width is always less than the cut width – given any linear ordering of the vertices yielding cut width W, we can obtain a carving of width W*

by taking a maximally unbalanced binary tree, and assigning vertices to leaves as dictated by the linear ordering. The cuts corresponding to internal nodes then correspond exactly to the cuts occurring in the definition of cut width.

Corollary 3.62. *[3] For every n vertex graph G of carving width W, the corresponding Tseitin tautology has a regular resolution refutation of size $poly(n) \cdot 2^{O(W)}$, and uses comparable space.*

Sketch: There is a natural way to convert a carving into a refutation – for each node t, we will have an associated set of clauses on the variables $Cut(t)$:

For t a leaf corresponding to vertex v, we will simply take all the axioms associated to vertex v.

For t an internal node, we will take the clauses associated to its children t_1, t_2, which are on variables $Cut(t_1)$ or $Cut(t_2)$, and make all derivations possible which yield clauses on $Cut(t)$ and only which resolve on variables from $Cut(t_1) \cup Cut(t_2) \setminus Cut(t)$.

To analyze this, we observe there are at most $3^{|Cut(t)|}$ clauses derived for any node, and each one can be derived from clauses at the children by resolving on at most $2W$ variables, hence in 2^{2W} time. There are only $2n$ nodes in total since there are n leaves, and so there is $poly(n) \cdot exp(O(W))$ total work, and at most $poly(n) \cdot exp(O(W))$ active clauses at any point. We omit the proof that this actually does result in deriving a contradiction at the end – Aleknovich and Razborov refer to (Krajicek 1992, Theorem 4.2.1). It is straightforward to see this when considering only Tseitin tautologies, since then the clauses associated to t will always correspond to a parity constraint on the variables $Cut(t)$, and the derivation of t from t_1, t_2 corresponds to addition of linear equations; Aleknovich and Razborov in fact establish this claim for any tautology. □

Lemma 3.63. *For every graph G on n nodes with carving width W, the corresponding Tseitin tautology has a tree-like refutation of rank at most $W \log_{\frac{3}{2}} n$. In particular, it has a refutation with size $n^{W \log \frac{3}{2}}$, using at most $W \log_{\frac{3}{2}} n + 1$ clause space.*

Proof. Given an optimal carving T of G, we use the classic $\frac{1}{3}, \frac{2}{3}$ lemma for binary trees to find an edge which represents a balanced cut in T. Suppose that t is the lower vertex

of this edge. (More explicitly: observe that if we define a function f on nodes of the tree s.t. $f(t) = |v(t)|$, then f is a subadditive function on the tree, that is $f(t) \leq f(t_1) + f(t_2)$, where t_1, t_2 are the children of t. f of each leaf is 1, and f of the root is n, therefore there exists t such that $\frac{n}{3} \leq f(t) \leq \frac{2n}{3}$.)

First observe that the induced subgraph on $v(t)$, and the induced subgraph on $V \setminus v(t)$, both have carving width at most W, and at most $\frac{2}{3}n$ vertices by assumption. Therefore by induction hypothesis, Tseitin tautologies on these graphs have refutations of rank at most $W(\log_{\frac{3}{2}} n - 1)$. This implies that every clause containing exactly the variables of $Cut(t)$ has a tree-like derivation of rank at most $W(\log_{\frac{3}{2}} n - 1)$, since any such clause falsifies at least one component of $G \setminus Cut(t)$.

If we may start with all clauses on exactly $Cut(t)$, it is easy to see that there is a tree-like refutation of rank $|Cut(t)| \leq W$, by branching on each variable of $Cut(t)$ in succession. By replacing the leaves in this refutation with corresponding small rank tree-like derivations of these clauses, we get a tree-like refutation of rank at most $W \log_{\frac{3}{2}} n$ as desired. □

It is worth pointing out that while this lemma shows that small space proofs exist, it generally does not yield a small space method for finding them. The results of [3] show that it is possible to find a good carving or branch decomposition in small space; however, they left open the question of how to exploit this to solve small width SAT instances efficiently in small space. Important progress has been made by [11], [8].

Some algorithmic upper bounds from the literature

While we have described two different resolution *proofs* for Tseitin tautologies, it remains to be seen whether these upper bounds really correspond to the executions of actual algorithms. There has been a series of SAT algorithms [3, 11, 8] based on branch width or tree width that achieve this.

Claim 3.64. The branch-width based algorithm of Aleknovich and Razborov [3] gives an efficient implementation of Lemma 3.7, matching the space and time usage up to constants in the exponent. A recent algorithm of [8] achieves this as well.

Claim 3.65. The small space variation of [8] achieves a runtime matching the size of the proof in Lemma 3.8, up to constants in the exponent, and achieving memory usage which is polynomial in $n = |G|$. The branch width-based algorithm [11] achieves memory usage which is near linear in $|G|$, after a near optimal branch decomposition has been computed. Both results essentially appeal to the branch decomposition routine of [3] to achieve their results.

A further contribution of [8] is to provide a smooth interpolation between the high space parameters and the low space parameters. Their results are phrased in terms of tree-width parameterized SAT, and they show that time T and space S are feasible when

$$\alpha \left(\log T / TW(\phi) \log |\phi|\right) + \beta \log S \geq TW(\phi) + \gamma \log |\phi|,$$

for appropriate constants α, β, and γ. They conjecture that this essentially cannot be improved. Specifically, they conjecture that any SAT algorithm that runs in time T with $\log T = o(TW(\phi) \log(|\phi|))$ requires space exponential in $TW(\phi)$.

We note that one way to prove that this is true, at least for backtracking algorithms, would be to improve the lower bound result given in Lemma 3.17 for Tseitin tautologies, by improving the exponent to match the small space upper bounds.

4 Time Space Tradeoffs in Polynomial Calculus

4.1 Introduction

The satisfiability problem is of paramount importance to theoretical computer science. Despite a strong belief in the theory community that the problem is intractable in the worst case, in practice there are many important and successful approaches to solving it. Applied SAT solving has matured significantly in the last 10–15 years, and SAT solvers are now routinely used to solve real-world instances with hundreds of thousands, or even millions, of variables. Today, practitioners often think of SAT as an easy problem to reduce to, rather than a hard problem to reduce from.

Because the SAT algorithms used in practice depend crucially on complex heuristics, essentially the only known way to analyze their worst-case performance is by means of proof complexity. In this approach, the detailed heuristics are abstracted away, and instead the focus is on the proofs which these algorithms generate. Such proofs can be thought of as summarizing the transcripts of the computations, containing only the *reasoning* which took place. Despite this apparently significant loss of information, proof complexity nevertheless has managed to give tight exponential lower bounds on the worst-case running time on approaches for SAT used in practice by lower-bounding proof size. Note that there are many other results in other areas of a similar flavour—rather than directly trying to give lower bounds against, e.g., Turing machines and circuit families, one considers models which contain a canonical algorithm as well as all "nearby" algorithms in some sense. Instead of finding a concrete "bad example" against one algorithm, which might potentially be fixed or avoided, lower bounds in this style show that the whole approach has inherent limitations. For comparison, see, e.g., other work on linear programming hierarchies [42], semidefinite programming hierarchies [105], and algorithmic paradigms [1].

One important recent direction in proof complexity concerns size-space trade-offs. This research is partly driven by concerns about time and memory usage of SAT solvers—in practice, space consumption can be almost as much of a bottleneck as running time—

but is also motivated by the fundamental importance of time and space complexity in computation. Time-space trade-offs have historically been one of the most productive directions in computational complexity, and there have been many results in both Boolean and algebraic settings. Typically, the strongest such results show that in a variety of models, the product of time and space in any computation of some function is at least *nearly quadratic* in the input length, giving lower bounds against sublinear space. However, in many applications one has much more than just linear space available, and it is natural to ask whether one can show that there are problems that are solvable in polynomial time but for which any polynomial-time computation must require a large polynomial amount of space. The trade-off results presented in the current paper are the first in an algebraic setting which obtain superpolynomial time blow-up even in the superlinear space regime and also exponential blow-up for sublinear (but polynomial) space, and which still take place in a model which captures practical algorithms.

Our focus in this paper is on the proof systems *polynomial calculus* and *resolution*. Resolution is arguably the most well-studied proof system in proof complexity, and is directly connected with modern SAT solvers based on DPLL [55, 54] with clause learning, also known as *conflict-driven clause-learning (CDCL)* solvers [12, 76]. These algorithms use heuristic-driven backtracking search combined with a dynamic programming technique, and so far have clearly been the most successful approach to solving SAT in practice. Polynomial calculus instead takes an algebraic view of SAT. In this proof system, the disjunctive clauses of a CNF formula are translated into polynomials, and computations in the ideal generated by these polynomials show whether they have a common root or not, corresponding to a satisfying assignment for the formula. It was shown in [46] that such an algebraic approach might be significantly better than resolution-based approaches on some instances, and would never be much worse, so it might ultimately lead to a more "well-rounded" SAT solver. Phrased in the language of proof complexity, the extra expressive power of polynomials can lead to significantly more efficient proofs in terms of size, and perhaps also space. It was suggested in [46] that if our understanding of suitable heuristics could be improved, these algebraic tech-

niques could become competitive with resolution and provide a way around some of the bottlenecks encountered for CDCL solvers.

Intriguingly, however, despite significant progress on algebraic SAT solvers such as PolyBori[35], the gains of the polynomial approach anticipated by [46] have largely failed to materialize. Our research sheds light on one aspect of this, in that it investigates deeper the question of how much the expressive power of polynomials could reasonably be expected to translate into computational efficiency. We show that essentially all time-space trade-offs known for resolution also extend to polynomial calculus, and even to the stronger proof system *polynomial calculus resolution (PCR)* that unifies polynomial calculus and resolution, thus casting doubt on hopes of a generic improvement obtained by using polynomials. Based on what we know now, there seems not to exist any generic transformation of PCR proofs which improves time, improves space, or trades time and space in a way which outperforms what is possible in resolution.

We remark that the issue of time-space trade-offs for SAT is also connected to recent work of the third author on width-parameterized SAT [7], and our improved lower bounds help to strengthen the support contributed by [14] to their thesis.

For more information about proof complexity in general two good references are [13, 107], while the upcoming survey [85] by the second author focuses specifically on time-space trade-offs. A recent and very comprehensive reference on SAT solving is [32].

4.1.1 Previous Work

The resolution proof system appeared in [33] and began to be investigated in connection with automated theorem proving in the 1960s [54, 55, 103]. Despite the apparent simplicity of this proof system, the first superpolynomial lower bounds on proof size were obtained only in 1985 by Haken [64] after decades of study. Truly exponential size lower bounds were later proven by Urquhart [113] and Chvatál and Szemerédi [43]. The repertoire of size lower bound techniques remains fairly limited, however, including random restrictions [16, 64], the size-width method [31], and the pseudowidth

technique first employed by Raz [98] and further developed by Razborov [102].

Polynomial calculus was defined by Clegg, Edmonds and Impagliazzo [46]. In a technical break-through, Razborov [101] obtained degree lower bounds. This result was simplified by Impagliazzo et al. [72], who also showed that degree lower bounds imply proof size lower bounds in polynomial calculus.

The study of space in resolution was initiated by Esteban and Torán [57] and was later extended to a more general setting including other proof systems by Alekhnovich et al. [4]. Intuitively, the (clause) space of a resolution proof is the maximal number of clauses one needs to keep in memory while verifying the proof. Perhaps somewhat surprisingly, it turns out that linear space is enough to refute any unsatisfiable CNF formula, and a sequence of papers [4, 26, 57] have proven matching lower bounds.

Regarding trade-offs between size and space, some results in restricted settings were obtained in [25, 82] and strong trade-offs for full, unrestricted resolution were reported in the paper [30] involving the second author. These trade-offs only apply for space smaller than the linear worst-case upper bound, however. The recent work [14] by the first author with co-authors presented trade-off results that extend even to superlinear space.

Turning to polynomial calculus and PCR, the space measure (measuring the number of monomials, which is the natural generalization of clause space in resolution) has been quite poorly understood until very recently. While nontrivial space lower bounds were established already in [4], these bounds crucially work only for formulas of unbounded width, and it was only in [58] that space lower bounds for k-CNF formulas were shown. In a very recent paper [34] building on and developing the techniques in [4, 58], optimal linear lower bounds on PCR space were finally obtained (for random k-CNF formulas and pigeonhole-style formulas over bipartite expanders), but many intriguing questions about this measure remain open.

As to trade-offs between size and space, we are not aware of any such results for PC or PCR except the recent paper [69] involving the second author. An important distinction here, however, is that in order to speak about a "true" trade-off we want to

find formulas which have proofs in small size and also in small space, but for which any proof optimizing one of the measures provably has to pay a stiff penalty with respect to the other measure. While [69] exhibits formulas for which any proofs in small space must have very large size, no such small-space proofs are known to exist. (In fact, it would seem more likely that there are no small-space proofs for these formulas and that the small-size proofs are also optimal with respect to space—this is known to be the case in resolution for very similar formulas.)

As noted above, degree is an important auxiliary measure in PC and PCR, playing a role similar to that of width in resolution. However, whereas the relationship between size and degree in PC/PCR is known to be analogous to that between length and width in resolution, it is open whether monomial space and degree behave with respect to each other as clause space and width do in resolution.

4.1.2 Our Results

In this paper, we extend the trade-offs in [25, 30, 14], i.e., essentially all known trade-offs for resolution,[5] to polynomial calculus and PCR. Our first result is that there is a strong trade-off between degree and monomial space in polynomial calculus and PCR, completely analogous to the trade-off between width and clause space in resolution. (We refer to Sections 4.3 and 4.3.6 for precise definitions of terminology and notation used below.)

Theorem 4.1. *There is a family of explicitly constructible 3-CNF formulas F_n of size $\Theta(n)$ that can be refuted in polynomial calculus in degree $Deg_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}(1)$ and also in monomial space $Sp_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}(1)$, but such that for any PCR refutation $\pi_n : F_n \vdash \bot$ it holds that $Sp(\pi_n) \cdot Deg(\pi_n) = \Omega(n/\log n)$.*

What this theorem says is that although the formulas F_n can be refuted in essentially minimal degree and essentially minimal space even in PC, when we optimize one of these measures the other has to blow up to almost worst possible in PCR (the worst-

[5] It also seems likely that the trade-offs in [82] would carry over to PC/PCR, but since these results are clearly more artificial we have not looked into this.

case upper bound for both measures is linear in n). This result follows by studying the same so-called pebbling formulas as in [25] and doing a careful analysis of the proofs in [30], which yields a very useful generalization of the techniques there.

Our first set of time-space trade-off results follow by applying the same generalization of [30] to other pebbling formulas. Combining this with random restrictions, we obtain trade-offs where the upper bounds hold for PC (and resolution) while the lower bounds apply for the stronger PCR proof system. There is a slight loss in the parameters as compared to the results for resolution in [30], however, which is due to the random restriction argument, and in particular we do not get tightly matching upper and lower bounds. The trade-offs obtained are still fairly dramatic, though, and a nice extra feature is that they also hold even if we allow the PCR refutations to use exponentially stronger *semantic* rules where anything that follows semantically from what is currently in memory can be derived in one single step instead of by a sequence of syntactic steps.

As in [30], we get a whole collection of trade-offs, and we only give two concrete examples here. The first example is that for arbitrarily small but growing space complexity, there can be superpolynomial size-space trade-offs for PC and PCR.

Theorem 4.2. *Let $g(n) = \omega(1)$ be any arbitrarily slowly growing function*[6] *and fix any $\varepsilon > 0$. Then there are explicitly constructible 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that the following holds:*

- *The formulas F_n are refutable in polynomial calculus in total space $TotSp_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}(g(n))$.*
- *There are PC refutations π_n of F_n in simultaneous size $S(\pi_n) = \mathrm{O}(n)$ and total space $TotSp(\pi_n) = \mathrm{O}\left(\left(n/g(n)^2\right)^{1/3}\right)$.*
- *Any PCR refutation of F_n in monomial space $\mathrm{O}\left(\left(n/(g(n)^3 \log n)\right)^{1/3-\varepsilon}\right)$ must have superpolynomial size.*

[6] Technically speaking, we also need $g(n) = \mathrm{O}\left(n^{1/7}\right)$ here, i.e., that $g(n)$ does not grow too quickly. This restriction is inconsequential since for faster-growing functions other results presented in this paper yield stronger trade-offs anyway.

Note that this trade-off is quite robust in the sense that for the whole range of space from $\omega(1)$ up to almost $n^{1/3}$ the proof size required is superpolynomial. Note also that the trade-off result is nearly tight in the sense that the superpolynomial lower bound on size in terms of space reaches up to very close to where the linear upper bound kicks in.

As a second example, we state a trade-off where the proof size blows up exponentially when space is optimized.

Theorem 4.3. *There is a family of explicitly constructible* 6*-CNF formulas* $\{F_n\}_{n=1}^{\infty}$ *of size* $\Theta(n)$ *such that the following holds:*

1. *The formulas* F_n *are refutable in PC in total space* $TotSp_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}\big(n^{1/11}\big)$.
2. *There are PC refutations* π_n *of* F_n *in size* $S(\pi_n) = \mathrm{O}(n)$ *and total space* $TotSp(\pi_n) = \mathrm{O}\big(n^{3/11}\big)$.
3. *Any PCR refutation of* F_n *in monomial space at most* $n^{2/11}/(10 \log n)$ *must have size at least* $\big(n^{1/11}\big)!$.

As in [30], the fact that we are working with pebbling formulas means that we can only get time-space trade-offs in the sublinear space regime using these techniques, however. For our second set of time-space trade-off results, we instead study so-called Tseitin formulas and lift the trade-offs in [14] from resolution to PCR. In resolution, these are the only known trade-off lower bounds which hold for superlinear space. Quantitatively, what they show is that if the space is reduced below a polynomial factor of the size of the smallest known proofs, the size must grow as a super-constant power of the optimal size. Besides strengthening this result to PCR, we modify the construction and simplify the technical analysis significantly, which allows us to obtain our trade-offs for 8-CNF formulas, and not just for CNF formulas of unbounded width as in [14].

Theorem 4.4. *Let* $\mathbb{F}$ *be a field of odd characteristic. There is an explicitly constructible family of* 8*-CNF formulas* $\{F_{n,w}\}$*, with* $1 \leq w \leq n^{1/4}$*, which are of size* $\Theta(n)$ *and have the following properties:*

1. *The formulas F_n have resolution refutations π_n in (short) length $L(\pi_n) \leq n^{O(1)}2^w$ and clause space $Sp(\pi_n) \leq 2^w + n^{O(1)}$.*

2. *They also have resolution refutations π'_n in (small) clause space $Sp(\pi'_n) = \mathrm{O}\big(w \log n\big)$ and length $L(\pi'_n) \leq 2^{\mathrm{O}(w \log n)}$.*

3. *For any PCR refutation π_n of F_n over $\mathbb{F}$, the proof size is bounded by $S(\pi_n) = \left(\frac{2^{\Omega(w)}}{Sp(\pi_n)}\right)^{\Omega\left(\frac{\log\log n}{\log\log\log n}\right)}$.*

In fact, the parameter w in $F_{n,w}$ is the *tree-width* of the formula, and this is the reason for the connection with [7] discussed above. In this paper, it was shown that the resolution upper bounds in Theorem 4.4 can in fact be obtained by a tree-width based algorithm with little overhead, and furthermore that a smooth trade-off upper bound exists between the two ranges. It was conjectured in [7] that this algorithm cannot be improved, which if true would have significant computational complexity consequences.

The lower bounds in Theorem 4.4 can be interpreted as evidence supporting at least a weak form of the conjecture—it places hard limits on how much a restricted class of algorithms could concievably improve over the algorithm in [7]. While an important open question in this regard is improving the exponent obtained in the lower bound argument, it is also interesting from the standpoint of the conjecture to generalize the lower bound to stronger proof systems, since this will cover a broader class of algorithms.

4.2 Outline of This Chapter

The rest of this chapter is organized as follows. In Section 4.3, we give a detailed overview of our results and describe the main technical ingredients in the proofs. Formal proof complexity preliminaries are given in Section 4.3.6. In Section 4.3.7, we do a careful study of the techniques in [30], and our generalization immediately allows us to derive space-degree trade-offs for polynomial calculus in Section 4.4. In Section 4.5, we prove time-space trade-offs for polynomial calculus in the sublinear

regime. In Section 4.6, we show an isoperimetric inequality for a certain kind of graphs which is instrumental for obtaining CNF formulas with strong trade-off properties, and in Section 4.7 we extend the time-space trade-offs for superlinear space in [14] from resolution to PCR. Finally, Section 4.8 contains concluding remarks including a discussion of some of the many fascinating problems in this area that remain open.

4.3 Overview of Results and High-Level Proofs

Generally speaking, time-space trade-off results are usually established by some variation of the following plan:

1. Formalize a notion of *work* or *progress* specific to the model and the problem.
2. Divide the time period of a hypothetical computation into a large number of equal-sized *epochs*.
3. Prove the following claims:
   (a) If the epochs are small, then no single epoch makes very much progress.
   (b) If the space is small, then not much progress can be carried over from one epoch to the next.
   (c) To solve the problem, the computation needs to make substantial progress summed over all epochs.
4. Conclude that if the computation is too short and uses too little space, then this leads to a contradiction.

This approach has been implemented in a wide variety of models, including graph pebbling, straight line programs, branching programs, et cetera. To obtain quantitatively strong trade-offs, i.e., trade-offs exhibiting superpolynomial blow-up, in addition it can be necessary to subdivide into epochs *recursively*. Frequently, this kind of refined strategy can only be carried out directly in more limited models. One contribution of our work is that we manage to realize such a strategy in a model which is significantly more

general than what has previously been possible. To achieve this, we make careful use of restriction and reduction arguments.

In our first set of trade-offs, which extends the results of [30], we combine random restrictions with a space-faithful projection technique, showing that if there existed PCR refutations which were very efficient with respect to time and space on a certain kind of pebbling formulas, then there would be pebbling strategies for the underlying graphs which would be very efficient as well. Thus we are able to lift graph pebbling lower bounds to PCR.

In fact, our result is more general in that we obtain a kind of generic "hardness amplification" result for CNF formulas. We show that if a formula has a mild form of trade-off in resolution, then by making appropriate syntactic substitutions we obtain another formula which has strong trade-off properties in the stronger proof system PCR. Pebbling then comes into the picture simply because pebbling formulas have exactly the form of weak trade-offs in resolution that we need.

The main technical problem which we overcome is how to reduce PCR refutations of the substituted formulas to resolution refutations of the original formulas in a way that preserves space. In resolution, it is possible to construct space-preserving reductions without using restrictions. Unfortunately, these reductions provably fail for stronger proof systems such as cutting planes and PCR (and even PC), but it turns out that by using random restrictions we can salvage enough of this approach to get strong trade-offs for PCR.

In our second set of trade-offs, which extends the results of [14], we make two contributions. Firstly, we simplify and strengthen the main result in [14] by studying a slightly different formula family with appropriately chosen random restrictions. As a result of this, we can prove trade-offs for k-CNF formulas rather than formulas of asymptotically growing width. Secondly, and more substantially, we manage to implement the overall strategy of [14] in the context of PCR.

This part of our work is different from the generalization of [30] in that it does not obtain a trade-off by reducing to another model of computation—instead, we carry

out the plan outlined above directly in the proof system. In [14], this is achieved by using a *semantic measure* of complexity of clauses as a progress measure. One of the crucial technical steps is to prove an inequality showing that not only are clauses representing different progress levels within a certain range all wide, but that they are also "pairwise wide" in that for any pair of such clauses each clause contains many variables not occurring in the other clause. In the context of polynomial calculus, we would need to prove an analogous result using degree instead of width, but sadly such a claim simply is not true.

We circumvent this obstacle by examining the binomial technique of [39] for degree lower bounds in polynomial calculus. This technique is based on the observation that PC refutations of *binomial* systems—i.e., where each initial polynomial is a sum of two monomials—have a special form. Binomial systems are never hard with respect to size or space, but can be hard with respect to degree. The paper [39] obtains degree lower bounds by constructing an explicit pseudoideal, and also gives low-degree reductions from other non-binomial systems to binomial systems. In this way, it is possible to get degree lower bounds for non-binomial systems, which can in turn be used to obtain size lower bounds.

For our purposes, however, we need much more than just degree lower bounds. We therefore refine the technique of [39] by combining the ideas behind the low-degree reduction and the pseudoideal construction. For any PCR refutation of a Tseitin contradiction, we construct a simulation of it by a restricted form of PC which refutes a "Fourier transformed" version of the formula. This simulation does *not* preserve size or space, but it allows us to obtain a suitable measure of progress for size-space trade-off results. This is because in this restricted setting, the semantic measure is much better behaved, and thanks to this we can prove an analogue of the lemma in [14] discussed above. However, due to the change of variables which occurs it is not true that the simulation commutes with restriction; it is not possible to, e.g., kill the monomials of the "shadow proof" obtained from the simulation with restrictions and argue that the resulting proof simulates the restriction of the original proof. Instead, we use restrictions to

eliminate monomials in the original proof, and use key properties of the simulation to show that they cannot reappear in the shadow proof, thus limiting the progress which can be made during its epochs. We can then carry out the progress measurement argument in the shadow proof to obtain a contradiction, given that the original proof was too short and used too little space.

Thus, by carrying out different parts of the argument of [14] in different contexts and mediating between them with this simulation, we are able to establish quantitatively equivalent lower bounds in full PCR. The only other techniques known for degree lower bounds are from [6, 101]; as far as we are aware none of these techniques yield simulations, nor can they be used to obtain time-space trade-offs in the manner described here.

In the rest of this section, we give precise definitions of the formulas we study as well as a more detailed overview of the proof techniques. The intention is to provide a "roadmap" for the formal proofs that will follow in later sections. However, the reader that wants to get to the detailed proofs right away can safely skip this section and start reading instead in Section 4.3.6 onwards, and we also refer to Section 4.3.6 for any definitions or notation missing below.

4.3.1 Substitution Formulas

Let F be a CNF formula over variables $x, y, z, \ldots$ and let $f : \{0,1\}^d \to \{0,1\}$ be a Boolean function over d variables. Then we can obtain a new CNF formula by substituting $f(x_1, \ldots, x_d)$ for every variable x (where we assume that $x_1, \ldots, x_d$ are new variables that do not appear anywhere else) and then expand to conjunctive normal form. We will write $F[f]$ to denote the resulting *substitution formula*. For example, for the disjunctive clause $C = x \vee \overline{y}$ and the binary exclusive or function $f(x_1, x_2) = x_1 \oplus_2 x_2$ we have

$$\begin{aligned} C[\oplus_2] = \; & (x_1 \vee x_2 \vee y_1 \vee \overline{y}_2) \wedge (x_1 \vee x_2 \vee \overline{y}_1 \vee y_2) \\ & \wedge (\overline{x}_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2) \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee y_2) \enspace . \end{aligned} \tag{4.1}$$

One important observation is that if we hit $F[\oplus_2]$ with a random restriction ρ that sets one of x_1 and x_2 to a random value for every x and leaves the other variable unset, then $F[\oplus_2]\restriction_\rho$ will be the formula F except possibly for sign flips of the literals. It is well known that restrictions preserve resolution and PCR refutations, and so for any refutation $\pi : F[\oplus_2] \vdash \bot$ we have that $\pi\restriction_\rho$ is a refutation of F (modulo sign flips). It is not hard to show that if in addition π has small length/size, then it is likely that $\pi\restriction_\rho$ does not have any wide clauses (in resolution) or high-degree monomials (in PCR). This will be useful in what follows.

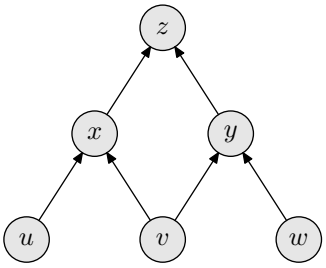
4.3.2 Pebbling Contradictions

Pebbling is a tool for studying time-space relationships by means of a game played on directed acyclic graphs (DAGs). Pebble games were originally devised for studying programming languages and compiler construction, but found a broad range of applications in computational complexity, during the 70s and 80s, which has expanded further during the last decade to cover also proof complexity. An excellent survey of pebbling up to ca. 1980 is [90], and some more recent developments are covered in the second author's upcoming survey [83].

The way pebbling results have been used in proof complexity has mainly been by studying so-called *pebbling contradictions*. These are CNF formulas encoding the pebble game played on a DAG G by postulating the sources to be true and the sink to be false, and specifying that truth propagates through the graph according to the rules of the pebble game.

Definition 4.5 (Pebbling contradiction [31]). Let G be a DAG with source vertices S and a unique sink vertex z. Identify every vertex $v \in V(G)$ with a propositional logic variable v. The *pebbling contradiction* over G, denoted Peb_G, is the conjunction of the following clauses:

- for all $s \in S$, a unit clause s (*source axioms*),
- For all non-source vertices v with immediate predecessors $pred(v)$, the clause $\bigvee_{u \in pred(v)} \overline{u} \vee v$ (*pebbling axioms*),



(a) Pyramid graph Π_2 of height 2.

$$
\begin{aligned}
& u \\
\wedge\ & v \\
\wedge\ & w \\
\wedge\ & (\overline{u} \vee \overline{v} \vee x) \\
\wedge\ & (\overline{v} \vee \overline{w} \vee y) \\
\wedge\ & (\overline{x} \vee \overline{y} \vee z) \\
\wedge\ & \overline{z}
\end{aligned}
$$

(b) Pebbling contradiction Peb_{Π_2}.

Figure 1: Example pebbling contradiction.

- for the sink z, the unit clause $\overline{z}$ (*sink axiom*).

For an example of a pebbling contradiction, see the CNF formula in Figure 1(b) defined in terms of the graph in Figure 1(a). If G has n vertices and maximal indegree ℓ, the formula Peb_G is an unsatisfiable $(1+\ell)$-CNF formula with $n+1$ clauses over n variables.

To make the connection back to Section 4.3.1, two examples of substituted version of the pebbling formula in Figure 1(b) are the substitution with logical or in Figure 2(a) and with exclusive or in Figure 2(b).

4.3.3 Substitution Theorem and Trade-offs Based on Pebbling

A paradigm that has turned out to be fruitful in many contexts in proof complexity is to take a CNF formula family $\{F_n\}_{n=1}^{\infty}$ with interesting properties, tweak it by substituting some function $f(x_1, \ldots, x_d)$ for each variable x as described in Section 4.3.1, and then use this new formula family to prove the desired result. In particular, the time-space trade-offs in [30] are obtained in this way. The techniques in [30] were developed specifically for resolution and the more general k-DNF resolution proof system, but a careful analysis of the proofs reveals that most of the approach can be carried over to other proof systems in a more general setting. We present this general setting below in the hope that it can be useful as an approach for proving space lower bounds and time-space trade-offs for proof systems such as PCR and cutting planes analogous to those for resolution and k-DNF resolution in [30]. And indeed, as we shall see soon, a

$$
\begin{aligned}
&(u_1 \vee u_2) \\
\wedge\ &(v_1 \vee v_2) \\
\wedge\ &(w_1 \vee w_2) \\
\wedge\ &(\overline{u}_1 \vee \overline{v}_1 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{u}_1 \vee \overline{v}_2 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{u}_2 \vee \overline{v}_1 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{u}_2 \vee \overline{v}_2 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{v}_1 \vee \overline{w}_1 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{v}_1 \vee \overline{w}_2 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{v}_2 \vee \overline{w}_1 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{v}_2 \vee \overline{w}_2 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{x}_1 \vee \overline{y}_1 \vee z_1 \vee z_2) \\
\wedge\ &(\overline{x}_1 \vee \overline{y}_2 \vee z_1 \vee z_2) \\
\wedge\ &(\overline{x}_2 \vee \overline{y}_1 \vee z_1 \vee z_2) \\
\wedge\ &(\overline{x}_2 \vee \overline{y}_2 \vee z_1 \vee z_2) \\
\wedge\ &\overline{z}_1 \\
\wedge\ &\overline{z}_2
\end{aligned}
$$

(a) Substitution pebbling contradiction $Peb_{\Pi_2}[\vee_2]$ with respect to binary logical or.

$$
\begin{aligned}
&(u_1 \vee u_2) \\
\wedge\ &(\overline{u}_1 \vee \overline{u}_2) \\
\wedge\ &(v_1 \vee v_2) \\
\wedge\ &(\overline{v}_1 \vee \overline{v}_2) \\
\wedge\ &(w_1 \vee w_2) \\
\wedge\ &(\overline{w}_1 \vee \overline{w}_2) \\
\wedge\ &(u_1 \vee \overline{u}_2 \vee v_1 \vee \overline{v}_2 \vee x_1 \vee x_2) \\
\wedge\ &(u_1 \vee \overline{u}_2 \vee v_1 \vee \overline{v}_2 \vee \overline{x}_1 \vee \overline{x}_2) \\
\wedge\ &(u_1 \vee \overline{u}_2 \vee \overline{v}_1 \vee v_2 \vee x_1 \vee x_2) \\
\wedge\ &(u_1 \vee \overline{u}_2 \vee \overline{v}_1 \vee v_2 \vee \overline{x}_1 \vee \overline{x}_2) \\
\wedge\ &(\overline{u}_1 \vee u_2 \vee v_1 \vee \overline{v}_2 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{u}_1 \vee u_2 \vee v_1 \vee \overline{v}_2 \vee \overline{x}_1 \vee \overline{x}_2) \\
\wedge\ &(\overline{u}_1 \vee u_2 \vee \overline{v}_1 \vee v_2 \vee x_1 \vee x_2) \\
\wedge\ &(\overline{u}_1 \vee u_2 \vee \overline{v}_1 \vee v_2 \vee \overline{x}_1 \vee \overline{x}_2) \\
\wedge\ &(v_1 \vee \overline{v}_2 \vee w_1 \vee \overline{w}_2 \vee y_1 \vee y_2) \\
\wedge\ &(v_1 \vee \overline{v}_2 \vee w_1 \vee \overline{w}_2 \vee \overline{y}_1 \vee \overline{y}_2) \\
\wedge\ &(v_1 \vee \overline{v}_2 \vee \overline{w}_1 \vee w_2 \vee y_1 \vee y_2) \\
\wedge\ &(v_1 \vee \overline{v}_2 \vee \overline{w}_1 \vee w_2 \vee \overline{y}_1 \vee \overline{y}_2) \\
\wedge\ &(\overline{v}_1 \vee v_2 \vee w_1 \vee \overline{w}_2 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{v}_1 \vee v_2 \vee w_1 \vee \overline{w}_2 \vee \overline{y}_1 \vee \overline{y}_2) \\
\wedge\ &(\overline{v}_1 \vee v_2 \vee \overline{w}_1 \vee w_2 \vee y_1 \vee y_2) \\
\wedge\ &(\overline{v}_1 \vee v_2 \vee \overline{w}_1 \vee w_2 \vee \overline{y}_1 \vee \overline{y}_2) \\
\wedge\ &(x_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2 \vee z_1 \vee z_2) \\
\wedge\ &(x_1 \vee \overline{x}_2 \vee y_1 \vee \overline{y}_2 \vee \overline{z}_1 \vee \overline{z}_2) \\
\wedge\ &(x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee y_2 \vee z_1 \vee z_2) \\
\wedge\ &(x_1 \vee \overline{x}_2 \vee \overline{y}_1 \vee y_2 \vee \overline{z}_1 \vee \overline{z}_2) \\
\wedge\ &(\overline{x}_1 \vee x_2 \vee y_1 \vee \overline{y}_2 \vee z_1 \vee z_2) \\
\wedge\ &(\overline{x}_1 \vee x_2 \vee y_1 \vee \overline{y}_2 \vee \overline{z}_1 \vee \overline{z}_2) \\
\wedge\ &(\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee y_2 \vee z_1 \vee z_2) \\
\wedge\ &(\overline{x}_1 \vee x_2 \vee \overline{y}_1 \vee y_2 \vee \overline{z}_1 \vee \overline{z}_2) \\
\wedge\ &(z_1 \vee \overline{z}_2) \\
\wedge\ &(\overline{z}_1 \vee z_2)
\end{aligned}
$$

(b) Substitution pebbling contradiction $Peb_{\Pi_2}[\oplus_2]$ with respect to binary exclusive or.

Figure 2: Examples of substitution pebbling formulas for the pyramid graph Π_2.

simple special case of this approach combined with random restrictions already yields nontrivial trade-offs for PCR, albeit with some loss in the parameters as compared to the resolution trade-offs in [30].

The idea is as follows: Start with a CNF formula F which has a (weak) trade-off in resolution between length and variable space (i.e., the number of variables that any refutation must mention simultaneously at some point). Consider some proof system $\mathcal{P}$ and study the substitution formula $F[f]$, where f is chosen to have the right properties with respect to $\mathcal{P}$. Let π_f be any $\mathcal{P}$-refutation of $F[f]$. Intuitively, we want to argue that whatever π_f looks like, we can *extract* from this π_f a resolution refutation π of F with related properties. Our way of doing this is to look at the $\mathcal{P}$-configurations (i.e., snapshots of the blackboard in $\mathcal{P}$-refutations), define *projections* of these $\mathcal{P}$-configurations to clauses over $Vars(F)$, and then to show that such projections translate $\mathcal{P}$-refutations to resolution refutations. Roughly, our intuition for projections is that if, for instance, a $\mathcal{P}$-configuration $\mathbb{D}$ implies $f(x_1, \ldots, x_d) \vee \neg f(y_1, \ldots, y_d)$, then this should project the clause $x \vee \overline{y}$. It will be convenient for us, however, to relax this requirement a bit and allow other definitions of projections as well, as long as they are "in the same spirit." Generalizing [30], we show that any function satisfying the following properties will make this approach work.

Definition 4.6 (Projection). Let $f : \{0,1\}^d \to \{0,1\}$ be a fixed Boolean function. Let $\mathcal{P}$ be a sequential implicational[7] proof system with space measure $Sp(\cdot)$, and let $\mathbb{D}$ be any $\mathcal{P}$-configuration over $Vars\big(F[f]\big)$. Then a function $proj_f$ mapping $\mathcal{P}$-configurations $\mathbb{D}$ to sets of clauses $\mathbb{C}$ over $Vars(F)$ is an *f-projection* if it is:

Complete: If $\mathbb{D} \vDash C[f]$ then the clause C either is in $proj_f(\mathbb{D})$ or is derivable from $proj_f(\mathbb{D})$ by weakening.

Nontrivial: If $\mathbb{D} = \emptyset$, then $proj_f(\mathbb{D}) = \emptyset$.

[7] Briefly, we say that $\mathcal{P}$ is *sequential implicational* if a $\mathcal{P}$-refutation π is a *sequence* of lines $\pi = \{L_1, \ldots, L_\tau\}$ where each line is semantically implied by previous lines. Note that, e.g., extended Frege does *not* satisfy this property, since introducing a new extension variable as a shorthand for a formula declares an equivalence that is not the consequence of this formula, but cutting planes, PC and PCR do.

Monotone: If $\mathbb{D}' \vDash \mathbb{D}$ and $C \in proj_f(\mathbb{D})$, then C is in or is derivable from $proj_f(\mathbb{D}')$ by weakening.

Incrementally sound: Let A be a clause over $Vars(F)$ and let L_A be the encoding of some clause in $A[f]$ as a Boolean function of the type prescribed by $\mathcal{P}$. Then if $C \in proj_f(\mathbb{D} \cup \{L_A\})$, it holds for all literals $a \in Lit(A) \setminus Lit(C)$ that the clause $\overline{a} \vee C$ either is in $proj_f(\mathbb{D})$ or can be derived from $proj_f(\mathbb{D})$ by weakening.

In order for a projection to be of use, it should also somehow preserve space when going from the proof system $\mathcal{P}$ to resolution. This is captured by the next definition.

Definition 4.7 (Space-faithful projection). We say that $proj_f$ is *space-faithful of degree* K with respect to $\mathcal{P}$ if there is a degree-K polynomial Q such that $Q(Sp(\mathbb{D})) \geq \big|Vars(proj_f(\mathbb{D}))\big|$ holds for any $\mathcal{P}$-configuration $\mathbb{D}$ over $Vars\big(F[f]\big)$. we say that $proj_f$ is *exactly space-faithful*.

As we will show in Section 4.3.7, if we can define a space-faithful projection for a proof system $\mathcal{P}$ with respect to some space measure in $\mathcal{P}$, then resolution trade-offs between length and variable space in resolution for F are amplified to time-space trade-offs for $F[f]$ in $\mathcal{P}$. Viewed from this angle, the main technical contribution in [30] can be described as proving that certain projections are space-faithful for resolution and k-DNF resolution. Also, this means that in order to prove time-space trade-offs for, say, PCR or cutting planes, it would be sufficient to design space-faithful projections as defined above. The trade-offs would then follow by applying the projection machinery in an entirely black-box fashion. Although we do not use the full generality of this machinery in the current paper, we nevertheless believe that the development of this black box is an important technical contribution.

Unfortunately, for both PCR and cutting planes it seems very challenging to come up with space-faithful projections with respect to the most interesting space measures in these systems. However, there is a particular measure for which we are able to obtain space-faithful projections for a wide range of proof systems $\mathcal{P}$ (once our refined analysis of [30] reveals that this is what we should be aiming for), namely if we consider variable

space not only as the "target measure" in resolution but also in $\mathcal{P}$. Furthermore, for this measure we can pick the "substitution function" f to be the identity.

Lemma 4.8. *Let $\mathcal{P}$ be any sequential implicational proof system and fix f to be the identity function. Then there are exactly space-faithful projections from $\mathcal{P}$ to resolution with respect to variable space for any CNF formula F without making substitutions.*

This simple but powerful lemma (which we prove in Section 4.3.7) turns out to be sufficient to lift the resolution trade-offs between width and clause space in [25] to the PCR trade-offs between degree and monomial space in Theorem 4.1 (as explained in Section 4.4).

The next step is to combine Lemma 4.8 with substitution using exclusive or (over two or more variables). If π is a PCR refutation of $F[\oplus]$, then after hitting π with a restriction ρ as described above we get a PCR refutation of the original formula F that is very likely not to contain high-degree monomials. But if all monomials are of small degree, then small monomial space implies small variable space, and this means that we can prove a slightly weaker analogue for PCR of the substitution space theorem in [30] for resolution, as stated next.

Theorem 4.9 (Substitution space theorem for PCR). *Suppose that F is a CNF formula for which any syntactic resolution refutation in variable space at most s must make more than T axiom downloads.*[8] *Then any semantic PCR refutation of $F[\oplus]$ in monomial space at most $s/\log_{4/3} T$ must have size larger than T.*

Proof. Let $\pi : F \vdash \bot$ be a PCR refutation of $F[\oplus]$ in size T and monomial space s'. If we apply a random restriction ρ to $F[\oplus]$ as described above, then $\pi\restriction_\rho$ is a PCR refutation of F. Consider some fixed monomial m in π. By Lemma 4.28, $m\restriction_\rho$ has degree at most K except with probability $(3/4)^K$. Thus, by a union bound we can pick ρ so that $\pi\restriction_\rho$ is a PCR refutation of F in size at most T, monomial space at most s', and degree at

[8] It would have been nice to be able to use bounds on refutation length here rather than bounds on the number of axiom downloads. This is clearly *not* possible, however. The reason for this is that the proof refuting $F[\oplus]$ is allowed to use any arbitrarily strong *semantic* inference rules, and this can lead to exponential savings compared to syntactic resolution. But, happily, the bound in terms of axiom downloads turns out to be exactly what we need for our applications.

most $\log_{4/3} T$. This means that the variable space of this refutation is upper-bounded by $s' \log_{4/3} T$. Applying the projection in Lemma 4.8, this results in a resolution refutation doing at most T downloads and never exceeding variable space $s' \log_{4/3} T$. This is impossible if $s' \leq s/\log_{4/3} T$, and the theorem follows. □

The time-space trade-offs for PCR in sublinear space reported in Theorems 4.2 and 4.3, as well as several other trade-off results, now follow by applying Theorem 4.9 to pebbling formulas substituted with exclusive or. These formulas are all refutable in linear length and constant width simultaneously in resolution, which means that polynomial calculus can simulate these refutations in linear size. In this way, we get trade-off results where the upper bounds hold for syntactic versions of the weaker proof systems resolution and polynomial calculus, whereas the lower bounds hold for the stronger proof system PCR, even when this system is made stronger still by allowing semantic derivation steps.

4.3.4 Tseitin Contradictions

Tseitin contradictions [111] encode the principle that every undirected graph has even total degree.

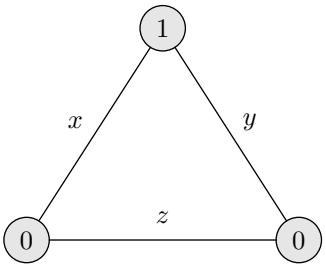
Definition 4.10. Let $G = (V, E)$ be an undirected graph and $\chi : V \to \{0, 1\}$ a function such that $\bigoplus_{v \in V} \chi(v)$ is odd. Identify each edge $e \in E$ with a variable x_e, and for a vertex $v \in V$ and value $b \in \{0, 1\}$ let

$$PARITY_{v,b} = \bigwedge \left\{ \bigvee_{e \ni v} x_e^{a(e)} \;\middle|\; \bigoplus_e (a(e) \oplus 1) \neq b \right\}$$

be the CNF representation of the constraint $\bigoplus_{e \ni v} x_e = b$. Then the *Tseitin contradiction* on (G, χ) is

$$Ts(G, \chi) = \bigwedge_{v \in V} PARITY_{v,\chi(v)} \ .$$

Since each edge is counted twice in $Ts(G, \chi)$, the parity constraints cannot all be satisfied if the overall parity of χ is odd. We will frequently suppress the reference



(a) Labelled triangle graph.

$$
\begin{aligned}
&(x \vee y) \\
\wedge\ &(\overline{x} \vee \overline{y}) \\
\wedge\ &(x \vee \overline{z}) \\
\wedge\ &(\overline{x} \vee z) \\
\wedge\ &(y \vee \overline{z}) \\
\wedge\ &(\overline{y} \vee z)
\end{aligned}
$$

(b) Corresponding Tseitin contradiction.

Figure 3: Example Tseitin contradiction.

to χ above, since when G is connected any two odd-parity functions yield equivalent formulas for all practical purposes.

When the degree of the graph is bounded by d, each local parity constraint for a vertex can be written as a CNF formula with at most 2^{d-1} clauses of width d, and hence $Ts(G)$ has at most $2^{d-1}|V|$ clauses in total. Figure 3(b) gives an example Tseitin contradiction generated from the graph in Figure 3(a).

4.3.5 Time-Space Trade-offs Based on Tseitin Contradictions

A useful tool when proving lower bounds in resolution are *semantic measures* of clause complexity as introduced by Ben-Sasson and Wigderson [31], i.e., measures of the form

$$\mu_{\mathcal{A}}(C) = \min\{|S| : S \subseteq \mathcal{A},\ S \models C\}\ , \tag{4.2}$$

where C is a clause and $\mathcal{A}$ is a collection of axioms (or sets of axioms). In the context of Tseitin contradictions $Ts(G)$, the semantic measure of a clause is defined to be the size of a smallest subset of the vertices of G such that the parity constraints over these vertices semantically imply the clause. Tseitin contradictions cannot be refuted without using parity constraint clauses for all vertices, so in the course of the proof information from all vertices must be aggregated. If the graph G has a large isoperimetric number—i.e., if every medium-sized set of vertices have many edges leaving the set—then the formula $Ts(G)$ will be hard to refute.

In [14] it was shown how to use the semantic measure as a progress measure to

get not only size lower bounds but also size-space trade-offs for Tseitin contradictions in resolution. Intuitively, clauses which are very wide cannot help with this much, because they rule out only a small fraction of assignments; we can think of them as "bottlenecks" which carry very little information and so cannot help with this aggregation process very much. The "bottleneck counting" strategy is to show a lower bound by showing that there must be many such bottlenecks in any proof. A simple way to do this counting was discovered by [16]: First, hit a formula and a hypothetical proof with a random restriction which kills wide clauses with high probability. Second, show that every proof of any (or most) of the restricted formulas must contain at least one wide clause. This works quite well for Tseitin contradictions, because any restriction gives a Tseitin formula on a smaller graph. This is where we make use of the semantic measure. It is easy to show that any proof must have a clause of intermediate semantic measure, and the key property of the Tseitin construction is that if the graph satisfies an isoperimetric inequality then this clause is wide. [14] developed this idea significantly to get time-space trade-offs for Tseitin contradictions in resolution. This strategy considers multiple ranges of intermediate complexity values for clauses, and requires quite specific and strong isoperimetric properties. The argument works for graphs that not only have a certain *extended isoperimetry* property, but also maintain this property after having a constant fraction of randomly chosen edges removed (corresponding to the formula being hit by a random restriction). This creates significant complications at a fairly low level of the argument, and necessitates the use of rather dense graphs, so that the trade-off can only be shown to hold for CNF formulas of unbounded width. We get a cleaner and simpler proof by instead considering multigraphs and using appropriate restrictions operating on them. This makes the argument more transparent and enables us to prove trade-offs for CNF formulas of *constant width* (which, as noted in, e.g., [4], is the preferred setting when studying space in proof complexity).

As in [14], our construction requires graphs with an extended isoperimetry property, which is formalized as follows.

Definition 4.11. Let $G = (V, E)$ be an undirected graph and (W, t_0, r) be associated

parameters. Call a vertex set $S \subseteq V$ of size $t_0 \leq |S| \leq |V|/2$ *medium-sized*. The *boundary* of S is $\delta(S) = \{(u, v) \in E \mid u \in S,\ v \in V \setminus S\}$, i.e., the set of edges with exactly one endpoint in S.

We say that G has the *extended isoperimetry property* with parameters (W, t_0, r) if any sequence of medium-sized sets of vertices $S_1, \ldots, S_k \subseteq V$ such that $|S_{i+1}| \geq r \cdot |S_i|$ for all i satisfies the inequality $\left|\bigcup_i \delta(S_i)\right| \geq k \cdot W$.

So-called *grid graphs* (with vertices indexed by integer coordinates (i, j) and edges to adjacent coordinates $(i, j \pm 1)$, $(i \pm 1, j)$) can be shown to have this property.

Lemma 4.12. *A $w \times \ell$ grid graph, where $4w^2 \leq \ell \leq 2^w$, satisfies the extended isoperimetry property with parameters $(w, 4w^3, 2 + \epsilon)$ for any $\epsilon > 0$ and any large enough w.*

For Tseitin contradictions, vertex set boundaries are related to the semantic complexity measure μ as follows.

Lemma 4.13 ([31]). *Let S be a minimal vertex set witnessing the complexity $\mu(C)$ of a clause C derived from $Ts(G)$ in resolution. Then $\delta(S) \subseteq Vars(C)$.*

We now formalize the idea that the semantic measure μ can be used as a tool to obtain time-space trade-offs. The following lemma is straightforward to prove by induction.

Lemma 4.14. *Fix any unsatisfiable CNF formula F with associated semantic complexity measure μ_F and any sequential implicational proof system. Let $\mu^*(\cdot) = \lfloor \log_2 \mu_F(\cdot) \rfloor$ and let $N = \mu^*(\bot)$. If a refutation of F is divided into consecutive subderivations, or* epochs*, and further subdivided recursively into subepochs to a recursive depth of h, then for any integer k at least one of the following cases apply:*

1. *There exists an epoch corresponding to a leaf in the recursive tree which contains formulas with at least $N \cdot k^{-h}$ distinct complexity values under μ^*.*

2. *There exists an epoch such that the formulas in memory during the breakpoints between the epochs in its immediate children contain formulas with at least k distinct complexity values under μ^*.*

To apply this lemma, consider Tseitin formulas over grid graphs G, do $\oplus_2$-substitution in $Ts(G)$ (which yields the formula $Ts(G'')$ over the multigraph G'' with two copies of each edge in G), and hit any resolution refutation of $Ts(G)[\oplus_2] = Ts(G'')$ with a random restriction as described in Section 4.3.1. Since G satisifies extended isoperimetry, the clauses in Lemma 4.14 are collectively wide. For this very reason, however, a random restriction would have been very likely to kill at least one of these clauses. Trading off parameters appropriately, we obtain strong time-space trade-offs.

This strategy breaks down in polynomial calculus; while naively one would hope to use the width/degree analogy and just carry out the plan above, unfortunately the existing degree lower bound machinery seems not to yield measures of progress with the properties we need for Lemma 4.14, despite some good candidates to fill the role.

One useful technique for showing degree lower bounds has been to do a linear transformation of the variables. In particular, in the context of PCR over a field of odd characterstic, consider rewriting the Tseitin parity constraints so that the variables take values in $\{+1, -1\}$ rather than $\{0, 1\}$. It is not hard to see that the degree needed to refute this "Fourier-transformed" $\{\pm 1\}$-Tseitin formula is the same as for the original $\{0, 1\}$-Tseitin formula, and [39] gave tight upper and lower degree bounds for the former exploiting the fact that its polynomials have a simple binomial form. Their approach to the lower bound constructs a "pseudoideal"—in other words, an obstruction to a low degree refutation—when the base graph has good expansion properties. However, while this machinery can easily establish that any refutation must contain many monomials corresponding to boundaries for vertex sets of many different sizes, by itself this does not suffice for time-space trade-offs. We must also be able to show how these monomials are related to one another, so that we can track the progress that a refutation makes.

We resolve the issue by modifying the plan. It turns out that a suitable progress measure does exist in the subsystem of binomial polynomial calculus. We adapt the ideas in [39] to construct a simulation of PCR refutations of a Tseitin formula by binomial polynomial calculus refutations of the Fourier-transformed formula, the simulation

having the following key property.

Lemma 4.15. *The simulation of PCR on $Ts(G)$ (in variables $\{x_e\}$) by binomial PC on the $\{\pm 1\}$-Tseitin formula (in variables $\{y_e\}$) is* conservative with respect to monomials*:*

- *If for some configuration of the simulated refutation no monomial appears which contains the set of variables $\{x_e \mid e \in E'\}$ for some $E' \subseteq E$, then the corresponding configuration of the simulating refutation does not contain any monomials containing all of $\{y_e : e \in E'\}$.*
- *If for some time period in the simulated refutation no monomial contains the set of variables $\{x_e \mid e \in E'\}$ for some $E' \subseteq E$, then the corresponding time period of the simulating refutation does not contain any monomials containing all of $\{y_e : e \in E'\}$.*

Since the simulation is *not* efficient with respect to size and space, the images of epochs under the simulation have wildly differing sizes in general. However, for the binomials we can recover a degree-analogue of the width lower bound in resolution without losing much, as stated next.

Lemma 4.16. *Suppose that $G = (V, E)$ has the extended isoperimetry property with parameters (w, t_0, r), and as before let $\mu^*(\cdot) = \lfloor \log_2 \mu(\cdot) \rfloor$. Then for any binomials $b_1, \ldots, b_k$ with distinct complexities between t_0 and $\mu^*(\bot)$ it holds that*

$$\left| \bigcup Vars(b_i) \right| \geq \Omega(k \cdot w) \ . \tag{4.3}$$

Using the semantic measure in binomial PC together with the division of the refutation into epochs in Lemma 4.14, we can obtain many monomials of collectively high complexity either within a single epoch or at a collection of breakpoints in the simulating refutation. Then we can apply Lemma 4.15 to lift these monomials back to the simulated refutation where they are unlikely to survive a restriction. This allows us to prove PCR time-space trade-offs.

sketch for Theorem 4.4. Let G be a grid graph satisfying Lemma 4.16 and let π be any PCR refutation of $Ts(G)[\oplus_2]$ in size $S(\pi) = T$ and space $Sp(\pi) = S$. Divide π into epochs with each epoch split into m equal subepochs to a recursive depth of h, with m and h to be determined later. Say that the *critical set* of monomials associated to an internal epoch consists of any monomial appearing at the breakpoints between its children, and that for a leaf epoch the critical set contains all monomials in the epoch.

Applying the random restriction ρ in Section 4.3.1, we get that $\pi\restriction_\rho$ is a refutation of $Ts(G)[\oplus]\restriction_\rho = Ts(G)$ (up to sign flips). Let π^* be the induced refutation of the $\{\pm 1\}$-Tseitin formula on G with corresponding induced recursive subdivision into epochs. Choose k such that $(\mu^*(\bot) - t_0)k^{-h} = k$. Combining the properties of μ^* with Lemma 4.14 implies that the critical set of some induced epoch of π^* contains at least k binomials of distinct complexity values. Thus, by Lemma 4.16 this critical set contains $2k$ monomials which collectively contain at least $\Omega(k \cdot w)$ variables. By Lemma 4.15, this holds for π^* also.

However, for any small set of monomials M, the probability that $M\restriction_\rho$ contains $2k$ monomials which collectively contain many variables is small. For any monomial m, the semantically equivalent clause is killed by ρ if and only if m is. Therefore by (the proof of) Lemma 4.28, the probability that any fixed $2k$-tuple of monomials all survive and have collective width W is at most $\exp(-\Omega(W))$. By a union bound over all $2k$-tuples of M, the probability that any $2k$ of the monomials in M have collectively $\Omega(k \cdot w)$ variables after the restriction is at most $|M|^{2k}\exp(-\Omega(k \cdot w))$.

Choose the parameter m so that $m^h = (T/S)$ and let $K = mS = Tm^{-h+1}$. For this choice of m the sizes of all critical sets of monomials are bounded by K. The probability for any critical set of any induced epoch to contain k distinct complexities with respect to μ^* after the restriction is at most $K^{2k}\exp(-k \cdot w)$, and there are at most m^h epochs. Set $h = k$. By a union bound, the probability that any epoch contains k complexities is at most $(mK^2\exp(-\Omega(w)))^k$. On the other hand, by Lemmas 4.14 and 4.15, this probability is 1. We conclude that $T \geq (\exp(\Omega(w))/S)^{\Omega(h)}$. Since since h and k may be set as large as $\log L/\log\log L$, and L may be set as large as $\Omega(\log |G|)$,

ultimately gives $T \geq (\exp(\Omega(w))/S)^{\Omega(\log\log n/\log\log\log n)}$. □

We thus manage to carry out the plan of [14] by establishing a nonlocal measure of progress for PCR, as it is clear that the semantic complexity of the underlying binomials of a polynomial cannot be determined simply by looking at the polynomial, but in general depends on the derivation we used to obtain the polynomial. This method of constructing progress measures could be useful in other contexts as well.

4.3.6 Preliminaries

For x a Boolean variable, a *literal over* x is either the variable x itself, called a *positive literal over* x, or its negation, denoted $\neg x$ or $\overline{x}$ and called a *negative literal over* x. Sometimes the notation x^1 and x^0 will be handy for positive and negative literals, respectively, where x^b is true if $x = b$. A *clause* $C = a_1 \vee \cdots \vee a_k$ is a disjunction of literals, and a *term* $T = a_1 \wedge \cdots \wedge a_k$ is a conjunction of literals. Below we will think of clauses and terms as sets, so that the ordering of the literals is inconsequential and that, in particular, no literals are repeated. A clause (term) containing at most k literals is called a *k-clause* (*k-term*). A *CNF formula* $F = C_1 \wedge \cdots \wedge C_m$ is a conjunction of clauses, and a *DNF formula* is a disjunction of terms. We will think of CNF and DNF formulas as sets of clauses and terms, respectively. A *k-CNF formula* is a CNF formula consisting of k-clauses, and a *k-DNF formula* consists of k-terms.

The *variable set* of a clause C, denoted $Vars(C)$, is the set of Boolean variables over which there are literals in C, and we write $Lit(C)$ to denote the set of literals in C. The variable and literal sets of a term are similarly defined and these definitions are extended to CNF and DNF formulas by taking unions. If V is a set of Boolean variables and $Vars(C) \subseteq V$ we say C is a clause *over* V and similarly define terms, CNF formulas, and DNF formulas over V.

We write α, β to denote truth value assignments. Truth value assignments are functions to $\{0, 1\}$, where we identify 0 with false and 1 with true. We have the usual semantics that a clause is true under α, or *satisfied* by α, if at least one literal in it is true, and a term is true if all literals evaluate to true. We write $\perp$ to denote the empty

clause without literals that is false under all truth value assignments. A CNF formula is satisfied if all clauses in it are satisfied, and for a DNF formula we require that some term should be satisfied. In general, we will not distinguish between a formula and the Boolean function computed by it.

If $\mathbb{C}$ is a set of Boolean functions we say that an assignment satisfies $\mathbb{C}$ if and only if it satisfies every function in $\mathbb{C}$. For $\mathbb{D}, \mathbb{C}$ two sets of Boolean functions over a set of variables V, we say that $\mathbb{D}$ *implies* $\mathbb{C}$, denoted $\mathbb{D} \models \mathbb{C}$, if and only if every assignment $\alpha : V \to \{0,1\}$ that satisfies $\mathbb{D}$ also satisfies $\mathbb{C}$. In particular, $\mathbb{D} \models \bot$ if and only if $\mathbb{D}$ is *unsatisfiable* or *contradictory*, i.e., if no assignment satisfies $\mathbb{D}$. If a CNF formula F is unsatisfiable but for any clause $C \in F$ it holds that the clause set $F \setminus \{C\}$ is satisfiable, we say that F is *minimally unsatisfiable*.

Definition 4.17 (Sequential implicational proof system). Let us say that $\mathcal{P}$ is a *sequential implicational* proof system if any $\mathcal{P}$-derivation π is a sequence of lines $\pi = \{L_1, \ldots, L_\tau\}$ where any derived line follows semantically from the lines used to derive it.

We remark that resolution, PC, PCR, cutting planes, Frege and most other proof systems usually studied are "implicational" in the sense of Definition 4.17, whereas, for instance, extended Frege is not.

Following the exposition in [57], we view a proof as similar to a non-deterministic Turing machine computation, with a special read-only input tape from which the clauses of the CNF formula F being refuted (the *axioms*) can be downloaded and a working memory where all derivation steps are made. Then the length of a proof is essentially the time of the computation and space measures memory consumption. The following definition is a straightforward generalization of [4].

Definition 4.18 (Refutation). For a sequential proof system $\mathcal{P}$, a $\mathcal{P}$-con*figuration* $\mathbb{D}$ is a set of lines L of the syntactic form prescribed by $\mathcal{P}$. A sequence of configurations $\{\mathbb{D}_0, \ldots, \mathbb{D}_\tau\}$ is a $\mathcal{P}$*-derivation* from a CNF formula F if $\mathbb{D} = \emptyset$ and for all $t \in [\tau]$, the set $\mathbb{D}_t$ is obtained from $\mathbb{D}_{t-1}$ by one of the following *derivation steps*:

Axiom Download $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_C\}$, where L_C is the encoding of a clause $C \in F$ in the syntactic form prescribed by the proof system (an *axiom*).

Inference $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L\}$ for some L inferred by one of the inference rules for $\mathcal{P}$ from a set of assumptions $L_1, \ldots, L_m \in \mathbb{D}_{t-1}$.

Erasure $\mathbb{D}_t = \mathbb{D}_{t-1} \setminus \{L\}$ for some $L \in \mathbb{D}_{t-1}$.

A $\mathcal{P}$-refutation $\pi : F \vdash \bot$ of a CNF formula F is a derivation $\pi = \{\mathbb{D}_0, \ldots, \mathbb{D}_\tau\}$ such that $\mathbb{D}_0 = \emptyset$ and $\bot \in \mathbb{D}_\tau$, where $\bot$ is the representation of contradiction (e.g. for resolution and $\mathcal{R}(k)$-systems the empty clause without literals).

If every line L in a derivation is used at most once before being erased (though it can possibly be rederived later), we say that the derivation is *tree-like*. This corresponds to changing the inference rule so that $L_1, \ldots, L_d$ must all be erased after they have been used to derive L.

To every refutation π we can associate a DAG G_π, with the lines in π labelling the vertices and with edges from the assumptions to the consequence for each application of an inference rule. There might be several different derivations of a line L during the course of the refutation π, but if so we can label each occurrence of L with a time-stamp when it was derived and keep track of which copy of L is used where. Using this representation, a refutation π can be seen to be tree-like if G_π is a tree.

Definition 4.19 (Refutation size, length and space). Given a size measure $S(L)$ for lines L in $\mathcal{P}$-derivations (which we usually think of as the number of symbols in L, but other definitions can also be appropriate depending on the context), the *size* of a $\mathcal{P}$-derivation π is the sum of the sizes of all lines in a derivation, where lines that appear multiple times are counted with repetitions (once for every vertex in G_π). The *length* of a $\mathcal{P}$-derivation π is the number of axiom downloads and inference steps in it, i.e., the number of vertices in G_π.[9] For a space measure $Sp_{\mathcal{P}}(\mathbb{D})$ defined for $\mathcal{P}$-configurations, the *space* of a derivation π is defined as the maximal space of a configuration in π.

[9]The reader who so prefers can instead define the length of a derivation $\pi = \{\mathbb{D}_0, \ldots, \mathbb{D}_\tau\}$ as the number of steps τ in it, since the difference is at most a factor of 2. We have chosen the definition above for consistency with previous papers defining length as the number of lines in a listing of the derivation.

If π is a refutation of a formula F in size S and space s, then we say that F can be refuted in size S and space s *simultaneously*. Similarly, F can be refuted in length L and space s simultaneously if there is a $\mathcal{P}$-refutation $\mathcal{P}$ with $L(\pi) = L$ and $Sp(\pi) = s$.

We define the *$\mathcal{P}$-refutation size* of a formula F, denoted $S_{\mathcal{P}}(F \vdash \bot)$, to be the minimum size of any $\mathcal{P}$-refutation of it. The *$\mathcal{P}$-refutation length* $L_{\mathcal{P}}(F \vdash \bot)$ and *$\mathcal{P}$-refutation space* $Sp_{\mathcal{P}}(F \vdash \bot)$ of F are analogously defined by taking the minimum with respect to length or space, respectively, over all $\mathcal{P}$-refutations of F.

When the proof system in question is clear from context, we will drop the subindex in the proof complexity measures. Let us next give formal definitions in the framework of Definition 4.18 of the proof systems that will be of interest in this paper. Below, the notation $\frac{G_1 \quad \cdots \quad G_m}{H}$ means that if $G_1, \ldots, G_m$ have been derived previously in the proof (and are currently in memory), then we can infer H. We will sometimes use notation $G_1, \ldots, G_m \vdash H$ for this as well, for convenience.

Definition 4.20 (k-DNF resolution). The *k-DNF resolution* proof systems are a family of sequential proof systems $\mathcal{R}(k)$ parameterized by $k \in \mathbb{N}^+$. Lines in a k-DNF-resolution refutation are k-DNF formulas and we have the following inference rules (where G, H denote k-DNF formulas, T, T' denote k-terms, and $a_1, \ldots, a_k$ denote literals):

k-cut $\dfrac{(a_1 \wedge \cdots \wedge a_{k'}) \vee G \qquad \overline{a}_1 \vee \cdots \vee \overline{a}_{k'} \vee H}{G \vee H}$, where $k' \leq k$.

$\wedge$-introduction $\dfrac{G \vee T \qquad G \vee T'}{G \vee (T \wedge T')}$, as long as $|T \cup T'| \leq k$.

$\wedge$-elimination $\dfrac{G \vee T}{G \vee T'}$ for any $T' \subseteq T$.

Weakening $\dfrac{G}{G \vee H}$ for any k-DNF formula H.

For standard resolution, i.e., $\mathcal{R}(1)$, the k-cut rule simplifies to the *resolution rule*

$$\frac{B \vee x \qquad C \vee \overline{x}}{B \vee C} \tag{4.4}$$

for clauses B and C. We refer to (4.4) as *resolution on the variable* x and to $B \vee C$ as the *resolvent* of $B \vee x$ and $C \vee \overline{x}$ on x. Clearly, in resolution the $\wedge$-introduction and

$\wedge$-elimination rules do not apply. It can also be shown that the weakening rule never needs to be used in resolution refutations, but it can be convenient to allow it to simplify some technical arguments in proofs.

For $\mathcal{R}(k)$-systems, the length measure is as defined in Definition 4.19, and for space we get the two measures *formula space* and *total space* depending on whether we consider the number of k-DNF formulas in a configuration or all literals in it, counted with repetitions. For standard resolution there are two more space-related measures that will be relevant, namely *width* and *variable space*. For clarity, let us give an explicit definition of all space-related measures for resolution that will be of interest.

Definition 4.21 (Width and space in resolution). The *width* $W(C)$ of a clause C is the number of literals in it, and the width of a CNF formula or clause configuration is the size of a widest clause in it. The *clause space* (as the formula space measure is known in resolution) $Sp(\mathbb{C})$ of a clause configuration $\mathbb{C}$ is $|\mathbb{C}|$, i.e., the number of clauses in $\mathbb{C}$, the *variable space*[10] $VarSp(\mathbb{C})$ is $|Vars(C)|$, i.e., the number of distinct variables mentioned in $\mathbb{C}$, and the *total space* $TotSp(\mathbb{C})$ is $\sum_{C\in\mathbb{C}}|C|$, i.e., the total number of literals in $\mathbb{C}$ counted with repetitions.

The width or space of a resolution refutation π is the maximum that the corresponding measures attain over any clause configuration $\mathbb{C} \in \pi$, and taking the minimum over all resolution refutations of a CNF formula F, we can define the width $W_{\mathcal{R}}(F \vdash \bot) = \min_{\pi:F\vdash\bot}\{W(\pi)\}$ of refuting F in resolution, and analogously the clause space $Sp_{\mathcal{R}}(F \vdash \bot) = \min_{\pi:F\vdash\bot}\{Sp(\pi)\}$, variable space $VarSp_{\mathcal{R}}(F \vdash \bot) = \min_{\pi:F\vdash\bot}\{VarSp(\pi)\}$, and total space $TotSp_{\mathcal{R}}(F \vdash \bot) = \min_{\pi:F\vdash\bot}\{TotSp(\pi)\}$ of refuting F.

Remark 4.22. When studying and comparing the complexity measures for resolution in Definition 4.21, as was noted in [4] it is preferable to prove the results for k-CNF formulas, i.e., formulas where all clauses have width upper-bounded by some constant.

[10] It should be noted that there is some terminological confusion in the literature here. The term "variable space" has also been used previously to refer to what is here called "total space." The terminology adopted in this paper is due to Alex Hertel and Alasdair Urquhart (see [65]), and we feel that their naming convention is the most natural one.

This is so since the width and space measures can "misbehave" rather artificially for formula families of unbounded width (see [82, Section 5] for a discussion of this). Since every CNF formula can be rewritten as an equivalent formula of bounded width by using auxiliary variables, it therefore seems natural to insist that the formulas under study should have width bounded by some constant.

Polynomial calculus (PC), was introduced in [46], though that paper used the name "Gröbner proof system." In a PC refutation, clauses are interpreted as multilinear polynomials. For instance, the requirement that the clause $x \vee y \vee \overline{z}$ should be satisfied gets translated to the equation $(1-x)(1-y)z = 0$ or $xyz - xz - yz + z = 0$, and we derive contradiction by showing that there is no common root for the polynomial equations corresponding to all the clauses.

Definition 4.23 (Polynomial calculus (PC)). Lines in a polynomial calculus proof are multivariate polynomial equations $p = 0$, where $p \in \mathbb{F}[x, y, z, \ldots]$ for some (fixed) field $\mathbb{F}$. It is customary to omit "$= 0$" and only write p. The derivation rules are as follows, where $\alpha, \beta \in \mathbb{F}$, $p, q \in \mathbb{F}[x, y, z, \ldots]$, and x is any variable:

Linear combination $\dfrac{p \qquad q}{\alpha p + \beta q}$

Multiplication $\dfrac{p}{xp}$

A PC refutation ends when 1 has been derived (i.e., $1 = 0$).

In the context of SAT solving and also in most proof complexity scenarios, PC also makes use of the following axioms:

Boolean axioms $\dfrac{}{x^2 - x}$ (forcing 0/1-solutions).

The *size* of a PC refutation is defined as the total number of monomials in the refutation (counted with repetitions), the *length* of a refutation is the number of polynomial equations, and the *(monomial) space* is the maximal number of monomials in any configuration (counted with repetitions). Another important measure is the *degree* of a refutation, which is the maximal (total) degree of any monomial.

The representation of a clause $\bigvee_{i=1}^{n} x_i$ as a PC polynomial is $\prod_{i=1}^{n}(1 - x_i)$, which means that the number of monomials is exponential in the clause width. This problem arises only for positive literals, however—a large clause with only negative literals is translated to a single monomial. This rather artificial space blow-up is a weakness of monomial space in polynomial calculus when compared to clause space in resolution. In order to obtain a cleaner, more symmetric treatment of proof space, in [4] the proof system *polynomial calculus resolution (PCR)* was introduced as a common extension of polynomial calculus and resolution. The idea is to add an extra set of parallell formal variables $x', y', z', \ldots$ so that positive and negative literals can both be represented in a space-efficient fashion.

Definition 4.24 (Polynomial calculus resolution (PCR)). Lines in a PCR-proof are polynomials over the ring $\mathbb{F}[x, x', y, y', z, z', \ldots]$, where as before $\mathbb{F}$ is some field. We have all the axioms and rules of PC plus the following axioms:

Complementarity $\dfrac{}{x + x' - 1}$ for all pairs of variables (x, x').

Size, length, and degree are defined as for polynomial calculus, and the (monomial) space of a PCR-refutation is again the maximal number of monomials in any configuration counted with repetitions.[11]

The point of the complementarity rule is to force x and x' to have opposite values in $\{0, 1\}$, so that they encode complementary literals. This means one can potentially avoid an exponential blow-up in size measured in the number of monomials (and thus also for space). Our running example clause $x \vee y \vee \overline{z}$ is rendered as $x'y'z$ in PCR. In PCR, monomial space is a natural generalization of clause space since every clause translates into a monomial as just explained.

It will be convenient for us to define the following subsystem of PC.

Definition 4.25 (Binomial PC). A *binomial* is a sum of two monomials, one or both of which may be zero. A system of equations will be called a *binomial system* if all of the

[11] We remark that in [4] space was defined as the number of *distinct* monomials in a configuration (i.e., not counted with repetitions), but we find this restriction to be somewhat arbitrary.

polynomial constraints $p = 0$ are binomials. A *binomial PC* derivation is one in which for every proof line $p = 0$, p is a binomial.

Because of the quite restrictive constraint that all axioms be binomials, binomial PC is studied without the Boolean axioms; as discussed in [39], binomial PC with Boolean axioms can be simulated efficiently by resolution. Thus, binomial PC should not be thought of as directly useful to SAT solving model, but more typically as a proof system to reduce to; if a good change of variables permits one to rephrase a formula as a binomial system it may be profitable to study it via binomial PC. It is also worth pointing out that there really can be no binomial PCR, since the extension axioms $\overline{x_i} = 1 + x_i$ are not binomials.

Definition 4.26. A *restriction* is a mapping ρ from some set of Boolean variables to constants and other variables. The restriction of a formula F, denoted by $F\restriction_\rho$, is the formula obtained by replacing the variables with their images under the restriction and performing local simplifications. The restriction of a set of formulas is defined as the collection of restricted formulas. The restriction of a polynomial p, denoted by $p\restriction_\rho$ is defined similarly. Notice that allowing a variable being mapped to another variable is a bit non-standard. Nevertheless, this non-standardness does not affect a fact that is well known for restrictions allowing mapping only to constants (a.k.a. partial assignments): proofs may also be restricted – if π is a proof of F, and ρ is a restriction, then there is an induced proof $\pi\restriction_\rho$ of $F\restriction_\rho$, obtained again by performing local simplifications to the formulas of the proof and to the proof structure.

Note that the definition above is slightly non-standard in that a restriction is not a partial assignment, but a combination of an assignment and a substitution (since variables can be "assigned" to other variables). It is straightforward to verify that this kind of "enhanced" restriction preserves derivations just as well as do standard restrictions.

For any formula F, there is a generic distribution of random restrictions to the variables $F[\oplus]$ which is often useful.

Definition 4.27. Let F be any formula. Define a random restriction ρ which maps variables of $F[\oplus]$ (as defined in Section 4.3.1) to constants and variables of F by indepen-

dently for each variable x of F, choosing one of x_1 and x_2 with equal probability to set to $\{0,1\}$ with equal probability, and the other to either x or $\overline{x}$ so that $\rho(x_1) \oplus \rho(x_2) \equiv x$.

This restriction sets a variable to a constant with constant probability, so it is a fairly dense restriction which can be used to kill very wide clauses. On the other hand it is very clean and always gives us back essentially the same formula. Note that it always holds that $F[\oplus]\restriction_\rho = F$.

We will use this next lemma throughout the paper, and throughout the paper ρ will refer to a random restriction chosen in this way.

Lemma 4.28. *Let C be any clause in the variables of $F[\oplus]$. Then,*

$$\Pr_\rho \left[\left| Vars(C\restriction_\rho) \right| \geq K \right] \leq \left(\frac{3}{4}\right)^K .$$

Proof. Suppose $\left|\{x : x_1, x_2 \in Vars(C\restriction_\rho)\}\right| < K$. Then the probability of the event is zero. Suppose not. Then there are at least K vars of C which ρ assigns independently, to 0 with probability at least $1/4$ and to 1 with probability at least $1/4$. Since for each such variable, there is a specific value which if chosen by ρ will result in $C\restriction_\rho = \top$ and $Vars(C\restriction_\rho) = \emptyset$. Therefore by independence, the probability that $Vars(C\restriction_\rho) \neq \emptyset$ is at most $\left(\frac{3}{4}\right)^K$, as desired. □

Note that while in some specific applications better results can be achieved with specially crafted random restrictions, if we do not care about constants in the exponent this kind of argument can often yield optimal results with a very simple proof. This technique has the advantage that often we will not need to think about $F[\oplus]$ directly and can instead focus on F.

In general, the admissible inferences in a proof system according to Definition 4.18 are defined by a set of syntactic inference rules. In what follows, we will also be interested in a strengthened version of this concept, which was made explicit in [4].

Definition 4.29 (Syntactic and semantic derivations). We refer to derivations according to Definition 4.18, where each new line L has to be inferred by one of the inference

rules for $\mathcal{P}$, as *syntactic* derivations. If instead *any line* L that is semantically implied by the current configuration can be derived in one atomic step, we talk about a *semantic* derivation.

Clearly, semantic derivations are at least as strong as syntactic ones, and they are easily seen to be superpolynomially stronger with respect to length for any proof system where superpolynomial lower bounds are known. This is so since a semantic proof system can download all axioms in the formula one by one, and then deduce contradiction in one step since the formula is unsatisfiable. Therefore, semantic versions of proof systems are mainly interesting when we want to reason about space or the relationship between space and length. But if we can prove lower bounds not just for syntactic but even semantic versions of proof systems, this of course makes these bounds much stronger.

Let us finally remark that although the measure of total space, considering the total number of symbols in memory, is perhaps a priori the most natural one, most papers on proof space have focused on space measured as the number of lines in memory (e.g., clauses, k-DNF formulas, or inequalities). However, as observed in [4], for strong enough proof systems, this "line space" measure is no longer interesting since just one unit of memory can contain a big AND of all formulas derived so far. The "line space" measure makes perfect sense for resolution and k-DNF resolution, and seems to do so also for cutting planes. For PC/PCR, however, measuring just the number of polynomial equations is not very meaningful, since every equation can be of exponential size and encode very much information. Instead, the natural generalization of clause space is monomial space.

4.3.7 Substituted Formulas, Projections, and Trade-offs

Let us start by recalling some key definitions from Section 4.3.3.

Definition 4.6 (restated). Let $f : \{0,1\}^d \to \{0,1\}$ be a fixed Boolean function. Let $\mathcal{P}$ be a sequential proof system, and let $\mathbb{D}$ denote an arbitrary $\mathcal{P}$-configuration over

$Vars\big(F[f]\big)$. Let $\mathbb{C}$ denote arbitrary sets of disjunctive clauses over $Vars(F)$. Then the function $proj_f$ mapping $\mathcal{P}$-configurations $\mathbb{D}$ to clauses $\mathbb{C}$ is an *f-projection* if it is:

Complete: If $\mathbb{D} \models C[f]$ then the clause C either is in $proj_f(\mathbb{D})$ or is derivable from $proj_f(\mathbb{D})$ by weakening.

Nontrivial: If $\mathbb{D} = \emptyset$, then $proj_f(\mathbb{D}) = \emptyset$.

Monotone: If $\mathbb{D}' \models \mathbb{D}$ and $C \in proj_f(\mathbb{D})$, then either $C \in proj_f(\mathbb{D}')$ or C is derivable from $proj_f(\mathbb{D}')$ by weakening.

Incrementally sound: Let A be a clause over $Vars(F)$ and let L_A be the encoding of some clause in $A[f]$ as a Boolean function of the type prescribed by $\mathcal{P}$. Then if $C \in proj_f(\mathbb{D} \cup \{L_A\})$, it holds for all literals $a \in Lit(A) \setminus Lit(C)$ that the clause $\overline{a} \vee C$ either is in $proj_f(\mathbb{D})$ or can be derived from $proj_f(\mathbb{D})$ by weakening.

Definition 4.7 (restated). Consider a sequential proof system $\mathcal{P}$ with space measure $Sp(\cdot)$. Suppose that $f : \{0,1\}^d \to \{0,1\}$ is a fixed Boolean function, and that $proj_f$ is an f-projection. Then we say that $proj_f$ is *space-faithful of degree K* with respect to $\mathcal{P}$ if there is a polynomial Q of degree at most K such that $Q(Sp(\mathbb{D})) \geq \big|Vars(proj_f(\mathbb{D}))\big|$ holds for any $\mathcal{P}$-configuration $\mathbb{D}$ over $Vars\big(F[f]\big)$. We say that $proj_f$ is *linearly space-faithful* if Q has degree 1, and that $proj_f$ is *exactly space-faithful* if we can choose $Q(n) = n$.

A special kind of projections are those that look not only on all of $\mathbb{D}$ "globally," but measure the semantic content of $\mathbb{D}$ more precisely.

Definition 4.30 (Local projection). If $proj_f$ is an f-projection, then its *localized version* $proj_f^L$ is defined to be $proj_f^L(\mathbb{D}) = \bigcup_{\mathbb{D}' \subseteq \mathbb{D}} proj_f(\mathbb{D}')$. If $proj_f = proj_f^L$, we say that $proj_f$ is a *local projection*.

It is easily verified that the localized version of a projection is indeed itself a projection in the sense of Definition 4.6.

4.3.8 Using Projections to Obtain Time-Space Trade-offs

We now show that if we can design a projection in accordance with Definition 4.6, then this projection can be used to extract resolution refutations from $\mathcal{P}$-refutations. Furthermore, if our projection is space-faithful, this extraction operation will preserve length-space trade-off (with some loss in parameters depending on how high the degree K is).

Lemma 4.31. *Let $\mathcal{P}$ be an implicational sequential proof system and let $f : \{0,1\}^d \to \{0,1\}$ be a Boolean function, and suppose that $proj_f$ is an f-projection. Then for any CNF formula F it holds that if $\pi_f = \{\mathbb{D}_0, \mathbb{D}_1, \ldots, \mathbb{D}_\tau\}$ is a semantic $\mathcal{P}$-refutation of the substitution formula $F[f]$, the sequence of sets of projected clauses $\{proj_f(\mathbb{D}_0), proj_f(\mathbb{D}_1), \ldots, proj_f(\mathbb{D}_\tau)\}$ forms the "backbone" of a resolution refutation π of F in the following sense:*

1. *$proj_f(\mathbb{D}_0) = \emptyset$.*
2. *$\bot \in proj_f(\mathbb{D}_\tau)$.*
3. *All transitions from $proj_f(\mathbb{D}_{t-1})$ to $proj_f(\mathbb{D}_t)$ for $t \in [\tau]$ can be accomplished in syntactic resolution in such a fashion that $VarSp(\pi) = \mathrm{O}\big(\max_{\mathbb{D} \in \pi_f}\{VarSp(proj_f(\mathbb{D}))\}\big)$, or, if $proj_f$ is a local projection, so that $VarSp(\pi) \leq \max_{\mathbb{D} \in \pi_f}\{VarSp(proj_f(\mathbb{D}))\}$.*
4. *The length of π is upper-bounded by π_f in the sense that the only time π performs a download of an axiom $C \in F$ is when π_f downloads some axiom $D \in C[f]$ from $F[f]$.*

On one hand, Lemma 4.31 is very strong in the sense that even *semantic* $\mathcal{P}$-refutations can be translated to *syntactic* resolution refutations. On the other hand, it would have been nice if the bound in part 4 of Lemma 4.31 could have been made into a true upper bound in terms of the length of π_f, but it is easy to see that this is *not* possible. The reason for this is precisely that the $\mathcal{P}$-proof refuting $F[f]$ is allowed to use any arbitrarily strong semantic inference rules, and this can lead to exponential savings compared to syntactic resolution. For a concrete example, just let F be an encoding of the pigeon-

hole principle and let π_f be the refutation that downloads all axioms of $F[f]$ and then derives contradiction in one step.

Before proving Lemma 4.31 let us see how it can be used to prove trade-offs provided that we can construct space-faithful projections.

Theorem 4.32. *Let $\mathcal{P}$ be an implicational sequential proof system with space measure $Sp(\cdot)$. Suppose $f : \{0,1\}^d \to \{0,1\}$ is a Boolean function such that there exists an f-projection which is space-faithful of degree K with respect to $\mathcal{P}$. Then if F is any unsatisfiable CNF formula and π_f is any semantic $\mathcal{P}$-refutation of the substitution formula $F[f]$, there is a resolution refutation π of F such that:*

- *The length of π is upper-bounded by π_f in the sense that π makes at most as many axiom downloads as π_f.*
- *The space of π is upper-bounded by π_f in the sense that $VarSp(\pi) = \mathrm{O}\big(Sp(\pi_f)^K\big)$.*

In particular, if there is no syntactic resolution refutation of F in simultaneous length $\mathrm{O}(L)$ and variable space $\mathrm{O}(s)$, then there is no semantic $\mathcal{P}$-refutation of $F[f]$ in simultaneous length $\mathrm{O}(L)$ and $\mathcal{P}$-space $\mathrm{O}\big(\sqrt[K]{s}\big)$.

Proof of Theorem 4.32. Let π_f be a semantic $\mathcal{P}$-refutation of $F[f]$, and let π be the resolution refutation we obtain by applying the the projection $proj_f$ on π_f as in Lemma 4.31. By part 4 of Lemma 4.31 we know that π makes at most as many axiom downloads as π_f. By part 3 of the lemma we have $VarSp(\pi) = \mathrm{O}\big(\max_{\mathbb{D}\in\pi_f}\{VarSp(proj_f(\mathbb{D}))\}\big)$. Fix some $\mathcal{P}$-configuration $\mathbb{D}$ maximizing the right-hand side of this expression. For this $\mathbb{D}$ we have $VarSp(proj_f(\mathbb{D})) = \mathrm{O}\big(Sp(\mathbb{D})^K\big) = \mathrm{O}\big(Sp(\pi_f)^K\big)$ according to Definition 4.7. The theorem follows. □

Clearly, the key to the proof of Theorem 4.32 is the claim that projections translate $\mathcal{P}$-refutations to resolution refutations. Let us substantiate this claim.

Proof of Lemma 4.31. Fix any sequential proof system $\mathcal{P}$, any f-projection $proj_f$, and any CNF formula F. Recall that we want to show that if $\pi_f = \{\mathbb{D}_0, \mathbb{D}_1, \ldots, \mathbb{D}_\tau\}$ is a semantic $\mathcal{P}$-refutation of the substitution formula $F[f]$, then the sequence of projected

clause sets $\{proj_f(\mathbb{D}_0), proj_f(\mathbb{D}_1), \ldots, proj_f(\mathbb{D}_\tau)\}$ is essentially a resolution refutation π except for some details that we might have to fill in when going from $proj_f(\mathbb{D}_{t-1})$ to $proj_f(\mathbb{D}_t)$ in the derivation.

Parts 1 and 2 of Lemma 4.31 are immediate from Definition 4.6, since we have $proj_f(\mathbb{D}_0) = proj_f(\emptyset) = \emptyset$ by nontriviality and $\bot \in proj_f(\mathbb{D}_\tau)$ by completeness (note that $\mathbb{D}_\tau \vDash \bot = \bigvee_{x^b \in \bot} f^b(\vec{x})$ and the empty clause clearly cannot be derived by weakening).

We want to show that a resolution refutation of F can get from $proj_f(\mathbb{D}_{t-1})$ to $proj_f(\mathbb{D}_t)$ as claimed in part 3 of the lemma. For brevity, let us write $\mathbb{C}_i = proj_f(\mathbb{D}_i)$ for all i, and consider the possible derivation steps at time t.

Inference Suppose $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_t\}$ for some L_t inferred from $\mathbb{D}_{t-1}$. Since $\mathcal{P}$ is an implicational proof system, it holds that $\mathbb{D}_{t-1} \vDash \mathbb{D}_t$, and since the projection is monotone by definition we can conclude that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ are derivable from $\mathbb{C}_{t-1}$ by weakening. We go from $\mathbb{C}_{t-1}$ to $\mathbb{C}_t$ in three steps. First, we erase all clauses $C \in \mathbb{C}_{t-1}$ for which there are no clauses $C' \in \mathbb{C}_t$ such that $C \subseteq C'$. Then, we derive all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ by weakening, noting that all clauses needed for weakening steps are still in the configuration. Finally, we erase the rest of $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$. At all times during this transition from $\mathbb{C}_{t-1}$ to $\mathbb{C}_{t-1}$, the variable space of the intermediate clause configurations is upper-bounded by $\max\{ VarSp(\mathbb{C}_{t-1}), VarSp(\mathbb{C}_t)\}$.

Erasure Suppose $\mathbb{D}_t = \mathbb{D}_{t-1} \setminus \{L_{t-1}\}$ for some $L_{t-1} \in \mathbb{D}_{t-1}$. Again we have that $\mathbb{D}_{t-1} \vDash \mathbb{D}_t$, and we can appeal to the monotonicity of the projection and proceed exactly as in the case of an inference above.

Axiom download So far, the only derivation rules used in the resolution refutation π that we are constructing are weakening and erasure, which clearly does not help π to make much progress towards proving a contradiction. Also, the only properties of the f-projection that we have used are completeness, nontriviality, and monotonicity. Note, however, that a "projection" that sends $\emptyset$ to $\emptyset$ and all other configurations to $\{\bot\}$ also

satisfies these conditions. Hence, the axiom downloads are where we must expect the action to take place, and we can also expect that we will have to make crucial use of the incremental soundness of the projection.

Assume that $\mathbb{D}_t = \mathbb{D}_{t-1} \cup \{L_A\}$ for a function L_A encoding some clause from the substitution clause set $A[f]$ corresponding to an axiom $A \in F$. We want to show that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ can be derived in π by downloading A, resolving (and possibly weakening) clauses, and then perhaps erasing A, and that all this can be done without the variable space exceeding $\mathit{VarSp}(\mathbb{C}_{t-1} \cup \mathbb{C}_t) \leq \mathit{VarSp}(\mathbb{C}_{t-1}) + \mathit{VarSp}(\mathbb{C}_t)$.

We already know how to derive clauses by weakening, so consider a clause $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that cannot be derived by weakening from $\mathbb{C}_{t-1}$. By the incremental soundness of the projection, it holds for all literals $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$ that the clauses $\overline{a} \vee C$ can be derived from $\mathbb{C}_{t-1}$ by weakening. Once we have these clauses, we can resolve them one by one with A to derive C.

Some care is needed, though, to argue that we can stay within the variable space bound $\mathit{VarSp}(\mathbb{C}_{t-1}) + \mathit{VarSp}(\mathbb{C}_t)$. Observe that what was just said implies that for all $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$ there are clauses $\overline{a} \vee C_a \in \mathbb{C}_{t-1}$ with $C_a \subseteq C$. In particular, we have $\overline{a} \in \mathit{Lit}(\mathbb{C}_{t-1})$ for all $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$. This is so since by the incremental soundness there must exist some clause $C' \in \mathbb{C}_{t-1}$ such that $\overline{a} \vee C$ is derivable by weakening from C', and if $\overline{a} \notin \mathit{Lit}(C')$ we would have that C is derivable by weakening from C' as well, contrary to assumption. Note furthermore that if the projection is local, then $\overline{a} \vee C_a \in \mathbb{C}_{t-1}$ implies that $\overline{a} \vee C_a \in \mathbb{C}_t$ as well, since no clauses can disappear from the projection when enlarging $\mathbb{D}_{t-1}$ to $\mathbb{D}_t$. Thus, for local projections we have $\mathit{VarSp}(\mathbb{C}_{t-1} \cup \{A\}) \subseteq \mathit{VarSp}(\mathbb{C}_t)$.

If it happens that all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ can be derived by weakening, we act as in the cases of inference and erasure above. Otherwise, to make the transition from $\mathbb{C}_{t-1}$ to $\mathbb{C}_t$ in a space-efficient fashion we proceed as follows.

1. Erase all clauses in $\mathbb{C}_{t-1} \setminus \mathbb{C}_t$ not used in any of the steps below.

2. Infer all clauses in $\mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that can be derived by weakening from $\mathbb{C}_{t-1}$.

3. Erase all clauses in $\mathbb{C}_{t-1} \setminus \mathbb{C}_t$ used in these weakening moves but not used in any further steps below.

4. Download the axiom clause A, and derive any clauses $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ such that $A \subseteq C$ by weakening.

5. For all remaining clauses $C \in \mathbb{C}_t \setminus \mathbb{C}_{t-1}$ that have not yet been derived, derive $\overline{a} \vee C$ for all literals $a \in \mathit{Lit}(A) \setminus \mathit{Lit}(C)$ and resolve these clauses with A to obtain C.

6. Erase all remaining clauses in the current configuration that are not present in $\mathbb{C}_t$, possibly including A.

Clearly, step 1 can only decrease the variable space, and steps 2 and 3 do not increase it. Step 4 can increase the space, but as was argued above we have $\mathit{Vars}(A) \subseteq \mathit{Vars}(C) \cup \mathit{Vars}(\mathbb{C}_{t-1})$ for every new clause C derived with the help of A. Step 5 does not change the variable space, and step 6 can only decrease it. It follows that the set of variables mentioned during these intermediate steps is contained in $\mathit{Vars}(\mathbb{C}_{t-1} \cup \mathbb{C}_t)$. If in addition the projection is local, we have $\mathbb{C}_{t-1} \subseteq \mathbb{C}_t$ and also $\mathit{Vars}(A) \subseteq \mathit{Vars}(\mathbb{C}_t)$, so in this case the variable space increases monotonically from $\mathbb{C}_{t-1}$ to $\mathbb{C}_t$.

Wrapping up the proof, we have shown that no matter what $\mathcal{P}$-derivation step is made in the transition $\mathbb{D}_{t-1} \rightsquigarrow \mathbb{D}_t$, we can perform the corresponding transition $\mathbb{C}_{t-1} \rightsquigarrow \mathbb{C}_t$ for our projected clause sets in resolution without the variable space going above $\mathit{VarSp}(\mathbb{C}_{t-1}) + \mathit{VarSp}(\mathbb{C}_t)$. Also, the only time we need to download an axiom $A \in F$ in our projected refutation π of F is when π_f downloads some axiom from $A[f_d]$. The lemma follows. □

4.3.9 Some Space-Faithful Projections

Let us pause and reflect on what Theorem 4.32 says. Suppose we have a family of CNF formulas F_n with lower bounds for refutation variable space in resolution, or with trade-offs between refutation length and refutation variable space (such as for instance pebbling contradictions over suitable graphs). Then we can lift these lower bounds and

trade-offs to stronger measures in a potentially stronger proof system $\mathcal{P}$, provided that we can find a Boolean function $f : \{0,1\}^d \to \{0,1\}$ and an f-projection $proj_f$ that is space-faithful with respect to $\mathcal{P}$.

Thus, at this point we can in principle forget everything about proof complexity. If we want to prove space lower bounds or time-space trade-offs for a proof system $\mathcal{P}$, we can focus on studying Boolean functions of the form used by $\mathcal{P}$ and trying to devise space-faithful projections for such functions.

Let us now briefly review how this is done in [30], and then prove Lemma 4.8. In what follows, let us write $Vars^d(V) = \{x_1, \ldots, x_d \mid x \in V\}$ for the set of variables resulting from substitution in a formula over variable set V. Also, we will abuse notation mildly and identify a set of disjunctive clauses with the Boolean function computed by these clauses.

Definition 4.33 (Precise implication [30]). Let f be a Boolean function of arity d, let $\mathbb{D}$ be a set of Boolean functions over $Vars^d(V)$, and let C be a disjunctive clause over V. If

$$\mathbb{D} \models C[f] \tag{4.5a}$$

but for all strict subclauses $C' \subsetneq C$ it holds that

$$\mathbb{D} \not\models C'[f] \ , \tag{4.5b}$$

we say that the clause set $\mathbb{D}$ implies $C[f]$ *precisely* and write

$$\mathbb{D} \rhd C[f] \ . \tag{4.6}$$

Definition 4.34 (Resolution projection [30]). Let f denote a Boolean function of arity d and let $\mathbb{D}$ be any set of Boolean functions over $Vars^d(V)$. Then we define

$$Rproj_f(\mathbb{D}) = \{C \mid \mathbb{D} \rhd C[f]\} \tag{4.7}$$

to be the *resolution projection* of $\mathbb{D}$. Also, we define $Rproj_f^*(\mathbb{D}) = \bigcup_{\mathbb{D}' \subseteq \mathbb{D}} Rproj_f(\mathbb{D}')$ to be the *local resolution projection* of $\mathbb{D}$.

Lemma 4.35. *The mapping $Rproj_f$ is an f-projection (for any sequential proof system $\mathcal{P}$).*

Proof. Suppose $\mathbb{D} \models C[f]$. Then we can remove literals from C one by one until we have some minimal clause $C' \subseteq C$ such that no more literal can be removed if the implication is to hold, and this clause C' is projected by $\mathbb{D}$ according to the definition. This proves both completeness and monotonicity for $Rproj_f$. Nontriviality is obvious.

For the incremental soundness, if $C \in Rproj_f(\mathbb{D} \cup \{L_A\})$ for an encoding L_A of some clause in $A[f]$, then this means, in particular, that $\mathbb{D} \cup \{L_A\} \models C[f]$. Consider any truth value assignment α such that $\alpha(\mathbb{D}) = 1$ but $\alpha\big(C[f]\big) = 0$. By assumption, $\alpha(L_A) = 0$. But this means that for all literals $a \in Lit(A)$ we have $\alpha\big(\overline{a}[f]\big) = 1$. Since this holds for any α, it follows for all $a \in Lit(A)$ that $\mathbb{D} \models (\overline{a} \vee C)[f]$, and we conclude by the completeness of the projection that the clause $\overline{a} \vee C$ is derivable by weakening from $Rproj_f(\mathbb{D} \cup \{L_A\})$. □

With this projection, and using Theorem 4.32, the main technical result in [30] can now be rephrased as follows, where we also need the following key definition.

Definition 4.36 (Non-authoritarian function [30]). We say that a Boolean function $f(x_1, \ldots, x_d)$ is *k-non-authoritarian*[12] if no restriction to $\{x_1, \ldots, x_d\}$ of size k can fix the value of f. In other words, for every restriction ρ to $\{x_1, \ldots, x_d\}$ with $|\rho| \leq k$ there exist two assignments $\alpha_0, \alpha_1 \supset \rho$ such that $f(\alpha_0) = 0$ and $f(\alpha_1) = 1$. If this does not hold, f is *k-authoritarian*. A 1-(non-)authoritarian function is called just *(non-)authoritarian*.

Theorem 4.37 ([30]). *If f is a non-authoritarian Boolean function, then the projection $Rproj_f^*$ is exactly space-faithful with respect to the resolution proof system.*

[12] Such functions have previously also been referred to as $(k+1)$-*robust* functions in [4].

We note in passing that [30] also proved a similar result, although with slightly worse parameters, for the so-called k-DNF resolution proof systems (provided that the substitution function is $(k+1)$-non-authoritarian).

Lemma 4.38 (Detailed version of Lemma 4.8). *If $\mathcal{P}$ is any implicational sequence proof system, then the projections $Rproj_f$ and $Rproj_f^*$ are both exactly space-faithful for variable space in $\mathcal{P}$.*

Proof. This is fairly straightforward. Consider first $Rproj_f$. If $Rproj_f$ is *not* exactly space-faithful, then there is a $\mathcal{P}$-configuration $\mathbb{D}$, a clause C, and a variable x such that $C \in Rproj_f(\mathbb{D})$ and $x \in Vars(C) \setminus Vars(\mathbb{D})$. Suppose without loss of generality that $C = C' \vee x$. By condition 4.5b in Definition 4.33 we have $\mathbb{D} \not\models C'$, so there is a truth value assignment α such that $\alpha(\mathbb{D}) = 1$ and $\alpha(C') = 0$. But we can flip α on x without affecting $\mathbb{D}$, since this variable does not occur, obtaining α' such that $\alpha'(\mathbb{D}) = 1$ and $\alpha'(C) = 0$. Contradiction.

For $Rproj_f^*$, we just focus on the subset $\mathbb{D}' \subseteq \mathbb{D}$ that is responsible for projecting C and then run the same argument. □

4.3.10 Designing Space-Faithful Projections for Stronger Proof Systems?

It seems like a very tempting approach to try to extend this projection framework to other proof systems than resolution and k-DNF resolution. In particular, it would be very interesting to see if one could prove space lower bounds and time-space trade-offs for cutting planes, polynomial calculus, or polynomial calculus resolution in this way. However, to do so another projection in the formal sense of Definition 4.6 than the one in Definition 4.34 would be needed. We conclude this section by explaining why the same projection as we used for resolution above will *not* work for PC or PCR. Namely, consider the following examples (where the original variables before substitution are $x[i]$ for $i = 1, 2, \ldots$).

Example 4.39. Using substitutions with $\oplus_2$, for polynomial calculus we have the exam-

ple

$$-1+\prod_{i=1}^{k} x[i]_1 x[i]_2 \tag{4.8}$$

showing that just two monomials can project the arbitrarily large conjunction $\overline{x[1]} \wedge \overline{x[2]} \wedge \cdots \wedge \overline{x[k]}$ if we use the projection in Definition 4.34.

Example 4.40. Let us also give a slightly more involved example for polynomial calculus resolution. For PCR, three monomials

$$-1+\prod_{i=1}^{k} x[i]_1 x[i]'_2 + \prod_{i=1}^{k} x[i]'_1 x[i]_2 \tag{4.9}$$

can project the arbitrarily large conjunction $x[1] \wedge x[2] \wedge \cdots \wedge x[k]$.

There are also similiar counter-examples that show why this projection does not work for cutting planes.

Somehow, the reason for these counterexamples is that the projection in Definition 4.34 allows the Boolean functions in the implication to be far too strong. These functions do not really imply just conjunctions of exclusive ors, but something much stronger in that they actually fix the variable assignments (to some particular assignment that happens to satisfy exclusive ors). Note that formulas $F[\oplus_2]$ do not speak about fixed variable assignments for, say, x_1 or x_2, but only about the value of $x_1 \oplus x_2$. Intuitively, therefore, the only way we can know something more about x_1 and x_2 than the value of $x_1 \oplus x_2$ is if the refutation has already derived contradiction and is now deriving all kinds of other interesting consequences from this. But before this happens, we would like to argue that any refutation must pass through a stage where all it can know about x_1 and x_2 is the value of $x_1 \oplus x_2$ and nothing more.

For this reason, we would like to find a more "fine-grained" definition of a projection that can capture only these weaker implications and discard too strong implications. It seems like a very interesting, though also quite challenging, question whether it is possible to prove space lower bounds and/or trade-offs between proof length/size and space for cutting planes, polynomial calculus, or polynomial calculus resolution by designing smarter projections than in Definition 4.34 that are space-faithful for these

proof systems. And note that Definition 4.6 provides quite a lot of flexibility here—what we have proven in this section is that any function satisfying the formal conditions in Definition 4.6 will automatically project $\mathcal{P}$-refutations to resolution refutations for any (implications sequential) proof system $\mathcal{P}$.

4.4 Trade-offs Between Space and Degree in PCR

In this short section, we show how Lemma 4.38 can be used to prove Theorem 4.1, which we restate here for reference.

Theorem 4.1. *There is a family of 3-CNF formulas F_n of size $\Theta(n)$ that can be refuted in polynomial calculus in degree $Deg_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}(1)$ and also in monomial space $Sp_{\mathcal{PC}}(F_n \vdash \bot) = \mathrm{O}(1)$, but such that for any PCR-refutation $\pi_n : F_n \vdash \bot$ it holds that $Sp(\pi_n) \cdot Deg(\pi_n) = \Omega(n/\log n)$.*

We start by recalling some facts about the trade-off between clause space and width in resolution, obtained by studying a particular kind of pebbling contradictions.

Theorem 4.41 ([25]). *There is a family of k-CNF formulas F_n of size $\Theta(n)$ such that $Sp(F_n \vdash \bot) = \mathrm{O}(1)$ and $W(F_n \vdash \bot) = \mathrm{O}(1)$ but $VarSp(F_n \vdash \bot) = \Omega(n/\log n)$.*

We need a bit more information about these formulas as stated next.

Observation 4.42 ([28]). *There are resolution refutations $\pi_n : F_n \vdash \bot$ of the formulas F_n in Theorem 4.41 with $L(\pi_n) = \mathrm{O}(n)$ and $W(\pi_n) = \mathrm{O}(1)$.*

Lemma 4.43 ([25]). *There are resolution refutations $\pi_n : F_n \vdash \bot$ of the formulas F_n in Theorem 4.41 with $L(\pi_n) = \mathrm{O}(n)$ and $Sp(\pi_n) = \mathrm{O}(1)$, and all clauses in π_n contain at most one positive literal.*

Since the resolution refutation in Observation 4.42 has constant width, polynomial calculus can simulate it in constant total degree. The resolution refutation in Lemma 4.43 is wide, which could be a problem for PC, but only if there are clauses containing many positive literals (by our choice of encoding of true and false in PC).

Since every clause in the refutation in Lemma 4.43 contains at most one positive literal, PC can simulate this refutation as well with at most a (small) constant factor blow-up in the monomial space as compared to the resolution clause space.

The following observation brings us to a point where we can conclude the proof.

Observation 4.44. *For any PCR-refutation π of a formula F it holds that $Sp(\pi) \cdot Deg(\pi) \geq VarSp(\pi) \geq VarSp_{\mathcal{PCR}}(F \vdash \bot)$.*

Proof. The refutation π never has more that $Sp(\pi)$ monomials in memory, and each monomial has degree at most $Deg(\pi)$. Thus the total number of distinct variables in memory at any point during the course of the PCR-refutation π is at most $Sp(\pi) \cdot Deg(\pi)$. □

And we clinch the argument by observing that the refutation variable space must be the same in PCR as in resolution.

Observation 4.45. *For any CNF formula F it holds that $VarSp_{\mathcal{PCR}}(F \vdash \bot) \geq VarSp_{\mathcal{R}}(F \vdash \bot)$.*

Proof. Apply Lemma 4.38 on $Rproj_f^*$ and then appeal to part 3 of Lemma 4.31. □

Putting all of this together, Theorem 4.1 follows.

4.5 Time-Space Trade-offs for PCR in the Sublinear Space Regime

Using the substitution theorem for PCR that we proved in Section 4.3.3, we can now go over the time-space trade-off results for resolution in [30] and lift most of them to PCR. The upper bounds will hold for total space syntactic resolution and polynomial calculus, whereas the lower bounds hold for monomial space in semantic PCR, i.e., a stronger space measure in a much stronger proof system. In contrast to [30], however, the upper and lower bounds in the trade-offs are no longer as tight, since the random restriction part of the argument leads to a loss of a logarithmic factor in the upper bound on the proof size. Let us recall our substitution theorem here for reference.

Theorem 4.9 (restated). *Suppose that F is a CNF formula for which any syntactic resolution refutation in variable space at most s must make more than T axiom downloads. Then any semantic PCR-refutation of $F[\oplus]$ in monomial space at most $s/\log_{4/3} T$ must have size larger than T.*

Given this theorem, there is a (literally) infinite supply of pebbling contradictions that it could be applied to in order to yield PCR time-space trade-offs. As was done also for the resolution trade-offs in [30], we try to simply state a few interesting examples here.

Theorem 4.46 (Trade-offs for constant space). *There are explicit 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that the following holds:*

- *The formulas F_n are refutable in syntactic resolution and polynomial calculus in total space $\mathrm{O}(1)$.*
- *For any $g(n) = \mathrm{O}\big(\sqrt{n}\big)$ there are syntactic resolution and polynomial calculus refutations π_n of F_n in simultaneous length/size $\mathrm{O}\big((n/g(n))^2\big)$ and total space $\mathrm{O}(g(n))$.*
- *For any semantic PCR-refutation $\pi_n : F_n \vdash \bot$ in monomial space $Sp(\pi_n) \leq g(n)$ it holds that $S(\pi_n) = \Omega\left(\big(n/(g(n)\log n)\big)^2\right)$.*

Theorem 4.46 follows by combining Theorem 4.9 with the seminal work on pebbling trade-offs by Lengauer and Tarjan [74] and the structural results on simulations of black-white pebblings by resolution in [84].

Our next result relies on a new pebbling time-space trade-off result in [84], building on earlier work [40, 41], which yields the rather striking statement that for any *arbitrarily slowly growing* non-constant function, there are explicit formulas of such (arbitrarily small) space complexity that nevertheless exhibit *superpolynomial* length-space trade-offs.

Theorem 4.47 (Trade-offs for growing space). *Let $g(n) = \omega(1)$, $g(n) = \mathrm{O}\big(n^{1/7}\big)$, be any arbitrarily slowly growing function and fix any $\varepsilon > 0$. Then there are explicit 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that the following holds:*

- *The formulas F_n are refutable in syntactic resolution and PC in total space $\mathrm{O}(g(n))$.*
- *There are resolution and PC refutations of F_n in simultaneous length/size $\mathrm{O}(n)$ and total space $\mathrm{O}\left(\left(n/g(n)^2\right)^{1/3}\right)$.*
- *Any PCR-refutation of F_n in monomial space $\mathrm{O}\left(\left(n/(g(n)^3 \log n)\right)^{1/3-\varepsilon}\right)$ must have superpolynomial size.*

All multiplicative constants hidden in the asymptotic notation depend only on ε.

The two theorems above focus on trade-offs for formulas of low space complexity, and the lower bounds on length obtained in the trade-offs are somewhat weak—the superpolynomial growth in Theorem 4.47 is something like $n^{g(n)}$. We next present a theorem that has both a stronger superpolynomial length lower bounds than Theorem 4.47 and an even more robust trade-off covering a wider (although non-overlapping) space interval. This theorem again follows by applying our tools to the pebbling trade-offs in [74].

Theorem 4.48 (Trade-off for medium-range space). *There are explicit 6-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\Theta(n)$ such that the following holds:*

- *The formulas F_n are refutable in syntactic resolution and PC in total space $\mathrm{O}(\log^2 n)$.*
- *There are syntactic resolution and PC refutations of F_n in simultaneous length in length $\mathrm{O}(n)$ and total space $\mathrm{O}(n/\log n)$.*
- *Any semantic PCR-refutation of F_n in monomial space $\mathrm{o}(n/\log^3 n)$ must have size $n^{\Omega(\log\log n)}$.*

Having presented trade-off results in the low-space and medium-space range, we conclude by presenting a result at the other end of the space spectrum. Namely, appealing one more time to [84], we can show that there are formulas of polynomial (but still sublinear) space complexity that exhibit not only superpolynomial but even exponential trade-offs.

Theorem 4.49 (Exponential trade-off). *There is a family of explicit* 6*-CNF formulas* $\{F_n\}_{n=1}^{\infty}$ *of size* $\Theta(n)$ *such that the following holds:*

1. *The formulas* F_n *are refutable in syntactic resolution and PC in total space* $\mathrm{O}\big(n^{1/11}\big)$.

2. *There are syntactic resolution and PC refutations* π_n *of* F_n *in simultaneous length/size* $\mathrm{O}(n)$ *and total space* $\mathrm{O}\big(n^{3/11}\big)$.

3. *Any semantic PCR-refutation of* F_n *in monomial space at most* $n^{2/11}/(10\log n)$ *must have size at least* $\big(n^{1/11}\big)!$.

As the knowledgeable reader might have noticed, however, we have no analogues of the strongest trade-offs in [30]. This is so because when the proof length in resolution becomes exponential, the log factor loss in the restriction argument becomes so big that it more or less cancels out the space lower bound.

4.6 Extended Isoperimetry

For any undirected graph $G = (V, E)$, and $S \subseteq V$, let $\delta(S)$ denote the set of *boundary edges*, $\delta(S) := \{(v, v') \in E : \text{exactly one of } v, v' \in S\}$. An isoperimetric inequality in graph theory refers to a lower bound on $\delta(S)$ as above which holds for any subset of S in G depending only on the cardinality of S. Classical work considered isoperimetric inequalities in Euclidean space; Harper's theorem gives a tight isoperimetric inequality in the hamming cube. Isoperimetry in graphs is closely related to graph expansion, see [68]. We require a variation on this idea, in which multiple sets of superincreasing sizes are considered.

Definition 4.50. Let $G = (V, E)$ be a undirected graph, and t_0 an associated parameter. A set of vertices of size between $t_0, |V|/2$ is called *medium sized*.

We say G satisfies the *extended isoperimetry* condition with parameters (W, t_0, r) if for any sequence of medium sized sets of vertices $S_1, \ldots, S_k \subseteq V$, where $\forall i \geq 2$, $|S_{i+1}| \geq r|S_i|$, it holds that $|\bigcup_i \delta(S_i)| \geq k \cdot W$.

We will now show that the grid has this property for appropriate parameters. In fact, W will be close to the width of the grid, so that there are sets of every medium size with only $O(W)$ boundary edges, and this implies that the lower bound we obtain above is tight.

As a starting point, we first show that all path graphs have the property. It is easiest to start with the infinite path graph. Let G_Z denote the graph defined by $V[G] := Z$, $E[G] := \{(i, i+1) : i \in Z\}$, that is, the undirected cayley graph on the integers with generator 1.

Lemma 4.51. *Let $S_1, \ldots, S_k$ be a nonempty sequence of finite nonempty subsets of the integers such that $|S_1|, \ldots, |S_k|$ is a superincreasing sequence. (That is, each successive value is at least twice as large as the previous.) Then, $|\bigcup_i \delta(S_i)| \geq k + c$, where boundary refers to the graph G_Z and c is the number of connected components of the subgraph induced by $\bigcup_i S_i$ in G_Z.*

Proof. Let $H_{k,c}$ denote the proposition for specific values of k, c. We prove that $H_{k,c}$ holds for all values by induction on k and c. The case that $k = 1, c = 1$ is trivial.

Suppose inductively that $H_{k,c}$ holds. We prove it for $H_{k,c+1}$. For any sets $S_1, \ldots, S_k$ such that $H_{k,c+1}$ applies $\bigcup_i S_i$ has at least two connected components, so there exists a point z not in any set which is between two components and such that $z - 1$ is in one of the sets. Consider the function $f : Z \to Z$ defined by

$$f(x) := \begin{cases} x & x < z \\ x - 1 & x \geq z \end{cases}.$$

That is, f contracts the points $z, z+1$ to one point. If $z + 1$ is not in any of the sets S_i, then f clearly preserves the size of every set, the number of boundary edges, and the number of connected components. So in this case, $H_{k,c+1}$ holds for $S_1 \ldots S_k$ if and only if it holds for $f(S_1), \ldots, f(S_k)$. Since there is a point in one of the S_i which is above z, after a finite number of applications of this we obtain an instance such that $z+1$ is in one of the sets. Now we handle this case. Since z is not a point in any set, we still have $|f(S_i)| = |S_i|$ for every set. Since there is a component of $\bigcup_i S_i$ containing

$z-1$ and component containing $z+1$, the images of these two components are one component afterwards, and no other collisions could have occurred so the number of components is reduced by exactly one. Further, there is at least one less edge in the $\bigcup_i \delta(f(S_i))$ compared with $\bigcup_i \delta(S_i)$ since the edges $(z-1,z)$ and $(z,z+1)$, which were boundary edges, collided after application of f, and no other edges collided or disappeared. By hypothesis that $H_{k,c}$ holds, we conclude that $|\bigcup_i \delta(f(S_i))| \leq k+c$, so $k+c+1 \leq |\bigcup_i \delta(S_i)|$, so $H_{k,c+1}$ holds for this instance as desired.

Suppose inductively that $H_{k,c}$ holds for all c. We prove $H_{k+1,1}$. For any sets $S_1, \ldots, S_{k+1}$ such that $H_{k,1}$ applies, let c' denote the number of connected components of $\bigcup_{i=1}^{k} S_i$. If $c' > 1$, then by $H_{k,c'}$, $|\bigcup_{i=1}^{k} \delta(S_i)| > k+1$, so $|\bigcup_{i=1}^{k+1} \delta(S_i)|$ is at least as large and $H_{k+1,1}$ is satisfied. Suppose $c'=1$. Then, the least and greatest points of $\bigcup_{i=1}^{k} S_i$ are at most $\sum_{i=1}^{k} |S_i| < |S_{k+1}|$ apart, so one of either the maximum or minimum point of S_{k+1} is an extreme point of $\bigcup_{i=1}^{k+1} S_i$ which is not in $\bigcup_{i=1}^{k} S_i$. Its boundary edge in the extreme direction is thus a boundary edge of S_{k+1} which is not a boundary edge of $\bigcup_{i=1}^{k} S_i$, which implies $|\bigcup_{i=1}^{k+1} \delta(S_i)| > |\bigcup_{i=1}^{k} \delta(S_i)| \geq k+1$, so $H_{k+1,1}$ holds. □

Lemma 4.52. *For any $\epsilon > 0$, for large enough n, the $n \times \ell$ grid, where $4n^2 \leq \ell \leq 2^n$, satisfies the extended isoperimetry condition with parameters $(W = n, t_0 = 4n^3, r = 2+\epsilon)$.*

Proof. For a set of vertices S, we call a column of the grid *full* if all its n vertices are included in S, *empty* if none of its vertices is included, and *partial* otherwise. If for some S_i, there are more than kn partial columns, each of which introduces at least one vertical boundary edge, thus the lemma holds. Therefore without loss of generality, the number of partial columns for every S_i is less than $kn < n^2$ (note that $k < n$).

It suffices to show that in each row we obtain at least k horizontal boundary edges, since an edge is a horizontal edge in at most one row and there are n rows. Fix any row, and for each S_i let S_i' denote the index number of columns which contain a vertex of S_i' from this row. If we can show that all ratios $|S_i'|/|S_{i-1}'|$ are at least 2, then the previous lemma implies that $|\bigcup \delta_{G_Z}(S_i')| \geq k+c$, where c is the number of components of

$\bigcup S'_i$ in G_Z. For every edge of this union *except* the edges $(0,1)$ and $(\ell, \ell+1)$ there is a corresponding horizontal edge of $\delta(S_i)$ in this row. If $|\bigcup \delta_{G_Z}(S'_i)|$ contains at most one of these two, then since $c \geq 1$ we obtain at least k horizontal edges as desired. If $|\bigcup \delta_{G_Z}(S'_i)|$ contains both of $(0,1)$, $(\ell, \ell+1)$, then $c \geq 2$. For suppose $c = 1$, then $1, \ell$ are in the same connected component of $\bigcup S'_i$, yet $|\bigcup S'_i|$ is strictly less than ℓ, since the seuence is superincreasing and hence $\sum_i |S'_i|$ is upper bounded by $2|S'_k|$. Thus in this case as well, there are at least k horizontal edges as desired.

Now we bound the ratios $|S'_i|/|S'_{i-1}|$. $|S'_i| \geq |S_i|/n - n^2$, since $|S'_i|$ is at least the number of full columns in S_i; and also $|S'_i| \leq |S_i|/n + n^2$, since there are at most $|S_i|/n + n^2$ columns which are full or partial. By assumption, $|S_{i+1}| \geq r|S_i|$. Therefore, $\forall i : 1 \leq i \leq k' - 1$

$$|S'_{i+1}|/|S'_i| \geq \frac{|S_{i+1}|/n - n^2}{|S_i|/n + n^2} \geq r(1 - n^2/t_0)^2 = r(1 - o(1))$$

here using that $t_0 = 4n^3$ is a lower bound on $|S_i|$.

Thus for any $\epsilon > 0$, for large enough n, $r(1 - o(1))$ will exceed two as required. This completes the proof. □

4.7 Trade-offs For Tseitin Tautologies

4.7.1 Overview

A long line of work [46, 72, 31] has resulted in several techniques in resolution for proving a size lower bound for proofs of a tautology from a width lower bound for proofs of that tautology – here, width refers to the maximum over all clauses in the proof of the number of variables in the clause. Proving lower bounds on proof width is significantly simpler, and many techniques exist which we will discuss later. These techniques take several forms – in the very well known work of Ben-Sasson & Wigderson [31], a very strong width lower bound for a formula is shown to immediately imply a very strong size lower bound for that formula. However, more relevant to us is in an older work of Beame & Pitassi [16], showing that if a formula has a nice distribution of random

restrictions associated to it such that the restricted formula satisfies a width lower bound of any kind, then that formula obtains a corresponding size lower bound. This argument often yields essentially tight size lower bounds at many different ranges of hardness, in contrast to the technique of Ben-Sasson & Wigderson. In [25], a generic technique was given for taking a formula F and obtaining a formula F' which has such a nice distribution of restrictions, all of which yield F. This technique is $\oplus$-substitution which we defined earlier. In the case of Tseitin Tautologies which we study, this technique gives essentially optimal size lower bounds – that is, the easy and tight width lower bound for F coupled with this argument gives a size lower bound for $F[\oplus]$ which is tight up to a constant in the exponent.

In the next section, we will show that the results of [14] can be cast in the following terms: If F satisfies an extended width lower bound property, then $F[\oplus]$ obeys a time-space trade-off lower bound, which in the high space regime agrees with (and thus extends) the size lower bound just described. In the case of Tseitin on the grid, the size lower bound is optimal up to constant factors, so this yields a true time-space trade-off result. To establish the extended width lower bound for Tseitin formulas on a grid, it is sufficient to augment existing techniques for width lower bounds with the extended isoperimetric property of the grid described in Section 4.6. Our arguments will show that any graph with extended isoperimetry would do here. Additionally, this improves over the results of [14] by permitting us to carry out the argument on k-CNFs for $k = O(1)$ rather than growing with the size of the formula.

In the subsequent section, we will discuss how to extend this paradigm to give lower bounds in PCR. In PCR a roughly analogous size vs. width paradigm exists, where degree is substituted for width. In [39], a beautiful technique for this was to take linear substitutions for the variables of a formula, and show a degree lower bound for proofs of the substituted formula. For Tseitin Tautologies, the most natural map to take is the Fourier Transform, and a series of other technical insights related to binomial PC results in a very intuitive degree lower bound result for Fourier Transformed Tseitin and hence standard Tseitin. In many ways this result is a natural extension of the corresponding

results for Resolution. We revisit these techniques and show that with additional careful analysis, they can be married with the ideas of Section 4.7.5 to give essentially the same results for PCR as we obtained for Resolution in Section 4.7.3.

4.7.2 Resolution Refutations of Tseitin Formulas

A standard technique described by [31] to show width lower bounds is to take advantage of the following measure of progress of any proof, which is definable in any proof system. Let A be a set of contradictory formulas. For ϕ a proof line in a refutation of A, let

$$\mu_A(\phi) := \min_{\substack{S \subseteq A \\ S \models \phi}} |S| .$$

That is, we ignore issues of whether ϕ can be feasibly proven from A and just consider how many of the axioms in A we need to *semantically* imply ϕ to judge roughly how valuable it is in the proof.

This simple measure enjoys nice properties. We always have $\mu_A(a) = 1$ for $a \in A$, and μ is always *subadditive*: If $\phi_1, \phi_2 \vdash \phi_3$ is a step of any proof in a sound proof system, then $\mu_A(\phi_1) + \mu_A(\phi_2) \geq \mu_A(\phi_3)$. This is because if the derivation is sound, then the union of the minimium sets of axioms for ϕ_1, ϕ_2 will semantically imply both ϕ_1 and ϕ_2, and thus ϕ_3. So for instance, if π is any refutation of A, there will always exist a "medium complexity" formula $\phi \in \pi$ such that $\mu_A(\bot)/3 \leq \mu(\phi) < 2\mu_A(\bot)/3$. For suppose there was not – then the first time a formula of complexity exceeding $2\mu_A(\bot)/3$ was derived in the refutation, it was derived from two formulas of complexity less than $\mu_A(\bot)/3$, contradicting subadditivity.

An important technical insight concerning such measures [31] is that if A is well chosen and the lines of the proof system we consider have restricted expressive power, then such a "medium complexity" ϕ will necessarily be complicated as a Boolean formula. In the case of resolution, ϕ will be a clause with many variables, and so in this way we obtain a width lower bound for refutations of A. Generally, it is sensible to

choose A to be minimally contradictory, since this gives us $\mu_A(\bot) = |A|$, and for any contradictory A we can usually think of an interesting minimally contradictory subset. In the sequel we'll restrict attention to this case.

Several authors [96, 14] observed that in fact this argument shows much more; there is also a ϕ' with $\frac{\mu_A(\phi')}{\mu_A(\bot)} \in [1/6, 1/3)$, a ϕ'' with $\frac{\mu_A(\phi')}{\mu_A(\bot)} \in [1/12, 1/6)$, etc. While the existence of several collectively wide clauses can be useful for some applications, to obtain time-space trade-offs we need to use more than this. For the sequel, it is convenient to transition to the following definition.

Definition 4.53. A function μ' mapping formulas of a proof system to $\mathbb{N}$ is *strongly bounded* if, whenever $\phi_1, \phi_2 \vdash \phi_3$ follows in one proof step, then $\mu'(\phi_3) \leq 1 + \max(\mu'(\phi_1), \mu'(\phi_2))$.

Such a μ' is a *strongly bounded complexity measure for* A if it is strongly bounded and $\mu'(a) = 0$ for all $a \in A$.

Of course, it is easy to see that if μ_A is a subadditive complexity measure as defined before, then $\mu'_A := \lfloor \log_2 \mu_A \rfloor$ is a strongly bounded complexity measure for A. μ' corresponds to the "complexity levels" as appeared in [14].

The following simple observation appears implictly in previous work, and gives a generic way in which such a measure can give rise to a notion of "progress" occurring in different epochs (time periods) of a proof.

Definition 4.54. Let π be a proof in a sequential proof system which is divided recursively into epochs. For E an epoch, say that its *critical set* of proof lines is

- If E is an internal epoch, the set of proof lines appearing in memory at breakpoints between sons of E
- If E is a leaf epoch, the set of all proof lines appearing in E.

Lemma 4.55 (Subdivision argument). *Let A be a set of formulas from a sequential proof system $\mathcal{P}$ such that there is a strongly bounded complexity measure μ' for A, and $\mu'(\bot) = L$. If a refutation of A is divided recursively into epochs to a recursive depth of h, then one of the following holds.*

1. *There exists a leaf epoch containing $L \cdot k^{-h}$ formulas with distinct values under μ'.*

2. *There exists an internal epoch such that the critical set of proof lines contains at least k formulas with distinct values under μ'.*

Proof. Say that an internal epoch represents a "progress gap" of g if for some ℓ, h, $g = h - \ell$ every formula ϕ in the initial configuration has complexity at most ℓ, but the final configuration contains a formula of complexity at least h. By definition, the entire proof represents a progress gap of at least L.

Suppose that the second condition fails. Then for any internal epoch representing a gap of g, one of its children represents a gap of at least g/k; since only k complexities appear at breakpoints between children of this epoch, there exist by averaging a successive pair of these values ℓ', h' with $h' - \ell' > g/k$, and so the first time a value of complexity $\geq h'$ appears at such a breakpoint, the epoch immediately preceeding it begins with only formulas of complexity $\leq \ell'$ and so this epoch respresents a progress gap of g/k.

By induction, we conclude that some leaf epoch represents a progress gap of at least $L \cdot k^{-h}$; let ℓ, h be according to the definition. At the beginning of the epoch we have only formulas of complexty at most ℓ, but at the end we have a formula of complexity at least h, so by strong boundedness, there is also a formula in this epoch of complexity $h-1$. Consider the first such formula. Again by strong boundedness, there is a formula of this epoch of complexity $h-2$, and so on, repeating the argument until a formula of complexity $\ell + 1$ is found. Thus this leaf epoch contains formulas of at least $L \cdot k^{-h}$ complexities. This completes the proof. □

For the class of Tseitin formulas, the Ben-Sasson Wigderson style complexity measure described is very well studied. A well known application is that if a graph has no balanced cut of size less than W, where balanced means between $1/3$ and $2/3$ of the vertices on either side, then the corresponding Tseitin formula has a clause of width at least W in any refutation. Follow-up work [27, 5] has shown that this is generally

tight. The idea is to show that if C is a clause then every variable on the boundary of S_C appears in C. To carry out the [14] strategy we need to prove this or something similar. We will also need a stronger version for when we ultimately handle polynomial calculus, so we will state this lemma in slightly greater generality, which we can do without complicating the proof.

Lemma 4.56. *Let ϕ be a disjunction of $\mathbb{F}_2$-linear equations in the variables x_e of a Tseitin formula on graph G. Let S be a minimal subset of the vertices such that $\{PARITY_v\}_{v \in S} \models \phi$. Then, $Vars(\phi) \supseteq \{x_e : e \in \delta(S)\}$.*

Proof. The assumptions imply that S is a minimal subset such that $\{PARITY_v\}_{v \in S} \wedge \neg\phi \models \bot$. Since ϕ is a disjunction of $\mathbb{F}_2$ equations, $\neg\phi$ is a conjunction of $\mathbb{F}_2$ equations, and our assumption is that $\{PARITY_v\}_{v \in S} \wedge \neg\phi$ is an inconsistent system of equations. Without loss of generality, we may assume that $\neg\phi$ is satisfiable, since if not then minimality of S implies that S is empty and the claim is vacuous.

By basic linear algebra this implies that it is possible to derive $0 = 1$ from the system via $\mathbb{F}_2$-linear combinations. Suppose that in this linear combination, one of the equations $PARITY_v$ has a coefficient of 0. Then, when we remove that equation, the linear combination would still derive $0 = 1$ from the subsystem, so that subsystem is inconsistent, contradicting minimality of S. Therefore each $PARITY_v$ equation has a coefficient of 1 in this linear combination, or in otherwords, the sum of the $PARITY_v$ equations are inconsistent with $\neg\phi$. If there is a variable of this sum which does not appear in $\neg\phi$, then clearly the sum and $\neg\phi$ are satisfiable if $\neg\phi$ is. So we must have $Vars(\sum_{v \in S} PARITY_v) \subseteq Vars(\neg\phi)$. Now we use the structure of the Tseitin equations. Every variable x_e occurs in exactly two equations $PARITY_v$, and so if $e \in \delta(S)$, then x_e occurs in exactly one of the summands in $\sum_{v \in S} PARITY_v$. Therefore it does not cancel in the sum and is a variable of $\sum_{v \in S} PARITY_v$. This completes the proof. □

The previous lemma gives a connection between isoperimetry in a graph and the following standard complexity measure [31] for Tseitin.

Definition 4.57 (Complexity Measure). Let C be a clause in the variables of a Tseitin formula on a connected graph G. Then define the measure μ by

$$\mu(C) := \min_{\substack{S \subseteq V \\ \{PARITY_v\}_{v \in S} \models C}} |S| .$$

Further, let S_ϕ denote any fixed set S achieving the minimum above for ϕ.

Observation 4.58. *The following statements are true,*

1. *For a any axiom in any representation of Tseitin, $\mu(a) = 1$.*
2. *$\mu(\bot) = |V|$.*
3. *μ is subadditive.*
4. *$\{x_e : e \in \delta(S_C)\} \subseteq Vars(C)$.*

Proof. The first three follow directly from the discussion. To see the fourth, observe that a clause can be thought of as a disjunction of $\mathbb{F}_2$-linear equations, so Lemma 4.56 applies. □

4.7.3 Time-Space Trade-off for Resolution

Now we are ready to show that extended isoperimetry implies a time-space trade-off.

Proposition 4.59. *If a graph $G = (V, E)$ satisfies extended isoperimetry with parameters W, t_0, and μ' is the strongly bounded complexity measure defined from the previous complexity measure μ via*

$$\mu'(C) = \begin{cases} 0 & \mu(C) < t_0 \\ L & \mu(C) > |V|/2 \\ \log_2(\mu(C)/t_0) & \textit{otherwise} \end{cases}$$

then for any k clauses $C_1, \ldots C_k$ with distinct values under μ' between 0 and L,

$$\left|\bigcup Vars(C_i)\right| \geq \Omega(kW)\ . \tag{4.10}$$

Proof. Order the C_i by value of μ'. If the C_i have distinct values under μ', then $\mu'(C_{i+3}) \geq \mu'(C_i) + 3$, and by definition of μ', $\mu(C_{i+3}) \geq 4 \cdot \mu_{C_i}$. This implies $|S_{C_{i+3}}| \geq 4 \cdot S_{C_i}$, and extended isoperimetry implies $|\bigcup_i \delta(S_{C_{3i+1}})| \geq kW/3$. By Observation 4.58, this implies $|\bigcup_i Vars(C_i)| \geq kW/3$ as desired. □

Theorem 4.60. *If F is any CNF with a strongly bounded complexity measure μ for F satisfying Equation 4.10, and $\mu(\bot) = L$, then $F[\oplus]$ satisfies $(2^{\Omega(W)}/S)^{\Omega\left(\frac{\log\log L}{\log\log\log L}\right)}$.*

Proof. Let π be any proof of $F[\oplus]$. Divide π into epochs recursively to a recursive depth of h, dividing each epoch into m equal subepochs, h, m to be determined later. Now consider the random restriction ρ. Since $F[\oplus]\restriction_\rho = F$, $\pi\restriction_\rho$ is a refutation of F. Choose k such that $Lk^{-h} = k$. Since μ' is a strongly bounded complexity measure, Lemma 4.55 implies that the critical set of some epoch of $\pi\restriction_\rho$ contains at least k clauses of distinct complexity values, with probability one.

However, for M any small set of clauses, the probability that $M\restriction_\rho$ contains k distinct complexities is seen to be small. If a collection of clause each is not restricted to a constant by ρ, then their disjunct is not restricted to a constant either. Therefore by Lemma 4.28, the probability that any fixed k-tuple of clauses all survive and have collective width $\geq X$ is at most exponentially small in X. In our case, the collective width is at least $\Omega((k-2)\cdot W)$, by Proposition 4.59. By a union bound over all k-tuples of M, the probability that any k of them have distinct complexity values is at most $|M|^k 2^{-\Omega((k-2)W)}$.

Choose the parameter m so that $mS = T/m^{h-1}$, so that all critical sets of epochs have the same size. Choose h so that $h = k$ and $k^h = L$. The probability that any critical set of any epoch contains k complexities after the restriction is at most $(mS\exp(-\Omega(W)))^k$, and there are at most m^h epochs. By a union bound, the probability that any critical set of any epoch contains k complexities after the restriction

is at most $(m^2 S \exp(-\Omega(W)))^k$. Since we know this must happen with certainty, we conclude $m^2 \geq 2^{\Omega(W)}/S$, or $T \geq \left(2^{\Omega(W)}/S\right)^{\Omega(k)}$. Here k is such that $k^k = L$, so in terms of our L our exponent exceeds $\log L / \log \log L$, as desired. □

In Section 4.6 we gave a simple proof that the $n \times \ell$ grid satisfies extended isoperimetry with parameters $(W = n, t_0 = 4n^3, r = 2 + \epsilon)$, so this theorem yields lower bounds of the form $\left(2^{\Omega(n)}/S\right)^{\Omega\left(\frac{\log \log n}{\log \log \log n}\right)}$. For a discussion of the relevant upper bounds on these formulas, see Section 4.3.5.

4.7.4 PCR Refutations of Tseitin Formulas

Our main result shows that actually, the trade-off lower bounds we just prove hold in PCR for these formulas. In the prevailing philosophy, resolution size lower bounds via width lower bounds are analogous to PCR size lower bounds via degree lower bounds, as we discussed. Thus, one would expect that an "extended degree lower bound" analogous to Equation 4.10 would imply time-space trade-offs via the subdivision and restriction argument we just saw. However, degree lower bounds for PCR are significantly more challenging to prove in general, and many of the techniques that have been developed (e.g. [6]) appear ill-suited for obtaining an extended degree lower bound. We illustrate one elegant solution which builds on the techniques of [39] for degree lower bounds. The crux of the argument is a pair of simulations, which while in general not "efficient" as measured by most standard proof complexity measures, are efficient with respect to degree. We revisit these simulations and give a refined analysis. The simulations convert a PCR refutation to a binomial PC refutation. Using a standard Ben-Sasson Wigderson style complexity measure, we are able to extend degree lower bounds in this setting directly from Lemma 4.28 and extended isoperimetry. The flavor of the final proof changes slightly from what we saw in resolution – ultimately we do not establish a strongly bounded complexity measure for PCR with the extended degree lower bound, but we do establish such a measure for binomial PC, and modulo the simulations, this is enough for the subdivide-and-restrict approach to work.

4.7.5 PCR Simulations: Fourier-Transformed Tseitin

Essential to our argument is a pair of simulations connecting PCR refutations of standard Tseitin to refutations of a "fourier transformed" version of Tseitin. These simulations were exploited by [39] for proving degree lower bounds, and are also essential for our time-space trade-off lower bounds.

First, we introduce an alternate formulation of Tseitin as a system of binomials over a field of odd characteristic.

The previous CNF will be called $\{0,1\}$-Tseitin (suppressing G, χ for now). Following [39], we define an alternate instance $\{+1,-1\}$-Tseitin, formed by defining fourier transformed variables $y_e := 1 - 2x_e$, that is, $x_e = 0 \iff y_e = 1$, $x_e = 1 \iff y_e = -1$, and expressing the parity constraints in this new basis. Now, the value of a linear equation $\bigoplus x_e$ corresponds to the value of $\prod y_e$ under this correspondence, due to the fourier transform.

Definition 4.61. Given G, χ, the PCR instance of $\{+1,-1\}$-Tseitin is defined by variables

$$\begin{aligned} \text{Variables: } \forall e \in E & \qquad y_e \\ \{\text{+1,-1}\}\text{-Constraints: } \forall e \in E & \qquad y_e^2 - 1 = 0 \\ \text{Parity Constraints: } \forall v \in V, & \qquad \prod_{e \sim_G v} y_e = (-1)^{\chi(v)} \end{aligned}$$

The primary reason why $\{+1,-1\}$-Tseitin is preferable is that every axiom above is a binomial, which was not the case for $\{0,1\}$-Tseitin, and binomial systems are much simpler.

4.7.6 PCR Simluations: Reductions

Definition 4.62. Let π be a PCR refutation of $\{0,1\}$-Tseitin. We define an induced / simulated PC refutation π' of $\{-1,1\}$-Tseitin in the straightforward way, maintaining the invariant that whenever π has a configuration $\mathcal{P}$, we will have a corresponding configuration with polynomials in the variables y_e which are semantically equivalent modulo $y_e = 1 - 2x_e$.

- When π downloads an axiom, π' downloads a semantically equivalent axiom in the new basis, by downloading one of the $\{-1,1\}$ axioms and weakening it.
- When π performs a weakening step, we perform a weakening and a linear combination step, and an erasure step. (when $p \vdash x_i \cdot p$ is inferred, we must simulate this with $p' \vdash y_i \cdot p'$ followed by $p', y_i \cdot p' \vdash (1 - 2y_i) \cdot p'$ to obtain a semantically equivalent polynomial)
- When π performs a linear combination step, we do the same to the corresponding polynomials.
- When π erases an polynomial, we erase the analogous polynomial.

The correctness of this simulation should be clear, and it should all be clear that when π is in fact a refutation, π' will also be a refutation.

For [39] the crucial property of this simulation was that it did not increase the degree – since the "fourier transform" represents a linear substitution of the variables, if there was no monomial of large degree before the substitution will not introduce one. For us, the crucial property of this simulation is that it is "conservative with respect to monomials".

Claim 4.63. The simulation of PCR on $\{0,1\}$-Tseitin by PC on $\{+1,-1\}$-Tseitin is *conservative with respect to monomials*;

- If for some *configuration* of the simulated proof no active monomial contains the set of variables $\{x_e : e \in E'\}$ for some $E' \subseteq E$, then the corresponding configu-

ration of the simulating proof does not contain any active monomials containing all of $\{y_e : e \in E'\}$.

- If for some *time period* in the simulated proof no active monomial contains the set of variables $\{x_e : e \in E'\}$ for some $E' \subseteq E$, then the corresponding time period of the simulating proof does not contain any active monomials containing all of $\{y_e : e \in E'\}$.

Proof. The only steps which introduce new monomials are download and weakening steps. The only steps which cause monomials to dissappear from the original proof are erasure steps. It is trivial to see that none of these will cause of violation of conservativity. □

Our second simulation is more sophisticated; it requires a clever algebraic argument which will allow us to work with polynomials of a simple form. This lemma is an adaptation of the main technical lemma of [39].

Lemma 4.64. *Suppose $\mathcal{B}$ is a set of binomials with coefficients in a field. Suppose p is a polynomial which can be written as a linear combination of elements of $\mathcal{B}$. Then by repeatedly taking linear combinations of binomials of $\mathcal{B}$ which each time yield a binomial and adding them to $\mathcal{B}$, it is possible to obtain a larger $\mathcal{B}'$ such that p is a no-cancellation linear combination of elements of $\mathcal{B}'$, i.e. for some coefficients α, $p = \sum_{b \in \mathcal{B}'} \alpha_b \cdot b$ and in this sum every monomial which appears in a summand appears in p.*

Proof. The proof is by induction on the number of cancelling monomials. We show that if p can be written as a sum with k cancelling monomials, then some number of steps can be performed for $\mathcal{B}$ to obtain $\mathcal{B}'$ such that p is now expressable as a sum with only $k - 1$ cancelling monomials.

Let m denote some monomial which cancels. Consider the subsum of the sum yielding p above obtained by selecting only the binomials which contain m. Because

we are working over a field, this subsum may be expressed

$$\sum_{k=1}^{T} \beta_k (m - t_k) \ ,$$

where β_k are nonzero field coefficients and t_k is some term, by factoring out the coefficient on m from each binomial in the subsum. Since m cancels in this sum by assumption, $\sum \beta_b = 0$. Now, we claim that the subsum can be rewritten

$$\sum_{k=2}^{T} \beta_k (t_1 - t_k) \ .$$

This is because

$$\begin{aligned}
\sum_{k=2}^{T} \beta_k (t_1 - t_k) &= \left(\sum_{k=2}^{T} \beta_k \right) t_1 - \sum_{k=2}^{T} \beta_k t_k \\
&= -\beta_1 t_1 - \sum_{k=2}^{T} \beta_k t_k \\
&= -\sum_{k=1}^{T} \beta_k t_k \\
&= \left(\sum_{k=1}^{T} \beta_k m \right) - \sum_{k=1}^{T} \beta_k t_k \\
&= \sum_{k=1}^{T} \beta_k (m - t_k)
\end{aligned}$$

Clearly, $t_1 - t_k$ is a linear combination of $\beta_1(m - t_1)$ and $\beta_k(m - t_k)$. Thus, if we derive $t_1 - t_k$ for each $2 \leq k \leq T$ and add it to $\mathcal{B}$, the subsum we considered can also be derived as a linear combination of the additional binomials, without containing the monomial m in any summand. Now, in the original sum for p, replace the subsum we considered with this new sum. We obtain a new sum that yields p, but which never mentions m, and we could not have introduced any new monomials since all we did was take linear combinations of the old summands. Thus p is a linear combination with one fewer noncancelling monomial. This completes the proof. □

Definition 4.65 (BGIP simulation). Let π' be a PC refutation of $\{-1, 1\}$-Tseitin. We define an induced / simulated binomial PC refutation π'' following the ideas of [39].

The simulation invariant is that whenever π' has a configuration $\mathcal{P}$, we will have a corresponding configuration in which every polynomial $p \in \mathcal{P}$ will be a no-cancellation linear combination of currently active binomials. That is, every polynomial p will be a linear combination of binomials from the simulating configuration, and no monomials will cancel in this sum. In the following, we will assume inductively that this is the case and fix such a linear combination for each p, and for each p the binomials in its linear combination will be termed the binomials *underlying* p. Note that this simulation *is* efficient, broadly speaking, but again this won't actually be important for our proof.

- When π' downloads an axiom, π'' downloads the same axiom, which is a binomial.
- When π' performs a weakening step on p to obtain result r, we perform the same weakening step for each underlying binomial. In the next configuration, these binomials underly r.
- When π' performs a linear combination step $p, q \vdash \alpha p + \beta q$ to obtain result r, we observe that r is now a linear combination of the binomials underlying p and q, with cancellation. By Lemma 4.64, r is also a linear combination without cancellation of binomials which may be inferred by linear combinations from the set of binomials underlying p, q, so π'' performs these linear combination inferences, and in the next configuration, these binomials underly r.
- When π' performs an erasure step, π'' erases any binomials which no longer underly any polynomial.

Observation 4.66. *If π' is a refutation, π'' is also a refutation.*

Proof. Suppose a configuration $\mathcal{P}$ of π' contains the contradition $1 = 0$. Then the corresponding binomial configuration contains $c = 0$ for some nonzero field element c – for suppose not, then there cannot be a no-cancellation linear combination of its binomials yielding 1. □

Claim 4.67. The BGIP simulation of PC on $\{+1, -1\}$-Tseitin by binomial PC is conservative with respect to monomials;

- If for some *configuration* of the simulated proof no active monomial contains the set of variables $\{y_e : e \in E'\}$ for some $E' \subseteq E$, then the corresponding configuration of the simulating proof does not contain any active monomials containing all of $\{y_e : e \in E'\}$.
- If for some *time period* in the simulated proof no active monomial contains the set of variables $\{y_e : e \in E'\}$ for some $E' \subseteq E$, then the corresponding time period of the simulating proof does not contain any active monomials containing all of $\{y_e : e \in E'\}$.

Proof. Download steps will obviously never pose a problem. A single weakening step for the PC proof may correspond to many weakening steps for the binomial proof, but no other varieties of steps occur, so it is straightforward to see that conservativity will hold here. Similarly, a single linear combination step for the PC proof introduces no new monomials, and is simulated by a series of linear combination steps in the binomial proof, with no other varieties of steps occuring. So inference steps will not pose a problem.

Finally consider erasure steps. If after an erasure step in the simulated proof there is a monomial in some binomial of the simulating proof which does not appear in the simulated proof, then this binomial cannot be underlying any polynomial of the simulated proof, since in any no-cancellation sum this monomial would remain. Thus by the end of the simulating erasures, every monomial in the simulating proof appears in the simulated proof, so conservativity holds. □

4.7.7 Time-Space Trade-off for PCR

As discussed in the overview, binomial PC is simple enough that we can obtain a strongly bounded complexity measure with the extended degree lower bound by generalizing the standard complexity measure (definition above).

Definition 4.68. Let $b = 0$ be a binomial PC proof line in the variables of $\{+1, -1\}$-Tseitin on graph $G = (V, E)$. Define the binomial complexity measure

$$\mu(b = 0) := \min_{S \subseteq V} |S| \,.$$
$$\forall e \in E, (x_e = 0 \wedge y_e = 1)$$
$$\vee (x_e = 1 \wedge y_e = -1)\},$$
$$\{PARITY_v\}_{v \in S} \models b = 0$$

Further, let S_b denote any fixed set S achieving the minimum above.

The first three parts of Observation 4.58 follow easily in this setting as well.

Observation 4.69. *μ is a subadditive complexity measure in the sense of Ben-Sasson & Wigderson [31]. That is,*

1. *For any axiom $b = 0$ of $\{+1, -1\}$-Tseitin, $\mu(b = 0) = 1$.*
2. *$\mu(\perp) = |V|$.*
3. *μ is subadditive.*

To obtain the last, we appeal to Lemma 4.56.

Corollary 4.70. *For any $b = 0$ binomial PC proof line in the variables of $\{+1, -1\}$-Tseitin,*

$$Vars(b = 0) \supseteq \delta(S_b) \,.$$

Proof. Suppose the two terms of the binomial b are t_1, t_2. Modulo the $\{+1, -1\}$ constraints for the y_e variables, $t_1 + t_2 = 0$ is logically equivalent to $t_1 t_2 = c$, for some field constant c, because these constraints imply that the square of any variable is one, so we may move all variables from t_2 to t_1 or vice versa by multiplying. If both terms are nonzero c will be nonzero, but if one of the terms is zero then c will be zero. However, the case that $c = 0$ is trivial, because already it implies that $b = 0$ contradicts the $\{+1, -1\}$ constraints. So in this case $\mu(b = 0) = 0$ and the claim is vacuous. In fact,

this argument shows that $c = \pm 1$. Additionally, we can reduce all terms modulo the squared powers, so that without loss $b = 0$ is of the form $m - c = 0$ for m a monomial with all variables of degree 1 and $c = \pm 1$.

We would like to rephrase the condition extension constraints $y_e = 1 - 2x_e, \{PARITY_v\}_{v \in S} \models m - c = 0$ so that we can apply Lemma 4.56 and be done. To do this, we reverse our "fourier transform" dictionary. As we saw before, products such as m correspond to sums under this transform, so modulo the extension variable constraints, $m = \pm c$ is the same as $\oplus_{e:y_e \in Vars(m)} x_e = c'$ for c' corresponding to c. We then drop the extension constraints since no other formulas remain in the y_e so the logical consequence follows without them, and so we have a system over $\mathbb{F}_2$ as desired. By Lemma 4.56, the transformed image of m contains the variables corresponding to the boundary, so by the definition of the transform, $Vars(m) \supseteq \{y_e : e \in \delta(S_b)\}$, as desired. □

The proof of the "extended degree" property is now exactly the same as the proof of the extended width property, Proposition 4.59.

Corollary 4.71. *If a graph $G = (V, E)$ satisfies extended isoperimetry with parameters W, t_0, and μ' is the strongly bounded complexity measure defined from the binomial complexity measure via*

$$\mu'(C) = \begin{cases} 0 & \mu(C) < t_0 \\ L & \mu(C) > |V|/2 \\ \log_2(\mu(C)/t_0) & \textit{otherwise} \end{cases}$$

then for any k clauses $C_1, \ldots C_k$ with distinct values under μ' between 0 and L,

$$\left| \bigcup Vars(C_i) \right| \geq \Omega(kW) \,. \tag{4.12}$$

Now we can prove the main result.

Proof of Theorem 4.4. Let π be any proof of $\tau_G[\oplus]$, in size $T = S(\pi)$ and monomial space $S = MSp(\pi)$.

Divide π into epochs recursively to a recursive depth of h, dividing each epoch into m equal subepochs, h, m to be determined later. Now consider the random restriction ρ. Since $\tau_G[\oplus]\restriction_\rho = \tau_G$ (up to replacing variables with their negations), $\pi\restriction_\rho$ is a refutation of τ_G. Let π' be the induced refutation of $\{+1, -1\}$-Tseitin on G, with induced recursive subdivision into epochs. Choose k such that $Lk^{-h} = k$. Since μ' is a strongly bounded complexity measure, Lemma 4.55 implies that the critical set of some induced epoch of π' contains at least k binomials of distinct complexity values, with probability one. Thus, by Corollary 4.71 this critical set contains $2k$ monomials which collectively contain at least $\Omega((k-2)W)$ variables. By conservativity of the simulations, the critical set of the inducing epoch in $\pi\restriction_\rho$ contains at least $2k$ monomials which collectively contain at least $\Omega((k-2)W)$ variables.

However, for M any small set of monomials, the probability that $M\restriction_\rho$ contains $2k$ monomials of which collectively contain many variables is seen to be small. For any monomial m, the semantically equivalent clause is killed (restricted to a constant) by ρ if and only if m is. Consider now a set of $2k$ monomials. They all survive ρ if and only if the disjunct of their corresponding clauses does, and this disjunct contains every variable that any one of themc ontains. Therefore by Lemma 4.28, the probability that any fixed $2k$-tuple of monomials all survive and have collectively more than X variables is at most exponenitally small in X. By a union bound over all $2k$-tuples of M, the probability that any $2k$ of M's monomials have them have collectively $\Omega((k-2)W)$ variables after the restriction is at most $|M|^{2k}2^{-\Omega((k-2)W)}$.

Choose the parameter m so that $mS = T/m^{h-1}$, so that all critical sets of epochs have the same size. Choose h so that $h = k$ and $k^h = L$. The probability that any critical set of any epoch contains k complexities after the restriction is thus at most $\left((mS)^2 2^{-\Omega(W)}\right)^k$, and there are at most $m^h = m^k$ epochs. By a union bound, the probability that any critical set of any epoch contains k complexities is at most $\left(m^3 S^2 2^{-\Omega(W)}\right)^k$. Since we know this must happen with certainty, this probability is at least one. We conclude $m \geq \left(2^{\Omega(W)}/S\right)$, or $T \geq \left(2^{\Omega(W)}/S\right)^{\Omega(k)}$. Here k is such that $k^k = L$, so in terms of our L our exponent exceeds $\log L/\log\log L$, as desired. $\square$

4.8 Concluding Remarks

In this paper, we report the first trade-off results for polynomial calculus and PCR which rule out simultaneous optimization of different proof complexity measures, in particular proof size and proof space. Loosely speaking, what our results say is that in the worst case, it is impossible to do any meaningful simultaneous optimization of size and space in polynomial calculus. PC and PCR are still not very well understood proof systems, however, and there remain several interesting open problems.

One such problem, which has seen exciting developments lately, is proving unconditional lower bounds on space in PC and PCR. It is only very recently that [34, 58] obtained lower bounds for k-CNF formulas, and these bounds all require $k \geq 4$. Intriguingly, there are still no nontrivial lower bounds for any family of 3-CNF formulas. Also, the space complexity of, for instance, Tseitin contraditions is open.

Another question is how far the analogies go between size, (monomial) space and degree in PC/PCR on the one hand and length, (clause) space and width in resolution on the other. In resolution, we know that clause space is an upper bound on width [10], that small width does not say anything about space complexity [29], and that there can be very strong trade-offs between these two measures [25]. In this paper, we have shown that exactly the same kind of trade-off holds between degree and monomial space in PC and PCR, but we do not know whether monomial space is an upper bound on degree or whether small degree says anything about the space complexity.

For our time-space trade-off results in sublinear space based on pebbling formulas, it would be very satisfying to remove the loss in the parameters resulting from having to take the logarithm of the proof size. This loss is inherent in the restriction argument, but for resolution it is known how to avoid restrictions completely and instead use the projection machinery in Section 4.3.7 together with the right kind of substitutions in the formulas to get tight trade-offs. It would be very interesting if something similar could be made to work for PC and PCR, since this would give tight trade-offs (for sublinear space) for these two proof systems and also yield new unconditional lower bounds on space similar to what is currently known for resolution.

Looking beyond polynomial calculus, another proof system that would be very interesting to understand is cutting planes. Here the open questions abound. Perhaps most obviously, it would be desirable to prove size lower bounds by some other technique than the interpolation used in [92], for instance, for Tsetin formulas or random k-CNF formulas.

As far as we are aware, there are no space lower bounds or "true" time-space trade-offs known for cutting planes. However, the recent results in [69] could be interpreted to suggest that pebbling formulas of the right flavour should inherit time-space trade-offs properties from the graphs in terms of which they are defined not only for the resolution proof system but also extending to cutting planes. If true, this would mean that the so-called black-white pebble game in [49] could be used to obtain strong trade-offs not only for resolution, k-DNF resolution and PC/PCR, but also for cutting planes.

Finally, it is known that PCR and cutting planes are both strictly stronger than resolution with respect to proof size, and it would seem natural to expect that PCR and cutting planes should both be stronger than resolution with respect to space as well. As far as we are aware, though, this is open. It would be nice to separate PCR from resolution with respect to space by finding a k-CNF formula that has low monomial space complexity in PCR but large clause space complexity in resolution, and similarly for cutting planes with respect to resolution.

5 Strong ETH holds in Regular Resolution

We obtain asymptotically sharper lower bounds on resolution complexity for k-CNF's than was known previously. We show that for any large enough k there are k-CNF's which require resolution width $(1 - O(k^{-1/4}))n$, regular resolution size $2^{(1-O(k^{-1/4}))n}$, and

general resolution size $(3/2)^{(1-\tilde{O}(k^{-1/4}))n}$.

The SAT problem is a canonical NP-complete problem. Non-trivial algorithms for SAT have ramifications both for the theory of computation and in applications such as hardware and protocol verification and planning. Despite a huge amount of attention from both theoretical and empirical perspectives, the exact difficulty of SAT remains somewhat mysterious. While quantitative improvements in SAT algorithms continue to be made, in many ways a wide variety of different algorithmic techniques have yielded similiar time bounds in a qualitative sense. The Exponential Time Hypothesis (ETH) and Strong Exponential Time Hypothesis (SETH) were introduced to give a precise meaning to the question of whether further improvements will be only quantitative, or substantially different [70]. These hypotheses have beeen shown to have other significant consequences in complexity, such as limits on the k-SUM problem from computational geometry, exponential algorithms for other NP-complete problems, limits to improving algorithms in parameterized complexity, and so on. Formally, ETH is the statement that, for $k \geq 3$, k-CNF SAT does not have algorithms running in time $2^{o(n)}$. Closely related is the strong ETH – that the savings possible for k-SAT goes to 0 as k goes to infinity, or, equivalently, that $k(n)$-SAT does not have deterministic algorithms in time $2^{(1-o(1))n}$ for any function $k(n) = \omega(1)$. That is, it is not even possible to get a constant polynomial advantage over brute force search that is indendent of k.

Both forms of the conjecture have a natural appeal, although there is admittedly little formal evidence for either. However, there are an increasing variety of interesting and non-trivial algorithms for SAT that seem to use unrelated algorithmic techniques ([88, 87, 67, 106, 71]), but all have roughly the same savings over exhaustive search : $\Theta(1/k)$ fractional savings over exhaustive search for k-SAT.

Empirically, it has also been observed that even tuned SAT solvers that solve 3-SAT formulas with millions of variables have difficulty with even small random k-SAT formulas for moderate k, such as 5 or 6 [104]. So SETH seems at least to be true for commonly used algorithms. Since we do not know how to show that problems in NP require even super-polynomial worst-case complexity, it seems that we are incredibly distant from any possible proof of ETH or SETH.

The challenge of lower bounds is to reason about arbitrary algorithms, including ones that are counter-intuitive. We might be able to confirm these hypotheses for at least some categories of algorithms that work in an intuitive fashion and are similar to those in use. Propositional proof complexity offers a general technique to do this. Many algorithmic techniques, when run on an unsatisfiable instance, implicitly define a "proof" that no solution exists, that can be formalized in a corresponding proof system. Then size lower bounds on the minimum proofs for a tautology in the proof system provide a lower bound on the time required by any algorithm in the family on the negation of the tautology. Using this method, ETH has been established for any algorithm that can be formalized within the resolution system, which includes many of the most successful empirical SAT-solving techniques. More precisely, lower bounds of the form $2^{\Omega(n)}$ are known for many natural proof systems like Resolution and Polynomial Calculus [113, 72]. which correspond naturally to important families of SAT algorithms.(Weakly exponential lower bounds 2^{n^ϵ} are also known for more exotic proof systems.) So we at least know that to negate ETH, we need to go beyond some of the standard algorithm design methods.

This raises the question of whether we can also get results establishing Strong ETH for similar classes of algorithms. A first result along these lines was by Pudlák and Impagliazzo [95], who showed that tree-like resolution requires size $2^{(1-\epsilon)n}$ for any ϵ, for k-CNFs of size cn where c, k are functions of ϵ. Here, we get a similar lower bound for regular resolution, a sub-system of resolution that is strictly more powerful than tree-like resolution that formalizes algorithms using the Davis-Putnam procedure [53]. Specifically, we show that there are k-CNF formulas which require regular resolution

proofs of size $2^{(1-O(k^{-1/4}))n}$. In particular, we get a somewhat improved and simplified version of the Pudlák-Impagliazzo lower bound. While we have not been able to show the same for general resolution, we do get a substantially improved exponential lower bound for general resolution, which approaches $(3/2)^n$ as k grows. An interesting interpretation is that this class of algorithms are now provably slower than Grover's quantum SAT algorithm [61].

The exact complexity of SAT has taken on an even greater significance in theoretical computer science due to the recent results of Williams [114], that show that even minute savings for circuit SAT can be used to prove circuit lower bounds. While this holds for general circuit SAT rather than k-SAT, we can often relate SAT problems for different classes of circuits. For example, if the AC^0 SAT algorithm of Impagliazzo, Matthews and Paturi [71] were to be substantially improved, it can be proved using Williams' results that $NEXP \not\subseteq NC^1$.

5.1 Techniques

Resolution has been intensively studied at least since the work of Davis and Putnam [53] in the early sixties. Despite its apparent simplicity, no exponential lower bounds were known until Haken's result [62] in 1985. Today there remain only a small number of techniques to give lower bounds in Resolution – Random Restrictions [62, 18], the Size-Width Tradeoffs [24], and the Pseudowidth technique which originally appeared in work of Raz [97] and was further developed by Razborov [99, 100].

One of the fundamental building blocks of previous lower bounds research has been resolution width lower bounds for systems of $\mathbb{F}_2$-linear equations; whether studied in the form of Tseitin tautologies, or random k-XOR CSPs, this result as appears in [24, 36] is essential to a great deal of subsequent work, in polynomial calculus [19], Lasserre hierarchy lower bounds [105, 60], and other results. Generally speaking, expanding systems of $\mathbb{F}_2$ linear equations require resolution width $\Omega(n)$. However, the width to refute these is not $(1-\epsilon)n$ as one might hope (for proving lower bounds), but rather there is always an upper bound of $n/2 + o(n)$. Ben-Sasson and Impagliazzo [19] gave

a probabilistic construction of a width $n/2$, size $2^{n/2}$ refutation of such a system, based on adding the $\mathbb{F}_2$ equations in a random order to obtain $1 = 0$ and simulating the linear algebra proof in resolution. They showed that with high probability, no intermediate equation has more than $n/2 + o(n)$ variables, due to random cancellations in the sum, thus the simulation results in a resolution proof of the claimed parameters.

Impagliazzo and Pudlák's result is also for a family of $\mathbb{F}_2$ linear equations, so to obtain size lower bounds of $2^{(1-\epsilon)n}$ they had to do significant work to overcome the fact that the width lower bounds are only $n/2$. They analyzed proof size via a Prover Adversary game which they introduced, and their technique works by considering not just the widest clause in the proof, but also for a series of subsequent smaller clauses which occur in the proof. They are able to show that the total combined width of all such clauses encountered approaches n, and their technique crucially exploits to obtain their lower bound.

In this paper, we consider equations over $\mathbb{F}_p$ rather than $\mathbb{F}_2$. When we add random linear combinations of equations over $\mathbb{F}_p$, a variable cancels with probability only $1/p$ rather than $1/2$, so the argument which gives width upper bound of $n/2$ for $\mathbb{F}_2$ in this case can only yield $(1 - 1/p)n$. Thus it is natural to guess that as p gets large the true width will approach n. One apparent drawback of this is that encoding $\mathbb{F}_p$ equations as boolean CSPs can result in significant complications, and it seems inevitable that one will have to think hard about partially constrained $\mathbb{F}_p$ linear systems and technical results from additive combinatorics. By a judicious choice of encoding scheme, we manage to avoid this and obtain an unexpectedly simple proof.

The width lower bound $(1-\epsilon)n$ which we thus obtain immediately implies the tree-like size lower bounds which we desire. To extend this to DAG-like proof systems, a natural idea is to employ a generalization of the Impagliazzo Pudlák prover adversary game [95] which was developed in [14], there with the goal of sharp time space tradeoffs in Regular Resolution. In this argument, a probabilistic adversary interacts with the proof, inducing a distribution of random paths through the proof DAG. A counting argument based on this can be used to obtain size lower bounds, in a manner similar

to the bottleneck counting argument first introduced by Haken [62]. In the full result of [14], this adversary is replaced with a random restriction argument in order to obtain results for General Resolution. In this work, we succeeded in adapting the techniques there to obtain sharp size lower bounds in regular resolution, but also we managed to dramatically simplify it in this context.

Finally, we introduce new random restriction techniques which are useful to get stronger lower bounds in general resolution. Rather than adhering to the usual paradigm of using random restrictions to kill wide clauses, we examine the width lower bound more carefully to give a tighter analysis.

In the next section we define the relevant proof systems and the preliminaries which we will need. In section (3), we prove the width lower bound, which implies the tree-like size bound. In section (4), we give an overview of the techniques for the main result, and together with a lemma from section (3) deduce the size lower bound for resolution.

5.2 Preliminaries

5.2.1 Basic Definitions

We consider Boolean formulas over a set of variables $\{x_1, \ldots, x_n\}$. As usual, a literal is a Boolean variable or its negation, a clause is a disjunction of literals, and a CNF is a conjunction of clauses. We think of clauses as being specified by their sets of literals, and CNFs as specified by their sets of clauses. For a clause C, we write $Vars(C)$ to be the set of variables appearing in C. The *width* $w(C)$ of a clause C is $|Vars(C)|$ and the width of a set or sequence of clauses F, is the maximum width of clauses in F. The *size* of a CNF formula F is the total number of literal occurrences in the formula, i.e., $\sum_{C \in F} w(C)$.

One of the simplest and most widely studied propositional proof systems is resolution which operates with clauses and has one rule of inference, the resolution rule: $\frac{A \vee x \qquad B \vee \overline{x}}{A \vee B}$. We say that the variable x is *resolved* in this instance of the resolution rule. A *resolution refutation* of a CNF formula (a set of clauses) is a sequence of clauses

ending in the unsatisfiable empty clause $\perp$, each of which is either a clause of from the formula (an "axiom") or follows from two previous clauses via the resolution rule. (The term *resolution proof* is used more generally to refer to any inference of this sort that may not necessarily result in $\perp$.) Every resolution proof naturally corresponds to a directed acyclic graph (DAG), known as the proof DAG, in which every clause derived via the resolution inference rule has a directed edge between a derived clause and each of its antecedents, oriented to show dependence. (Note that, formally, a resolution proof corresponds to one of possibly many topological sorts of its proof DAG.)

The *size* or *length* of a resolution proof is the total number of clauses in the proof.

A resolution proof is *tree-like* if its proof DAG has the structure of a tree. A resolution proof is *regular* if along each path in the proof DAG, each variable is resolved at most once. The unrestricted model is often called *general* resolution for contrast with regular and tree-like resolution.

It is easy to see that a tree-like proof of minimum size is regular without loss of generality.

Clauses are permitted to appear multiple times in a resolution proof; in general resolution this is unnecessary when only proof size is a concern, but in restricted forms this becomes important.

Resolution is *sound and complete* in that every CNF formula is unsatisfiable if and only if it has a (tree-like) resolution refutation.

A *restriction* is a partial assignment of truth values to variables of a formula, resulting in some simplification. Formally a restriction is a mapping $\rho : X \to \{0, 1, \star\}$. Restrictions on X can be identified with partial assignments on X by viewing unassigned inputs as being mapped to $\star$ and vice versa.

The restriction of a clause C by ρ, denoted by $C|_\rho$ is the clause obtained from C by setting the value of each $x \in \rho^{-1}(\{0, 1\})$ to $\rho(x)$, and leaving each $x \in \rho^{-1}(\star)$ as a variable. The restriction of a set of clauses is defined by restricting each one. The restriction of a resolution refutation of a CNF is a refutation of its restriction.

5.3 Finite Field Expanding Matrices

We will need the following variation on the concept of edge expansion, isoperimetry, etc.

Definition 5.1. Say that a matrix A over a field $\mathbb{F}_p$ is an $\mathbb{F}_p$ expander with parameters (r_1, r_2, c) if, for every column vector v of support size $r_1 \leq |v| \leq r_2$, the support of Av satisfies $c \leq |Av|$.

When A is the incidence matrix of a graph, thought of over $\mathbb{F}_2$, and v is the characteristic vector of a set of vertices, it is easy to see that Av indicates the boundary edges, so $\mathbb{F}_p$ expansion generalizes the familiar notion from graph theory. On the other hand, whereas for graphs with m vertices and n edges, we can only reasonably hope that balanced cuts will contain say half of the edges, from $\mathbb{F}_p$ expanders we can hope for more.

In particular for a random sparse $n \times n$ $\mathbb{F}_p$-matrix, and a small constant δ, we could reasonably expect to obtain a $(\delta n, 2\delta n, (1 - 1/p)n)$-expander. As a rough heuristic in support of this, the function computed by this matrix will look like a random map, at least on input vectors of relatively large hamming weight. At the same time, there are relatively few vectors in F_p^n of hamming weight between δn and $2\delta n$. Instead, the typical vectors have weight $(1 - 1/p)n$. Thus the chance that every vector of weight between $\delta n, 2\delta n$ maps to such a vector is high.

For our purposes it is not important to get a deterministic construction of $\mathbb{F}_p$ expanders, so we prove only existence by a probabilistic argument.

Lemma 5.2. *Fix any integers d, p, $d > 100$, p at least a suitably large multiple of $\sqrt{d}$. For any large enough n, let $\gamma = n/\sqrt{d}$. There is an $n + 1 \times n$ matrix over $\mathbb{F}_p$ such that*

- *Each row is supported on exactly d entries.*
- *The matrix is an $(\gamma n, 3\gamma n, (1 - 4/p)(1 - 1/\sqrt{d})n)$-$\mathbb{F}_p$ expander.*
- *No linear combination of fewer than $3\gamma n$ rows is the zero vector.*

Proof. Let B denote a random $\{0,1\}$-valued matrix of dimensions $n+1 \times n$, in which rows are independently chosen from the uniform distribution on vectors of support d, and let A denote the random $\mathbb{F}_p$ matrix in which nonzero $\mathbb{F}_p$ values are substituted for the ones of B independently.

First we show that with high probability over B, any set S of rows of size $n/\sqrt{d}$ has ones in at least $(1 - O(1/\sqrt{d}))n$ columns. For any column, the chance that it is missed is at most $(1 - d/n)^{|S|} \leq \exp(-d|S|/n)$. The chance that any set of δn columns are missed is therefore at most $\exp(-d\delta|S| + H(\delta)n)$, so for $\delta = \frac{1}{\sqrt{d}}$, the chance that any set S does not expand so much is $\ll 2^{-n}$. We conclude that with high probability every set of at least $n/\sqrt{d}$ rows has ones in at least $(1 - 1/\sqrt{d})n$ columns.

Now we show that with high probability over A, the result is $(\gamma n, 3\gamma n, (1-4/p)(1-1/\sqrt{d})n)$-$\mathbb{F}_p$ expanding. There are at most $(3ed^{1/2})^{n/\sqrt{d}} \cdot (p-1)^{3n/\sqrt{d}}$ vectors v which we must be concerned about. For each one, there are at least $(1 - \frac{1}{\sqrt{d}})n$ independent coordinates of Av which take nonzero values with probability at least $1 - \frac{1}{p-1}$. The probability that any $4/p$ fraction of these takes value zero is at most $\left(\frac{1}{p-1}\right)^{(4/p)(1-\frac{1}{\sqrt{d}})n} \cdot (ed^{1/2})^{n/\sqrt{d}} \cdot (p-1)^{3n/\sqrt{d}}$.

Since p was chosen to be at least a large enough multiple of $\sqrt{d}$, we conclude that with high probability A is $\mathbb{F}_p$ expanding as desired.

By a relatively standard calculation it can be shown that sets of rows of size $\leq n/\sqrt{d}$ expand by a factor $\Omega(\sqrt{d})$ at least in B, and that in this case, the probability over A that any vector supported on such a small set does not have image equal to zero is very small, using arguments similar to the above. If vectors supported on fewer than $n/\sqrt{d}$ positions don't have image zero, and $\mathbb{F}_p$ expansion holds between that value and $3n/\sqrt{d}$, then no vector of support $\leq 3n/\sqrt{d}$ has image zero, so no subset of $3\gamma n$ rows has the zero vector as a nontrivial linear combination. □

5.4 Width Bound

We will be interested in defining unsatisfiable linear systems usin $\mathbb{F}_p$ expanders.

To obtain width lower bounds, we will give an analysis based on the *semantic mea-*

sure of proof lines, following the technique standardized by Ben-Sasson and Wigderson [24].

Definition 5.3. The *semantic measure* of a proposition P with respect to a (refutation of) a conjunction $\bigwedge_{i=1}^{m} A_i$ is

$$\mu(P) := \min_{S \subseteq [m] : \bigwedge_{i \in S} A_i \models P} |S| \ ,$$

that is, the size of the minimal subset of the axioms A_i which semantically implies S.

Observation 5.4. *(Ben-Sasson Wigderson '01)*

- *For any axiom clause C of ϕ, $\mu(C) = 1$.*
- *For any inference $C_1, C_2 \vdash C_3$, $\mu(C_1) + \mu(C_2) \geq \mu(C_3)$.*
- *For any minimally unsatisfiable conjunction $\bigwedge_{i=1}^{m} A_i$, $\mu(\bot) = m$.*

As an immediate corollary of these observations, in any refutation, in any proof system, of any minimally unsatisfiable set of m constraints, there must exist a proof line C such that $\mu(C) \in [m/3, 2m/3]$.

It is easy to see that for any matrix A as in the corollary, we can choose a vector $\vec{b}$ such that $A\vec{y} = \vec{b}$ is not satisfiable. Such a system will be *close* to minimally unsatisfiable, since if any subset S of the equations is contradictory, there must be a linear combination of them which produces $0 = 1$, and we argued before that this does not happen when $|S| \leq 3\gamma n$.

Claim 5.5. There are unsatisfiable linear systems $A\vec{y} = \vec{b}$ consisting of $n + 1$ $\mathbb{F}_p$-equations on n variables, each containing $O(p^2)$ variables, in which A is an $(\gamma n, 3\gamma n, (1 - 4/p)n)$-$\mathbb{F}_p$ expander, and no subset of $\leq 3\gamma n$ equations is contradictory, where $\gamma = \Theta(1/p)$.

To express an $\mathbb{F}_p$ expanding system as a CNF, we encode variables as follows. For each $\mathbb{F}_p$ variable y_i we will have p^2 boolean variables x_{ij}, thought of as taking values

$\{0,1\}$, with the intended meaning that $y_i = \sum_j x_{ij} \mod p$. Thus, y_i does not determine the x_{ij}, and if a partial assignment π to the x_{ij} assigns only $p^2 - p$ variables, y_i is still completely unconstrained.

Definition 5.6. For $A\vec{y} = \vec{b}$ an $\mathbb{F}_p$ linear system over variables $y_1, \ldots, y_n$, let CNF ϕ denote the following conjunction in variables $x_{ij}, 1 \leq i \leq n, 1 \leq j \leq p^2$:

$$\bigwedge_{k=1}^{m} \left\{ \sum_i A_{k,i} \sum_j x_{ij} = b_k \pmod p \right\} .$$

Naturally we replace each equation above with its trivial CNF representation, which, if each equation in the linear system has fanin ℓ has only ℓp^2 variables.

Theorem 5.7. *Let ϕ be a CNF corresponding to a linear system as in Claim 5.5 via Definition 5.6. Then any resolution refutation of ϕ requires width $(1 - 5/p)np^2$, and ϕ is an $O(p^4)$-CNF.*

Proof. Let C be any clause containing fewer than $(1 - 5/p)$ of the variables. We show that if C has semantic complexity between $3\gamma/2n$ and $3\gamma n$, we contradict $\mathbb{F}_p$-expansion.

By a markov argument, there are at least $4/p$ of the y_i variables such that at least $1/p$ of their $x_{i,j}$ are unassigned. Let ρ denote the restriction corresponding to $\neg C$.

Say that a y_i variable is free if at least p of its $x_{i,j}$ variables are unassigned by ρ, and let ρ^* denote any extension of ρ whose domain is the domain of ρ, as well as all $x_{i,j}$ variables for non-free y_i's.

Then in terms of the y_i variables, ρ^* corresponds semantically to a restriction which assigns all non-free variables, and leaves the free variables unset.

Thus, a subset $\{A_i\}_{i \in S}$ of the equations semantically implies C if and only if for every such ρ^*, $\{A_i|_{\rho^*}\}_{i \in S}$ implies a contradiction, and it minimally implies C if and only if every A_i is needed for some ρ^*.

An $\mathbb{F}$-linear system is unsatisfiable if and only if there is an $\mathbb{F}$-linear combination in the equations which gives $1 = 0$. If for some equation $E|_{\rho^*} = (1 = 0)$, then E only contains the variables assigned by ρ^*. That is, for each ρ^*, there exists a linear

combination of the axioms $\{A_i\}_{i \in S}$ supported only on the non-free variables of ρ, and for every $i \in S$, by minimality, some ρ^*'s combination has a nonzero coefficient for A_i.

Therefore, take a random linear combination of the equations corresponding to each ρ^*. Then the resulting equation is supported again on only the non-free variables of ρ. We show that this equation is a nontrivial linear combination of many of the A_i, contradicting $\mathbb{F}_p$-expansion. Each A_i occurs in some equation, thus, in the resulting random linear combination, its coefficient is distributed uniformly over $\mathbb{F}_p$. By averaging, there exists such a combination which results in at least $(1 - 1/p)$ of the A_i having nonzero coefficients. Thus there is a linear combination supported on between $(1 - 1/p)3/2\gamma n \geq \gamma n$ and $3\gamma n$ equations of A with fewer than $(1 - 4/p)$ of the y_i variables, contradicting $\mathbb{F}_p$ expansion as desired.

□

Since width is at most the base two log of Tree-like Size [24], this immediately implies $2^{(1-o(1))n}$ lower bounds for tree-like resolution, as mentioned previously.

5.5 Regular Resolution

To obtain size lower bounds for Regular Resolution, we adapt and simplify the probabilistic adversary technique introduced in [14]. At a high level, this is a variation on the bottleneck counting argument introduced by Haken [62]. In this argument, a rule is given which maps assignments to particular clauses in the proof, at which significant "work" is done thinking about this assignment. The task is to show that we can map a large number of assignments in such a way that only a small number map to any particular clause, which implies that there are many clauses. In Haken's work this map is described explicitly – in the more modern form of the argument, due to Beame and Pitassi, a random restriction argument is used to hide these details. The bottleneck counting argument is of fundamental importance in computational complexity theory; besides underlying much of modern proof complexity, the bottleneck counting approach was also employed by Cook and Haken for monotone circuit lower bounds [63], and a report of Simon and Tsai [108] illustrated how closely related it is with the method

of approximations used in other contexts. The high level plan is executed in part using ideas from [95].

Theorem 5.8. *Any regular resolution refutation of ϕ as defined in Theorem 5.7 has size at least $2^{(1-5/p)np^2}$.*

Proof. We define a probabilistic process, which one may think of as an adversary in the sense of [95], which interacts with the proof, and which we think of as taking place at a particular clause in the proof at every step. The process begins at the final clause, $\perp$, and in each step moves to one of the two parents of the current clause, until at some point it stops, at some clause somewhere in the middle of the proof. The path which is followed by the process depends on what the current and parent clauses look like, what the history of the process is, and some random coins. We will show that over the random coins of the process, the probability that it stops at any particular clause of the proof is extremely small – from this we will deduce that there are many clauses.

We will think of the process as building up a truth assignment by assigning one variable at a time – in the step corresponding to a clause C, if C is deduced by resolving on variable x, the process will either assign $x = 0$ and move to the clause containing the literal x, or assign $x = 1$ and move to the caluse containing the literal $\overline{x}$. Thus if π denotes the partial assignment corresponding to all previous assignments made by the process, it always maintains the invariant that π falsifies the current clause. Crucially, by regularity, the variable x is always unassigned by π.

The rule by which we will assign variable x at each step is as follows:

- If variable x corresponds to a *free* $\mathbb{F}_p$ variable of ϕ (at least $p+1$ of its boolean variables are unset by π), then x is assigned randomly
- Otherwise, we choose x so as to maximize the semantic complexity of the clause for the next round.

The crucial claim regarding this process is that in each step, the semantic complexity of the occupied clause cannot decrease by more than a factor of two. Let us prove this.

In the second case above, this is easy to see, because it is a standard application of subadditivity of the semantic measure. In the first case, we claim that the semantic measure cannot decrease at all – this is because if x corresponds to a free variable, then the two parent clauses each semantically entail one of $C \cup \{x = 0\}, C \cup \{x = 1\}$, but these clauses are semantically implied by a set S of equations if and only if the corresponding restrictions of S are unsatisfiable, and since x is a free variable, the corresponding restriction in terms of the $\mathbb{F}_p$ variables is the same for $C, C \cup \{x = 0\}$, and $C \cup \{x = 1\}$. Thus all three of these clauses are of the same semantic complexity, and the two parent clauses are each of at least this complexity.

Since at the beginning, the contradiction clause $\perp$ at which the process begins has semantic complexity $\geq 3\gamma n$, at the end of any path, any axiom has semantic complexity 1, and in any step the measure at most halves, at some point in the process we must walk to a clause of semantic complexity between $2\gamma n$ and γn. The first time that this happens, the process is defined to stop at this clause.

What is the probability that the process stops at any particular clause C? Since π must falsify C by the time we walk to C, the process can only walk to C if it assigns all the variables which appear in C consistently with $\neg C$. By $\mathbb{F}_p$ expansion and the width argument from before, C can only have semantic complexity between γn and $2\gamma n$ if at least $(1 - 4/p)n$ of the $\mathbb{F}_p$ variables are non-free according to C – thus C assigns at least $(1 - 1/p)$ of the bits corresponding to these variables to particular values. All of these bits must be queried along any path which reaches C, and for each $\mathbb{F}_p$ variable, for the first $(1 - 1/p)$ of these bits, the process will respond randomly. In the event that it assigns some bit incorrectly, it can never reach C, and for each such queried bit, this happens with probability $1/2$ at least.

Claim 5.9. For any path from the root σ, and any clause C, say that the number of freedoms of a $\mathbb{F}_p$ variable with respect to σ, C is the number of bits corresponding to this variable which appear in C and not $\operatorname{dom} \sigma$, minus p, and is zero if this value is less than zero. The probability that the adversary, having followed σ, reaches C is at most two the minus to the total number of freedoms for all $\mathbb{F}_p$ variables.

Proof. By induction on σ. The base case is that σ is large and has zero freedoms, in which case the probability bound is trivial. Let σ be any path in the proof, leading to a clause which is deduced by resolving on variable x. In case that assigning x does not reduce the number of freedoms, by inductive hypothesis applied to the paths $\sigma \cup \{x = 0\}$ and $\sigma \cup \{x = 1\}$ the probability bound we want holds at both of these, and by averaging it holds for σ. In case that assigning x does reduce the number of freedoms, the bound obtained inductively is only a factor two worse than what we claim at σ. Since assigning x does reduce the number of freedoms, its $\mathbb{F}_p$ variable is still free so x is assigned randomly by the adversary. If the adversary assigns x to satisfy C, then he can never reach C, so for one of $\sigma \cup \{x = 0\}, \sigma \cup \{x = 1\}$ the probability to reach C is zero. We conclude that the probability that the adversary reaches C from σ is two the minus the total number of freedoms as claimed. □

Now apply the claim in case that σ is the empty path which starts and ends at $\perp$, and with C any possible stopping point. Since the total number of freedoms initially is at least $(1 - 4/p)(1 - 1/p) \geq (1 - 5/p)$, by our bound on the total number of freedoms, there is at most a $2^{-(1-5/p)np^2}$ probability that the adversary stops at any particular clause C. Since the process always stops at some clause, this implies that there are at least $2^{(1-5/p)np^2}$ clauses in any refutation of the tautology ϕ, which is on np^2 variables. □

5.6 General Resolution

In this section, we give size lower bounds in General Resolution, using the analysis of the width lower bound in the first section, together with a quantitatively improved random restriction argument.

We can abstract the argument as follows.

Definition 5.10. For $\mathcal{C}$ a set of constraints in n boolean variables $x_1, \ldots, x_n$, and $f : \{0,1\}^m \to \{0,1\}^n$ a boolean function $f : \vec{y} \mapsto \vec{x}$, we define the f-substituted set of constraints $\mathcal{C}[f]$ in variables $y_1, \ldots, y_m$ as the set $\{C(f(\vec{y})) : C \in \mathcal{C}\}$.

When f is n parallel copies of the parity function on ℓ bits, we denote this more succinctly by $\mathcal{C}[\oplus^\ell]$.

The idea of $\oplus$-substitution has been used in many contexts. Intuitively, this kind of indirection makes things harder for resolution in part because parity constraints are hard to express efficiently in CNF.

Definition 5.11. Let C be a clause in the variables $y_1, \ldots, y_m$ of $\mathcal{C}[f]$. Let $C = \bigvee C_i$ where C_i is the part of C_i in the variables corresponding to x_i. Define the *projection* C_i' of C_i by

$$C_i' := \begin{cases} x_i & C_i \models (f(\vec{y}))_i = 1 \\ \overline{x_i} & C_i \models (f(\vec{y}))_i = 0 \\ 1 & \text{otherwise} \end{cases}.$$

Define the projection C' of C by $\bigvee C_i'$.

Our next observation explains how to determine the semantic complexity of clauses in substituted formulas in terms of the original formula.

Observation 5.12. *The semantic complexity of C with respect to $\mathcal{C}[\oplus^\ell]$ is the same as the semantic complexity of C' with respect to $\mathcal{C}$.*

Proof. It is easy to see that for any assignment to $y_1, \ldots, y_m$ satisfying C, its image under f satisfies C'. By the definition of projection, it is also true that any assignment to $x_1, \ldots, x_n$ which satisfies $\mathcal{C}'$ may be lifted to an assignment satisfying C. □

Our next lemma uses this to bound the probability that the projection of a restricted clause is wide.

Lemma 5.13. *Fix any $\ell \geq 2$, and CNF ϕ in variables $x_1, \ldots, x_n$. Let ρ denote an iid random restriction which sets the variables of $\phi[\oplus^\ell]$ to $0, 1, \star$ with equal probability, conditioned on never setting all y variables associated to any x_i to a constant. For any clause C in the y variables, the probability that the projection $C|_\rho'$ has width at least $(1-\epsilon)n$ is at most $\left(\frac{2}{3}\right)^{(1-\epsilon)n\ell - O((2/3)^\ell)n - O(H(\epsilon))n}$, where H is the binary entropy function.*

Proof. Fix an arbitrary clause C and consider the width of the projection of the restricted clause, $(C|_\rho)'$. We first show that this is small with high probability. We can analyze this by considering each variable one at a time. Let $(C|_\rho)'_i$ denote the portion of $C|_\rho$ associated to x_i, as in the definition of projection.

$$(C|_\rho)' = \bigvee_i (C|_\rho)'_i \,.$$

Traditionally, restrictions are only thought to kill clauses by setting variables of the clause to true. Thanks to the definition of projection, we can also think of killing variables of a clause by setting its *non-variables* to $\star$, since in this case the projection of that portion of the restricted clause will be trivial.

Claim 5.14. For any C,

$$\Pr_\rho[(C|_\rho)'_i \neq 1] \leq (2/3)^\ell \cdot (1-(2/3)^\ell)^{-1} \,.$$

Proof. For each variable in C_i, if ρ sets it oppossite to its value in C_i, then $C_i|_\rho = 1$, and $(C|_\rho)'_i = 1$. For each variable y_j associated to x_i but not appearing in C_i, if it set to $\star$, then any satisfying assignment to $C_i|_\rho$ may be flipped on this variable, still satisfying $C_i|_\rho$ but having opposite parity, thus $C_i|_\rho \not\models \oplus y_j = 1, C_i|_\rho \not\models \oplus y_j = 0$, hence $(C|_\rho)'_i = 1$. Thus if we ignore the conditioning, the probability that $(C|_\rho)'_i \neq 1$ is at most $(2/3)^\ell$. The event which we condition away has probability at most $(2/3)^\ell$, hence doing so can only increase probabilities by a fraction $(1-(2/3)^\ell)$. □

The width of $C|'_\rho$ is the sum of the widths of $(C|_\rho)'_i$, and ρ acts independently on each C_i, so by a union bound over all subsets of $(1-\epsilon)n$ of the variables $x_1, \ldots, x_n$,

$$\Pr_\rho[\mathit{Vars}\big(C|'_\rho\big) \geq (1-\epsilon)n] \leq$$

$$\binom{n}{(1-\epsilon)n} \left((2/3)^\ell (1-(2/3)^\ell)^{-1}\right)^{(1-\epsilon)n} \,.$$

Using the identity $-\ln(1-x) \leq x + x^2$ for $x < 1/2$,

$$\leq (2/3)^{(1-\epsilon)n\ell - O((2/3)^\ell)n - O(H(\epsilon))n},$$

where H is the binary entropy function. This completes the proof of Lemma 5.13. □

This can be used to obtain size lower bounds. If ϕ is such that any clause of intermediate semantic complexity is wide, then the above shows that it is very unlikely that a restriction of a clause in a proof of $\phi[\oplus^\ell]$ has intermediate semantic complexity.

Corollary 5.15. *For any small enough $\epsilon > 0$, there exist $O\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon}\right)$-CNF formulas on n variables which have resolution complexity at least $\left(\frac{3}{2}\right)^{(1-\epsilon)n}$.*

Proof. First we choose ℓ to simplify the bound above. Take $\ell = O(\log\frac{1}{\epsilon})$, and using the fact that $\lim_{\epsilon\to 0}\frac{H(\epsilon)}{\epsilon\log\frac{1}{\epsilon}} = O(1)$, observe that with this choice of ℓ, for small enough ϵ, $\epsilon + \frac{1}{\ell}\left((2/3)^\ell + O(H(\epsilon))\right) = O(\epsilon)$, so the above lemma implies a probability bound of $\left(\frac{2}{3}\right)^{(1-O(\epsilon))n}$.

Let $k = O(1/\epsilon^4)$, and apply the lemma above when ϕ is any k-CNF such that clauses of intermediate semantic complexity have width at least $(1-O(k^{-1/4}))n = (1-\epsilon)n$, as we obtained from Theorem 5.7. Then $\phi[\oplus^\ell]$ is a $k\ell$-CNF, which we will show has resolution complexity $\left(\frac{3}{2}\right)^{(1-\epsilon)n}$ Consider any hypothetical resolution refutation of size less than this, and apply random restriction ρ. By a union bound, for some such ρ every clause of the restricted proof has a projection of width less than $(1-\epsilon)n$, which implies none of the projections have intermediate semantic complexity.

Since ρ is conditioned never to set all variables to constants, it is always true that $\phi[\oplus^\ell]|_\rho$ is a substitution of ϕ. By (the proof of) Observation 5.12, this implies the restricted proof of $\phi[\oplus^\ell]|_\rho$ thus obtained has no clause of semantic complexity intermediate between γn and $2\gamma n$, contradicting subadditivity of the semantic measure. Since $k\ell = O\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon}\right)$ this shows $\phi[\oplus^\ell]$ satisfies the claim.

□

5.7 Concluding Remarks

We have demonstrated that there exist tautologies on n variables which require resolution width $(1-\epsilon)n$ and regular resolution proofs of size $2^{(1-\epsilon)n}$, for any $\epsilon > 0$. Moreover, these tautologies may be taken to be k-CNF's for $k = O(\frac{1}{\epsilon})^2$. In general resolution we obtain lower bounds of $(3/2)^{(1-\epsilon)n}$ for $k = O(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon})$.

A good question is how closely this can be made to match the performance of k-SAT algorithms like PPSZ. Can we get results for general resolution matching those we obtained in regular resolution? Can we find k-CNF's which require resolution width $(1-O(1/k))n$?

References

[1] M. Alekhnovich, A. Borodin, J. Buresh-Oppenheim, R. Impagliazzo, A. Magen, and T. Pitassi. Toward a model for backtracking and dynamic programming. *Computational Complexity*, pages 1–62, 2012.

[2] M. Alekhnovich, J. Johannsen, T. Pitassi, and A. Urquhart. An exponential separation between regular and general resolution. Technical Report TR01-56, Electronic Colloquium in Computation Complexity, `http://www.eccc.uni-trier.de/eccc/`, 2001.

[3] M. Alekhnovich and A. A. Razborov. Satisfiability, branch-width and tseitin tautologies. In *Proceedings 43nd Annual Symposium on Foundations of Computer Science*, pages 593–603, Vancouver, BC, November 2002. IEEE.

[4] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002. Preliminary version appeared in *STOC '00*.

[5] Michael Alekhnovich and Alexander A. Razborov. Satisfiability, branch-width and tseitin tautologies. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science (FOCS '02)*, pages 593–603, November 2002.

[6] Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. *Proceedings of the Steklov Institute of Mathematics*, 242:18–35, 2003. Available at `http://people.cs.uchicago.edu/~razborov/files/misha.pdf`. Preliminary version appeared in *FOCS '01*.

[7] Eric Allender, Shiteng Chen, Tiancheng Lou, Periklis Papakonstantinou, and Bangsheng Tang. Width-parameterized SAT: Time-space tradeoffs. Technical Report TR12-027, Electronic Colloquium on Computational Complexity (ECCC), March 2012.

[8] Eric Allender, Shiteng Chen, Tiancheng Lou, Periklis A Papakonstantinou, and Bangsheng Tang. Width-parameterized sat: Time-space tradeoffs. *Theory of Computing*, 10(12):297–339, 2014.

[9] A. Atserias and V. Dalmau. A combinatorial characterization of resolution width. In *Proceedings Eighteenth Annual IEEE Conference on Computational Complexity*, pages 239–247, Aarhus, Denmark, July 2003.

[10] Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *Journal of Computer and System Sciences*, 74(3):323–334, May 2008. Preliminary version appeared in *CCC '03*.

[11] Fahiem Bacchus, Shannon Dalmao, and Toniann Pitassi. Solving# sat and bayesian inference with backtracking search. *Journal of Artificial Intelligence Research*, 34:391–442, 2009.

[12] Roberto J. Bayardo Jr. and Robert Schrag. Using CSP look-back techniques to solve real-world SAT instances. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI '97)*, pages 203–208, July 1997.

[13] Paul Beame. Proof complexity. In Steven Rudich and Avi Wigderson, editors, *Computational Complexity Theory*, volume 10 of *IAS/Park City Mathematics Series*, pages 199–246. American Mathematical Society, 2004.

[14] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 213–232, May 2012.

[15] Paul Beame, Henry Kautz, and Ashish Sabharwal. Towards understanding and harnessing the potential of clause learning. *Journal of Artificial Intelligence Research*, 22:319–351, 2004.

[16] Paul Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS '96)*, pages 274–282, October 1996.

[17] Paul W. Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 794–806, Santa Fe, NM, November 1994. IEEE.

[18] Paul W. Beame and Toniann Pitassi. Simplified and improved resolution lower bounds. In *Proceedings 37th Annual Symposium on Foundations of Computer Science*, pages 274–282, Burlington, VT, October 1996. IEEE.

[19] E. Ben-Sasson and R. Impagliazzo. Random CNF's are hard for the polynomial calculus. In *Proceedings 40th Annual Symposium on Foundations of Computer Science*, pages 415–421, New York,NY, October 1999. IEEE.

[20] E. Ben-Sasson, R. Impagliazzo, and A. Wigderson. Near-optimal separation of treelike and general resolution. *Combinatorica*, 24(4):585–603, 2004.

[21] E. Ben-Sasson and J. Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proceedings 49th Annual Symposium on Foundations of Computer Science*, pages 709–718, Philadelphia,PA, October 2008. IEEE.

[22] E. Ben-Sasson and J. Nordström. Understanding space in resolution: Optimal lower bounds and exponential tradeoffs. Technical Report TR09-034, Electronic Colloquium in Computation Complexity, `http://www.eccc.uni-trier.de/eccc/`, 2009.

[23] E. Ben-Sasson and J. Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. Technical Report arXiv:1008.1789, arxiv, 2010.

[24] E. Ben-Sasson and A. Wigderson. Short proofs are narrow – resolution made simple. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 517–526, Atlanta, GA, May 1999.

[25] Eli Ben-Sasson. Size space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, May 2009. Preliminary version appeared in *STOC '02*.

[26] Eli Ben-Sasson and Nicola Galesi. Space complexity of random formulae in resolution. *Random Structures and Algorithms*, 23(1):92–109, August 2003. Preliminary version appeared in *CCC '01*.

[27] Eli Ben-Sasson and Russell Impagliazzo. Random CNF's are hard for the polynomial calculus. *Computational Complexity*, 19:501–519, 2010. Preliminary version appeared in *FOCS '99*.

[28] Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of treelike and general resolution. *Combinatorica*, 24(4):585–603, September 2004.

[29] Eli Ben-Sasson and Jakob Nordström. Short proofs may be spacious: An optimal separation of space and length in resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pages 709–718, October 2008.

[30] Eli Ben-Sasson and Jakob Nordström. Understanding space in proof complexity: Separations and trade-offs via substitutions. In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, pages 401–416, January 2011. Full-length version available at http://eccc.hpi-web.de/report/2010/125/.

[31] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48(2):149–169, March 2001. Preliminary version appeared in *STOC '99*.

[32] Armin Biere, Marijn J. H. Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, February 2009.

[33] Archie Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.

[34] Ilario Bonacina and Nicola Galesi. Pseudo-partitions, transversality and locality: A combinatorial characterization for the space measure in algebraic proof systems. Technical Report TR12-119, Electronic Colloquium on Computational Complexity (ECCC), September 2012.

[35] Michael Brickenstein and Alexander Dreyer. PolyBoRi: A framework for Gröbner-basis computations with Boolean polynomials. *Journal of Symbolic Computation*, 44(9):1326–1345, September 2009.

[36] S. Buss, D. Grigoriev, R. Impagliazzo, and T. Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. In *Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing*, pages 547–556, Atlanta, GA, May 1999.

[37] S. R. Buss. *Bounded Arithmetic*. Bibliopolis, Napoli, 1986. Volume 3 of Studies in Proof Theory.

[38] S. R. Buss. Polynomial-size Frege and resolution proofs of *st*-connectivity and Hex tautologies. *Theoretical Computer Science*, 357(1–3):35–52, 2006.

[39] Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, March 2001. Preliminary version appeared in *CCC '99*.

[40] David A. Carlson and John E. Savage. Graph pebbling with many free pebbles can be difficult. In *Proceedings of the 12th Annual ACM Symposium on Theory of Computing (STOC '80)*, pages 326–332, 1980.

[41] David A. Carlson and John E. Savage. Extreme time-space tradeoffs for graphs with small space requirements. *Information Processing Letters*, 14(5):223–227, 1982.

[42] M. Charikar, K. Makarychev, and Y. Makarychev. Integrality gaps for sherali-adams relaxations. In *Proceedings of the 41st annual ACM symposium on Theory of computing (STOC '09)*, pages 283–292, 2009.

[43] Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM*, 35(4):759–768, October 1988.

[44] Edmund Clarke, Armin Biere, Richard Raimi, and Yunshan Zhu. Bounded model checking using satisfiability solving. *Formal methods in system design*, 19(1):7–34, 2001.

[45] M. Clegg, J. Edmonds, and R. Impagliazzo. Using the Gröbner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, pages 174–183, Philadelphia, PA, May 1996.

[46] Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, pages 174–183, May 1996.

[47] Stephen A. Cook and Robert A. Reckhow. On the lengths of proofs in the propositional calculus. In *Conference Record of Sixth Annual ACM Symposium on Theory of Computing*, pages 135–148, Seattle, WA, April-May 1974.

[48] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44(1):36–50, 1977.

[49] Stephen A. Cook and Ravi Sethi. Storage requirements for deterministic polynomial time recognizable languages. *Journal of Computer and System Sciences*, 13(1):25–37, 1976.

[50] M. Davis. *Computability and Unsolvability*. McGraw-Hill, 1958.

[51] M. Davis, editor. *The Undecidable*. Raven Press, Hewlett, NY, 1965.

[52] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem proving. *Communications of the ACM*, 5:394–397, 1962.

[53] M. Davis and H. Putnam. A computing procedure for quantification theory. *Communications of the ACM*, 7:201–215, 1960.

[54] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem proving. *Communications of the ACM*, 5(7):394–397, July 1962.

[55] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.

[56] J. L. Esteban and J. Toran. Space bounds for resolution. In *(STACS) 99: 16th Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563 of *Lecture Notes in Computer Science*, pages 530–539, Trier, Germany, March 1999. Springer-Verlag.

[57] Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171(1):84–97, 2001. Preliminary versions of these results appeared in *STACS '99* and *CSL '99*.

[58] Yuval Filmus, Massimo Lauria, Jakob Nordström, Neil Thapen, and Noga Ron-Zewi. Space complexity in polynomial calculus. In *Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC '12)*, pages 334–344, June 2012.

[59] Z. Galil. On the complexity of regular resolution and the Davis-Putnam procedure. *Theoretical Computer Science*, 4:23–46, 1977.

[60] D. Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259:613–622, 2001.

[61] Lov K. Grover. A fast quantum mechanical algorithm for database search. In Gary L. Miller, editor, *STOC*, pages 212–219. ACM, 1996.

[62] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–305, 1985.

[63] A. Haken and S. A. Cook. An exponential lower bound for the size of monotone real circuits. *Journal of Computer and System Sciences*, 58:326–335, 1999.

[64] Armin Haken. The intractability of resolution. *Theoretical Computer Science*, 39(2-3):297–308, August 1985.

[65] Alexander Hertel. *Applications of Games to Propositional Proof Complexity*. PhD thesis, University of Toronto, May 2008. Available at http://www.cs.utoronto.ca/~ahertel/.

[66] P. Hertel and T. Pitassi. Exponential time/space speedups for resolution and the pspace-completeness of black-white pebbling. In *Proceedings 48th Annual Symposium on Foundations of Computer Science*, pages 137–149, Berkeley, CA, October 2007. IEEE.

[67] Timon Hertli. 3-sat faster and simpler - unique-sat bounds for ppsz hold in general. In Rafail Ostrovsky, editor, *FOCS*, pages 277–284. IEEE, 2011.

[68] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–562, 2006.

[69] Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, pages 233–248, May 2012.

[70] R. Impagliazzo and R. Paturi. On the complexity of k-SAT. *Journal of Computer and System Sciences*, 67:367–375, 2001.

[71] Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac^0. In Yuval Rabani, editor, *SODA*, pages 961–972. SIAM, 2012.

[72] Russell Impagliazzo, Pavel Pudlák, and Jiri Sgall. Lower bounds for the polynomial calculus and the Gröbner basis algorithm. *Computational Complexity*, 8(2):127–144, 1999.

[73] J. Krajíček. *Bounded Arithmetic, Propositional Logic and Complexity Theory*. Cambridge University Press, 1996.

[74] Thomas Lengauer and Robert Endre Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *Journal of the ACM*, 29(4):1087–1130, October 1982. Preliminary version appeared in *STOC '79*.

[75] João P. Marques-Silva and Karem A. Sakallah. Grasp – a new search algorithm for satisfiability. In *Proceedings of the International Conference on Computer-Aided Design*, pages 220–227, San Jose, CA, November 1996. ACM/IEEE.

[76] João P. Marques-Silva and Karem A. Sakallah. GRASP—a new search algorithm for satisfiability. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '96)*, pages 220–227, November 1996.

[77] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In *Proceedings of the 38th Design Automation Conference*, pages 530–535, Las Vegas, NV, June 2001. ACM/IEEE.

[78] J. Nordström. Narrow proofs may be spacious: Separating space and width in resolution. *SIAM Journal on Computing*, 39(1):59–121, 2009.

[79] J. Nordström and J. Håstad. Towards an optimal separation of space and length in resolution. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, pages 701–710, Victoria, BC, May 2008.

[80] Jakob Nordström. A simplified way of proving trade-off results for resolution. *Information Processing Letters*, 109(18):1030–1035, 2009.

[81] Jakob Nordstrom. Pebble games, proof complexity, and time-space trade-offs. *arXiv preprint arXiv:1307.3913*, 2013.

[82] Jakob Nordström. A simplified way of proving trade-off results for resolution. *Information Processing Letters*, 109(18):1030–1035, August 2009. Preliminary version appeared in ECCC report TR07-114, 2007.

[83] Jakob Nordström. New wine into old wineskins: A survey of some pebbling classics with supplemental results. Manuscript in preparation. To appear in *Foundations and Trends in Theoretical Computer Science*. Current draft version available at http://www.csc.kth.se/~jakobn/research/, 2012.

[84] Jakob Nordström. On the relative strength of pebbling and resolution. *ACM Transactions on Computational Logic*, 13(2), 2012. To appear. Preliminary version appeared in *CCC '10*.

[85] Jakob Nordström. Pebble games, proof complexity and time-space trade-offs. *Logical Methods in Computer Science*, 2012. To appear. Available at http://www.csc.kth.se/~jakobn/research/.

[86] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[87] Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-sat. *J. ACM*, 52(3):337–364, 2005.

[88] Ramamohan Paturi, Pavel Pudlák, and Francis Zane. Satisfiability coding lemma. In *FOCS*, pages 566–574. IEEE Computer Society, 1997.

[89] Wolfgang J. Paul and Robert E. Tarjan. Time-space trade-offs in a pebble game. *Acta Informatica*, 10:111–115, 1978.

[90] Nicholas Pippenger. Pebbling. Technical Report RC8258, IBM Watson Research Center, 1980. Appeared in Proceedings of the 5th IBM Symposium on Mathematical Foundations of Computer Science, Japan.

[91] T. Pitassi and R. Raz. Lower bounds for regular resolution proofs of the weak pigeonhole principle. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, pages 347–355, Hersonissos, Crete, Greece, July 2001.

[92] Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, September 1997.

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

[94] Pavel Pudlák and Russell Impagliazzo. A lower bound for dll algorithms for k-sat (preliminary version). In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 128–136. Society for Industrial and Applied Mathematics, 2000.

[95] Pavel Pudlák and Russell Impagliazzo. A lower bound for dll algorithms for k-sat (preliminary version). In David B. Shmoys, editor, *SODA*, pages 128–136. ACM/SIAM, 2000.

[96] Pavel Pudlák and Russell Impagliazzo. A lower bound for DLL algorithms for k-SAT (preliminary version). In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00)*, pages 128–136, January 2000.

[97] R. Raz. Resolution lower bounds for the weak pigeonhole principle. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 553–562, Montreal, Quebec, Canada, May 2002.

[98] Ran Raz. Resolution lower bounds for the weak pigeonhole principle. *Journal of the ACM*, 51(2):115–138, 2004. Preliminary version appeared in *STOC '02*.

[99] A. A. Razborov. Improved resolution lower bounds for the weak pigeonhole principle. Technical Report TR01-055, Electronic Colloquium in Computation Complexity, http://www.eccc.uni-trier.de/eccc/, 2001.

[100] A. A. Razborov. Resolution lower bounds for the weak functional pigeonhole principle. Technical Report TR01-075, Electronic Colloquium in Computation Complexity, http://www.eccc.uni-trier.de/eccc/, 2001.

[101] Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, December 1998.

[102] Alexander A. Razborov. Resolution lower bounds for the weak functional pigeonhole principle. *Theoretical Computer Science*, 1(303):233–243, June 2003.

[103] John Alan Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, January 1965.

[104] The international SAT Competitions. http://www.satcompetition.org.

[105] G. Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, pages 593–602, 2008.

[106] Uwe Schöning. A probabilistic algorithm for k-SAT and constraint satisfaction problems. In *Proceedings 40th Annual Symposium on Foundations of Computer Science*, pages 410–414, New York,NY, October 1999. IEEE.

[107] Nathan Segerlind. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 13(4):482–537, December 2007.

[108] Janos Simon and Shi-Chun Tsai. On the bottleneck counting argument. *Theor. Comput. Sci.*, 237(1-2):429–437, 2000.

[109] Martin Tompa. Time-space tradeoffs for computing functions, using connectivity properties of their circuits. *Journal of Computer and System Sciences*, 20:118–132, April 1980.

[110] G. S. Tseitin. On the complexity of derivation in the propositional calculus. In A. O. Slisenko, editor, *Studies in Constructive Mathematics and Mathematical Logic, Part II*. 1968.

[111] Grigori Tseitin. On the complexity of derivation in propositional calculus. In A. O. Silenko, editor, *Structures in Constructive Mathematics and mathematical Logic, Part II*, pages 115–125. Consultants Bureau, New York-London, 1968.

[112] Grigori Tseitin. On the complexity of derivation in propositional calculus. In J. Siekmann and G. Wrightson, editors, *Automation of Reasoning*, pages 466–483. Springer, Berlin, 1983.

[113] Alasdair Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, January 1987.

[114] Ryan Williams. Non-uniform acc circuit lower bounds. In *IEEE Conference on Computational Complexity*, pages 115–125. IEEE Computer Society, 2011.

[115] Hantao Zhang. Sato: An efficient propositional prover. In *Proceedings of the International Conference on Automated Deduction, LNAI*, volume 1249, pages 272–275, July 1997.

[116] Lintao Zhang, Conor F. Madigan, Matthew H. Moskewicz, and Sharad Malik. Efficient conflict driven learning in a boolean satisfiability solver. In *Proceedings of the International Conference on Computer-Aided Design*, pages 279–285, San Jose, CA, November 2001. ACM/IEEE.