THE UNIVERSITY OF CHICAGO

COMPLEXITY MEASURES, SEPARATIONS, AND AUTOMATABILITY IN PROOF COMPLEXITY

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

BY
THEODOROS PAPAMAKARIOS

CHICAGO, ILLINOIS
AUGUST 2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Alexander Razborov, for his help, support and guidance throughout my time in the University of Chicago. I would like to thank the members of my committee, Aaron Potechin and Madhur Tulsiani. I would like to thank László Babai, Janos Simon, Costas Koutras, Stathis Zachos and Stavros Cosmadakis for their help and kindness throughout my studies at the University of Chicago, the University of Athens and the University of Patras. Finally, I would like to thank my parents.

ABSTRACT

With the rise of computer science, questions like "can we solve a problem?" got a quantitative counterpart: "how hard is it to solve a problem?". Proof complexity deals with the quantitative version of "can we prove a theorem?", namely, the question "how hard is it to prove a theorem?". The systematic study of the latter question for propositional proof systems started with Cook and Reckhow [24, 25], initiating a line of research which investigates how different proof systems for propositional logic compare with each other with respect to the minimum size needed to prove tautological formulas. A question that had remained open in the classification of Cook and Reckhow is whether cut-free sequent calculus can polynomially simulate resolution, in other words, whether adding atomic cuts to cut-free sequent calculus can super-polynomially decrease the size of proofs. We answer this question negatively, by showing a super-polynomial separation between cut-free sequent calculus as a system for refuting sets of clauses, and resolution.

Now, while size is certainly the most well studied, and arguably the most important measure of the complexity of proofs, other complexity measures have emerged, along with a line of study about relations between them, lack of relations thereof, and the inherent impossibility of optimizing two different measures at once. We contribute to this line of research by identifying two new clusters of known proof complexity measures equal up to polynomial and $\log n$ factors. The first cluster contains the logarithm of tree-like resolution size, regularized clause and monomial space, and clause space, ordinary and regularized, in regular and tree-like resolution. Consequently, separating clause or monomial space from the logarithm of tree-like resolution size is equivalent to showing strong trade-offs between clause space and length, and equivalent to showing super-critical trade-offs between clause space and depth. The second cluster contains resolution width, Σ_2 space (a generalization of clause space to depth-2 Frege systems), ordinary and regularized, and the logarithm of tree-like $R(\log)$ size. As an application, we improve a known size-space trade-off for polynomial

calculus with resolution. We further show a quadratic lower bound on tree-like resolution size for formulas refutable in clause space 4, and introduce a measure intermediate between depth and the logarithm of tree-like resolution size that might be of independent interest.

A third contribution of the dissertation is on the topic of automatability, that is the topic of how hard it is to find short proofs of a statement in a proof system. Significantly refining earlier results, Atserias and Müller [6] showed that the problem of automatability is NP-hard for resolution. Subsequently, this result was extended to stronger proof systems, including cutting planes, Res(k), and algebraic proof systems. We extend it to bounded-depth Frege, showing that for any $d > 0$, depth-d Frege systems are NP-hard to automate.

CHAPTER 1

INTRODUCTION

It might strike one as strange pondering the influence Platonism, and ultimately a Pythagorean mysticism, had during the scientific revolution, which served as an intellectual basis for the industrial revolution. Equally odd is the fact that at the foundations of the third industrial revolution, that is the rise of computers, lies mathematical logic and the attempt to understand the limits thereof.

The limits of mathematical theories become apparent once one adopts the view that proofs are mathematical object themselves. More specifically, a proof is a syntactic object that one should be able to algorithmically verify that it is indeed a proof. In the language of computability, the set of all provable statements in a theory is recursively enumerable, and that already sets serious limits to what can be proved within the theory.

The focus of this dissertation is proof complexity, that is the study of proofs within the framework of complexity theory. Complexity theory differs from its parent discipline, computability, in a radical way. Whereas the fundamental questions in computability got definite answers shortly after their formulation, the analogous questions in complexity theory have remained widely open for more than five decades after they were posed. When quantifiers are polynomially bounded, the same questions get drastically harder. REC vs. RE becomes P vs. NP, the question of what can be proved in theories of arithmetic becomes what can be proved in theories of bounded arithmetic, and so on.

The main goal in proof complexity is to show lower bounds in proof systems for propositional logic. From the point of view of complexity theory, propositional logic is the most interesting case. Here, as formulated by Cook and Reckhow [25], we require proofs not only to be algorithmically verifiable, but verifiable in polynomial time. So, NP is the class of language membership in which has proofs of polynomial size, and since the set of propositional tautologies is a coNP-complete set, asking whether there is a proof system with polynomial

size proofs of all tautologies is the same as asking whether $\mathsf{NP} = \mathrm{co}\mathsf{NP}$. Another reason why propositional logic is interesting, is that it can be viewed as the non-uniform analogue of theories of bounded arithmetic, in a way that showing lower bounds for what can be proved in propositional logic sheds light on the limits of those theories. Lastly, proof systems for propositional logic naturally correspond to algorithmic paradigms for solving SAT, so that lower bounds for the proof systems give lower bounds on the complexity of the corresponding algorithms.

1.1 Thesis overview

The rest of this chapter contains an overview of the dissertation's contribution, stating its main results. We continue in Chapter 2 by setting up the proofs-as-algorithms paradigm for the proof systems we shall be dealing with. In Chapter 3, we define various measures of the complexity of proofs and explore the relations among them. In Chapter 4, we investigate the relationship between cut-free sequent calculus and resolution and show a super-polynomial separation of the two systems for refuting sets of clauses. Chapter 5 is devoted to the topic of automatability, that is the topic of how hard is to find short proofs of a statement. More specifically, we show that bounded depth Frege systems are NP-hard to automate. We end with Chapter 6 with a summary, focusing on open problems.

Chapter 3 is based on [54], Chapter 4 is based on [52] and Chapter 5 is based on [53].

1.2 Contributions

1.2.1 Space characterization of complexity measures and size-space trade-offs

The most natural, arguably also the most important, measure of the complexity of a proof is its size, and indeed, much of the research in proof complexity has concentrated on proof size lower bounds. But given in particular their role in proof systems of practical significance,

several other natural complexity measures have been considered, and that has led to a considerable line of study about relations between them (*simulations*), lack of relations thereof (*separations*) and the inherent impossibility of optimizing two different measures at once (*trade-offs*).

A measure that directly emerged from the study of proof size lower bounds is width. The width of a resolution proof is the number of literals in the largest clause occurring in the proof. Its importance was accentuated by Ben-Sasson and Wigderson [14], who, building on the earlier works of Clegg et al. [23] and Impagliazzo et al. [44] showing an analogous result for polynomial calculus, showed that a short resolution proof can be transformed into a narrow one. Namely, we have that for a CNF formula in n variables,

$$W(F \vdash \bot) \leq \log S_{TR}(F \vdash \bot) + W(F), \tag{1.1}$$

$$W(F \vdash \bot) \leq O\left(\sqrt{n \log S_R(F \vdash \bot)}\right) + W(F). \tag{1.2}$$

$W(F \vdash \bot)$, $S_{TR}(F \vdash \bot)$ and $S_R(F \vdash \bot)$ stand here for the minimum width, tree-like size and DAG-like size respectively of refuting F in resolution; similar notation is employed throughout the section. $W(F)$ is the maximum width of a clause in F.

The study of the space requirements of proofs started in the early 2000's [29, 1], and space measures have since been realized to be important measures of the complexity of proofs, having natural connections with other complexity measures.

To define the space complexity of proofs, we imagine having a memory which at a given moment in time stores a portion of the proof we are producing. Then to make progress, we either delete some of the memory's contents, or we add to the memory either an axiom or a formula derivable from the current contents of the memory. The size of the memory used by a proof—different ways to measure the size give rise to different space measures—is the space complexity of the proof.

Esteban and Torán [29] showed that a short tree-like resolution proof can be transformed

into a resolution proof of small clause space:

$$\mathrm{CSpace}(F \vdash \bot) \leq \log S_{TR}(F \vdash \bot). \tag{1.3}$$

Atserias and Dalmau [5] demonstrated the first instance of the relationship between space and width, showing that a resolution proof having small clause space can be transformed into a narrow one:

$$W(F \vdash \bot) \leq \mathrm{CSpace}(F \vdash \bot) + W(F). \tag{1.4}$$

It is worth noting that (1.3) and (1.4) taken together provide a refinement of (1.1) and that, viewed this way, we relate two *sequential* measures (tree-like size and width) with a *space* measure as an intermediate. We will see more examples of such an interplay.

More recently, Bonacina [16] showed that for total space in resolution we have

$$W(F \vdash \bot) \leq O\left(\sqrt{\mathrm{TSpace}(F \vdash \bot)}\right) + W(F), \tag{1.5}$$

and Galesi et al. [35] showed a weakened version of (1.4), but for the analogue of clause space, called monomial space, in stronger proof systems operating with polynomials (or in fact even arbitrary Boolean functions of monomials):

$$W(F \vdash \bot) \leq O\left((\mathrm{MSpace}(F \vdash \bot))^2\right) + W(F). \tag{1.6}$$

Razborov [60] introduced regularized space complexity measures—for a space complexity measure μ, its regularized version μ^* is defined by multiplying the measure in question μ by the logarithm of the proof length—and identified a big cluster of ordinary and regularized space complexity measures, including total space and variable space, that are all equal up

to polynomial and $\log n$ factors to proof depth in resolution:

$$D(F \vdash \bot) \approx \text{TSpace}(F \vdash \bot) \approx \text{TSpace}^*(F \vdash \bot) \approx VSpace^*(F \vdash \bot). \tag{1.7}$$

We contribute to this line of research by identifying two more clusters of complexity measures each characterized by a regularized space measure. The cumulative picture combining both previously known and our results is summarized in Figure 1.1. There, an arrow from

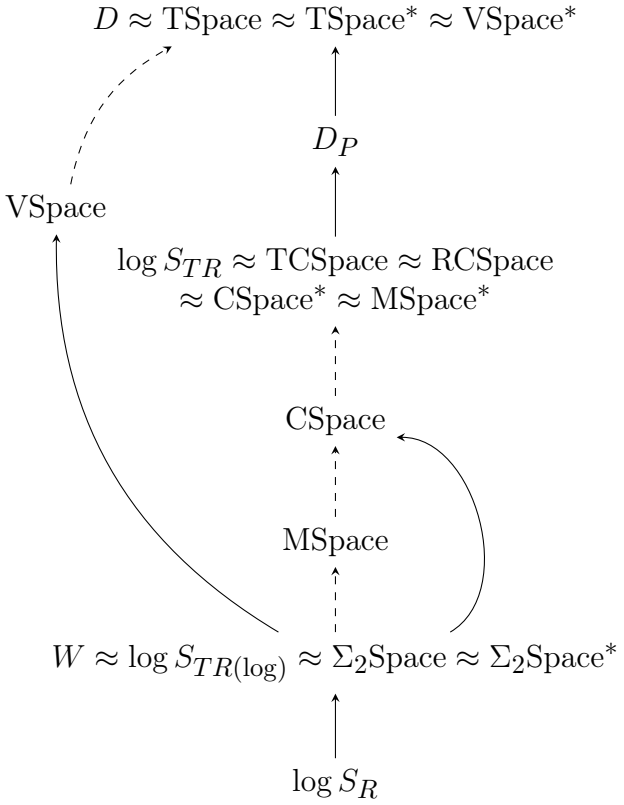


Figure 1.1: The landscape of complexity measures

μ_1 to μ_2 means that $\mu_1(F \vdash \bot) \leq (\mu_2(F \vdash \bot) \log n)^{O(1)}$ for any CNF F in n variables. A solid arrow from μ_1 to μ_2 indicates that a separation between μ_1 and μ_2 is known, that is, it additionally indicates that there exists a sequence $\{F_k(x_1, \ldots, x_{n_k})\}$ of unsatisfiable CNFs such that $\mu_2(F_k \vdash \bot) \geq (\mu_1(F_k \vdash \bot) \log n_k)^{\omega(1)}$. The non-trivial separations $W \not\leq \log S_R$, $W \not\leq \text{CSpace}$, $\text{VSpace} \not\leq \text{CSpace}$ and $\log S_{TR} \not\leq D$ are shown in [20], [10], [51, 12] and [68]

respectively.

The first new cluster is characterized by regularized clause and monomial space and contains the logarithm of tree-like resolution size, and clause space in tree-like and regular resolution (TCSpace and RCSpace in Figure 1.1), as well as their regularized versions. This, enables us on one hand to re-derive and strengthen previously known space lower bounds. In particular, we get that:

> There are unsatisfiable CNF formulas F of size $O(n)$ with $S(F \vdash \bot) \leq O(n)$, $W(F \vdash \bot) \leq O(1)$ and $\text{MSpace}^*(F \vdash \bot) \geq \Omega(n/\log n)$.

This is an improvement on the previously known bounds $\Omega(n^{2/11})$ [8], $\Omega(n^{1/4})$ [42] and $n^{1/2}/(\log n)^{O(1)}$ [40] on regularized monomial space. Unlike these previous results, our proof is remarkably simple.

On the other hand, it allows us to rephrase the existence of strong trade-offs between clause (or monomial) space and size in resolution in terms of separating clause space from the logarithm of tree-like size. Recently, trade-offs of this kind have been discovered between other measures (see e.g. [59, 60, 33]). Whether however strong trade-offs exist between clause space and size remains open.

Our second cluster is characterized by Σ_2 space and regularized Σ_2 space. Σ_2 space is an extension of clause space—a measure for depth 1 proofs—to depth 2 Frege systems, introduced in [54]. In this model, memory configurations are *arbitrary* sets of DNFs, and we charge k for every individual k-DNF in the memory. Clearly, $\Sigma_2\text{Space} \leq \text{CSpace}$ and $\Sigma_2\text{Space}^* \leq \text{CSpace}^*$. We then strengthen the Atserias-Dalmau bound (1.4) by replacing CSpace with Σ_2Space and continue to show that *both* ordinary and regularized versions of Σ_2 space are actually equivalent to resolution width.

We get this way that surprisingly, the open question of whether we have strong trade-offs between space and size for clause space gets a relatively easy negative solution once we get to depth 2 Frege systems. We have also been able to locate in this cluster another interesting

size measure: the size of tree-like proofs in the system $R(\log)$, which gives a somewhat unexpected generalization of (1.1). Note that $R(\log)$ being a proof system of depth between 1 and 2, this shows that strong size-space trade-offs are ruled out even before we reach depth 2. We have not been able to retrieve the equivalence of width and tree-like size in $R(\log)$ from the literature in exactly this form: it appears in [48] for k-DNF resolution, where k is constant, and it is implicit in [45].

Finally, we show that a tree-like resolution proof can be seen as describing a *pebbling* strategy, with the size of the proof corresponding to the time and its positive depth (a measure introduced in [54]—D_P in Figure 1.1) corresponding to the space of the pebbling strategy. We are able in this way, with an eye towards separating CSpace from $\log S_{TR}$, to get:

> There are CNF formulas F of size $O(n)$ with $\mathrm{CSpace}(F \vdash \bot) = 4$ and $S_{TR}(F \vdash \bot) \geq \Omega(n^2/\log n)$.

1.2.2 A super-polynomial separation of cut-free sequent calculus and resolution

Since their inception by Gentzen [38], sequent calculus and cut elimination have been ubiquitous tools in proof theory, and more generally in mathematical logic. Cuts play the role of intermediate lemmas in a proof, and cut elimination is a process of transforming a proof into one without intermediate lemmas.

On the philosophical side, when applicable, cut elimination states that a proof of a theorem can be made purely *analytic*, that is, not using concepts outside those within the statement of the theorem or the theory in which it is proven. On a more technical side, proofs without cuts are particularly benign objects to work with, and as such, arguments about proofs often begin with a reduction to the cut-free case. For instance, in classical unprovability results such as Gentzen's consistency proof or the unprovability of the termi-

nation of Goodstein sequences, one one starts with a purported proof of the statement the provability of which is under examination, and through a cut-elimination process arrives at an impossible proof.

In proof complexity, most, if not all, lower-bound results, can been seen in a similar manner: One starts with a purported proof of low complexity, and through a cut-elimination process, arrives at a proof of impossible complexity.

Now, it is well known that eliminating cuts from a proof must necessarily result in a proof of enormous size. And this is true even if the original proof only contained cuts of low complexity, e.g. formulas of a bounded number of variables or formulas of bounded depth. In propositional logic, we get a strict hierarchy of proof systems by allowing more and more formulas as cuts (see Figure 1.2).

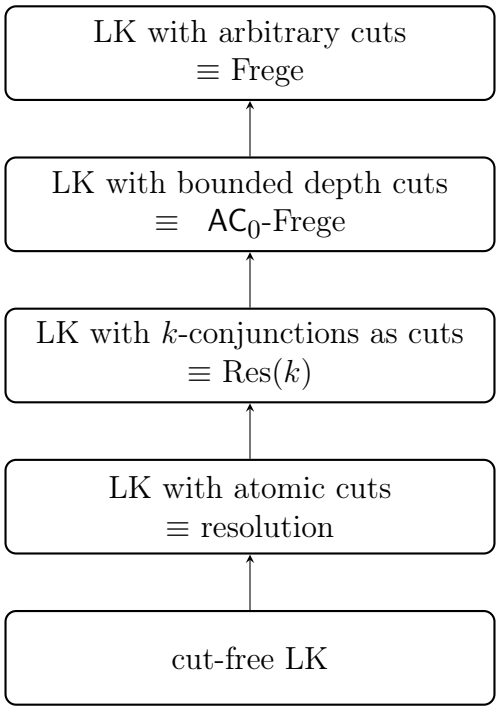


Figure 1.2: Proof systems as subsystems of sequent calculus

On the top of Figure 1.2 lies sequent calculus (abbreviated as LK) with arbitrary formulas as cuts. This is a system polynomially equivalent with Frege systems. These are very powerful systems; proving non-trivial lower bounds for them is completely out of reach of

current methods. Next we have sequent calculus with formulas of bounded depth as cuts, known as bounded-depth Frege, or AC_0-Frege. AC_0-Frege stands on the verge of systems for which exponential lower bounds are known. In particular, there are statements that only have exponential AC_0-Frege proofs, but have Frege proofs of polynomial size. In other words, allowing arbitrary cuts instead of cuts of bounded depth results in exponentially shorter proofs. Next we have sequent calculus with conjunctions of at most k variables as cuts, which is known as $R(k)$ or Res(k), and resolution, which is the same as Res(1), that is cuts can only be propositional variables. And again, there are formulas exponentially separating Res(k) from Res($k+1$). We will show a super-polynomial separation between resolution and cut-free sequent calculus. That is to say, even allowing propositional variables as cuts in cut-free sequent calculus, gives super-polynomially shorter proofs.

The question of whether cut-free sequent calculus can polynomially simulate resolution is a question existing since the beginnings of proof complexity. It was first raised in [24] and iterated e.g. in [67]. Cook and Reckhow [24] show that in the tree-like case, there are examples where resolution can have exponentially smaller proofs. Arai, Pitassi and Urquhart [4] point out that the answer may heavily depend on how clauses are represented. A clause consisting of the literals, say $\ell_1, \ell_2, \ell_3, \ell_4$, can be seen as either a single disjunction of arity four, or as a series of applications of binary disjunctions, for example $(\ell_1 \vee \ell_2) \vee (\ell_3 \vee \ell_4)$, and this can have a profound impact on the complexity of sequent calculus proofs. The result of Cook and Reckhow above applies in the case where clauses are seen as single disjunctions of unbounded arity, or the case where the order in which the binary disjunctions are applied is fixed. If we are free to choose the order, then tree-like cut-free sequent calculus can quasi-polynomially simulate tree-like resolution, and this is optimal [4]. In the DAG-like case, and if we are free to choose the order in which binary disjunctions are applied, Reckhow [61] shows that cut-free sequent calculus can polynomially simulate regular resolution, and Arai [3] shows that it can polynomially simulate resolution for refuting k-CNF formulas F, where k grows

at most logarithmically as a function of the size of the shortest resolution refutation of F.

We define the width of a sequent calculus proof as the maximum number of formulas occurring in a sequent of the proof. This definition extends in a natural way the concept of the width of a resolution proof to stronger proof systems. Furthermore, it allows for a simple, abstract characterization of sequent calculus width generalizing the characterization of Atserias and Dalmau for resolution width [5]. Using this characterization, we show a quadratic gap between resolution width and cut-free sequent calculus width. Resolution is a sequent calculus system that has only atomic cuts, so this says that including atomic cuts in cut-free sequent calculus can shorten the width of proofs. Utilizing then this gap, we show a super-polynomial separation between cut-free sequent calculus and resolution size. Namely, we show:

Theorem 1.2.1. *There are CNF formulas F of size s that have resolution refutations of size $O(s)$, but any cut-free sequent calculus refutation of F has size* $\exp\left(\Omega((\log s/\log\log s)^2)\right)$.

To put it in other words, atomic cuts can super-polynomially decrease the size of proofs. This result applies only when clauses are seen as disjunctions of unbounded arity. There is a way to write the clauses in our examples using binary disjunctions, so that the resulting formulas have linear size cut-free sequent calculus refutations. Thus, as it was already known for the tree-like case, the complexity of sequent calculus proofs can depend on how disjunctions are represented.

We furthermore note that our characterization of cut-free sequent calculus width for refuting CNF formulas coincides with the concept of dynamic satisfiability, introduced by Esteban, Galesi and Messner [28] as a tool for proving space lower bounds in resolution and Res(k). It is easily seen that dynamic satisfiability is a weakened version of resolution width. How much weaker however is a question that has not been addressed. We show that it is strictly weaker, the quadratic gap between resolution and cut-free sequent calculus width being a quadratic gap between the two. Moreover, our basic construction extends to stronger

versions of dynamic satisfiability used to prove monomial space lower bounds in algebraic proof systems [17, 18], allowing us to make progress towards separating resolution width from monomial space. In particular, we show a quadratic separation between resolution width and monomial space; no separation between the two was known before.

1.2.3 Non-automatability of bounded-depth Frege systems

Much of proof complexity is centered around the question of whether short proofs of a statement exist. A question of equal importance—central to automated theorem proving—is whether proofs are easy to find. Clearly, if a statement does not have short—i.e. of polynomial size—proofs in a proof system $\mathbf{P}$, then searching for a proof in $\mathbf{P}$ will necessarily take super-polynomial time. But in the case short proofs do exist, can one be found in polynomial time? This motivates the definition of automatability: A proof system $\mathbf{P}$ is automatable if there is an algorithm that given a formula F, finds a $\mathbf{P}$-proof of F in time polynomial in the size of the shortest $\mathbf{P}$-proof of F.

Apart from being a natural notion in itself, the notion of automatability is well connected to other important threads in proof complexity, for example feasible interpolation, canonical pairs and reflection principles. Relying on reflection principles (in particular, investigating how hard it is for a proof system to show lower bounds for itself!), it was shown by Atserias and Müller [6] that automating resolution is as hard as possible, significantly strengthening the non-automatability result of Alekhnovich and Razborov [2]. Namely, Atserias and Müller show that resolution is automatable if and only if $\mathsf{P} = \mathsf{NP}$. This result was subsequently extended to stronger systems, namely cutting planes [39], Res(k) [37], various algebraic proof systems [27]. We extend it to even bounded-depth Frege systems, showing:

Theorem 1.2.2. *If $\mathsf{P} \neq \mathsf{NP}$, then for any $d > 0$, depth-d Frege systems are not automatable.*

It should be noted that the non-automatability of bounded-depth Frege systems was known under a stronger assumption, namely that the Diffie-Hellman key exchange protocol

cannot be broken with circuits of subexponential size [19]. We improve on [19] on three fronts. First, the assumption $\mathsf{P} \neq \mathsf{NP}$ is much weaker, in particular, it is as weak as possible. Secondly, the result of [19] only works for sufficiently large d, while ours works for all d. Finally, our result requires proving new lower bounds for bounded-depth Frege, unlike the approach of [19]. However, still, this result and ours are incomparable: [19] rules out even the weak automatability of bounded-depth Frege systems, which is to say that no system polynomially simulating a depth-d Frege system is automatable.

CHAPTER 2

PROOFS, ALGORITHMS AND GAMES

A proof of a statement can be seen in two main ways. First, it can be seen as a derivation where, starting with some axioms, we arrive at the conclusion via a series of inferences. Alternatively, it can be seen as the execution of an algorithm which fails trying to falsify the given statement. The latter can be also viewed as a winning strategy in a game played by two players, where one of the players claims that the given statement is false and the other tries to refute that claim.

Let us try to illustrate this correspondence with an example taken from Plato's Meno. There, Socrates speaks with a boy. The boy claims that by doubling the side of a square, the length of which is two feet, we get a square the area of which is twice the area of the initial square, i.e. eight feet. Socrates goes on and refutes this claim, by asking a series of questions, forcing the boy to a contradiction:

> *Socrates.* Wouldn't this line [a side of the initial square] become double itself if we add another of the same length at this point [the end of the line]?
>
> *Boy.* Of course.
>
> *Socrates.* So from that, you say, the eight foot area will be made, if there are four such lines?
>
> *Boy.* Yes.
>
> *Socrates.* Let us then draw from this four equal lines. Is that what you say the eight foot area is or is it something else?
>
> *Boy.* This is exactly it.
>
> *Socrates.* Doesn't this picure contain four areas, each of which is equal to that four foot area?
>
> *Boy.* Yes.

Socrates. So how large does it become? Isn't it four times as large?

Boy. Of course it is.

Socrates. Can therefore, something four times as large, be twice as large?

Boy. No by Zeus.

We present this proof as a tree in Figure 2.1. We start at the root of the tree with the

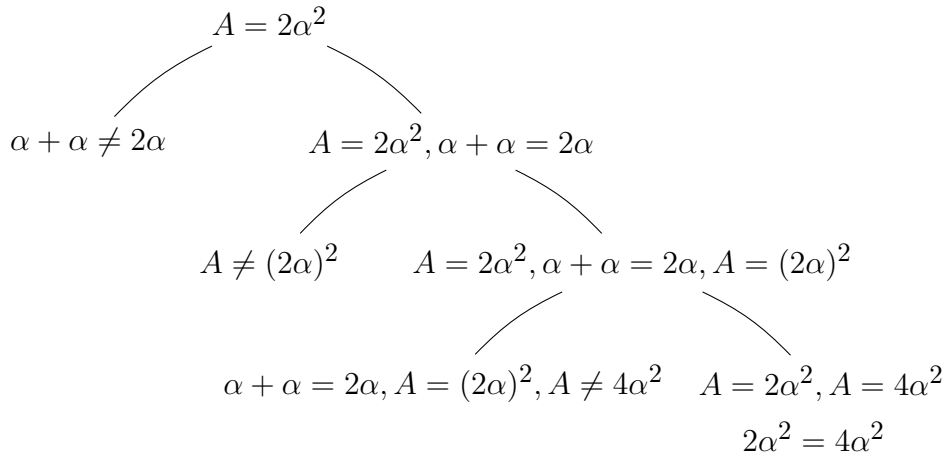


Figure 2.1: Socrates's proof as a game

boy's assertion that by doubling the side α of a square we get a square whose area A is twice as large as the area α^2 of the initial square. Socrates then asks, whether adding a line of the same length next to α will result in a line of double the length. If the boy answers no, he immediately arrives at a contradiction and the dialogue ends. If he answers yes, then Socrates asks whether A is the area made by this new line. Again the boy has to admit that this is indeed what A is, and Socrates asks whether that area contains four squares each of area α^2. The boy has to say yes, at which point he arrives at a contradiction: He maintains that A is both twice and four times the area α^2.

But how can this proof be turned into a derivation? The answer is simple: We replace every assertion at each node of this tree by its negation. This is done in Figure 2.2. This way the leaves become axioms and intermediate nodes become inferences. More specifically, we start at the bottom right, with the axiom $2\alpha^2 \neq 4\alpha^2$, that something twice as large

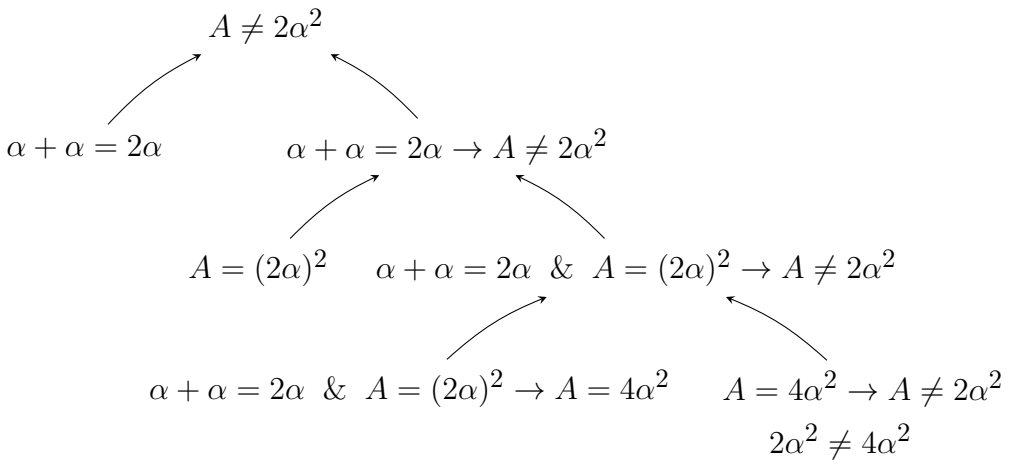


Figure 2.2: Socrates's proof as a derivation

cannot be four times as large. From this we infer $A = 4\alpha^2 \rightarrow A \neq 2\alpha^2$, that is, if A is four times the length α^2, then it cannot be two times α^2. Then from this and the axiom $\alpha + \alpha = 2\alpha \ \ \& \ \ A = (2\alpha)^2 \rightarrow A = 4\alpha^2$ stating that if we know that adding two lines of length α next to each other results in a line of twice the length α and A is the area of a square whose sides have length twice the length of α, then A contains four squares each of area α^2, we infer $\alpha + \alpha = 2\alpha \ \ \& \ \ A = (2\alpha)^2 \rightarrow A \neq 2\alpha^2$. Now from this assertion and the premise that A is the area of a square whose sides have length twice the length of α, $\alpha + \alpha = 2\alpha \rightarrow A \neq 2\alpha^2$ is inferred. Finally, from the latter and the axiom stating that adding two lines of length α next to each other results in a line of twice the length α, we infer the conclusion $A \neq 2\alpha^2$.

2.1 Sequent calculus

2.1.1 LK proofs as derivations

The correspondence between proofs, algorithms and games is best illustrated for systems based on logic by the sequent calculus, called Logistische Kalkül and abbreviated as LK by Gentzen. In its proofs-as-derivations formulation, LK operates with *sequents*. A sequent is a

tuple of the form (Γ, Δ), where Γ and Δ are finite sets of formulas. Traditionally, a sequent (Γ, Δ) is written as $\Gamma \to \Delta$. This is to remind us its semantic interpretation: $\Gamma \to \Delta$ is to be read as "if all formulas in Γ are true, then at least one formula in Δ is true". where Γ and Δ are finite sets of formulas.

Let us present the rules of the system's propositional part. In what follows, A, B represent arbitrary propositional formulas, and $\Gamma, \Delta, \Gamma', \Delta'$ represent finite sets (possibly empty) of propositional formulas. Sets occurring in sequents are written in a quite plain manner: We write Γ, A instead of $\Gamma \cup \{A\}$, A instead of $\{A\}$, A, B instead of $\{A, B\}$, $\Gamma \to$ instead of $\Gamma \to \emptyset$ and so no.

The axioms of LK are all sequents of the form $A \to A$. Of the inference rules, first we have a rule, which allows us to add formulas to the left or right part of a sequent. This rule is called the *thinning* or *weakening* rule and has the form

$$\frac{\Gamma \to \Delta}{\Gamma' \to \Delta'},$$

i.e., from $\Gamma \to \Delta$ derive $\Gamma' \to \Delta'$, where $\Gamma \subseteq \Gamma'$ and $\Delta \subseteq \Delta'$. Next, we have rules for each connective. These come in pairs; a connective is treated differently according to which side of a sequent it appears. The rules for the connectives $\wedge$, $\vee$ and $\neg$ are shown in Table 2.1. These rules are called *analytic*, and they already form a complete proof system for proving tautologies; we shall call this system *cut-free* LK or LK^-.

Finally, there is the *cut rule*:

$$\frac{\Gamma, A \to \Delta \qquad \Gamma \to \Delta, A}{\Gamma \to \Delta}. \tag{2.1}$$

So, already having a proof of, say $\to B$, we may use it to prove $\to A$: $\to A$ can be derived from $B \to A$ and $\to B$ via the cut rule, and now to prove $\to A$, we need to prove the weaker

$$\neg\mathrm{L}: \ \frac{\Gamma, \neg A \to \Delta, A}{\Gamma, \neg A \to \Delta} \qquad \neg\mathrm{R}: \ \frac{\Gamma, A \to \Delta, \neg A}{\Gamma \to \Delta, \neg A}$$

$$\wedge\mathrm{L}_1: \ \frac{\Gamma, A \wedge B, A \to \Delta}{\Gamma, A \wedge B \to \Delta} \qquad \wedge\mathrm{R}: \ \frac{\Gamma \to \Delta, A \wedge B, A \qquad \Gamma \to \Delta, A \wedge B, B}{\Gamma \to \Delta, A \wedge B}$$

$$\wedge\mathrm{L}_2: \ \frac{\Gamma, A \wedge B, B \to \Delta}{\Gamma, A \wedge B \to \Delta}$$

$$\vee\mathrm{R}_1: \ \frac{\Gamma \to \Delta, A \vee B, A}{\Gamma \to \Delta, A \vee B} \qquad \vee\mathrm{L}: \ \frac{\Gamma, A \vee B, A \to \Delta \qquad \Gamma, A \vee B, B \to \Delta}{\Gamma, A \vee B \to \Delta}$$

$$\vee\mathrm{R}_2: \ \frac{\Gamma \to \Delta, A \vee B, B}{\Gamma \to \Delta, A \vee B}$$

Table 2.1: The analytic LK rules

formula $B \to A$. B can be anything. It doesn't need to have any intuitive relation to A, but even as such, it might be the case that a proof of $B \to A$ is much shorter than a proof of $\to A$. Gentzen's cut-elimination theorem says that there is always an effective procedure of eliminating all applications of the cut rule from a proof, making it purely analytic. We refer to the formula A in applications of rule (2.1) as the formula being cut, or as the *cut formula*.

An LK derivation of a sequent σ is a derivation of σ starting with axioms and applying the rules of LK. We imagine starting with the axioms at the bottom, as in Figure 2.2, and going to the top by applying the LK rules. More formally, an LK derivation of σ is a sequence consisting of sequents, called the *lines* of the derivation, that ends with σ, and such that every sequent in the sequence is either an axiom, or results from previous sequents by one of the LK rules. An LK^- derivation is an LK derivation that never uses the cut rule.

We may allow, besides axioms $A \to A$, non-logical axioms to participate in a derivation as well, in which case we speak of derivations from a set of formulas. We write $S \vdash_{\mathrm{LK}} \sigma$ (the subscript is often omitted when it is clear what the underlying system is) if there is an LK derivation of σ from S.

The *size* $S(\pi)$ of an LK derivation π is the total number of symbols occurring in π. The

length of π is the number of sequents in π. Finally, we may view a derivation as a directed acyclic graph (*DAG* for short), by drawing edges from premises to conclusions in applications of the inference rules. If the DAG corresponding to a derivation is a tree, we shall refer to the derivation as being *tree-like*.

Let us end this section by mentioning an important property of LK derivations that we are going to use repeatedly, namely that LK derivations are closed under taking restrictions. A *restriction*, or partial truth assignment, is an assignment of truth values, 0 or 1, to a set of variables. Applying a restriction ρ to a sequent σ, we get a sequent $\sigma|_\rho$, by substituting every variable $x \in \mathrm{dom}(\rho)$ in σ with $\rho(x)$ and eliminating all constants, using the identities:

$$A \vee 0 = A,\ A \vee 1 = 1,\ A \wedge 0 = 0,\ A \wedge 1 = A,\ \neg 0 = 1,\ \neg 1 = 0,$$
$$\Gamma, 1 \to \Delta = \Gamma \to \Delta,\ \Gamma, 0 \to \Delta = 1,\ \Gamma \to \Delta, 1 = 1,\ \Gamma \to \Delta, 0 = \Gamma \to \Delta$$

Lemma 2.1.1. *For an LK derivation $\pi = \sigma_1, \ldots, \sigma_t$ and a restriction ρ to some of the variables occurring in π, $\pi|_\rho \stackrel{\text{def}}{=} \sigma_1|_\rho, \ldots, \sigma_t|_\rho$ is an LK derivation.*

2.1.2 LK as an algorithm

Following Smullyan [65], let us write the sequent

$$A_1, \ldots, A_k \to B_1, \ldots, B_\ell$$

as

$$T\,A_1, \ldots, T\,A_k, F\,B_1, \ldots, F\,B_\ell.$$

That is, we annotate the formulas appearing on the left side of the sequent by T, the formulas appearining on its right side by F, and conjoin the two sides to form a single set. T and F stand for true and false respectively—$T\,A$ should be thought of as asserting that A is true

and $F\,A$ as asserting that A is false.

Annotated formulas that are not annotated variables are naturally divided into two groups: those of a conjunctive and those of a disjunctive type. Formulas of the form $T\,A \wedge B$, $F\,A \vee B$, $T\,\neg A$ or $F\,\neg A$ belong to the former group, and those of the form $T\,A \vee B$ or $F\,A \wedge B$ to the latter. We use the letter "α" to stand for an arbitrary annotated formula of conjunctive type, and the letter "β" to stand for an arbitrary annotated formula of disjunctive type. We define the components α_i of a formula α and the components β_i of a formula β as shown in Table 2.2.

α	α_1	α_2	β	β_1	β_2
$T\,A \wedge B$	$T\,A$	$T\,B$	$F\,A \wedge B$	$F\,A$	$F\,B$
$F\,A \vee B$	$F\,A$	$F\,B$	$T\,A \vee B$	$T\,A$	$T\,B$
$T\,\neg A$	$F\,A$				
$F\,\neg A$	$T\,A$				

Table 2.2: Smullyan's notation

These provisions allow on the one hand for an extremely concise description of the rules of Table 2.1; they can be written as:

$$\frac{S, \alpha, \alpha_1}{S, \alpha}, \qquad \frac{S, \alpha, \alpha_2}{S, \alpha}, \qquad \frac{S, \beta, \beta_1 \qquad S, \beta, \beta_2}{S, \beta}.$$

More importantly, they reveal an algorithmic interpretation of LK. An LK proof, seen from the top to the bottom, i.e. from the sequent $\sigma \coloneqq A_1, \ldots, A_k \to B_1, \ldots, B_\ell$ it is proving to the axioms, describes the execution of an algorithm that tries to find a truth assignment (or more generally a model) that falsifies σ. The algorithm begins with σ written as $T\,A_1, \ldots, T\,A_k, F\,B_1, \ldots, F\,B_\ell$, asserting that there is an assignment that makes all A_i true and all B_i false, or equivalently, an assignment that falsifies σ. Then it keeps expanding this set, by applying the LK rules in reverse, that is from the conclusion to the premises. This expansion takes the form of a tree (or a DAG if we identify nodes labelled by the same set).

At any point, we may choose a leaf labelled by S, α and add to it a single child labelled by S, α, α_i, as any assignment satisfying α must also satisfy every α_i. Or we may choose a leaf labelled by S, β and add to it two children, one labelled by S, β, β_1 and the other by S, β, β_2, as any assignment satisfying β must either satisfy β_1 or β_2. The weakening rule allows the algorithm to forget information: We may add to a leaf labelled by S, a child labelled by a subset of S. Finally, the cut rule allows us to add to a leaf labelled by S, two children, one labelled by $S, T\ A$ and the other by $S, F\ A$ for any formula A, as every assignment must satisfy either A or $\neg A$. This may greatly facilitate the search procedure. If at any point a set of the form $T\ A, F\ A$ is reached, then the search process may terminate at that particular branch, as no assignment can set A to both true and false. Notice that the contradiction $T\ A, F\ A$ corresponds to the axiom $A \to A$. A tree (or DAG) constructed this way, every branch of which ends with a leaf labelled by a set of the form $T\ A, F\ A$, is an LK proof of σ.

A depth-first implementation of the algorithm described above is shown as Algorithm 1 below. Algorithm 1 is called on a sequent represented as a set of annotated formulas. It

Algorithm 1 The LK algorithm

procedure LK(S)
  if S contains both $T\ A$ and $F\ A$ for some formula A **then**
    return false
  if for every $\alpha \in S$, all $\alpha_i \in S$ and for every $\beta \in S$, there is a $\beta_i \in S$ **then**
    return true
  go to either **1**, **2** or **3**
  1. select an $S' \subseteq S$ and **return** LK(S')
  2. select an arbitrary formula A and **return** LK($S, T\ A$) **or** LK($S, F\ A$)
  3. select an $A \in S$
  if $A = \alpha$ **then**
    select a component α_i and **return** LK(S, α_i)
  if $A = \beta$ **then**
    return LK(S, β_1) **or** LK(S, β_k)

chooses at each recursive call non-deterministically what rule to apply and which formula to apply it to. If false is returned then there is an LK proof of our initial sequent; if no such proof exists, then there is an assignment that falsifies our initial sequent and Algorithm 1 is

able to find it, returning true. As presented, line 1, corresponding to the weakening rule, is redundant. However, incorporating memoization, that is the ability to stop the search when a set S has already been encountered in a previous recursive call that has returned, effectively identifying nodes labelled by the same set, this line makes it possible to greatly prune the search for a falsifying assignment. In other words, DAG-like proofs may be shorter than tree-like proofs. The key point in analyzing the correctness of Algorithm 1 (or equivalently the completeness of LK), is that when at the base case true is returned, we can create an assignment consistent with S by setting for each formula A, A to true if $T\,A \in S$, and false otherwise:

Lemma 2.1.2. *Suppose S is a set of annotated formulas such that*

1. *there is no formula A for which $T\,A \in S$ and $F\,A \in S$;*
2. *$\alpha \in S \implies$ for every α_i, $\alpha_i \in S$;*
3. *$\beta \in S \implies$ there is a β_i such that β_i.*

Then S is satisfiable.

Proof. Let τ be an assignment of truth values, 0 or 1, to all variables occurring in S, defined by setting

$$\tau(x) := \begin{cases} 1 & \text{if } T\,x \in S, \\ 0 & \text{otherwise.} \end{cases}$$

We show, by structural induction, that every element of S is made true by τ. It is immediate that every annotated variable is made true by τ. For every α-formula of S, it holds that for every α_i, $\alpha_i \in S$. By the induction hypothesis, every α_i is made true by τ, hence the same is true for α. Similarly, for every β-formula of S, it holds that there is a β_i such that $\beta_i \in S$, and by the induction hypothesis, that β_i is made true by τ. Hence β must be made true by τ as well. □

2.1.3 LK proofs as games

The third way of seeing LK proofs is to view them as games, or, more precisely, as winning strategies, in the following game. There are two players whom we call prover and adversary. Much like the example in the beginning of the chapter, given a statement in the form of a sequent

$$\sigma_0 = A_1, \ldots, A_k \to B_1, \ldots, B_\ell,$$

the adversary wants to show that σ_0 is false, while prover's aim is to prove the adversary wrong. The game starts with σ_0 written as

$$S_0 = T\,A_1, \ldots, T\,A_k, F\,B_1, \ldots, F\,B_\ell,$$

that is, the game starts with the adversary's claim that σ_0 is false. In every round, the prover either deletes some formulas in the current set S, or selects an α-formula in S and adds a component of it to S, or selects a β-formula, in which case the adversary adds a component of it to S. If we allow the cut rule, the prover may also choose an arbitrary formula A and the adversary must respond by adding either $T\,A$ or $F\,A$ to S. If at any point S contains $T\,A$ and $F\,A$ for some A, then the prover wins. If on the other hand the adversary is able to always respond never reaching such a contradiction, then he wins. Let us note that in the case we consider non-tautological formulas to be axioms as well, viz. we look at derivations from a set of premises, then the prover would also win if S stipulated that such an axiom is falsified.

Of course if σ_0 is provable, then the prover can always win, an LK proof of σ_0 describing her winning strategy. We will mainly be interested in the case where the adversary has a winning strategy. Most, if not all, lower-bound results in proof complexity can be seen as providing a winning strategy for the adversary where we impose certain restrictions on the prover. Now, whereas a winning strategy for the prover is described by an LK proof, to

describe a winning strategy for the adversary, we need to provide answers to all possible queries of the prover. The following definition does exactly that.

Definition 2.1.1 [65]**.** We call S_0 *analytically consistent* if there is a collection $\mathcal{S}$ of sets of annotated formulas that contains S_0 and such that for each $S \in \mathcal{S}$:

1. for any formula A, S does not contain both $T\,A$ and $F\,A$;

2. $S' \subseteq S \implies S' \in \mathcal{S}$;

3. $\alpha \in S \implies S, \alpha_i \in \mathcal{S}$ for every component α_i of α;

4. $\beta \in S \implies S, \beta_i \in \mathcal{S}$ for some component β_i of β.

If the following condition is also satisfied, then we call S_0 *synthetically consistent* with respect to the set $\mathcal{C}$:

5. $S, T\,A \in \mathcal{S}$ or $S, F\,A \in \mathcal{S}$ for any formula $A \in \mathcal{C}$.

In particular, we have:

Proposition 2.1.1. *S_0 is analytically consistent if and only if the adversary has a winning strategy in the game starting on S_0, where the prover does not make use of the cut rule. S_0 is synthetically consistent with respect to $\mathcal{C}$ if and only if the adversary has a winning strategy in the game starting on S_0, where the prover only uses cuts from $\mathcal{C}$.*

The following theorem puts in a formal way the correspondence between the proofs-as-derivations and proofs-as-games paradigms:

Theorem 2.1.1. *S_0 is analytically consistent if and only if there is no LK^- derivation of σ_0. It is synthetically consistent with respect to $\mathcal{C}$ if and only if there is no LK proof of σ_0 in which every cut formula belongs to $\mathcal{C}$.*

Proof. Let us only show the former sentence. Suppose first that S_0 is analytically consistent, and let $\mathcal{S}$ be the corresponding winning strategy of the adversary. We will show that in every LK^- derivation (not necessarily beginning with axioms $A \to A$) π of σ_0, there is an initial sequent (i.e. one that is not inferred by other sequents) which, when written as as set of annotated formulas, is in $\mathcal{S}$. From the first condition of Definition 2.1.1, that sequent is not an axiom, thus π is not a proof.

Base case. If π contains just σ_0, then we are done since $S_0 \in \mathcal{S}$.

Inductive step. Take some initial sequents $\sigma_1, \ldots, \sigma_r$ from which a sequent σ is derived via an inference rule ρ, and let $S_1, \ldots, S_r$ and S be the corresponding sets of annotated formulas. Remove $\sigma_1, \ldots, \sigma_r$ to get the derivation π'. From the induction hypothesis, there is an initial sequent in π' that belongs to $\mathcal{S}$. If that sequent is not σ, then it also appears in π and we are done. Otherwise, we have the following cases according to what rule ρ is:

Case 1. If it is the weakening rule, and thus $r = 1$ and $S_1 \subseteq S$, then from the second condition of Definition 2.1.1, $S_1 \in \mathcal{S}$.

Case 2. If ρ is the α-rule, and thus $r = 1$, $\alpha \in S$ and $S_1 = S, \alpha_i$ for some α_i, then from the third condition of Definition 2.1.1, $S_1 \in \mathcal{S}$.

Case 3. If ρ is the β-rule, and thus $\beta \in S$ and each S_i is of the form S, β_i, then from the fourth condition of Definition 2.1.1, some S_i belongs to $\mathcal{S}$.

Now suppose that there is no LK^- proof of σ_0. We set $\mathcal{S}$ to be the collection of all sets of annotated formulas whose corresponding sequents cannot be derived in LK^-. Clearly $S_0 \in \mathcal{S}$. We will show that $\mathcal{S}$ satisfies the conditions 1–4 of Definition 2.1.1. For each $S \in \mathcal{S}$, first S cannot contain $T\ A$ and $F\ A$ for some A, as the sequent σ corresponding to S is a weakening of an axiom. Similarly, if $S' \subseteq S$, then it must be that $S' \in \mathcal{S}$, since the sequent corresponding to S' is a weakening of σ. If $\alpha \in S$, then for each α_i it must be that $S, \alpha_i \in \mathcal{S}$, since the corresponding sequent follows from σ via the α-rule. Finally, if $\beta \in S$, then there

must be a β_i such that $S, \beta_i \in \mathcal{S}$, since σ can be derived from the sequents corresponding to S, β_i for all β_i. □

Note that to show that the existence of a winning strategy of the adversary for the game played on S_0 implies that S_0 is satisfiable, in other words to show the completeness of LK, we need to notice that such a winning strategy will contain a set that includes S_0 and satisfied the conditions of Lemma 2.1.2.

2.2 Interpolation games

2.2.1 LK$^-$ proofs as communication protocols

Suppose we have a formula of the form

$$A(X, Y) \wedge B(X, Z)$$

where X, Y and Z are disjoint sets of variables, A contains only variables from $X \cup Y$ and B contains only variables from $X \cup Z$. Furthermore, suppose that $A(X, Y)$ and $B(X, Z)$ are satisfiable, but their conjunction $A(X, Y) \wedge B(X, Z)$ is unsatisfiable.

Now, imagine there are two players, Alice and Bob. Alice receives an assignment $\tau_A : X \to \{0, 1\}$ under which $A(X, Y)$ is satisfiable, and Bob receives an assignment $\tau_B : X \to \{0, 1\}$ under which $B(Y, Z)$ is satisfiable. Since the conjunction $A(X, Y) \wedge B(X, Z)$ is unsatisfiable, there must exist an $x \in X$ such that $\tau_A(x) \neq \tau_B(x)$. We will see that an LK$^-$ proof of the unsatisfiability of $A(X, Z) \wedge B(X, Z)$, i.e. a LK$^-$ derivation of $A(X, Z), B(X, Z) \to$, can be seen as a communication protocol that the two players can follow to find such an x.

Take an LK$^-$ derivation of $A(X, Z), B(X, Z) \to$. First, we replace each sequent in the derivation by the set of annotated formulas stating that the sequent is falsified. The two players will follow a root-to-leaf path in the proof DAG, maintaining the invariant:

If $S = S_A \cup S_B$ is the set that appears in the current vertex, where S_A contains those formulas that are subformulas of A and S_B contains those formulas that are subformulas of B, then S_A is satisfied by an assignment that extends Alice's assignment τ_A and S_B is satisfied by an assignment that extends Bob assignment τ_B.

At the root of the proof DAG, we have that $S_A = \{T\ A(X,Z)\}$, $S_B = \{T\ B(X,Z)\}$ and the invariant holds. Now suppose that the invariant holds at an intermediate vertex labelled by $S = S_A \cup S_B$. If S is inferred from S' via either the weakening or the α-rule, then the invariant holds at S'. If S is inferred from the sets S, β_1 and S, β_2 via the β-rule, then we have two cases according to whether $\beta \in S_A$ or $\beta \in S_B$. If $\beta \in S_A$, then since S_A is satisfiable under Alice's assignment τ_A, then it must be that either S, β_1 or S, β_2 is satisfiable under τ_A. Alice finds which of the two is satisfiable, sends a bit to Bob to make this information available to him, and the two players move to the corresponding vertex. Similarly, if $\beta \in S_B$, then Bob finds which of the S_B, β_1 and S_B, β_2 is satisfiable under his assignment, sends a bit to Alice to make this information available to her, and the two players move to the corresponding vertex. When the two player reach an axiom, say $T\,x, F\,x$, then it must be that $T\,x \in S_A$ and $F\,x \in S_B$, or $T\,x \in S_B$ and $F\,x \in S_A$, so x is a variable for which $\tau_A(x) \neq \tau_B(x)$.

2.2.2 LK$^-$ proofs as circuits

Now, suppose we are given an LK$^-$ proof of the unsatisfiability of $A(X,Y) \wedge B(X,Z)$ and we are given an assignment $\tau : X \to \{0,1\}$ to the common variables. We assume as in the previous section that sequents are written as sets of annotated formulas, and we assume for simplicity that all axioms in the proof are of the form $T\,x, F\,x$ for a variable x. Again, there are two players, Alice and Bob, who will follow a root-to-leaf path in the proof DAG. The two players start at the root. At a vertex labelled by $S = S_A \cup S_B$, if S is inferred by either

the weakening or the α-rule, then there is only one choice: the two players move to the vertex labelled by the set from which S is inferred. If S is inferred from S, β_1 and S, β_2 by the β-rule, then if $\beta \in S_A$ it is Alice's choice which of the two vertices the players move to, and if $\beta \in S_B$, then it is Bob's choice which of the two vertices the players move to. When the two players reach an axiom $T\,x, F\,x$, then:

- If $x \in Y$, then Bob wins;
- if $x \in Z$, then Alice wins;
- if $x \in X$, $T\,x \in S_A$ and $F\,x \in S_A$, then Bob wins;
- if $x \in X$, $T\,x \in S_B$ and $F\,x \in S_B$, then Alice wins;
- if $x \in X$, $T\,x \in S_A$ and $F\,x \in S_B$, then if $\tau(x) = 1$ Alice wins, otherwise Bob wins;
- if $x \in X$, $T\,x \in S_B$ and $F\,x \in S_A$, then if $\tau(x) = 1$ Bob wins, otherwise Alice wins.

That is, whoever player is forced to a contradiction, it is a win for the other player.

It is easy to compute which of the two players has a winning strategy. Let $I_v : \{0,1\}^n \to \{0,1\}$ be a function which, given values to all variables in X, returns 1 if Alice has a winning strategy and 0 if Bob has a winning strategy, when the two players start at a vertex v of the proof DAG. If v is labelled by an axiom $T\,x, F\,x$, then

- If $x \in Y$, then $I_v = 0$;
- if $x \in Z$, then $I_v = 1$;
- if $x \in X$, $T\,x \in S_A$ and $F\,x \in S_A$, then $I_v = 0$;
- if $x \in X$, $T\,x \in S_B$ and $F\,x \in S_B$, then $I_v = 1$;
- if $x \in X$, $T\,x \in S_A$ and $F\,x \in S_B$, then $I_v = x$;

- if $x \in X$, $T\,x \in S_B$ and $F\,x \in S_A$, then $I_v = \neg x$.

If v is inferred by a single vertex u, then $I_v = I_u$. Finally, if v is inferred by two vertices u_ℓ and u_r, then if v is controlled by Alice (i.e. when at v, Alice chooses which of u_ℓ and u_r the two players will move to), then

$$I_v = I_{u_\ell} \vee I_{u_r},$$

and if v is controlled by Bob, then

$$I_v = I_{u_\ell} \wedge I_{u_r}.$$

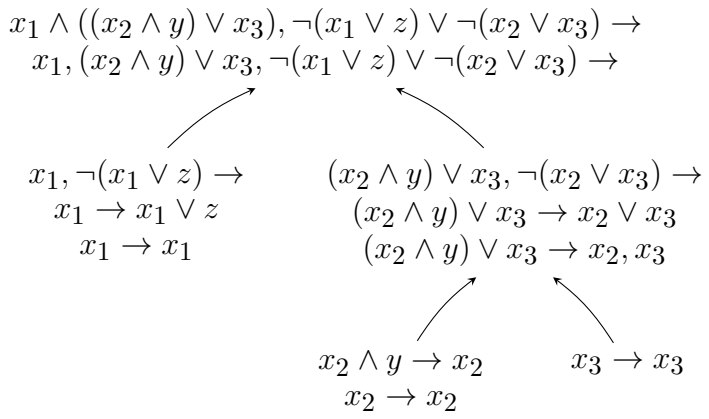
The function I_r, where r is the root of the proof DAG is known as an *interpolant* of $A(X,Y) \wedge B(X,Z)$ as it points to whichever of A and B becomes unsatisfiable when values are given to the variables in X. The construction of I_r from a cut-free proof of the unsatisfiability of $A(X,Y) \wedge B(X,Z)$ is known as Craig's interpolation theorem.

Figure 2.3 illustrates the different ways of seeing an LK^- proof.

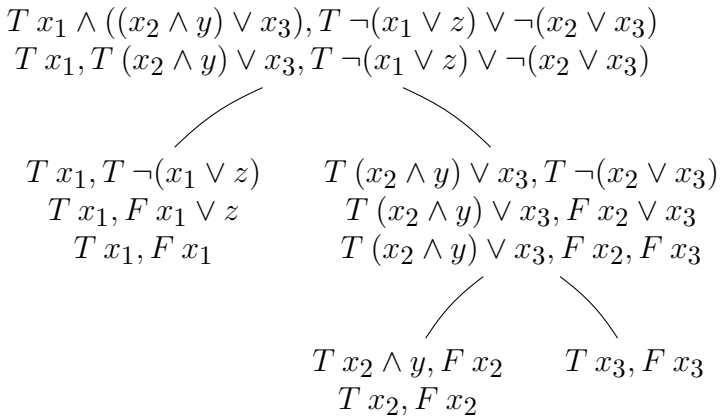
2.3 Bounded-depth Frege systems

Bounded-depth Frege systems are subsystems of LK where there is a bound on the depth of formulas occurring in proofs. For their definition, it is convenient to allow formulas have connectives $\bigvee$ and $\bigwedge$ of unbounded arity. The inference rules of LK for these two connectives are shown in Table 2.3. Or more succinctly, extending the uniform notation of Section 2.1.2 to encompass connectives of unbounded arity, we can write the analytic LK rules as:

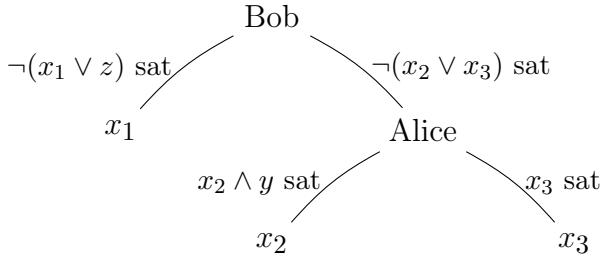
$$\frac{S, \alpha, \alpha_j}{S, \alpha}, \qquad \frac{S, \beta, \beta_1 \qquad \ldots \qquad S, \beta, \beta_k}{S, \beta}.$$



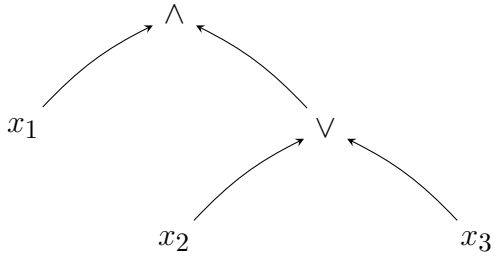
(a) As a derivation



(b) As an algorithm or a game



(c) As a communication protocol



(d) As a circuit

Figure 2.3: Different ways of seeing a cut-free proof

$$\bigwedge \text{L}: \ \frac{\Gamma, \bigwedge_i A_i, A_j \to \Delta}{\Gamma, \bigwedge_i A_i \to \Delta}$$

$$\bigwedge \text{R}: \ \frac{\Gamma \to \Delta, \bigwedge_i A_i, A_1 \qquad \ldots \qquad \Gamma \to \Delta, \bigwedge_i A_i, A_k}{\Gamma \to \Delta, \bigwedge_i A_i}$$

$$\bigvee \text{L}: \ \frac{\Gamma, \bigvee_i A_i, A_1 \to \Delta \qquad \ldots \qquad \Gamma, \bigvee_i A_i, A_k \to \Delta}{\Gamma, \bigvee_i A_i \to \Delta}$$

$$\bigvee \text{R}: \ \frac{\Gamma \to \Delta, \bigvee_i A_i, A_j}{\Gamma \to \Delta, \bigvee_i A_i}$$

Table 2.3: The analytic LK rules for $\bigvee$ and $\bigwedge$

The *depth* of a formula is defined by:

$$\begin{aligned} d(x) &= 0, \\ d(\neg A) &= d(A), \\ d\left(\bigvee_i A_i\right) &= 1 + \max_i d(A_i), \\ d\left(\bigwedge_i A_i\right) &= 1 + \max_i d(A_i). \end{aligned}$$

For a positive integer d, LK_d, also called depth-d Frege, is the subsystem of LK, where sequents in a proof can only contain formulas of depth $d-1$.[1]

The following important theorem, due to Krajícek, states that LK_d and tree-like LK_{d+1} are polynomially equivalent systems.

Theorem 2.3.1 [45]**.** *We have:*

1. An LK_d derivation of σ from S of size s can be transformed into a tree-like LK_{d+1}

1. Many authors define LK_d to be the subsystem of LK, where sequents can only contain formulas of depth d. We chose to follow the convention in which d refers to the depth of the lines of derivations, instead of the depth of the formulas occurring in their sequents.

derivation of σ from S of size $O(s^3)$.

2. *A tree-like LK_{d+1} derivation of a sequent σ containing depth d-formulas from a set of depth-d formulas S of size s can be transformed into an LK_d derivation of σ from S of size $O(s^2)$.*

Proof. For 1, let $\pi = \sigma_1, \ldots, \sigma_t$ be an LK_d derivation of σ from S of size s. For a sequent

$$\sigma = A_1, \ldots, A_k \to B_1, \ldots, B_\ell,$$

we set

$$\sigma' \coloneqq \neg A_1 \vee \cdots \vee A_k \vee \neg B_1 \vee \cdots B_\ell$$

to be σ written as a formula. We first note that for every $i \in [t]$, there is a tree-like derivation $\mathbf{T}_i$ of

$$\sigma'_1, \ldots, \sigma'_{i-1} \to \sigma'_i$$

in LK^- of length linear in the number of premises of the inference that gave σ_i in π. This can be seen by a case analysis depending on what inference rule gave σ_i in π. Having the derivations $\mathbf{T}_1, \ldots, \mathbf{T}_t$, we can derive $\to \sigma'_t$ from S, as shown in Figure 2.4. σ_t can then be derived from $\to \sigma'_t$ using size linear in the size of σ_t.

For 2, let $\mathbf{T}$ be a tree-like LK_{d+1} derivation of σ from a set of depth-d formulas S. We show, by induction on $\mathbf{T}$, that for any sequent written as a set of annotated formulas

$$T, \alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_\ell$$

in $\mathbf{T}$, where T contains all depth-d formulas occurring in the sequent, there is an LK_d derivation of

$$T, \alpha_{11}, \ldots, \alpha_{1p_1}, \ldots, \alpha_{k1}, \ldots, \alpha_{kp_k}$$

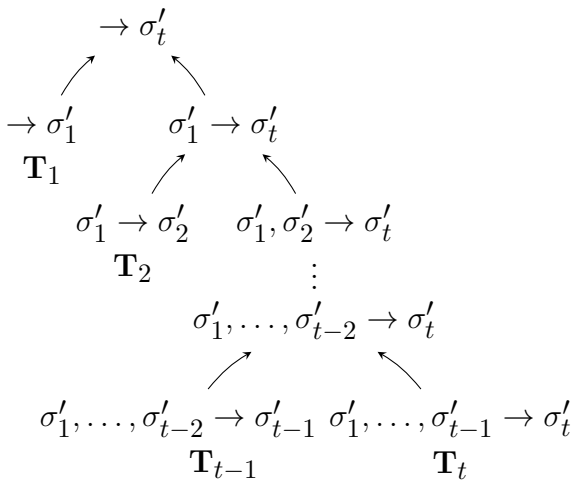


Figure 2.4: Turning an LK_d proof into a tree-like LK_{d+1} proof

from

$$S, \overline{\beta_{11}}, \ldots, \overline{\beta_{1q_1}}, \ldots, S, \overline{\beta_{\ell 1}}, \ldots, \overline{\beta_{\ell q_\ell}}$$

of length linear in the length of $\mathbf{T}$. Here $\overline{\beta}$ is the annotated formula resulting from β by changing its sign from T to F or from F to T. This can be proved by a case analysis depending on what rule gave $T, \alpha_1, \ldots, \alpha_k, \beta_1, \ldots, \beta_\ell$. 2 follows since σ does not contain formulas of depth $d+1$. □

2.3.1 Resolution, $R(k)$ and $R(\log)$

A *literal* is a propositional variable or the negation of a propositional variable. We let $\overline{x} \stackrel{\text{def}}{=} \neg x$ and $\overline{\neg x} \stackrel{\text{def}}{=} x$. Literals are often written in the form x^ε, where $x^1 := x$ and $x^0 := \neg x$. A *clause* is a disjunction (possibly empty) $\bigvee_i \ell_i$ of literals, and *term* is a conjunction (possibly empty) $\bigwedge_i \ell_i$ of literals. For a clause $C = \ell_1 \vee \cdots \vee \ell_w$, we define the term $\overline{C} \stackrel{\text{def}}{=} \overline{\ell_1} \wedge \cdots \wedge \overline{\ell_w}$; similarly for a term $t = \ell_1 \wedge \cdots \wedge \ell_w$, $\overline{t} \stackrel{\text{def}}{=} \overline{\ell_1} \vee \cdots \vee \overline{\ell_w}$. A conjunction of clauses is called a *CNF formula*, and a disjunction of terms is called a *DNF formula*. The *width* of a clause or a term is the number of literals it contains. The *width*, $W(F)$, of a CNF or DNF formula F is the width of the largest clause or term in it. A CNF or DNF formula of width at most w

is called *w-CNF* or *w-DNF* respectively.

Resolution is the system LK_1 restricted to derivations of the form $C_1, \ldots, C_m \vdash \bot$. That is, derivations of a contradiction (encoded by the empty sequent) from a set of clauses (encoded as sequents). Such derivations, showing the unsatisfiability of a set of formulas, are called *refutations*.

Now, notice that in refutations like this, the only applicable rules of LK are the weakening and the cut rule. Moreover, notice that sequents occurring in such refutations are comprised of literals, and such sequents can be written as clauses. We can hence get the following formulation of resolution: A resolution refutation of a CNF formula F is a sequence of clauses such that every clause in the sequence is either a clause of F, or results from previous clauses via one of the rules:

$$\frac{C}{C \vee D}, \quad \frac{C \vee x \qquad D \vee \neg x}{C \vee D}. \tag{2.2}$$

These are the weakening and the cut rule operating on clauses. The cut rule operating on clauses is also called the resolution rule, and the cut variable x in applications of the resolution rule is also referred to as the variable being *resolved*. A resolution derivation in which every variable is resolved at most once in every root-to-leaf path of the proof DAG is called *regular*.

We will be interested in systems extending resolution, intermediate between depth 1 and depth 2 Frege. To define such systems, let us first rewrite depth-2 Frege as a system operating on DNF formulas. Its rules and axioms become:

$$\frac{}{x \vee \neg x}, \quad \frac{G}{G \vee H}, \quad \frac{G \vee t_1 \qquad H \vee t_2}{G \vee H \vee (t_1 \wedge t_2)}, \quad \frac{G \vee t \qquad H \vee \bar{t}}{G \vee H},$$

where G and H are DNF formulas, t, t_1, t_2 and $t_1 \wedge t_2$ are terms. This system is known in literature as *DNF resolution*.

We get subsystems of DNF resolution by restricting the lines to be w-DNFs for some w. In particular, for a non-decreasing function $f : \mathbb{N} \to \mathbb{N}$, $R(f)$ is the subsystem of DNF resolution where each DNF in a proof of size s is required to have width at most $f(s)$. $R(k)$ where k is a constant function is also known as $\mathrm{Res}(k)$. For a non-constant function f, we will be mainly interested in the case $f = \log$; the system $R(\log)$ is also known as $\mathrm{LK}_{1.5}$, or depth-1.5 Frege.

2.4 Polynomial calculus

The correspondence between satisfiability algorithms and proof systems is expressed by various duality theorems for algorithms or proof systems based on algebra. We will be interested in *polynomial calculus*, which is a proof system whose corresponding satisfiability problem is, given a system of polynomials $(P_1, \ldots, P_k)$ in $\mathbb{F}[x_1, \ldots, x_n]$, whether $P_1, \ldots, P_k$ have a common root in $\mathbb{F}^n$.

A typical approach to this problem is first, to find a suitable basis of the ideal $\langle P_1, \ldots, P_l \rangle$ generated by $P_1, \ldots, P_k$. This involves two elementary operations:

1. taking linear combinations of polynomials, and
2. multiplying a polynomial by a variable,

or, written as inference rules:

$$\frac{P \qquad Q}{\alpha P + \beta Q}, \quad \frac{P}{xP}. \tag{2.3}$$

Clearly, if 1 is derived from $P_1, \ldots, P_k$ using the above rules, in other words $1 \in \langle P_1, \ldots, P_l \rangle$, then $P_1, \ldots, P_k$ cannot have a common root in $\mathbb{F}^n$. Moreover, if $\mathbb{F}$ is algebraically closed and $P_1, \ldots, P_k$ do not have a common root in $\mathbb{F}^n$, then 1 can be derived from $P_1, \ldots, P_k$. This is Hilbert's Nullstellensatz.

We can turn this system into a system for refuting sets of clauses as follows. First, we introduce, for every variable x, its dual variable $\overline{x}$. This is not necessary, but it is more than a matter of convenience; in particular, it enables the resulting system to polynomially simulate resolution. Secondly, we add the axioms

$$\frac{}{x^2 - x}, \quad \frac{}{x + \overline{x} - 1} \tag{2.4}$$

for every variable x to the system. These axioms restrict the space of solutions to 0-1 solutions, and also force the variables x and $\overline{x}$ to have complementary values. Notice that due to the presence of the axioms $x^2 - x$, $\mathbb{F}$ does not have to be algebraically closed for the system to be complete; if it is not algebraically closed, we can take its algebraic closure, and 0-1 solutions remain the same. Finally, for every clause $C = \bigvee_i \ell_i$, we encode C as the monomial $\prod_i \overline{\ell_i}$.

To summarize, fix a field $\mathbb{F}$, and let S be a set of clauses. A polynomial calculus with resolution derivation—*PCR* derivation for short—from S, is a sequence of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$, such that each polynomial in the sequence is either a clause in S represented as a monomial, or an axiom (2.4), or results from previous polynomials in the sequence via one of the two rules (2.3). A derivation from S ending with 1 is called a refutation of S. The *size* of a PCR derivation $\pi = P_1, \ldots, P_t$ is the total number of *distinct* monomials it contains. The *length* of π is t, and the *degree* of π is the maximum degree of a polynomial occurring in it.

2.5 Intuitionistic systems

Intuitionistic sequent calculus, abbreviated as *LJ*, is the subsystem of LK obtained by restricting all sequents $\Gamma \to \Delta$ occurring in a proof to have at most one formula in their right hand side Δ. The rules of the system are shown in Figure 2.4.

$$\text{Axioms: } \frac{}{A \to A}$$

$$\text{Weakening: } \frac{\Gamma \to \Delta}{\Gamma' \to \Delta'}, \text{ where } \Gamma \subseteq \Gamma', \Delta \subseteq \Delta', |\Delta'| \leq 1$$

$$\neg\text{L}: \frac{\Gamma \to A}{\Gamma, \neg A \to} \qquad \neg\text{R}: \frac{\Gamma, A \to}{\Gamma \to \neg A}$$

$$\wedge\text{L}_1: \frac{\Gamma, A \wedge B, A \to \Delta}{\Gamma, A \wedge B \to \Delta} \qquad \wedge\text{R}: \frac{\Gamma \to A \qquad \Gamma \to B}{\Gamma \to A \wedge B}$$

$$\wedge\text{L}_2: \frac{\Gamma, A \wedge B, B \to \Delta}{\Gamma, A \wedge B \to \Delta}$$

$$\vee\text{R}_1: \frac{\Gamma \to A}{\Gamma \to A \vee B} \qquad \vee\text{L}: \frac{\Gamma, A \vee B, A \to \Delta \qquad \Gamma, A \vee B, B \to \Delta}{\Gamma, A \vee B \to \Delta}$$

$$\vee\text{R}_2: \frac{\Gamma \to B}{\Gamma \to A \vee B}$$

$$\text{Cut: } \frac{\Gamma \to A \qquad \Gamma, A \to B}{\Gamma \to B}$$

Table 2.4: The system LJ

The proofs-as-algorithms paradigm carries over to LJ, however here, we have an algorithm that searches not for truth assignments falsifying the derived sequent, but more general models, namely Kripke model (see [32, 69]). We will take the more traditional view according to which a sequent $\Gamma \to A$ is interpreted as "there is a construction which, given proofs of all formulas in Γ, produces a proof of A". The rules of LJ are interpreted accordingly; for instance the cut rule

$$\frac{\Gamma \to A \qquad \Gamma, A \to B}{\Gamma \to B}$$

states that

> if there is a construction f_1 which, given proofs of all formulas in Γ, produces a proof of A, and there is a construction f_2 which, given proofs of all formulas

in Γ and A, produces a proof of B, then there is a construction f_3 which, given proofs of all formulas in Γ, produces a proof of B.

f_3 is easy to describe: apply f_1 to Γ to get A, and then apply f_2 to Γ, A to get B.

Let us try to illustrate this idea for the tree-like version of the system LJ_1. Sequents in LJ_1 are of the form $x_1, \ldots, x_k \to y$, where $x_1, \ldots, x_k, y$ are propositional variables. Sequents of this form are called *Horn clauses*, and CNF formulas whose clauses are Horn are called *Horn formulas*.

Notice that the only applicable rules in LJ_1 are the cut and the weakening rules. Moreover, since applying the resolution rule to two Horn clauses always gives a Horn clause, LJ_1 derivations from a set of Horn clauses are the same as resolution derivations. We are going to consider tree-like LJ_1 refutations from a set S, and we will assume that the weakening rule is never applied except at the premises of S—a tree-like LJ_1 (or resolution) refutation can always be put in this form. For such a refutation $\mathbf{T} = \Gamma_1 \to \Delta_1, \ldots, \Gamma_t \to \Delta_t$, we define its *positive depth* to be $D_P(\mathbf{T}) \stackrel{\text{def}}{=} \max_i |\Gamma_i|$.

Now, consider the following pebbling game, played on a set H of Horn clauses. There is a limited amount of pebbles. Pebbles are placed on the variables occurring in H according to the rules:

1. A pebble can be placed on a variable y if $x_1, \ldots, x_k \to y$ is a clause of H, and all $x_1, \ldots, x_k$ have pebbles on them. In particular, a pebble can be always placed on any variable y such that $\to y$ is a clause of H.

2. A pebble can be removed from a variable at any time.

A *configuration* of the pebbling game is a set of variables of H. A *pebbling strategy* is a sequence of configurations, each resulting from the previous one by one of the above rules. We say that a pebbling strategy *refutes* H if it ends with a configuration where for some clause $x_1, \ldots, x_k \to$ of H, all variables $x_1, \ldots, x_k$ are pebbled. Note that if H is unsatisfiable, then such a clause $x_1, \ldots, x_k \to$ must exist.

Theorem 2.5.1. *Let H be an unsatisfiable set of Horn clauses. A tree-like LJ_1 (or resolution) refutation $\mathbf{T}$ of H of size s and positive depth d can be turned into a pebbling strategy that, starting with the empty configuration, refutes H in at most s steps and using at most $d+1$ pebbles.*

Proof. We will show, by induction on $\mathbf{T}$, that every subtree of $\mathbf{T}$ deriving a clause $\Gamma \to \Delta$, describes a pebbling strategy that, starting with pebbles on all variables of Γ, either refutes H, or ends with a configuration which has pebbles on all variables of Γ and Δ. Thus, if Δ is empty then the former must occur; in particular, the strategy corresponding to the empty clause $\to$ will start with no pebbles on the variables of H and will refute H.

Suppose that $\Gamma \to \Delta$ is at a leaf. If there are variables $x_1, \ldots, x_k$ in Γ such that $x_1, \ldots, x_k \to$ is a clause of H, then that leaf describes a strategy that, starting with pebbles on all variables in Γ, immediately refutes H. Otherwise, there must be variables $x_1, \ldots, x_k$ in Γ and a variable $y \in \Delta$ such that $x_1, \ldots, x_k \to y$ is a clause of H. Then the strategy at that leaf is to put a pebble on y.

If $\Gamma \to \Delta$ is not at a leaf, then consider its left and right subtrees $\mathbf{T}_1$ and $\mathbf{T}_2$ proving $\Gamma, x \to \Delta$ and $\Gamma \to x$ respectively. The strategy corresponding to $\Gamma \to \Delta$ is the following. First follow $\mathbf{T}_2$'s strategy. If that strategy either refutes H or places a pebble on one of Γ's variables, then we are done. Otherwise, when the strategy of $\mathbf{T}_2$ is concluded, there are pebbles on Γ and x. Remove all other pebbles and follow the strategy of $\mathbf{T}_1$.

The number of steps of the pebbling strategy corresponding to $\to$ is at most the size of $\mathbf{T}$ (in fact it is exactly the number of leaves of $\mathbf{T}$ in case H is not refuted before reaching $\to$), and the number of pebbles used is at most the positive depth of $\mathbf{T}$ plus one. □

CHAPTER 3

COMPLEXITY MEASURES

The focus in this chapter is the study of complexity measures for systems revolving around resolution. A primary goal is to present and explain the relations of Figure 1.1. We start with introducing the width measure for general LK proofs. Then we turn to width in resolution and demonstrate its relation with resolution size. We continue with introducing the notion of the space complexity of proofs and show the relations between space, width and size. A notable feature of our analysis is using width in LK^- instead of width in resolution in many of the relations involving width. On one hand, this shows that many of the simulations, e.g. the simulation of tree-like resolution size by width of Ben-Sasson and Wigderson [14], or the simulation of clause space by width of Atserias and Dalmau [5], are cut-elimination constructions in disguise. On the other hand, it simplifies and refines the relations. We end the chapter with a discussion on lower bounds for the bottom two clusters of Figure 1.1.

3.1 Width

3.1.1 The width of LK proofs

We define the *width* of an LK derivation $\pi = \Gamma_1 \to \Delta_1, \ldots, \Gamma_t \to \Delta_t$ to be $W(\pi) \stackrel{\text{def}}{=} \max_i |\Gamma_i| + |\Delta_i|$. We will be interested in the minimum width needed to derive a sequent. From a construction dual to that of the first part of Theorem 2.3.1, we can see that every tautological sequent can be derived in constant width in LK. The concept of the minimum width needed to derive a sequent becomes interesting once we consider restrictions of LK, in particular when we restrict the cut rule. In such cases, the minimum width needed to derive a sequent can be characterized using the following modification of Definition 2.1.1:

Definition 3.1.1. We call a set of annotated formulas S_0 *analytically k-consistent* if there is a set of sequents $\mathcal{S}$ containing S_0 and such that for each $S \in \mathcal{S}$:

1. for any formula A, S does not contain both $T\,A$ and $F\,A$;

2. $S' \subseteq S \implies S' \in \mathcal{S}$;

3. $|S| < k \;\&\; \alpha \in S \implies S, \alpha_i \in \mathcal{S}$ for every component α_i of α;

4. $|S| < k \;\&\; \beta \in S \implies S, \beta_i \in \mathcal{S}$ for some component β_i of β.

If the following condition is also satisfied, then we call S_0 *synthetically k-consistent* with respect to the set $\mathcal{C}$:

5. $|S| < k \implies S, T\,A \in \mathcal{S}$ or $S, F\,A \in \mathcal{S}$ for any formula $A \in \mathcal{C}$.

In the game interpretation of Definition 3.1.1, there is a bound k, and the prover tries to force the adversary to a contradiction while maintaining the bound k on the cardinality of the configurations of the game. A k-consistency property describes a winning strategy of the adversary, that is, a strategy the adversary can follow so that the prover can never force him to a contradiction provided she maintains the bound k. The adversary can win in the game played on S_0 precisely in the case there is no width k derivation of S_0. More precisely, analogously to Theorem 2.1.1, we have:

Theorem 3.1.1. *Suppose that $|S_0| \leq k$. Then S_0 is analytically k-consistent if and only if every LK^- derivation of the sequent σ_0 corresponding to S_0 has width more than k. It is synthetically k-consistent with respect to $\mathcal{C}$ if and only if every LK derivation of σ_0 in which every cut formula belongs to $\mathcal{C}$ has width more than k.*

3.1.2 Resolution width and size

The importance of the width measure stems from the fact that showing width lower bounds is one of the main tools for showing size lower bounds. In the case of resolution, size and width are directly related. The width of a resolution derivation is the width of the largest clause in the derivation. We denote by $W_R(F \vdash \bot)$ and $S_R(F \vdash \bot)$ the minimum width and

size respectively of a resolution refutation of F, and denote by $S_{TR}(F \vdash \bot)$ the minimum size of a tree-like resolution refutation of F. The following theorems are due to Ben-Sasson and Wigderson.

Theorem 3.1.2 [14]**.** *For any unsatisfiable CNF formula F,*

$$W_R(F \vdash \bot) \leq \log S_{TR}(F \vdash \bot) + W(F).$$

Theorem 3.1.3 [14]**.** *For any unsatisfiable CNF formula over n variables,*

$$W_R(F \vdash \bot) \leq O\left(\sqrt{n \log S_R(F \vdash \bot)}\right) + W(F).$$

Theorem 3.1.2 can be seen as a special case of the second part of Theorem 2.3.1, in the sense that the two constructions are essentially the same. This is no coincidence; both constructions are cut-elimination constructions, and can be seen as special cases of Gentzen's cut-elimination construction.

To see that Theorem 3.1.2 is a cut-elimination construction, let us first present LK^- for refuting CNF formulas as a system operating on clauses. First, notice that we cannot consider LK^- derivations of the empty sequent $\to$, as there are no analytic rules that can be applied to infer $\to$. What we are going to do is, instead of considering derivations $C_1, \ldots, C_m \vdash \bot$, consider derivations $\vdash C_1, \ldots, C_m \to$. The only analytic inference rule of LK that can be applied in such derivations is $\bigvee$L. Moreover, there is no reason to always carry the clauses $C_1, \ldots, C_m$ in sequents. We may as well delete them from every sequent, but keep in our mind that they are implicitly there. What remains are sequents of the form $\ell_1, \ldots, \ell_r \to$, where the ℓ_i's are literals, and such sequents can be written as clauses. To be explicit, the axioms of the resulting system are clauses of the form $x \vee \neg x$, and the inference rules are the

weakening rule and $\bigvee$L written as

$$\frac{C \vee \overline{\ell_1} \quad \cdots \quad C \vee \overline{\ell_r}}{C}, \tag{3.1}$$

where C and D are clauses and $\ell_1 \vee \cdots \vee \ell_r$ is a clause of the formula we are refuting.

So an LK$^-$ refutation of a CNF formula F is a derivation of the empty clause starting with axioms $x \vee \neg x$ and using the weakening and rule (3.1). The width of such a derivation is the width of the largest clause occurring in it. We denote by $W_{\text{LK}^-}(F \vdash \bot)$ and $S_{\text{LK}^-}(F \vdash \bot)$ the minimum width and size respectively of an LK$^-$ refutation of F.

Now, returning to the point that Theorem 3.1.2 is a cut-elimination construction, we have:

Theorem 3.1.4. *For any unsatisfiable CNF formula F,*

$$W_{\text{LK}^-}(F \vdash \bot) \leq \log S_{TR}(F \vdash \bot) + 1.$$

Proof. We will construct, by induction on s, for every tree-like resolution refutation $\mathbf{T}$ of F of size s, an LK$^-$ refutation of F of width at most $\log s + 1$.

If $\mathbf{T}$ has size 1, then there is an LK$^-$ refutation of F of width 0; if $\mathbf{T}$ has size 3 then there is an LK$^-$ refutation of F of width 2.

For the inductive step, suppose that $\mathbf{T}$ has size $s > 3$. Let $\mathbf{T}_1$ and $\mathbf{T}_2$ be the subderivations of $\mathbf{T}$, deriving $\neg x$ and x respectively, for some variable x. One of $\mathbf{T}_1$ and $\mathbf{T}_2$, say $\mathbf{T}_1$, must have size at most $s/2$. $\mathbf{T}_1|_{x=1}$ is a refutation of $F|_{x=1}$ of size at most $s/2$, and $\mathbf{T}_2|_{x=0}$ is a refutation of $F|_{x=0}$ of size less than s. From the induction hypothesis, there are LK$^-$ refutations π_1 and π_2 of $F|_{x=1}$ and $F|_{x=0}$ of width $\log s$ and $\log s + 1$ respectively. Start with π_2. To every application of Rule (3.1), add the extra premise $C \vee x$. But x can be derived from π_1, and hence all those clauses $C \vee x$ can be derived via the weakening rule

from x: We can do the same to π_1, now adding $C \vee \overline{x}$ as the extra premise, and moreover add to every clause the variable x. The refutation obtained when we combine π_1 and π_2 is a valid LK^- refutation of F and has width at most $\log s + 1$. □

A benefit of Theorem 3.1.4 compared to Theorem 3.1.2 is that the additive factor $W(F)$ disappears. This means in particular that Theorem 3.1.4 can be used to show size lower bounds for CNF formulas that have large width, whereas Theorem 3.1.2 could not.

Theorem 3.1.3 can be stated for LK^- as well:

Theorem 3.1.5. *For any unsatisfiable CNF formula F over n variables,*

$$W_{\mathrm{LK}^-}(F \vdash \bot) = O\left(\sqrt{n \log S_{\mathrm{LK}^-}(F \vdash \bot)}\right).$$

Proof. The construction will be the same as that of Theorem 3.1.4. The problem here is that it is not clear what variable to choose to recurse on. The trick is to choose the variable that appears more often in the clauses of the proof.

For an LK^- refutation π of a CNF and a $d \geq 0$, let π^* be the set of clauses in π of width greater than d. We call the clauses in π^* the *fat* clauses of π. We show, by induction on n, that for any CNF F in n variables, any LK^- refutation π of F and any integers $d, b \geq 0$,

$$|\pi^*| < a^b \implies W_{\mathrm{LK}^-}(F \vdash \bot) \leq d + b,$$

where $a = (1 - d/(2n))^{-1}$. The theorem follows taking π to be of minimum size and setting $d \coloneqq \left\lceil \sqrt{n \log S(\pi)} \right\rceil$.

If $n = 1$, then there is an LK^- refutation of F of width 2, and in the case that $b + d < 2$ the implication becomes trivially true.

For the inductive step, suppose that $n > 1$, let $d, b \geq 0$, and let π be an LK^- refutation of F with $|\pi^*| < a^b$. If $b = 0$, then π itself is a refutation of width at most $d + b$. Suppose $b > 0$. There are $2n$ literals, so there must be some literal, say the variable x, appearing

in at least $d|\pi^*|/(2n)$ clauses in π^*. $\pi|_{x=1}$ is an LK$^-$ refutation of $F|_{x=1}$ with at most $|\pi^*|\,(1-d/(2n)) < a^{b-1}$ fat clauses, so by the induction hypothesis (notice that a is a decreasing function of n), there is an LK$^-$ refutation π' of $F|_{x=1}$ of width at most $d+b-1$. Furthermore, $\pi|_{x=0}$ is an LK$^-$ refutation of $F|_{x=0}$ with less than a^b clauses, so by the induction hypothesis, there is an LK$^-$ refutation π'' of $F|_{x=0}$ of width at most $d+b$. Combining π' and π'' as in the proof of Theorem 3.1.4, we get an LK$^-$ refutation of F of width at most $d+b$. □

Notice that in Theorem 3.1.4 we have LK$^-$ in the left hand side and resolution in the right hand side. It is tempting to speculate on whether the same is also true for Theorem 3.1.5, that is whether we can replace $S_{\text{LK}^-}(F \vdash \bot)$ with $S_R(F \vdash \bot)$. After all, the only place in the proof of Theorem 3.1.5 where we need LK$^-$ in the right hand side is the case $b = 0$. We will see in Chapter 4 that this is not possible.

3.2 Space

3.2.1 Space measures for resolution, PCR and LK$_2$ proofs

To define the space requirements of proofs, we imagine having a memory, which at a given moment in time stores a portion of the proof we are producing. That is, the content of the memory at a given time, called a *memory configuration* (or simply configuration), will be a set of lines in the proof system we are working with. Then to make progress, we either delete some of lines currently in the memory, or we add to the memory either an axiom or a formula derivable from the current contents of the memory. The size of the memory used by a proof—different ways to measure the size give rise to different space measures—is the space (complexity) of the proof.

More formally, a derivation in *configurational form* from a set S in a proof system $\mathbf{P}$ is a sequence $\mathcal{M}_1, \ldots, \mathcal{M}_t$ of memory configurations, such that $\mathcal{M}_1 = \emptyset$ and each $\mathcal{M}_i$ results

from $\mathcal{M}_{i-1}$ by one of the following:

Axiom Download: $\mathcal{M}_i = \mathcal{M}_{i-1} \cup \{A\}$, where A is either an axiom of $\mathbf{P}$ or $A \in S$;

Erasure: $\mathcal{M}_i = \mathcal{M}_{i-1} \setminus \{L\}$, where $L \in \mathcal{M}_{i-1}$;

Inference: $\mathcal{M}_i = \mathcal{M}_i \cup \{L\}$, where L can be derived from $\mathcal{M}_{i-1}$ by one of the inference rules of $\mathbf{P}$.

Let $\pi = \mathcal{M}_1, \ldots, \mathcal{M}_t$ be a configurational proof. π is called a refutation if $\bot \in \mathcal{M}_t$. π is said to be *tree-like* if, whenever a formula is used as a premise in an inference rule, it is immediately erased from the memory. Finally, π is said to be *regular* if a variable cannot be resolved more than once on any path in (the DAG resulting from the expansion of) π.

The *clause space* of a configuration in resolution, or a subsystem of resolution, is the number of clauses it contains, its *variable space* the number of *distinct* variables it contains, and its *total space* the total number of literals, counting repetitions, it contains. For DNF resolution, we will be interested in what we call Σ_2 *space* of a configuration. The Σ_2 space of a configuration $\mathcal{M} = \{G_1, \ldots, G_s\}$, where the G_i's are DNF formulas, is defined as the sum of widths:

$$\Sigma_2\text{Space}(\mathcal{M}) \stackrel{\text{def}}{=} W(G_1) + \ldots + W(G_s).$$

For PCR, we will consider the *monomial space* of a configuration, which is the number of *distinct* monomials in it.

For a space measure μ on configurations and a derivation $\pi = \mathcal{M}_1, \ldots, \mathcal{M}_t$ in configurational form, we let

$$\mu(\pi) \stackrel{\text{def}}{=} \max\{\mu(\mathcal{M}_i) \mid i \in [t]\}.$$

As before, for a space complexity measure μ, we write $\mu(F \vdash \bot)$ for the minimum value of $\mu(\pi)$, taken over all refutations in the proof system associated with μ of F.

Finally, *regularized* versions of space complexity measures are defined by multiplying the

measure in question by the logarithm of the proof length. That is, for a space measure μ on proofs, we define its regularized version μ^* by

$$\mu(\pi) \stackrel{\text{def}}{=} \mu(\pi) \log |pi|.$$

Regularized space measures were considered in [10] and [60].[1] The latter paper also contains the suggestion that the "right" level of precision when comparing measures of this kind is up to polynomial and $\log n$ factors, where n is the number of variables of the formula being refuted. We will say that, for complexity measures μ_1, μ_2, μ_1 *simulates* μ_2, and write $\mu_1 \preceq \mu_2$, if

$$\mu_1(F \vdash \bot) \leq (\mu_2(F \vdash \bot) \log n)^{O(1)}$$

for any CNF F in n variables. We call μ_1 and μ_2 *equivalent*, and write $\mu_1 \approx \mu_2$, if $\mu_1 \preceq \mu_2$ and $\mu_2 \preceq \mu_1$. Clearly $\preceq$ is transitive, and this implies that $\approx$ is an equivalence relation and $\preceq$ imposes a partial order on its equivalence classes.

This notion of simulation brings structure to the framework of comparing proof complexity measures that is faithful to both previous research and open problems. Regarding the choice to include $\log n$ factors in $\preceq$, $\log n$ factors appear naturally e.g. in simulations involving regularized space measures, so that we wouldn't want to consider for example a separation of $O(1)$ vs. $\Omega(\log n)$ involving such measures to be a true separation. We also note that this aligns well with standard practice in *computational* complexity, where in most major results about space, a $\log n$ lower bound on it is included as an assumption.

It might be tempting to think that for any space measure μ, it is always the case that $\mu(F \vdash \bot) \approx \mu^*(F \vdash \bot)$, as the regularization factor is just a logarithm. It could be however that restricting μ must necessarily give rise to refutations π of F of length $|\pi| \gg \exp(\mu(F \vdash \bot) \log n)$. We call trade-offs of this kind *strong*, and should be contrasted with trade-off

1. [60] uses the term "amortized"; we use, as in [54], them perhaps more appropriate term "regularized" instead.

results in e.g. [13, 8]. We will see that such trade-offs do not exist for Σ_2 space. We consider the question of whether strong size-space trade-offs exist for clause, monomial or variable space to be one of the most important open questions raised by this work.

3.2.2 Regularized clause and monomial space and tree-like resolution size

We show in this section that regularized clause and monomial space, tree-like clause space (i.e. clause space in tree-like resolution), and regular clause space (i.e. clause space in regular resolution) are all equivalent, and equivalent to the logarithm of tree-like resolution size. That is, we have that for any unsatisfiable CNF formula,

$$\begin{aligned}\mathrm{CSpace}^*(F \vdash \bot) \approx \mathrm{MSpace}^*(F \vdash \bot) \approx \mathrm{TCSpace}(F \vdash \bot)\\ \approx \mathrm{RCspace}(F \vdash \bot) \approx \log S_T(F \vdash \bot),\end{aligned}$$

where TCSpace and RCSpace stand for tree-like clause space and regular clause space respectively.

Before going to the theorems, let us fix some notation. For a restriction ρ and a formula F, we write $\rho \models F$ if every total extension of ρ satisfies F or, in other words, if $F|_\rho$ is *semantically* equal to 1. For a set of formulas S, $\rho \models S$ means $\rho \models \bigwedge_{F \in S} F$, and for two sets of formulas S and T, we write $S \models T$ if all total assignments satisfying every formula in S also satisfy every formula in T. For a clause C, we denote by ρ_C the minimal restriction such that $\rho_C \models \overline{C}$. Dually, for a partial assignment ρ let C_ρ be the maximal clause with the property $\rho \models \neg C_\rho$.

Theorem 3.2.1. *For any unsatisfiable CNF formula F over n variables,*

$$\log S_{TR}(F \vdash \bot) \leq 2\mathrm{MSpace}^*(F \vdash \bot)\log(n+1),$$
$$\mathrm{TCSpace}(F \vdash \bot) \leq 2\left(\mathrm{MSpace}^*(F \vdash \bot) + 1\right).$$

Proof. The proof is analogous to the construction in [60] showing that depth is upper bounded by regularized variable space. Let $\mathcal{M}_1, \ldots, \mathcal{M}_t$ be a refutation of F in configurational form, of monomial space s. We show, by induction on d, that for every interval $[i..j] \subseteq [1..t]$ with $j > i$, $j - i \leq 2^d$, and for every clause D such that $\rho_D \models \mathcal{M}_i$ and $\rho_D \models \neg\mathcal{M}_j$, it holds that

$$S_{TR}(F \vdash D) \leq (n+1)^{ds}$$

and, moreover, the assumed tree-like resolution proof can be carried out in clause space at most $ds + 2$. The theorem follows by taking $[i..j] \coloneqq [1..t]$, $d \coloneqq \lceil \log t \rceil$ and $D \coloneqq \bot$.

Suppose that $d = 0$, so that $j = i + 1$. The statement is vacuously true except when the step consists in downloading an axiom C from F, simply because in all other cases we have $\mathcal{M}_i \models \mathcal{M}_{i+1}$ and hence D with the specified properties does not even exist. Let D be a clause for which $\rho_D \models \mathcal{M}_i$ and $\rho_D \models \neg(\mathcal{M}_i \cup \{C\})$. Then we necessarily must have $\rho_D \models \neg C$, which is equivalent to saying that D is a weakening of C.

For the inductive step, suppose that $d > 0$, let $[i..j] \subseteq [1..t]$ be any interval with $j - i \leq 2^d$, $j > i+1$, and let D be a clause such that $\rho_D \models \mathcal{M}_i$ and $\rho_D \models \neg\mathcal{M}_j$. Set $k \coloneqq i + \lceil (j-i)/2 \rceil$, so that $k - i \leq 2^{d-1}$ and $j - k \leq 2^{d-1}$. Let the list $m_1, \ldots, m_s$ contain all monomials occurring in $\mathcal{M}_k$. For a clause A and a monomial $m = \ell_1 \ldots \ell_r$, consider the tree shown in Figure 3.1 designating a derivation of A in resolution; it is obtained from the obvious decision tree

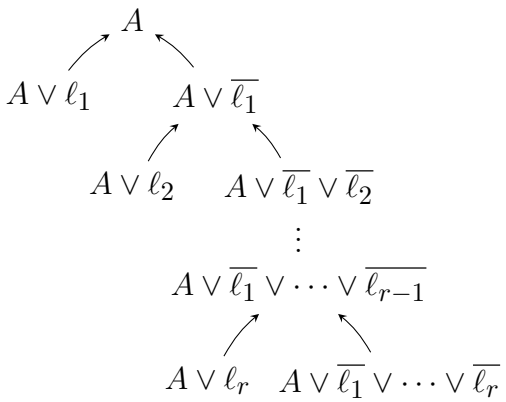


Figure 3.1: The tree $\mathbf{T}_{A;m}$

deciding m by reversing edges and weakening the result by A. Denote this tree by $\mathbf{T}_{A;m}$.

Let us now describe the required tree-like resolution proof of D. Start with $\mathbf{T}_{D;m_1}$. To every leaf of $\mathbf{T}_{D;m_1}$ labelled by a clause D', append the tree $\mathbf{T}_{D';m_2}$. Continue this process for all $m_1, \ldots, m_s$. If at any point during this construction, a forbidden disjunction containing a variable and its negation occurs, then we delete that node and merge its sibling with its parent. The resulting tree $\mathbf{T}$ has at most $(n+1)^s$ leaves, and each of its leaves is labelled by a clause E such that $\rho_E \models \mathcal{M}_k$ or $\rho_E \models \neg\mathcal{M}_k$. From the induction hypothesis, there are tree-like resolution proofs of all those clauses from F, of size $(n+1)^{(d-1)s}$. Therefore, there is a tree-like resolution proof of D from F of size $(n+1)^{ds}$.

To see that this proof can be carried out in clause space at most $ds+2$, notice that the proof designated by $\mathbf{T}$ can be carried out in clause space $s+2$. Proceed with this proof, and whenever a clause at its leaves is downloaded, keep all current clauses in memory (there are at most s of them—the maximum clause space is hit when the parent of two leaves is brought to memory), and derive it in clause space at most $(d-1)s+2$. The fact that such a derivation exists is guaranteed by the induction hypothesis. The resulting proof has clause space at most $s+(d-1)s+2 = ds+2$. □

Remark 3.2.1. Let us note that the above construction works for any *sound* system whose configurations are Boolean functions of monomials. In particular, it works for the purely semantic system called functional calculus [1]. Even more generally, it works for any proof system in which small configurations have low decision tree complexity.

For the rest of the relations, we claim that for any unsatisfiable CNF formula F over n

variables, we have

$$\begin{aligned}
\mathrm{RCSpace}(F \vdash \bot) &\leq \mathrm{TCSpace}(F \vdash \bot) \\
&\leq \log S_T(F \vdash \bot) + 2 \\
&\leq 2\left(\mathrm{MSpace}^*(F \vdash \bot)\log(n+1) + 1\right) \\
&\leq 2\left(\mathrm{CSpace}^*(F \vdash \bot)\log(n+1) + 1\right) \\
&\leq 2\left(\mathrm{RCSpace}^*(F \vdash \bot)\log(n+1) + 1\right) \\
&\leq 2\left(\left(\mathrm{RCSpace}(F \vdash \bot)\right)^2 \log(n+1)\log(2n) + 1\right).
\end{aligned}$$

The first inequality follows from the observation that every tree-like refutation can be pruned to the regular form, and this operation does not increase its space. The second inequality is [29, Theorem 2.1], and the third is Theorem 3.2.1. The fourth and the fifth inequalities are obvious. Finally, the last inequality follows from [29, Corollary 4.2]. Namely, Esteban and Torán show that if π is a resolution refutation, in configurational form, of clause space s and depth d, then

$$|\pi| \leq \binom{d+s}{s}. \tag{3.2}$$

Taking π to be a regular resolution refutation of minimum clause space, we get, since a regular refutation must have depth at most n,

$$\mathrm{RCSpace}^*(F \vdash \bot) \leq \left(\mathrm{RCSpace}(F \vdash \bot)\right)^2 \log(2n).$$

As a byproduct, we get that TCSpace $\approx$ RCSpace. This comes in sharp contrast with the situation for size, where there is an exponential separation between tree-like and regular resolution [11].

We also see from (3.2) that, somewhat surprisingly, instead of regularizing clause space

by multiplying it by the logarithm of *size*, we could have as well used a much weaker regularization multiplying by the logarithm (!) of depth, and the resulting measure would still be in this cluster. This allows us to rephrase the main open problem of whether $\mathrm{CSpace}(F \vdash \bot) \approx \mathrm{CSpace}^*(F \vdash \bot)$ as whether there exists a *super-critical* (in the sense of [59]) trade-off between clause space and depth, that is, whether restricting clause space must necessarily result in proofs of depth $\gg n$.

The remaining simulation in this cluster of complexity measures involves the positive depth measure, denoted by D_P, which was introduced in Section 2.5, is:

Theorem 3.2.2. *For any unsatisfiable CNF formula F,*

$$\mathrm{TCSpace}(F \vdash \bot) \leq D_P(F \vdash \bot) + 2.$$

Proof. The argument is a refinement of the argument in [29] showing that tree-like clause space is bounded by depth. We show, by induction on $\mathbf{T}$, that if $\mathbf{T}$ is a tree-like resolution proof of a clause E from F of positive depth d, then there is a tree-like resolution proof, in configurational form, of E from F of clause space at most $d + 2$.

If $\mathbf{T}$ has size at most 2, then $d \leq 1$, and $\mathrm{TCSpace}(F \vdash \bot) \leq 3$. Otherwise, let $\mathbf{T}_1$ and $\mathbf{T}_2$ be the sub-derivations of $\mathbf{T}$ deriving the two clauses E_1 and E_2 respectively from which E is derived via an application of the resolution rule and possibly applications of the weakening rule. One of $\mathbf{T}_1$ and $\mathbf{T}_2$, say $\mathbf{T}_1$, must have positive depth at most $d-1$. From the induction hypothesis, there is a tree-like proof π_1 of E_1 of clause space at most $d + 1$, and a tree-like proof π_2 of E_2 of clause space at most $d + 2$. Deriving first E_2 using π_2, and then, keeping E_2 in memory, deriving E_1 using π_1, we get a proof of E of clause space at most $d + 2$. □

3.2.3 Resolution width and Σ_2 space

Next we show the simulations in the bottom cluster of Figure 1.1, namely that for any unsatisfiable CNF formula,

$$W(F \vdash \bot) \approx \log S_{TR(\log)}(F \vdash \bot) \approx \Sigma_2\text{Space}(F \vdash \bot) \approx \Sigma_2\text{Space}^*(F \vdash \bot).$$

Both the results and their proofs will be cleaner if we consider the minimum width of resolutions refutations of F that also use rule (3.1):

$$\frac{C \vee \overline{\ell_1} \quad \cdots \quad C \vee \overline{\ell_r}}{C},$$

where $\ell_1 \vee \cdots \vee \ell_r$ is a clause of F, as well as axioms $x \vee \neg x$. As was already noted, this has the effect of eliminating the additive factor $W(F)$ that comes in simulations involving W_R. Apart from that, adding (3.1) to resolution, does not add extra power to the system. As shown in Figure 3.2, the resolution rule can efficiently simulate rule (3.1) (more generally, the cut rule can efficiently simulate any analytic LK rule). Moreover, it does so, increasing

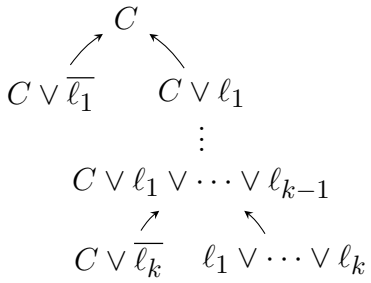


Figure 3.2: Resolution simulating rule (3.1)

size by a constant factor, and width by an additive factor of $W(F)$. In particular, we have

$$S_R(F \vdash \bot) = O\left(S_{\mathrm{LK}^-}(F \vdash \bot)\right) \tag{3.3}$$

$$W_R(F \vdash \bot) \leq W_{\mathrm{LK}^-}(F \vdash \bot) + W(F) - 1. \tag{3.4}$$

Now, for our simulations, we will need the following "locality" property of DNF resolution.

Lemma 3.2.1. *Let ρ be a restriction. For each of the inference rules of DNF resolution, if both premises contain a term satisfied by ρ, then ρ satisfies some term in the conclusion.*

The main theorem of this section says that as long as we transition from depth 1 Frege to depth 2 Frege, then not only width continues to be smaller than space, but in fact it becomes almost equal to it.

Theorem 3.2.3. *For any unsatisfiable CNF formula F,*

$$\frac{1}{5}\Sigma_2\mathrm{Space}(F \vdash \bot) \leq W(F \vdash \bot) \leq \Sigma_2\mathrm{Space}(F \vdash \bot).$$

Proof. Let $\mathcal{M}_1, \ldots, \mathcal{M}_t$ be a DNF resolution refutation of F, of Σ_2 space s. We will construct a sequence $\mathbf{T}_1, \ldots, \mathbf{T}_t$ of derivations in the system "resolution plus the rule (3.1)". The property we are going to maintain is that for every clause D labelling a leaf of $\mathbf{T}_i$, either D is a weakening of an axiom $x \vee \neg x$ (call such a leaf an *axiom leaf*), or the following hold:

1. for every $G \in \mathcal{M}_i$, ρ_D satisfies some term of G;
2. $W(D) \leq \Sigma_2\mathrm{Space}(\mathcal{M}_i)$.

$\mathbf{T}_1$ has one vertex labelled by the empty clause. Now suppose we have constructed $\mathbf{T}_{i-1}$ such that 1 and 2 hold for all non-axiom leaves. For every such leaf v labelled by a clause D, do the following.

- *Axiom Download*: Suppose that $\mathcal{M}_i = \mathcal{M}_{i-1} \cup \{C\}$, where $C = \ell_1 \vee \cdots \vee \ell_r$ is either a clause of F or an axiom $x \vee \overline{x}$. Then for every $j \in [r]$, add to v a child labelled by $D \vee \ell_j$.

- *Erasure*: Suppose that $\mathcal{M}_i \subseteq \mathcal{M}_{i-1}$. Add to v a single child labelled by a clause $E \subseteq D$ such that $W(E) \leq \Sigma_2\text{Space}(\mathcal{M}_i)$ and for every $G \in \mathcal{M}_i$, ρ_E satisfies some term of G.

- The case of an *inference* is immediately taken care of by Lemma 3.2.1, D does not change.

Since $\bot \in \mathcal{M}_t$, $\mathbf{T}_t$ cannot contain any non-axiom leaves and hence defines a refutation of F. Also, it is clear from the construction and property 2 above that any clause D appearing in it must satisfy

$$W(D) \leq \max_i \Sigma_2\text{Space}(\mathcal{M}_i) = s.$$

Hence $W(F \vdash \bot) \leq s$.

For the converse inequality, suppose that $C_1, \ldots, C_t$ is a refutation in the system "resolution plus the rule (3.1)", of width w. For every $i \in [t-1]$, set $G_i \coloneqq \bigvee_{j=1}^{i} \overline{C_j}$. Each G_i is a w-DNF. For our small space refutation, we start by deriving G_{t-1} and $G_{i-1} \vee C_i$ for each $i \in [t-1]$. Having derived these formulas, we can use a series of cuts to derive the empty clause: From $G_{t-1} = G_{t-2} \vee \overline{C_{t-1}}$ and $G_{t-2} \vee C_{t-1}$ we can derive G_{t-2} by cutting on C_{t-1}, then from G_{t-2} and $G_{t-3} \vee C_{t-2}$ we can derive G_{t-3}, and so on. Notice that G_{t-1} contains a tautology of size $O(w)$, and hence has a derivation whose complexity depends only on w; one can sees in particular that G_{t-1} has a tree-like proof of Σ_2 space at most $3w$. To derive $G_{i-1} \vee C_i$, notice that either C_i is a clause of F, in which case we can immediately derive $G_{i-1} \vee C_i$, or $G_{i-1} \vee C_i$ is a tautology. In the case $G_{i-1} \vee C_i$ is a tautology, then C_i will be the result of applying either the resolution rule or the weakening rule or rule (3.1) to some clauses among $C_1, \ldots, C_{i-1}$. In either case, it can be checked that $G_{i-1} \vee C_i$ has a tree-like

proof of Σ_2 space at most $3w$. We therefore see that that the overall refutation of F can be carried in Σ_2 space at most $5w$. □

Remark 3.2.2. Notice that the refutation constructed for the upper bound in Theorem 3.2.3 only uses the resolution rule when an axiom $x \vee \neg x$ is downloaded. In the absence of such axioms, the refutation is cut-free; in particular, we have

$$W_{\mathrm{LK}^-}(F \vdash \bot) \leq \mathrm{CSpace}(F \vdash \bot).$$

Remark 3.2.3. In the refutation of Σ_2 space at most $5w$ constructed in the second part of Theorem 3.2.3, there is a constant number of formulas in every configuration. This implies that *a posteriori*, DNF resolution will retain its power in terms of space even if we restrict the *formula space* (the maximum number of DNFs in a configuration) to a constant. This in turn immediately implies, also a posteriori, that we can balance our definition of Σ_2 space replacing in it $W(G_1) + \ldots + W(G_s)$ with $s \cdot \max(W(G_1), \ldots, W(G_s))$ (since s is a constant, these expressions differ by at most a constant multiplicative factor), and the resulting measure will still be equivalent to Σ_2 space. We are not aware of a direct proof of this simulation by-passing width.

We get from Theorem 3.2.3 that strong length-space trade-offs conjectured for variable, clause and monomial space (see [60] and [54]), are ruled out for DNF resolution. In particular, we get:

Corollary 3.2.1. *For any unsatisfiable CNF formula F with n variables,*

$$\Sigma_2\mathrm{Space}^*(F \vdash \bot) \leq O\left((\Sigma_2\mathrm{Space}(F \vdash \bot))^2 \log n\right).$$

Proof. Let $s \coloneqq \Sigma_2\mathrm{Space}(F \vdash \bot)$. By the first part of Theorem 3.2.3, F has a width $O(s)$ refutation. We apply to this refutation the construction from the second part of Theo-

rem 3.2.3 in which we can clearly assume $t \leq n^{O(s)}$ (since all clauses in the sequence can be assumed to be different). By an easy inspection, the length of the resulting refutation will still be $n^{O(s)}$. Therefore,

$$\Sigma_2\text{Space}^*(F \vdash \bot) \leq O(s^2 \log n). \qquad \square$$

Corollary 3.2.2. *If F has a constant Σ_2 space refutation, then it has a refutation of constant Σ_2 space and polynomial length.*

Proof. The refutation constructed in the proof of Corollary 3.2.1 will in our case also have constant Σ_2 space. $\square$

Let us finally deal with the remaining measure, size in tree-like $R(\log)$.

Theorem 3.2.4. *Let F be an unsatisfiable CNF formula over n variables. Then*

$$\Sigma_2\text{Space}(F \vdash \bot)^{1/2} \leq \log S_{T,R(\log)}(F \vdash \bot) \leq O(W(F \vdash \bot) \log n).$$

Proof. For the upper bound, let π be a resolution refutation of F of width $w := W(\vdash_F \bot)$. Apply to it the construction in the second part of the proof of Theorem 3.2.3 once again. By inspection (cf. the proof of Corollary 3.2.1), this refutation is tree-like, has size $n^{O(w)}$ and every term occurring in it has width at most w. Padding it with dummy formulas if necessary, we can assume that it has size $\geq 2^w$ which makes it into a tree-like $R(\log)$ refutation of the required size.

For the lower bound, the argument is an adaptation of the argument in [29] showing (1.3). Namely, by pebbling, a tree-like proof $\mathbf{T}$ of size $s > 1$ can be turned into a proof in configurational form, where each configuration contains at most $\log s$ formulas occurring in $\mathbf{T}$. If $\mathbf{T}$ is a refutation in $R(\log)$, then all terms occuring in $\mathbf{T}$ have width at most $\log s$, so the resulting refutation has Σ_2 space $(\log s)^2$. $\square$

Remark 3.2.4. Let us note that the equivalence of width and tree-like size in $R(\log)$ can be

seen as the case $d = 1.5$ of Theorem 2.3.1.

3.3 Lower bounds

3.3.1 Width lower bounds

Let F_n be a family of unsatisfiable CNF formulas which is uniform, in the sense that it is the propositional translation of some first order formula Φ. For example, if Φ is the conjunction of the formulas

$$\forall x \exists y\, R(x,y),$$
$$\forall x \forall y \forall z\, R(x,y) \wedge R(y,z) \rightarrow R(x,z),$$
$$\forall x\, \neg R(x,x),$$

then, restricting the range of quantifiers to $[n]$, replacing each $\forall x$ with $\bigwedge_{i\in[n]}$, each $\exists x$ with $\bigvee_{i\in[n]}$, and each $R(i,j)$ with the propositional variable R_{ij}, we get that F_n consists of the clauses

$$\bigvee_{j\in[n]} R_{ij}, \quad \text{for } i \in [n]$$
$$R_{ij} \wedge R_{jk} \rightarrow R_{ik}, \quad \text{for } i,j,k \in [n]$$
$$\neg R_{ii}, \quad \text{for } i \in [n].$$

In general, it might be that the propositional translations of Φ are not CNF formulas, but Φ can be always put in a form so that F_n is a CNF formula (see [62, 47]).

Now, a way to get width lower bounds for such formulas F_n is as follows: First, we find a model $\mathbf{M}$ in which Φ is satisfied. Notice that since all F_n are unsatisfiable, $\mathbf{M}$ has to be infinite. In the example above, all F_n are unsatisfiable since a finite graph every vertex of

which has an outgoing edge must contain a cycle, but $(\mathbb{N}, \leq)$ is a model in which Φ is satisfied. We then use $\mathbf{M}$ to construct a winning strategy for the adversary for the width game played on F_n. Such a strategy shows that a prover with limited width cannot distinguish between finite models in which Φ is falsified and $\mathbf{M}$; in other words the adversary can fool the prover into thinking that it is Φ over $\mathbf{M}$ what he's refuting.

Variations of the following theorem are stated for tree-like resolution size in [62] and for tree-like size in DNF resolution in [46, 47]. We state it for $W(F \vdash \bot)$.

Theorem 3.3.1. *If Φ is satisfied in an infinite model $\mathbf{M}$, then*

$$W(F_n \vdash \bot) > n/r,$$

where r is the maximum arity of a relation symbol occurring in Φ.

Proof. Assume for simplicity that Φ only has one relation symbol R of arity r. Let $\mathcal{N}_n$ be the collection of all tuples $(\mathbf{N}, \rho)$, where $\mathbf{N}$ is a model and ρ an assignment of truth values to some of the atomic formulas of Φ such that:

1. the domain N of $\mathbf{N}$ is a subset of $[n]$;
2. $\mathbf{N}$ is isomorphic to an induced submodel of $\mathbf{M}$;
3. it holds that

$$\rho(R(i_1, \ldots, i_r)) = 1 \implies i_1, \ldots, i_r \in N \ \ \& \ \ R^{\mathbf{N}}(i_1, \ldots, i_r),$$
$$\rho(R(i_1, \ldots, i_r)) = 0 \implies i_1, \ldots, i_r \in N \ \ \& \ \ \neg R^{\mathbf{N}}(i_1, \ldots, i_r).$$

We say that a tuple $(\mathbf{N}', \rho') \in \mathcal{N}_n$ extends $(\mathbf{N}, \rho) \in \mathcal{N}_n$, and write $(\mathbf{N}, \rho) \preceq (\mathbf{N}', \rho')$, if $\rho \subseteq \rho'$ and $\mathbf{N}$ is a submodel of $\mathbf{N}'$.

It is easy to see that for each $(\mathbf{N}, \rho) \in \mathcal{N}_n$,

1. if $|N| \leq n - r$ and C is a clause of F_n, then there is a tuple $(\mathbf{N}', \rho') \in \mathcal{N}_n$ such that $(\mathbf{N}, \rho) \preceq (\mathbf{N}', \rho')$ and ρ' satisfies C;

2. if $|N| < n$ and $u \in [n]$, then there is a tuple $(\mathbf{N}', \rho') \in \mathcal{N}_n$ such that $(\mathbf{N}, \rho) \preceq (\mathbf{N}', \rho')$ and $N' = N \cup \{u\}$

Now, for a tuple $p = (\mathbf{N}, \rho) \in \mathcal{N}_n$, we set S_p to be the set of literals such that $x \in S_p \iff \rho(x) = 1$ and $\neg x \in S_p \iff \rho(x) = 0$. The collection

$$\mathcal{C} \coloneqq \{S_p \mid p \in \mathcal{N}_n\}$$

satisfies that for each $S \in \mathcal{C}$,

1. S does not contain a variable and its negation;

2. $S' \subseteq S \implies S' \in \mathcal{C}$;

3. if $|S| \leq n/r - 1$ and C is a clause of F_n, then there is an $S' \supseteq S$ in $\mathcal{C}$ that contains a literal of S;

4. if $|S| \leq n/r - 1$, then for every variable x of F_n either $S \cup \{x\} \in \mathcal{C}$ or $S \cup \{\neg x\} \in \mathcal{C}$.

The theorem follows from Theorem 3.1.1. □

Theorem 3.3.1 can be complemented by noticing that in the case Φ is not satisfied by any model, then there is a first-order LK^- derivation of $\Phi \to$ of some height, say h. The propositional translations of this derivation will have height h as well, hence will also have width at most h, for every n. Therefore, in the case Φ is not satisfied by any model, then for every n there is an LK^- refutation of F_n of width $O(1)$.

Let us end this section by noting that although Theorem 3.3.1 can be used to show lower bounds on Σ_2 space or tree-like size in $R(\log)$, it cannot be used to show lower bounds on DAG-like resolution size. n here refers not to the number of variables of F_n but to the

size of the models' domain we are restricting on, so in general Theorem 3.3.1 can only give sub-linear in the number of variables lower bounds. A common way to get around this is consider suitable restrictions on F_n so that n becomes linear in the number of variables (see [14]).

3.3.2 Lower bounds for tree-like resolution sisze

We next turn to lower bounds on measures in the middle cluster of Figure 1.1. We will focus on lower bounds for tree-like resolution size. Of course one can get lower bounds for tree-like resolution size from width lower bounds, as

$$W_{\mathrm{LK}^-}(F \vdash \bot) \leq \mathrm{CSpace}(F \vdash \bot) \leq \log S_{TR}(F \vdash \bot).$$

We will be interested instead in lower bounds on $S_{TR}(F \vdash \bot)$ for formulas F that can be refuted in small width.

A prime example of such formulas are "lifted" versions of the Horn formulas we encountered in Section 2.5. Let us start with the following theorem characterizing Horn formulas as those CNF formulas that can be refuted using the smallest possible width, namely 1, and the smallest possible clause space, namely 1. Note that to be able to speak about width 1 refutations, we have to incorporate the weakening rule into rule (3.1). That is, we use rule (3.1) in the form:

$$\frac{D \vee \overline{\ell_1} \qquad \dots \qquad D \vee \overline{\ell_r}}{D \vee E}.$$

To be able to speak about clause space 1 refutations, we need to consider a slightly modified version of configurational proofs, by incorporating the three rules, axiom download, erasure and inference into one

Three-in-one rule: from a configuration $\mathcal{M}$, infer any configuration $\mathcal{M}^* \subseteq \mathcal{M} \cup F \cup \{C\}$,

where C is obtained from clauses in $\mathcal{M}, F$ via a single application of the resolution or weakening rule.

Theorem 3.3.2. *Let F be a CNF formula. The following are equivalent:*

1. *F contains an unsatisfiable CNF formula that results from a Horn formula by negating some of its variables;*
2. $\mathrm{CSpace}(F \vdash \bot) \leq 1$*;*
3. $W_{\mathrm{LK}^-}(F \vdash \bot) \leq 1$.

Proof. For $1 \implies 2$, we can without loss of generality assume that F itself is an unsatisfiable Horn formula. We show, by induction on n, that every such CNF formula F can be refuted in clause space 1. The base case is trivial. Now, suppose that $n > 0$, and let y be a variable such that F contains the clause $\to y$. Such a clause must exist, for if every clause contained a negated variable, then we could satisfy F by setting every variable to false. Setting $y \coloneqq 1$, we get an unsatisfiable Horn formula $F|_{y:=1}$ with $n-1$ variables. From the induction hypothesis, there is a clause space 1 refutation of $F|_{y=1}$. Weakening every clause in it by $\overline{y}$ gives us a space 1 proof of $\overline{y}$ from F. Now we only have to download y and infer $\bot$.

The implication $2 \implies 3$ is proven by an argument similar to the first part of the proof of Theorem 3.2.3. Namely, a clause space 1 refutation of minimum length in the three-in-one model must necessarily be a sequence $\{D_1\}, \ldots, \{D_t\}$, where D_{i+1} is obtained by resolving D_i with a clause C_i in F. We construct a sequence $\mathbf{T}_1, \ldots, \mathbf{T}_t$ of width 1 LK^- refutations, in which axioms $x \vee \neg x$ are erased, and such that leaves of $\mathbf{T}_i$ are labelled by literals among $\overline{\ell_{i,1}}, \ldots, \overline{\ell_{i,r_i}}$, where $D_i = \ell_{i,1} \vee \ldots \vee \ell_{i,r_i}$. To get $\mathbf{T}_{i+1}$ from $\mathbf{T}_i$, we add to every leaf v of $\mathbf{T}_i$ a child labelled by $\overline{\ell}$ for every literal ℓ of C_i that does not conflict with the label of v.

Finally, for $3 \implies 1$, we again proceed by induction on the number of variables n of F. The base case is trivial. Suppose that $n > 0$. The fact that there is a width 1 LK^- refutation of F, forces F to have a one literal clause (since the refutation must start somewhere), say

ℓ. Setting $\ell := 1$, we get a width 1 refutation of $F|_{\ell:=1}$. From the induction hypothesis, a sub-formula G of $F|_{\ell:=1}$ is unsatisfiable Horn up to negating some variables. Let $\widehat{G}$ be the corresponding sub-formula of F; $\widehat{G}$ is obtained from G by restoring $\overline{\ell}$ to some of its clauses. Then $\widehat{G} \wedge \ell$ is an unsatisfiable Horn sub-formula of F. □

Our tree-like size lower bounds will rely on lower bounds on the *pebbling* complexity of DAGs. Pebbling a DAG G means, starting with no pebbles on the vertices of G, place a pebble in some designated vertex of G with no outgoing edges, called the *sink* of G, following the rules:

1. a pebble can be placed on a vertex v, if all immediate predecessors of v have pebbles on them;

2. a pebble can be removed from a vertex at any time.

In particular, our overall strategy is as follows:

1. We start with a family of DAGs $\{G_n \mid n \in \mathbb{N}\}$ that have high pebbling complexity.

2. We associate with every G_n, the Horn formula H_{G_n} consisting of all clauses $S \to x$ for every non-sink vertex x of G whose immediate predecessors are the vertices of S, and the clause $\to x$ for the sink vertex x of G.

3. Using Theorem 2.5.1, lower bounds on the pebbling complexity of G_n translate to typically linear, or close to linear, lower bounds for the positive depth and size of a tree-like resolution proof.

4. These linear lower bounds are then "lifted" to exponential lower bounds for a lifted version $\widetilde{H}_{G_n}$ of H_{G_n}; in all applications below $\widetilde{H}_{G_n}$ will always be the formula, denoted by $H_{G_n}[\vee]$, which results from H_{G_n} substituting each of its variables by the or of two new variables.

The following theorem formulates the last step of the above strategy:

Theorem 3.3.3. *Let H be an unsatisfiable Horn formula on n variables. Suppose that every pebbling strategy that refutes H in s steps and using d pebbles, starting with no pebbles on its variables, satisfies*

$$(d-1)f(s) \geq g(n)$$

for non-decreasing positive functions f, g. Then

$$f(t)\log t \geq g(n),$$

where $t \coloneqq S_{TR}(H[\vee] \vdash 0)$.

Proof. Let $\mathbf{T}$ be a tree-like resolution refutation of $H[\vee]$ of size t, represented as in the proof of Theorem 2.5.1. That is, weakenings appear at leaves only, so that the positive depth of a clause $S \to y$ in $\mathbf{T}$ is exactly $|S|$.

Create a probability space of partial assignments by choosing independently for every variable x of H, which was substituted by $y \vee z$, one of y and z with probability $1/2$ and setting it to zero. Note that for any α from this space, $H[\vee]|_\alpha$ is identical to H up to renaming its variables and hence $\mathbf{T}|_\alpha$ is a refutation of H, again up to renaming variables. Let $D_1, \ldots, D_k$ be all clauses of positive depth at least $g(n)/f(t)$ occurring in $\mathbf{T}$, that is, all clauses that contain at least $g(n)/f(t)$ variables in the antecedent. We have that

$$P\left[\bigvee_{i=1}^{k}(D_i|_\alpha \neq 1)\right] \leq \sum_{i=1}^{k} P[D_i|_\alpha \neq 1] \leq k2^{-g(n)/f(t)} \leq t2^{-g(n)/f(t)}.$$

If $f(t)\log t < g(n)$, then the above probability is smaller than 1, which means that there is a point α in our sample space such that $\mathbf{T}|_\alpha$ is a tree-like resolution refutation of size at most t and positive depth $\leq g(n)/f(t)$. This, from Theorem 2.5.1, gives a pebbling strategy that refutes H in t steps using d pebbles, where $(d-1)f(t) < g(n)$. □

Plugging into Theorem 3.3.3 various DAGs from the literature with known bounds on their pebbling complexity and various functions f, we can get several corollaries. The first is a simplified proof of the separation by Ben-Sasson et al.

Corollary 3.3.1 [11]**.** *For every n, there is a formula F of size $O(n)$ that has DAG-like resolution refutations of size $O(n)$, and such that every tree-like resolution refutation of F requires size* $\exp(\Omega(n/\log n))$.

Proof. First notice that a constant width and linear size refutation of an unsatisfiable Horn formula always exists from Theorem 3.3.2, and such a refutation remains of constant width and linear size after substituting each x_i with $y_i \vee z_i$. Hence there is always a linear size refutation of $H[\vee_2]$. The lower bound follows by setting $f \coloneqq 1$ in Theorem 3.3.3, and using the graphs of [55] having constant in-degree and requiring $g(n) = \Omega(n/\log n)$ pebbles to pebble. □

Using Theorem 3.2.1, the formulas of Corollary 3.3.1 also give:

Corollary 3.3.2. *For every $n \geq 0$, there is a formula F of size $\Theta(n)$ that has a resolution refutation of size $O(n)$, width $O(1)$, and such that* $\mathrm{MSpace}^*(F \vdash \bot) \geq \Omega\left(n/\log n\right)$.

Proof. Corollary 3.3.1 demonstrates the existence of constant width CNFs having resolution refutations of size $O(n)$, width $O(1)$, and such that for every such CNF F, $\log S_{T,R}(F \vdash \bot) \geq \Omega(n/\log n)$. In fact, [11] shows that $\Omega(n/\log n)$ is also the lower bound on the number of points the Delayer can score in the Prover-Delayer game of [58] played on F. Now, it is proved in [30] that this number of points is precisely equal to $\mathrm{TCSpace}(F \vdash \bot)$ and then the result immediately follows from the second inequality in Theorem 3.2.1. □

Corollary 3.3.2 is an improvement of the lower bound

$$\mathrm{MSpace}^*(F \vdash \bot) \geq n^{1/2}/(\log n)^{O(1)}.$$

from [42, 40].

Corollary 3.3.1 is the strongest lower bound we can get on tree-like resolution size using Theorem 3.3.3. A drawback is that the formulas in it have large clause space as well. Theorem 3.3.3 can be used to give weaker lower bounds on tree-like resolution size, but for formulas of varying clause space. The next result was promised in the introduction. It should be compared with Theorem 3.3.2.

Theorem 3.3.4. *For every n, there is a formula F of size $O(n)$ that has tree-like resolution refutations of clause space 4, and such that every tree-like resolution refutation of F has size $\Omega(n^2/\log n)$.*

Proof. The lower bound follows by setting $f(t) := t$ in Theorem 3.3.3, and using the graphs of [49, Theorem 2.3.2] having linear size and exhibiting a $ds \geq g(n) = \Omega(n^2)$ time-space trade-off. These graphs can be pebbled using 3 pebbles, and that immediately gives that $\mathrm{CSpace}(\mathrm{Peb}_{G_n}[\vee_2] \vdash \bot) \leq O(1)$. By being more careful however, we can bring the space down to the minimum possible value for which a super-linear lower bound on tree-like resolution size is possible, which is 4 by Theorem 3.3.2.

More precisely, the above graphs have the following form. They contain two directed vertex-disjoint paths, let us call them U and L, and there are additional edges from vertices of U to vertices of L (see Figure 3.3). Moreover, there are no two vertices in U both with edges to the same vertex in L.

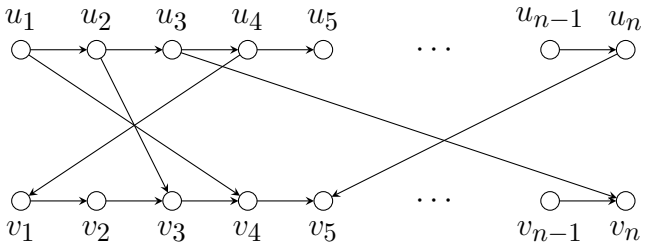


Figure 3.3: The form of the graphs giving the trade-off in Theorem 3.3.4

Let G be such a graph, and let $H \stackrel{\text{def}}{=} \mathrm{Peb}_G$ be the corresponding Horn formula. Call the

variables of the path U, $u_1, \ldots, u_n$ and the variables of L, $v_1, \ldots, v_n$ as in Figure 3.3, and suppose that to obtain $H[\vee_2]$, u_i is substituted by $x_i \vee y_i$ and v_i by $z_i \vee w_i$. We first note that any clause $x_i \vee y_i$ can be derived in clause space 3.

Indeed, after assigning $x_i \coloneqq 0,\ y_i \coloneqq 0$ in $H[\vee_2]$, we will get an unsatisfiable formula F such that if we additionally negate all its variables, the result will be Horn. Hence, by Theorem 3.3.2, $\mathrm{CSpace}(F \vdash \bot) \leq 3$. To get the desired derivation, we simply weaken all clauses in this refutation by appending $x_i \vee y_i$.

Notice that the derivations provided by Theorem 3.3.2 are tree-like. To show the bound $\mathrm{CSpace}(H[\vee_2] \vdash \bot) \leq 4$, we need to show, having derived $z_{i-1} \vee w_{i-1}$, how to derive $z_i \vee w_i$. Suppose that $\overline{u_j} \vee \overline{v_{i-1}} \vee v_i$ is a clause of H, so that $\overline{x_j} \vee \overline{z_{i-1}} \vee z_i \vee w_i$, $\overline{x_j} \vee \overline{w_{i-1}} \vee z_i \vee w_i$, $\overline{y_j} \vee \overline{z_{i-1}} \vee z_i \vee w_i$ and $\overline{y_j} \vee \overline{w_{i-1}} \vee z_i \vee w_i$ are clauses of $H[\vee_2]$. Notice that

$$\mathrm{CSpace}\left(\begin{array}{c}(z_{i-1} \vee w_{i-1}) \wedge \\ (x_j \vee y_j) \wedge \\ (\overline{x_j} \vee \overline{z_{i-1}} \vee z_i \vee w_i) \wedge \\ (\overline{y_j} \vee \overline{z_{i-1}} \vee z_i \vee w_i)\end{array} \vdash\ w_{i-1} \vee z_i \vee w_i\right) \leq 4.$$

In this derivation, all premises are immediately removed from memory after they are used as premises in an inference. Similarly, we have

$$\mathrm{CSpace}\left(\begin{array}{c}(w_{i-1} \vee z_i \vee w_i) \wedge \\ (x_j \vee y_j) \wedge \\ (\overline{x_j} \vee \overline{w_{i-1}} \vee z_i \vee w_i) \wedge \\ (\overline{y_j} \vee \overline{w_{i-1}} \vee z_i \vee w_i)\end{array} \vdash\ z_i \vee w_i\right) \leq 4.$$

Running the first derivation, deleting everything from memory except $w_{i-1} \vee z_i \vee w_i$ and then running the second derivation, deriving $x_j \vee y_j$ in clause space 3 whenever it is downloaded in these derivations, we get the desired clause space 4 derivation of $z_i \vee w_i$. $\square$

By using the construction from [49, Theorem 4.2.6], Theorem 3.3.4 can be further generalized to a lower bound $(n/\log n)^{\Omega(k)}$ on the tree-like resolution size of refuting formulas with clause space k. Let us further notice that the fact that the space 4 refutation in Theorem 3.3.4 is tree-like might be interesting, as typically tree-like resolution size lower bounds have been proven in the literature based on the prover-delayer game of [58], which also gives a lower bound for the clause space of tree-like resolution refutations (cf. Theorem 3.3.2).

CHAPTER 4

CUT-FREE SEQUENT CALCULUS AND RESOLUTION

Our aim in this chapter is to show that LK$^-$ for refuting CNF formulas cannot polynomially simulate resolution . To put it in different words, adding atomic cuts in cut-free sequent calculus can make proofs super-polynomially shorter. This answers a question posed by Cook and Reckhow [24]. We start with revisiting the notions of resolution width and LK$^-$ width for refuting CNF formulas, and notice that the latter can be characterized in terms of the concept of dynamic satisfiability introduced by Esteban, Galesi and Messner [28] as a tool for proving space lower bounds in resolution and Res(k). Using this characterization, we show a quadratic gap between LK$^-$ and resolution width, and utilize this gap to show a super-polynomial gap between LK$^-$ and resolution size. We further notice that our basic construction extends to stronger versions of dynamic satisfiability used to prove monomial space lower bounds in polynomial calculus [17, 18], allowing us to make progress towards separating resolution width from monomial space. In particular, we show a quadratic separation between resolution width and monomial space.

4.1 Resolution and LK$^-$ width

Recall that an LK$^-$ refutation of a CNF formula is a sequence of clauses ending with the empty clause and such that every clause in the sequence is either an axiom $x \vee \neg x$, or results from previous clauses via the rule

$$\frac{C \vee \overline{\ell_1} \quad \cdots \quad C \vee \overline{\ell_r}}{C},$$

where $\ell_1 \vee \cdots \vee \ell_r$ is a clause of F.

The definition of an analytic k-consistency property (Definition 3.1.1) for refutations of

this form, becomes, if we identify a set of literals with the minimal restriction that satisfies it:

Definition 4.1.1 [28]**.** Let F be a CNF formula, and let k be a natural number. F is said to be *k-dynamically satisfiable* is there is a non-empty set $\mathcal{R}$ of restrictions such that for every restriction $\rho \in \mathcal{R}$,

1. $\rho' \subseteq \rho \implies \rho' \in \mathcal{R}$;
2. if $|\mathrm{dom}(\rho)| < k$ and C is a clause of F, then there is a $\rho' \supseteq \rho$ in $\mathcal{R}$ that satisfies C.

In addition, Theorem 2.1.1 becomes:

Theorem 4.1.1. *A CNF formula F is k-dynamically satisfiable if and only if* $W_{\mathrm{LK}^-}(F \vdash \bot) > k$.

Similarly, the definition of a synthetic k-consistency property with respect to propositional variables, becomes:

Definition 4.1.2. Let F be a CNF formula, and let k be a natural number. We call F *atomically k-consistent* is there is a non-empty set $\mathcal{R}$ of restrictions such that for every restriction $\rho \in \mathcal{R}$,

1. ρ does not falsify any clause of F;
2. $\rho' \subseteq \rho \implies \rho' \in \mathcal{R}$;
3. if $|\mathrm{dom}(\rho)| < k$ and x is a variable of F, then there is a $\rho' \supseteq \rho$ in $\mathcal{R}$ with $x \in \mathrm{dom}(\rho)$.

Here condition 1 corresponds to the fact that we consider derivations $C_1, \ldots, C_m \vdash \bot$ from the set of clauses of F as premises. Theorem 2.1.1 becomes in this case the characterization of resolution width of Atserias and Dalmau [5]:

Theorem 4.1.2 [5]**.** *A CNF formula F is atomically k-consistent if and only if* $W_R(F \vdash \bot) > k - 1$.

In terms of games, in the game corresponding to LK^- width, the prover chooses in each round a clause of F, and the adversary responds by choosing a literal in that clause, which adds to the current restriction. The closure under subsets condition corresponds to the ability of the prover to delete at any round literals from the current restriction. The prover wins once the current restriction falsifies a clause of F.

In the game corresponding to resolution width, the prover chooses in each round a variable x of F. Then the adversary responds by either choosing x, in which case the current restriction is updated by adding $x \coloneqq 1$, or she chooses $\neg x$, and the current restriction is updated by adding $x \coloneqq 0$. Again, the prover may at any point delete literals from the current restriction, and the prover wins once the current restriction falsifies a clause of F.

We have already seen that, provided $W(F)$ is small, the prover in the game corresponding to resolution width is more powerful. Namely, we have

$$W_R(F \vdash \bot) \leq W_{\mathrm{LK}^-}(F \vdash \bot) + W(F) - 1. \tag{4.1}$$

In terms of games, the argument goes as follows: When the prover in the game corresponding to LK^- width selects a clause C, the prover in the game corresponding to resolution can start selecting, one by one the variables of C. If the game does not end, then the current restriction satisfies C, and then the prover can delete literals to match the restrictions in the two games.

We next deal with the question of how much more powerful the prover in the game corresponding to resolution width is. We will show she is strictly more powerful, by demonstrating a quadratic gap between $W_R(F \vdash \bot)$ and $W_{\mathrm{LK}^-}(F \vdash \bot)$.

4.2 A quadratic gap between resolution and LK$^-$ width

Let $F = \bigwedge_{i=1}^{s} C_i$ and $G = \bigwedge_{i=1}^{t} D_i$ be unsatisfiable CNF formulas. We define

$$F \times G \stackrel{\text{def}}{=} \bigwedge_{i=1}^{s} \bigwedge_{j=1}^{t} (C_i \vee D_j).$$

Notice that $F \times G$ is the CNF expansion of the formula $F \vee G$, which is also unsatisfiable.

Remarkably, LK$^-$ width and resolution width exhibit a different behavior with respect to this construction. This disparity ultimately relies on the fact that the cut rule gives us the ability to combine given proofs into a more complicated proof.

On one hand, we have:

Lemma 4.2.1. *If F and G are over disjoint sets of variables, then*

$$W_{LK^-}(F \times G \vdash \bot) \geq W_{LK^-}(F \vdash \bot) + W_{LK^-}(G \vdash \bot) - 1.$$

Proof. Suppose that F is k-dynamically satisfiable, G is ℓ-dynamically satisfiable, and let $\mathcal{A}$ and $\mathcal{B}$ respectively be sets witnessing this. We need to show that $F \times G$ is $(k+\ell)$-dynamically satisfiable, that is we need to find a set satisfying the conditions of Definition 4.1.1 for the parameter $k + \ell$. We claim that

$$\mathcal{C} \coloneqq \{\alpha \cup \beta \mid \alpha \in \mathcal{A} \ \ \& \ \ \beta \in \mathcal{B}\}$$

is such a set. Closure under subsets immediately follows from the fact that $\mathcal{A}$ and $\mathcal{B}$ are closed under subsets. For the second condition, suppose that $\gamma \in \mathcal{C}$, $|\gamma| < k + \ell$, and let $C_i \vee D_j$ be a clause of $F \times G$, where C_i is a clause of F and D_j is a clause of G. Since $\gamma \in \mathcal{C}$, there is an $\alpha \in \mathcal{A}$ and a $\beta \in \mathcal{B}$ such that $\gamma = \alpha \cup \beta$. Moreover, since $|\gamma| < k$ and F and G do not share variables, either $|\alpha| < k$ or $|\beta| < \ell$. In the first case, there is an $\alpha' \supseteq \alpha$ in $\mathcal{A}$

satisfying C_i, and thus $\rho' \cup \beta$ is a restriction in $\mathcal{C}$ satisfying $C_i \vee D_j$. In the second case, there is a $\beta' \supseteq \beta$ in $\mathcal{B}$ satisfying D_j, and thus $\alpha \cup \beta'$ is a restriction in $\mathcal{C}$ satisfying $C_i \vee D_j$. □

For resolution on the other hand, we have:

Lemma 4.2.2. $W_R(F \times G \vdash \bot) \leq \max\{W_R(F \vdash \bot) + W(G), W_R(G \vdash \bot)\}$.

Proof. Let π and ρ be resolution refutations of F and G respectively, both of minimum width. Replacing every clause C in π with $C \vee D_i$ we get a resolution proof π_i of D_i from $F \times G$. π_i has width at most $W_R(F \vdash \bot) + W(F)$. Replacing then every clause D_i in ρ with π_i we get a resolution refutation of $F \times G$ with the stated width. □

Choosing an appropriate seed and iterating, we get our result.

Theorem 4.2.1. *There are CNF formulas G with n^2 variables, size $O(n)^n$, and such that $W_R(G \vdash \bot) = O(n)$ and $W_{\mathrm{LK}^-}(G \vdash \bot) = \Omega(n^2)$.*

Proof. Let F be a CNF formula with n variables, width $O(1)$, size $\Theta(n)$, and such that $W_R(F \vdash \bot) = \Theta(n)$. Such formulas exist from e.g. [14]. Consider the formula $F^n \coloneqq F_1 \times \cdots \times F_n$, where the F_i's are copies of F over mutually disjoint sets of variables. From Lemma 4.2.2, $W_R(F^n \vdash \bot) = O(n)$. On the other hand $W_{\mathrm{LK}^-}(F \vdash \bot) = \Omega(n)$ from (4.1), and hence from Lemma 4.2.1, $W_{\mathrm{LK}^-}(F^n \vdash \bot) = \Omega(n^2)$. □

4.3 Separating resolution width from monomial space

The gap shown in the previous section can be extended to stronger versions of dynamic satisfiability that have been used to show monomial space lower bounds, thus showing a gap between resolution width and monomial space. The configurations in those are not restrictions as in Definition 4.1.1, but sets of restrictions. They will not be arbitrary sets however; they will have a certain structure. Namely, we call a set H of restrictions *admissible*,

if it is of the form

$$H = H_1 \times \cdots \times H_r \stackrel{\text{def}}{=} \{\rho_1 \cup \cdots \cup \rho_r \mid \rho_i \in H_i\},$$

where each H_i is a non-empty set of non-empty restrictions, for any two restrictions $\rho_i \in H_i$ and $\rho \in H_j$ for $i \neq j$, the domains of ρ_i and ρ_j do not intersect, and moreover, if a restriction $\rho \in H_i$ gives the value ϵ to a variable x, then there is also a restriction $\rho' \in H_i$ giving to x the value $1 - \epsilon$. The H_i's are called the *factors* of H; we write $\|H\|$ for their number. We write $H' \sqsubseteq H$ if every factor of H' is a factor of H.

Definition 4.3.1 [17, 18]**.** Let F be a CNF formula and let k be a natural number. We say that F is *k-extendible* if there is a non-empty set of admissible configurations $\mathcal{H}$ such that for each $H \in \mathcal{H}$,

1. if $H' \sqsubseteq H$, then $H' \in \mathcal{H}$;
2. if $\|H\| < k$ and C is a clause of F, then there is an $H' \sqsupseteq H$ in $\mathcal{H}$, such that every $\rho \in H'$ satisfies C.

Theorem 4.3.1 [17, 18]**.** *If F is k-extendible, then* $\mathrm{MSpace}(F \vdash \bot) \geq \lfloor k/4 \rfloor$.

Lemma 4.2.1 with the same proof applies here as well.

Lemma 4.3.1. *Let F and G be CNF formulas over disjoint sets of variables. If F is k-extendible and G is ℓ-extendible, then $F \times G$ is $(k + \ell)$-extendible.*

Proof. Let $\mathcal{H}$ and $\mathcal{I}$ be sets of admissible configurations witnessing the k and ℓ-extendibility of F and G. Since F and G are over disjoint sets of variables, we may assume that the domains for any of two restrictions in $\mathcal{H}$ and $\mathcal{I}$ do not intersect. Set

$$\mathcal{J} \coloneqq \{H \times I \mid H \in \mathcal{H} \ \ \& \ \ I \in \mathcal{I}\}.$$

Clearly, $\mathcal{J}$ is a set of admissible configurations. We claim that it satisfies the conditions of Definition 4.3.1 for the parameter $k+\ell$. Closure under $\sqsubseteq$ immediately follows from the fact that $\mathcal{H}$ and $\mathcal{I}$ are closed under $\sqsubseteq$. For the second condition, suppose that $J = H \times I \in \mathcal{J}$, $\|J\| < k+\ell$, and let $C_i \vee D_j$ be a clause of $F \times G$, where C_i is a clause of F and D_j is a clause of G. Since $\|J\| < k+\ell$, either $\|H\| < k$ or $\|I\| < \ell$. In the first case, there is an $H' \sqsupseteq H$ in $\mathcal{H}$ such that all restrictions in H' satisfy C_i. Then $H' \times I$ is an admissible configuration in $\mathcal{J}$ such that all restrictions in it satisfy $C_i \vee D_j$. The second case is analogous. □

Therefore, we get:

Theorem 4.3.2. *There are CNF formulas G with n^2 variables, size $O(n)^n$, and such that $W_R(G \vdash \bot) = O(n)$ and $\mathrm{MSpace}(G \vdash \bot) = \Omega(n^2)$.*

Proof. Again, let F be a CNF formula with n variables, width $O(1)$ and size $\Theta(n)$, that is $\Omega(n)$-extendible. Such formulas exist, see [17, 31, 18, 15]. The formulas $F^n \coloneqq F_1 \times \cdots \times F_n$, where the F_i's are copies of F over mutually disjoint sets of variables, have resolution width $O(n)$, and from Lemma 4.3.1 they are $\Omega(n^2)$-extendible, thus from Theorem 4.3.1 require $\Omega(n^2)$ monomial space. □

4.4 A super-polynomial separation between resolution and LK$^-$ size

Theorem 4.4.1. *There is a CNF formula F with n^2 variables and size $O(n)^n$, such that $S_R(F \vdash \bot) = O(n)^n$ but $S_{\mathrm{LK}^-}(F \vdash \bot) \geq \exp(\Omega(n^2))$.*

Proof. The formulas of Theorem 4.2.1 are such. The upper bound $S_R(F \vdash \bot) = O(n)^n$ follows from the construction of Lemma 4.2.2. The lower bound follows from the fact that $W_{\mathrm{LK}^-}(F \vdash \bot) = \Omega(n^2)$ and Theorem 3.1.5. Namely, Theorem 3.1.5 gives

$$S_{\mathrm{LK}^-}(F \vdash \bot) \geq \exp\left(\Omega\left(\frac{(W_{\mathrm{LK}^-}(F \vdash \bot))^2}{n^2}\right)\right) = \exp\left(\Omega\left(n^2\right)\right). \qquad \square$$

It is important to note that the $\exp\left(\Omega\left(n^2\right)\right)$ lower bound in Theorem 4.4.1 holds for the version of LK^- operating on clauses, where the clauses of the CNF formula F to be refuted are viewed as disjunctions of unbounded arity. It does not hold when the clauses of F are made up from binary disjunctions and moreover we are free to choose the order in which they are applied. If $\vee$ in the definition of of $F \times G$, is seen as a binary disjunction, then having derived $C_1, \ldots, C_n \rightarrow$ and $D_1, \ldots, D_t \rightarrow$, it is easy to see that we may derive from these sequents $C_1 \vee D_1, \ldots, C_1 \vee D_t, \ldots, C_s \vee D_1, \ldots, C_s \vee D_t \rightarrow$ in $s \cdot t$ steps, and in this case F^n in Theorem 4.4.1 has an LK^- refutation of size $n^{O(n)}$. An analogous situation occurs between the tree-like versions of LK^- and resolution [4]. But let us notice, concluding, that with binary disjunctions, LK^- cannot be seen as a system operating on clauses, and it becomes rather unnatural to compare it with resolution — it is not even clear, in this case, whether resolution can polynomially simulate LK^-. LK^- for clauses consisting of binary disjunctions is closer to resolution with limited extension, in which case resolution does polynomially simulate it [66].

CHAPTER 5

AUTOMATABILITY

We show in this chapter that for any $d > 0$, depth-d Frege systems are not automatable unless $\mathsf{P} = \mathsf{NP}$. This extends the result of Atserias and Müller [6] for resolution—or LK_1—to LK_d, for any $d > 0$.

The proof is a reduction from SAT. We want for any positive integer d, given a CNF formula F, to construct a formula G such that if F is satisfiable, then G has small depth-d refutations, whereas if F is unsatisfiable, then G requires large depth-d refutations.

For $d = 1$, [6] considers the formula $G \coloneqq \mathrm{Ref}(F, z)$ expressing that z encodes a resolution refutation of F. Then a "relativization" construction is applied to $\mathrm{Ref}(F, z)$ to get the formula $\mathrm{RRef}(F, z)$, stating that z is either itself, or contains a resolution refutation of F. It is shown by Pudlák [56] that if F is satisfiable, then the formula $\mathrm{Ref}(F, z)$ has short resolution refutations, and this readily extends to $\mathrm{RRef}(F, z)$ [6]. To show a lower bound for $\mathrm{RRef}(F, z)$ in the case F is unsatisfiable, first it is argued in [6] that there cannot be resolution refutation of $\mathrm{RRef}(F, z)$ having small index-width, where the index-width of a resolution refutation π of $\mathrm{RRef}(F, z)$ is defined as the maximum number of clauses of z contained in a clause of π. Then it is shown, following [26], that if $\mathrm{RRef}(F, z)$ had a small resolution refutation, then $\mathrm{Ref}(F, z')$, where the size of z' is polynomially related to the size of z, would have a resolution refutation of small index-width. Notice that arguing in terms of a variant of width, index-width in this case, is necessary. The same argument could not have worked for width, since resolution is automatable with respect to width, in the sense that a resolution refutation of F having width w can be found in time $n^{O(w)}$, where n is the number of variables of F.

To extend the above for the case $d > 1$, an idea is to employ the construction of [45] (see also [9]), replacing every variable of $\mathrm{RRef}(F, z)$ with Sipser functions, i.e. formulas of

the form

$$\bigwedge_{i_1}\bigvee_{i_2}\cdots\bigwedge_{i_k=1} x_{i_1,\ldots,i_k}$$

or

$$\bigvee_{i_1}\bigwedge_{i_2}\cdots\bigvee_{i_k=1} x_{i_1,\ldots,i_k},$$

for some suitable k. Following [45, 9], one gets a lower bound by repeated applications of Håstad's switching lemma [41], which reduce a size lower bound to essentially a width lower bound. In our case, we need a reduction in the base case of the argument to a lower bound for index-width, and trying to apply Håstad's switching lemma for index-width instead of width, one encounters several difficulties, a main one being that the variables encoding the clauses of z induce an exponential factor in the switching probability, making the lemma trivial. We are able to overcome these difficulties by applying the weaker Furst-Saxe-Sipser switching lemma [34], which can use restrictions that fix much less variables on average than Håstad's switching lemma. This will give a weaker lower bound, only polynomial in our case, which nonetheless is sufficient for the purposes of showing non-automatability.

5.1 Basic definitions

We define $\Sigma_d^{s,k}$ to be the class of all formulas F for which there is a depth-d formula G that is semantically equivalent to F, the outermost connective of G is $\bigvee$, and

1. G contains at most s subformulas of depth at least 2;

2. all depth-2 subformulas of G are either k-DNFs or k-CNFs.

Similarly, $\Pi_d^{s,k}$ is defined as the class of all formulas F for which there is a depth-d formula G semantically equivalent to F, the outermost connective of which is $\bigwedge$, satisfying the above two conditions.

A semantic depth-d (Frege) proof from a set of formulas S is a sequence of depth-d formulas $F_1, \ldots, F_t$ such that for every i, either $F_i \in S$, or there are $j, k < i$ such that $F_j, F_k \models F_i$. Notice that if S consists of depth-$(d-1)$ formulas, then there is a trivial depth-d proof of any valid consequence of S, as $\bigwedge S$ can be derived in $|S| - 1$ steps from S. Thus, under this formulation, depth-d proofs from S are interesting only if S contains depth-d formulas not in $\Pi_d^{s,k}$ for any s and k, and indeed, our results pertain to such proofs.

Let $\pi = F_1, \ldots, F_t$ be a semantic proof. If F_t is false under every assignment, π is called a refutation. The formulas $F_1, \ldots, F_t$ are called the lines of π. The length of π is t, and the size of π is the total number of symbols it contains. π is tree-like if its proof-DAG is a tree. Finally, the *variable width* of π is defined to be the quantity $\max_{i \in [t]} |\mathrm{Vars}(F_i)|$, where $\mathrm{Vars}(F_i)$ is the set of variables occurring in F_i.

Unlike size, variable width is an inherently semantic measure. In particular, it is independent of depth: any depth-d proof of variable width w can be transformed into a depth-1 proof of (variable) width $O(w)$. In fact, something stronger can be said. A *decision tree* is a binary tree the internal nodes of which are labelled by variables, and the edges by values 0 or 1. Nodes query variables and the edges going from a node to its children are labelled, one by the value 0 and the other by 1, giving an answer to that query. No variable is repeated in a branch so that branches correspond to restrictions, and each branch has a value, 0 or 1, associated with it, so that the decision tree represents a Boolean function. We denote the set of branches of $\mathbf{T}$ having the value v by $\mathrm{Br}_v(\mathbf{T})$. Specifically, we say that a decision tree $\mathbf{T}$ *represents a formula* F if for every branch π of $\mathbf{T}$ with value v, $F|_\pi \equiv v$. The *height* of a decision tree is the length of its longest branch. Notice that if a formula F is represented by a decision tree of height h, then $F \in \Sigma_2^{1,h} \cap \Pi_2^{1,h}$. We write $h(F)$ for the minimum height of a decision tree representing F. The following lemma is shown in [63] for $\mathrm{Res}(k)$ proofs, but holds for proofs of arbitrary depth, or for that matter, arbitrary sound proofs.

Lemma 5.1.1. *Let S be a set of clauses each containing at most h literals. If there is a*

semantic refutation of S each line of which is represented by a decision tree of height at most h, then there is a resolution refutation of S of width at most $3h$.

Proof. Let $F_1, \ldots, F_t$ be a semantic refutation of S and let $\mathbf{T}_i$ be a decision tree of height at most h representing F_i. We assume that $\mathbf{T}_t$ has a single node having the value 0. For a restriction π, let C_π be the minimal clause falsified by π. We will show that for every i, for every branch $\pi \in \mathrm{Br}_0(\mathbf{T}_i)$, we can derive C_π via a resolution proof of width at most $3h$. Notice that C_π for $\pi \in \mathrm{Br}_0(\mathbf{T}_t)$ is the empty clause, so this construction will give a refutation.

If F_i is a clause C in S, then every $\pi \in \mathrm{Br}_0(\mathbf{T}_i)$ must make every literal in C false, hence C_π is a weakening of C. Assume now that F_i is derived from F_j and F_k and we have derived all clauses C_π for $\pi \in \mathrm{Br}_0(\mathbf{T}_j) \cup \mathrm{Br}_0(\mathbf{T}_k)$. Let $\sigma \in \mathrm{Br}_0(\mathbf{T}_i)$, and let $\mathbf{T}$ be the tree resulting by appending a copy of $\mathbf{T}_k$ at the end of every branch $\pi \in \mathrm{Br}_1(\mathbf{T}_j)$ of $\mathbf{T}_j$. We will use $\mathbf{T}$ to extract a resolution proof of C_σ. More specifically, for every node u of $\mathbf{T}$ such that the path π_u from the root of $\mathbf{T}$ to u corresponds to a restriction that is consistent with σ, we will derive $C_\sigma \vee C_{\pi_v}$. When we reach the root of $\mathbf{T}$ we will have derived C_σ. If u is a leaf of $\mathbf{T}$, then we claim that C_{π_u} is a weakening of some clause C_π for $\pi \in \mathrm{Br}_0(\mathbf{T}_j) \cup \mathrm{Br}_0(\mathbf{T}_k)$. To see this, let $\pi_u = \pi_j \cup \pi_k$, where π_j is the part of π_u that belongs to $\mathbf{T}_j$ and π_k the part that belongs to $\mathbf{T}_k$. Since $F_j, F_k \models F_i$ and π_u is consistent with σ, it cannot be the case that both $\pi_j \in \mathrm{Br}_1(\mathbf{T}_j)$ and $\pi_k \in \mathrm{Br}_1(\mathbf{T}_k)$, otherwise a total assignment extending both π_u and σ would make F_j and F_k true, but F_i false. Suppose now that u is not a leaf of $\mathbf{T}$ and suppose that v and w are its children. Then either π_v and π_w are both consistent with σ, in which case $C_\sigma \vee C_{\pi_u}$ can be derived by resolving $C_\sigma \vee C_{\pi_v}$ and $C_\sigma \vee C_{\pi_w}$ on the variable labelling u, or one of the children, say v, will be consistent with σ and thus $C_\sigma \vee C_{\pi_u}$ will be identical to $C_\sigma \vee C_{\pi_v}$. □

A proof system $\mathbf{P}$ is called *automatable* [21] if there is an algorithm that, given a set of formulas S and a formula ϕ provable from S, outputs a $\mathbf{P}$-proof of ϕ from S in time

polynomial $r + s$, where r is the total size of S and s the size of the shortest σ-proof of ϕ from S.

The main theorem of this chapter is the fact that approximating the minimum size of a depth-d Frege refutation within a polynomial factor is NP hard:

Theorem 5.1.1. *For every integer $d > 0$, there is a polynomial-time computable function, which takes as input a CNF formula F with n variables and m clauses and integers $s, N > 0$ represented in unary, and returns a formula $G_d(F; s, N)$ of depth d such that*

1. *if F is satisfiable, then there is an LK_d refutation of $G_d(F; s, N)$ of size*

$$O\left(\left(N^{d+3}s^2n(m + s^2n^3)\right)^2\right);$$

2. *if F is not satisfiable, N is an increasing function of n and s is a polynomial in n, every semantic depth-d refutation of $G_d(F; s, N)$ must have size at least*

$$N^{\frac{1}{3}\left(\frac{\log s}{\log n}-2\right)^{\frac{1}{d-1}}}$$

for large enough n.

The NP hardness of automating depth-d Frege systems follows from Theorem 5.1.1 by setting $s \coloneqq n^{(3h)^{d-1}+2}$ and $N \coloneqq s$ for a large enough constant h (see Theorem 5.5.1).

We describe the reduction, constructing the formula $G_d(F; s, N)$ from F in Section 5.2. In Section 5.3, we show the upper bound of Theorem 5.1.1, and in Section 5.4 we show the lower bound. It is important to note that both bounds hold for semantic depth-d refutations. The reason we formulate the upper bound in terms of LK refutations is twofold. First, we are able to apply Theorem 2.3.1; we contend it is much cleaner to first give a depth-$(d+1)$ tree-like LK refutation of our formulas and then convert it to a depth-d refutation, rather than directly giving a depth-d refutation. Secondly, the notion of automatability is neither monotone nor

anti-monotone. Hence it is clear from Theorem 5.1.1 that the non-automatability result applies to any version intermediate between depth-d LK and depth-d semantic systems.

5.2 The formulas Ref

Let F be a CNF formula with n variables and m clauses. The key ingredient in the non-automatability result of [6] is expressing by a set of clauses $\mathrm{Ref}(F, s)$ the statement that there is a resolution refutation $D_1, \ldots, D_s$ of length s from the clauses of F.

The variables of $\mathrm{Ref}(F, s)$ are:

$$
\begin{array}{ll}
D[u, i, b], & u \in [s], i \in [n], b \in \{0, 1\}; \\
V[u, i], & u \in [s], i \in [n] \cup \{0\}; \\
I[u, j], & u \in [s], j \in [m] \cup \{0\}; \\
L[u, v], & u, v \in [s]; \\
R[u, v], & u, v \in [s].
\end{array}
$$

The meaning of $D[u, i, b]$ is that x_i^b appears in D_u. The meaning of $V[u, i]$ is that D_u is derived as a weakening of the resolvent of two previous clauses on x_i, and the meaning of $I[u, j]$ is that D_u is a weakening of the j-th clause of F. The meaning of $L[u, v]$ is that the left clause (i.e. that which contains $\neg x_i$) from which D_u was derived is D_v, and the meaning of $R[u, w]$ is that the right clause (i.e. that which contains x_i) from which D_u was derived is D_w. We also use the variables $V[u, 0]$ and $I[u, 0]$ to indicate whether D_u is derived from previous clauses or from an initial clause of F: in the former case, $I[u, 0]$ will be true and $V[u, 0]$ false, and in the latter $V[u, 0]$ will be true and $I[u, 0]$ false.

The clauses of $\mathrm{Ref}(F, s)$ encode the following conditions: For each $u, v \in [s]$, $i, i' \in [n]$,

$j \in [m]$ and $b \in \{0, 1\}$,

$$\exists! k\, V[u,k] \;\;\&\;\; \exists! k\, I[u,k] \;\;\&\;\; \exists! k\, L[u,k] \;\;\&\;\; \exists! k\, R[u,k]; \tag{5.1}$$

$$V[u,0] \iff \neg I[u,0]; \tag{5.2}$$

$$\neg L[u,v] \text{ for } v \geq u \;\;\&\;\; \neg R[u,v] \text{ for } v \geq u; \tag{5.3}$$

$$V[u,i] \;\;\&\;\; L[u,v] \implies D[v,i,0]; \tag{5.4}$$

$$V[u,i] \;\;\&\;\; R[u,v] \implies D[v,i,1]; \tag{5.5}$$

$$V[u,i] \;\;\&\;\; L[u,v] \;\;\&\;\; D[v,i',b] \;\;\&\;\; i \neq i' \implies D[u,i',b]; \tag{5.6}$$

$$V[u,i] \;\;\&\;\; R[u,v] \;\;\&\;\; D[v,i',b] \;\;\&\;\; i \neq i' \implies D[u,i',b]; \tag{5.7}$$

$$I[u,j] \;\;\&\;\; x_i^b \text{ appears in } C_j \implies D[u,i,b]; \tag{5.8}$$

$$\neg D[u,i,0] \vee \neg D[u,i,1]; \tag{5.9}$$

$$\neg D[s,i,b]. \tag{5.10}$$

It was shown, subsequent to [6], that $\mathrm{Ref}(F,s)$ is hard for resolution whenever F is unsatisfiable [36]. In [6], a variation, $\mathrm{RRef}(F,s)$, is used. $\mathrm{RRef}(F,s)$ expresses the fact that there is a resolution refutation $D_1, \ldots, D_s$ or one contained in $D_1, \ldots, D_s$, from the clauses of F. $\mathrm{RRef}(F,s)$ has the same variables as $\mathrm{Ref}(F,s)$ plus a new variable $P[u]$ indicating which of the indices $1, \ldots, s$ are active, i.e. are part of the refutation. The clauses of $\mathrm{RRef}(F,s)$ express the following conditions, which are those of $\mathrm{Ref}(F,s)$ conditioned on the fact that $P[u]$ is true, in addition to three new ones requiring $P[s]$ to be true, and $P[v]$ to be true

whenever $P[u]$ and $L[u,v]$ or $R[u,v]$ are true:

$$P[u] \implies \exists!k\, V[u,k] \;\&\; \exists!k\, I[u,k] \;\&\; \exists!k\, L[u,k] \;\&\; \exists!k\, R[u,k]; \tag{5.11}$$

$$P[u] \implies (V[u,0] \iff \neg I[u,0])\,; \tag{5.12}$$

$$P[u] \implies \neg L[u,v] \text{ for } v \geq u \;\&\; \neg R[u,v] \text{ for } v \geq u; \tag{5.13}$$

$$P[u] \implies (V[u,i] \;\&\; L[u,v] \implies D[v,i,0])\,; \tag{5.14}$$

$$P[u] \implies (V[u,i] \;\&\; R[u,v] \implies D[v,i,1])\,; \tag{5.15}$$

$$P[u] \implies \big(V[u,i] \;\&\; L[u,v] \;\&\; D[v,i',b] \;\&\; i \neq i' \implies D[u,i',b]\big)\,; \tag{5.16}$$

$$P[u] \implies \big(V[u,i] \;\&\; R[u,v] \;\&\; D[v,i',b] \;\&\; i \neq i' \implies D[u,i',b]\big)\,; \tag{5.17}$$

$$P[u] \implies \Big(I[u,j] \;\&\; x_i^b \text{ appears in } C_j \implies D[u,i,b]\Big)\,; \tag{5.18}$$

$$P[u] \implies (\neg D[u,i,0] \vee \neg D[u,i,1])\,; \tag{5.19}$$

$$P[s] \;\&\; \neg D[s,i,b]; \tag{5.20}$$

$$(P[u] \;\&\; L[u,v] \implies P[v]) \;\&\; (P[u] \;\&\; R[u,v] \implies P[v])\,. \tag{5.21}$$

Notice that giving truth values to the $P[u]$ variables (where $P[s] = 1$) reduces $\mathrm{RRef}(F,s)$ to $\mathrm{Ref}(F,s')$ where s' is the number of indices u for which $P[u] = 1$.

For an integer $k \geq 1$, we define $\mathrm{R}^k\mathrm{Ref}(F,s)$ as the formula resulting from substituting each variable $P[u]$ in $\mathrm{RRef}(F,s)$ with the conjunction $\bigwedge_{i=1}^{k} P_i[u]$ for new variables $P_1[u], \ldots, P_k[u]$. Note that $\mathrm{RRef}(F,s) = \mathrm{R}^1\mathrm{Ref}(F,s)$.

Now, let $d, N \geq 1$ be integers, and let x be a propositional variable. We associate with x $N^{d-1}\lceil\sqrt{N}/2\rceil$ new variables $x_{i_1,\ldots,i_d}$, where $i_1, \ldots, i_{d-1} \in [N]$ and $i_d \in [\lceil\sqrt{N}/2\rceil]$. The fact that we make i_d range over $[\lceil\sqrt{N}/2\rceil]$ instead of $[N]$ will be important later (specifically in

Lemma 5.4.1). The depth-d Sipser functions for x are defined by

$$S^{\wedge}_{d,N}(x) \stackrel{\text{def}}{=} \bigwedge_{i_1=1}^{N} \bigvee_{i_2=1}^{N} \cdots \bigwedge_{i_d=1}^{\lceil\sqrt{N}/2\rceil} x_{i_1,\ldots,i_d},$$
$$S^{\vee}_{d,N}(x) \stackrel{\text{def}}{=} \bigvee_{i_1=1}^{N} \bigwedge_{i_2=1}^{N} \cdots \bigvee_{i_d=1}^{\lceil\sqrt{N}/2\rceil} x_{i_1,\ldots,i_d}$$

if d is odd, and

$$S^{\wedge}_{d,N}(x) \stackrel{\text{def}}{=} \bigwedge_{i_1=1}^{N} \bigvee_{i_2=1}^{N} \cdots \bigvee_{i_d=1}^{\lceil\sqrt{N}/2\rceil} x_{i_1,\ldots,i_d},$$
$$S^{\vee}_{d,N}(x) \stackrel{\text{def}}{=} \bigvee_{i_1=1}^{N} \bigwedge_{i_2=1}^{N} \cdots \bigwedge_{i_d=1}^{\lceil\sqrt{N}/2\rceil} x_{i_1,\ldots,i_d}$$

if d is even.

We define $\mathrm{RRef}_{d,N}(F,s)$ to be the result of substituting every variable of the form $P[u]$ in $\mathrm{RRef}(F,s)$ with $S^{\wedge}_{d,N}(P[u])$ and every other variable x with $S^{\vee}_{d,N}(x)$. Notice that $\mathrm{RRef}_{d,N}(F,s)$ is a set of depth-$(d+1)$ formulas. But, as we want to prove statements about whether $\mathrm{RRef}_{d,N}(F,s)$ has or does not have small depth-d refutations, we must write it as a set of depth-d formulas. We may do that with only a polynomial increase in size, as the only clauses of non constant size of $\mathrm{RRef}(F,s)$ are those of the form $\neg P[u] \vee \bigvee_i X[u,i]$ corresponding to conditions (5.11), and these clauses will have depth-d after the substitution taking us from $\mathrm{RRef}(F,s)$ to $\mathrm{RRef}_{d,N}(F,s)$. Note that the conversion from $\mathrm{RRef}_{d,N}(F,s)$ written as a set of depth-d formulas to its equivalent set of depth-$(d+1)$ formulas can be carried in tree-like depth-$(d+1)$ LK in linear time. In particular, a tree-like depth-$(d+1)$ LK refutation of the latter set can be turned into a tree-like depth-$(d+1)$ LK refutation of the former set, increasing the size by at most a factor of N^3.

5.3 Upper bounds

We show in this section that if F is satisfiable, then $\mathrm{RRef}_{d,N}(F,s)$ has small LK_d refutations:

Proposition 5.3.1. *If F is a satisfiable CNF formula with n variables and m clauses, then there is an LK_d refutation of $\mathrm{RRef}_{d,N}(F,s)$ of size*

$$S = O\left(\left(N^{d+3}s^2n(m+s^2n^3)\right)^2\right).$$

In particular, if $m = O(s^2n^3)$, then $S = O(N^{2(d+3)}(sn)^8)$.

Proof. We start with a small tree-like LK_2 refutation of $\mathrm{RRef}(F,s)$. This refutation will be such that after the substitution with Sipser functions, we get a tree-like LK_{d+1} refutation of $\mathrm{RRef}_{d,N}(F,s)$, which in turn we can convert to a LK_d refutation of $\mathrm{RRef}_{d,N}(F,s)$ by Theorem 2.3.1.

Let α be an assignment that satisfies every clause of F. We set

$$T(u) \coloneqq P[u] \to \bigvee_{i=1}^{n} D[u,i,\alpha(x_i)].$$

What $T(u)$ says is that if $P[u]$ is true, then α satisfies the u-th clause in the refutation $\mathrm{Ref}(F,s)$ describes.

Our refutation of $\mathrm{RRef}(F,s)$ consists of $s-1$ stages, starting with stage 0. In the u-th stage, $T(1),\dots,T(s-u) \to 0$ will have been derived. Then we can use this formula, along with a derivation of $T(1),\dots,T(s-u-1) \to T(s-u)$, to derive $T(1),\dots,T(s-u-1) \to 0$. In the $s-1$-th stage, $T(1) \to 0$ will have been derived, at which point we can reach a contradiction by deriving $T(1)$.

A derivation of $T(1),\dots,T(v-1) \to T(v)$ is sketched in Figure 5.1. The formulas $I[v,j] \to T(v)$ for $j \in [m]$ can be immediately derived from the clauses $P[u] \wedge I[v,j] \to D[v,i,\alpha(x_i)]$, which are clauses corresponding to condition (5.18), as the fact that α satisfies

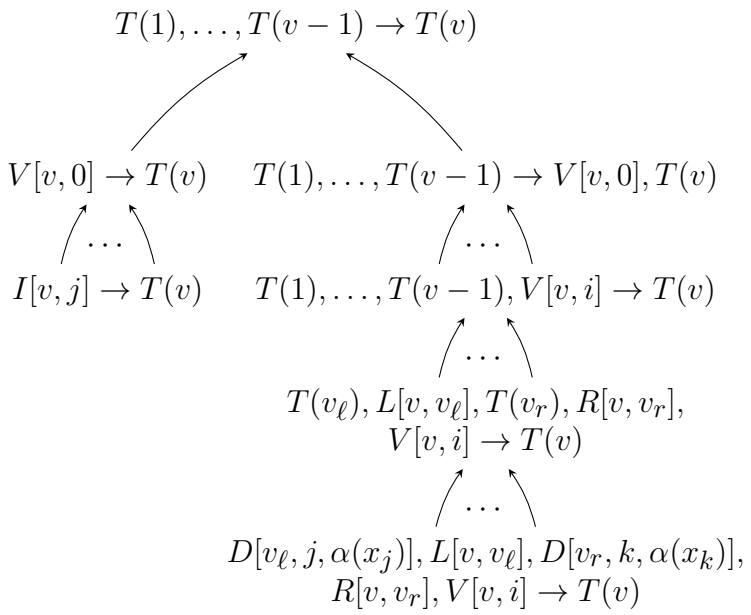


Figure 5.1: A sketch of a derivation of $T(1), \ldots, T(v-1) \to T(v)$

the clause C_j means that $x_i^{\alpha(x_i)}$ must belong to C_j for some i. These formulas can be in turn used along with the clauses (5.11) for $I[v,k]$ and (5.12) to derive $V[v,0] \to T(v)$. Now deriving

$$T(1), \ldots, T(v-1) \to V[v,0], T(v) \tag{5.22}$$

will allow us to derive $T(1), \ldots, T(v-1) \to T(v)$ by cutting on $V[v,0]$. We can derive (5.22) from the formulas

$$T(1), \ldots, T(v-1), V[v,i] \to T(v) \tag{5.23}$$

for $i \in [n]$ using the clauses (5.11) for $V[v,i]$. The formulas (5.23) can be in turn derived from the formulas

$$T(v_\ell), L[v,v_\ell], T(v_r), R[u,v_r], V[v,i] \to T(v) \tag{5.24}$$

for $v_\ell, v_r \in [s]$ using the clauses (5.11) for $L[v,k]$ and $R[v,k]$, (5.12) and (5.13). Finally, (5.24) can be derived from the formulas

$$D[v_\ell, j, \alpha(x_j), L[v, v_\ell], D[v_r, x_k, \alpha(x_k)], R[u, v_r], V[v, i] \to T(v), \tag{5.25}$$

for $j, k \in [n]$, which can be derived directly from the clauses (5.21) and either (5.14), (5.15) and (5.19) or (5.16) and (5.17) depending on whether $i = j = k$ or not.

We can see that the derivations of $T(1)$, $\overline{T(s)}$ and $T(1), \ldots, T(v-1) \to T(v)$ take at most $O(m + s^2n^3)$ steps, hence the overall refutation has size $O\left(s^2n(m + s^2n^3)\right)$.

Now, notice that after substituting every variable $P[u]$ in it with $S^{\wedge}_{d,N}(P[u])$ and every other variable x with $S^{\vee}_{d,N}(x)$, $T(v)$ becomes a depth-d formula. Hence we see that after the substitution, the refutation described above becomes a tree-like LK_{d+1} refutation of $\mathrm{RRef}_{d,N}(F, s)$. We can then get a depth-d refutation of $\mathrm{RRef}_{d,N}(F, s)$ of the required size by applying Theorem 2.3.1. □

5.4 Lower bounds

Lower bounds for depth-d Frege systems for $d > 1$, typically follow the following strategy:

1. We first show that the formulas we are trying to refute are robust; namely, after applying a restriction selected at random to them, then with high probability they cannot be refuted with proofs whose lines are, in a certain sense, simple.

2. Then we show, through the use of a switching lemma, that applying such a restriction to a short proof will result with high probability in a proof with simple lines.

Here we start with $\mathrm{RRef}_{d,N}(F, s)$, which after applying the restrictions will collapse to $\mathrm{Ref}(F, s')$, where s' is polynomially related to s. For the part of the overall strategy showing that there cannot be refutations with simple lines, we take, as in [6], simple to mean of small

index-width. We say that a variable of the form $D[u,i,b]$, $V[u,i]$, $I[u,j]$, $L[u,v]$ or $R[u,v]$ *mentions* the index u. The index-width of a clause in the variables of $\mathrm{Ref}(F,s)$ is defined as the number of indices mentioned by its variables, and the index-width of a resolution refutation of $\mathrm{Ref}(F,s)$ is the maximum index-width over its clauses. We have:

Theorem 5.4.1 [6]**.** *For all integers $n, s > 0$ with $s \leq 2^n$, and every unsatisfiable CNF F with n variables, every resolution refutation of* $\mathrm{Ref}(F,s)$ *has index-width at least $s/6n$.*

The proof of Theorem 5.4.1 is similar in spirit to the proof of Theorem 3.3.1. As in Theorem 3.3.1, we first find a model $\mathbf{M}$ in which $\mathrm{Ref}(F,s)$ becomes satisfiable. Here, we take $\mathbf{M}$ to be the brute force tree-like refutation of F of size 2^n. Then, using this model, we show that there is a strategy of the adversary such that a prover of small index-width, cannot distinguish between $\mathrm{Ref}(F,s)$ and $\mathrm{Ref}(F,2^n)|_{\mathbf{M}}$.

5.4.1 The robustness of $RRef_{d,N}$

We create a distribution on restrictions to the variables of $\mathrm{RRef}_{d,N}(F,s)$ as follows. Suppose d is odd (if d were even, we would exchange the roles of 0 and 1 in the following construction). For each $S^{\wedge}_{d,N}(x)$ formula in $\mathrm{RRef}_{d,N}(F,s)$, look at its bottom-most N^{d-1} $\bigwedge$ connectives. For each such connective, we decide to "preserve" it with probability $1/\sqrt{N}$, and not to preserve it with probability $1-1/\sqrt{N}$. For each of the preserved connectives, we leave its first variable unset and set the rest to 1. For each variable in the unpreserved connectives, we set it to 0 or 1 with probability 1/2 for each choice. The variables of $S^{\vee}_{d,N}(x)$ are set in the same way, except that the set variables of the preserved $\bigvee$ connectives are set to 0 instead of 1.

Under such restrictions, Sipser functions do not simplify much. For formulas F and G, in which each variable appears only once, we say that F *contains* G if we can get G from F by deleting some of its literals and/or renaming some of its variables.

Lemma 5.4.1. *For any $d \geq 2$, the probability that $S^{\nu}_{d,N}(x)|_{\rho}$, where $\nu \in \{\wedge, \vee\}$, does not contain $S^{\nu}_{d-1,N}(x)$ is at most $2^{-\Omega(\sqrt{N})}$.*

Proof. We show the lemma for $S^{\wedge}_{d,N}(x)$ and d odd. If $S^{\wedge}_{d,N}(x)|_{\rho}$ does not contain $S^{\wedge}_{d-1,N}$, then either one of its bottom-most $\bigwedge$ connectives takes the value 1, or in one of its depth-2 subformulas, less than $\sqrt{N}/2$ $\bigwedge$ connectives are preserved. The probability that a bottom-most $\bigwedge$ connective takes the value 1 is at most $2^{-\sqrt{N}/2}$ and the probability that this happens for at least one of the N^{d-1} bottom-most $\bigwedge$ connectives is at most

$$N^{d-1}2^{-\sqrt{N}/2} \leq 2^{-\Omega(\sqrt{N})}.$$

Now fix a depth-2 subformula A of $S^{\wedge}_{d,N}(x)$. The expected number of preserved $\bigwedge$ connectives in $A|_{\rho}$ is $N/\sqrt{N} = \sqrt{N}$, and by the Chernoff bound, the probability that there are less than $\sqrt{N}/2$ preserved $\bigwedge$ connectives is at most $2^{-\Omega(\sqrt{N})}$. The probability that at least one of the N^{d-2} depth-2 subformulas of $S^{\wedge}_{d,N}(x)$ has less than $\sqrt{N}/2$ preserved connectives is thus at most

$$N^{d-2}2^{-\Omega(\sqrt{N})} \leq 2^{-\Omega(\sqrt{N})}.$$

We conclude that the probability that $S^{\wedge}_{d,N}|_{\rho}$ does not contain $S^{\wedge}_{d-1,N}$ is at most $2^{-\Omega(\sqrt{N})}$.

□

5.4.2 The Furst-Saxe-Sipser switching lemma

Switching lemmas provide conditions under which a k-DNF formula "switches" to a ℓ-CNF formula after applying a restriction created at random. We will use the switching lemma of [34] and a variation tailored for $\mathrm{R}^k\mathrm{Ref}(F, s)$ due to [37].

Let G be a k-DNF formula over the set of variables X. Let $X_1, \ldots, X_r$ be a partition of X into r blocks, and let $\nu \in \{0, 1\}$. Consider the following distribution over restrictions on X: For each block X_i, we decide to "preserve" X_i with probability p, and not to preserve it

with probability $1-p$. For each preserved block, we leave one of its variables, say the first in the block, unset, and set all others to ν. For each unpreserved block, we set each of its variables to 0 or 1 with probability $1/2$ for each value.

We can extract the following lemma from [34, 64]. The lemma is implicit in [34, 64] with parameters obscured under a big O notation. We present it here in a more general, improved form, with explicit parameters, using decision trees along the lines of [63]. In what follows, ln denotes the natural logarithm; we preserve the notation log for the base 2 logarithm.

Lemma 5.4.2 (see [34, 64])**.** *If $phk2^k \ln N = o(N^{-\varepsilon})$ for some $\varepsilon \in (0,1)$, then*

$$P[h(G|_\rho) > kh] \leq o(N^{-\varepsilon h})\frac{2^{kh}-1}{2^h-1}.$$

Proof. The proof is by induction on k. If $k = 0$, G is a constant and can be represented by a decision tree of height 0. Suppose $k > 0$. We distinguish between two cases, G being wide and G being narrow. We call G wide if there are at least $h2^k \ln N$ terms in it such that no two of them contain variables from the same block. G is narrow if and only if it is not wide. If G is wide, then

$$\begin{aligned} P[h(G|_\rho) > kh] \leq P[G|_\rho \neq 1] &\leq \left(1 - \left(\frac{1-p}{2}\right)^k\right)^{h2^k \ln N} \\ &\leq e^{-(1-p)^k h \ln N} = N^{-(1-p)^k h} = o(N^{-\varepsilon h}). \end{aligned}$$

If G is narrow, then take a maximal set of terms such that no two of them contain variables from the same block, and let H be the set of blocks that contain a variable occurring in some term of this set. H contains at most $hk2^k \ln N$ blocks and every term of G contains some variable (or its negation) from some block in H. The probability of the event A that

ρ preserves more than h blocks in H is

$$P[A] \leq \binom{hk2^k \ln N}{h} p^h \leq (hk2^k \ln N)^h p^h = o(N^{-\varepsilon h}).$$

Now, let π be a restriction that sets the variables of all blocks in H, and let A_π be the event that π is consistent with ρ and $h((G|_\rho)|_\pi) > (k-1)h$. Notice that $G|_\pi$ is a $(k-1)$-DNF, so by the induction hypothesis,

$$P[A_\pi] \leq P[h((G|_\pi)|_\rho) > (k-1)h] \leq o(N^{-\varepsilon h}) \frac{2^{(k-1)h} - 1}{2^h - 1}.$$

Notice that a restriction ρ that preserves at most h blocks is consistent with at most 2^h restrictions π, so we get

$$\begin{aligned} P\left[A \cup \bigcup_\pi A_\pi\right] &\leq o(N^{-\varepsilon h}) + o(N^{-\varepsilon h}) 2^h \frac{2^{(k-1)h} - 1}{2^h - 1} \\ &= o(N^{-\varepsilon h}) \frac{2^{kh} - 1}{2^h - 1}. \end{aligned}$$

In the event

$$\left(A \cup \bigcup_\pi A_\pi\right)^c,$$

i.e. the event that ρ preserves at most h blocks in H and for all restrictions π consistent with ρ, $h((G|_\rho)|_\pi) \leq (k-1)h$, we can construct a decision tree of height at most kh representing $G|_\rho$ as follows: We query all variables belonging to some block in H left unset by ρ (since ρ preserves at most h blocks in H, there are at most h of them), and at each branch π of the resulting tree, we append a decision tree of minimum height representing $(G|_\rho)|_\pi$. □

We create a distribution on restrictions on the variables of $\mathrm{R}^\ell\mathrm{Ref}(F, s)$ as follows: For every index u and every $i \in [\ell]$, we set $P_i[u]$ to 0 or 1, with probability 1/2 for each value. Let U be the set of indices such that $P_i[u] = 1$ for all $i \in [\ell]$. For each variable x of $\mathrm{R}^\ell\mathrm{Ref}(F, s)$

not of the form $P_i[u]$ mentioning an index in U, we set x to 0 or 1, with probability 1/2 for each value.

For a decision tree $\mathbf{T}$ querying variables of $\mathrm{Ref}(F, s)$, we define the *index-height* of $\mathbf{T}$ as the maximum number of indices mentioned by variables over all branches that do not falsify axioms of $\mathrm{Ref}(F, s)$. For a formula G, We denote by $\hbar(G)$ the minimum index-height of a decision tree representing G.

The following lemma is from [37]. We give a proof because in [37] the lemma is stated not for $\mathrm{R}^\ell\mathrm{Ref}(F, s)$ but a variation, plus we view the following proof to be simpler.

Lemma 5.4.3 [37]**.** *Let F be a CNF formula in n variables, k and ℓ integers with $0 < k \leq \ell$, and G a k-DNF formula over the variables of $\mathrm{R}^\ell\mathrm{Ref}(F, s)$, where $s \leq 2^{\delta n}$ for some $\delta < 1$. Then for large enough n,*

$$P[\hbar(G|_\rho) > h] \leq 2^{-\frac{h}{n^{k-1}}\gamma(k)},$$

where $\gamma(0) = 1$, $\gamma(i) = (\log e)(i4^{i+1})^{-1}\gamma(i-1)$.

Proof. Let $h_i \coloneqq h\gamma(i-1)/(4n^{i-1})$. We will show, by induction on k, that for every k and ℓ with $k \leq \ell$, for every k-DNF formula G over the variables of $\mathrm{R}^\ell\mathrm{Ref}(F, s)$,

$$P\left[\hbar(G|_\rho) > \sum_{i=1}^{k} h_i\right] \leq 2^{-\frac{h}{n^{k-1}}\gamma(k)}$$

for large enough n.

If $k = 0$, F is a constant and can be represented by a decision tree of height 0. Suppose $k > 0$. We call G wide if there are at least h_k/k terms in G over disjoint sets of indices, and call G narrow otherwise. Suppose G is wide. A literal in a term t of G is satisfied with probability at least 1/4: Literals on a variable $P_i[u]$ are satisfied with probability 1/2. For any other literal x^ϵ of t mentioning the index u, since $k \leq \ell$, there must be a variable $P_i[u]$ not in t, which is made 0 with probability 1/2, in which case x^ϵ will be satisfied with

probability 1/2. Hence

$$\begin{aligned} P[\hbar(G|_\rho) > h] \leq P[G|_\rho \neq 1] &\leq (1 - 4^{-k})^{\frac{h\gamma(k-1)}{4kn^{k-1}}} \\ &\leq 2^{-\frac{h}{n^{k-1}}(\log e)(k4^{k+1})^{-1}\gamma(k-1)} \\ &= 2^{-\frac{h}{n^{k-1}}\gamma(k)}. \end{aligned}$$

Suppose now that G is narrow. Take a maximal set of terms over disjoint sets of indices, and let H be the set of indices that are mentioned by the terms of this set. Notice that $|H| \leq h_k$ and that every term of G contains some variable (or its negation) that mentions an index in H. Let π be a restriction that

1. sets all variables mentioning an index in H and leaves all other variables unset, and
2. does not falsify any axioms of $\mathrm{R}^\ell Ref(F, s)$.

The second condition means in particular that if U is the set of indices u for which π sets $P_i[u]$ to 1 for all i, then for all $u \in U$, there will be exactly one v such that $L[u, v]$ is true, exactly one v such that $R[u, v]$ is true, exactly one i such that $V[u, i]$ is true, and exactly one j such that $I[u, j]$ is true, making the total number of such π's to be at most

$$S^{|U|}2^{(|H|-|U|)n_0}$$

where $S \coloneqq s^2(n+1)(m+1)2^{2n}$ and n_0 is the number of variables of $\mathrm{R}^\ell\mathrm{Ref}(F, s)$ mentioning a fixed index u.

Let A_π be the event that π is consistent with ρ and $\hbar((G|_\rho)|_\pi) > \sum_{i=i}^{k-1} h_i$. We have that

$$\begin{aligned}
P[A_\pi] &= P\left[\hbar((G|_\rho)|_\pi) > \sum_{i=i}^{k-1} h_i \mid \rho \text{ con. with } \pi\right] P[\rho \text{ con. with } \pi] \\
&= P\left[\hbar((G|_\pi)|_\rho) > \sum_{i=i}^{k-1} h_i\right] P[\rho \text{ con. with } \pi] \\
&\leq 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell|U|} 2^{-(|H|-|U|)n_0}.
\end{aligned}$$

Hence, we get

$$\begin{aligned}
P\left[\bigcup_\pi A_\pi\right] &\leq \sum_\pi P[A_\pi] \\
&\leq \sum_{U\subseteq H} S^{|U|} 2^{(|H|-|U|)n_0} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell|U|} 2^{-(|H|-|U|)n_0} \\
&= \sum_{r=0}^{|H|} \binom{|H|}{r} S^r 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} 2^{-\ell r} \\
&= (S/2^\ell + 1)^{|H|} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)} \\
&\leq S^{|H|} 2^{-\frac{h}{n^{k-2}}\gamma(k-1)}.
\end{aligned}$$

Since $s \leq 2^{\delta n}$ for some $\delta < 1$, the quantity

$$S^{|H|} = \left(s^2(n+1)(m+1)2^{2n}\right)^{\frac{h}{4n^{k-1}}\gamma(k-1)}$$

will be at most $2^{\frac{\varepsilon h}{n^{k-2}}\gamma(k-1)}$ for some $\varepsilon < 1$ for large enough n, therefore

$$P\left[\bigcup_\pi A_\pi\right] \leq 2^{-\frac{h}{n^{k-1}}\gamma(k)}$$

for large enough n.

In the event $\left(\bigcup_\pi A_\pi\right)^c$, that is the event that for every π consistent with ρ, $\hbar((G|_\rho)|_\pi) \leq$

$\sum_{i=1}^{k-1} h_i$, we can construct a decision tree for $G|_\rho$ of index-height at most $\sum_{i=1}^{k} h_i$ as follows: We first query all variables mentioning an index in H left unset by ρ. Then, at each branch π of the resulting tree, we append a decision tree of minimum index-height representing $(G|_\rho)|_\pi$. □

5.4.3 The lower bound for $RRef_{d,N}$

Theorem 5.4.2. *For every integer $d > 0$, if F is an unsatisfiable CNF in n variables, N is an increasing function of n and s is a polynomial in n, every semantic depth-d refutation of* $\mathrm{RRef}_{d,N}(F, s)$ *has size at least*

$$N^{\frac{1}{3}\left(\frac{\log s}{\log n}-2\right)^{\frac{1}{d-1}}}$$

for large enough n.

Proof. Let $h := (1/3)(\log s/\log n - 2)^{1/(d-1)}$ and let $G_1, \ldots, G_t$ be a semantic depth-d refutation of $\mathrm{RRef}_{d,N}(F, s)$ of size at most N^h. We assume that each G_i is either a literal or a disjunction of its immediate subformulas. Let A be a depth-1 subformula of some G_i. A is a 1-DNF or a 1-CNF formula, so applying Lemma 5.4.2 to it (or its negation respectively) with $k = 1$ and $p = N^{-1/2}$ and using as blocks $X_1, \ldots, X_r$ the variables in the depth-1 subformulas of $\mathrm{RRef}_{d,N}(F, s)$, we get, since $N^{-1/2}3h \ln N = o\left(N^{-1/3}\right)$,

$$P[h(A|_\rho) > 3h] = o(N^{-h}).$$

Now, there are at most N^h depth-1 subformulas A in the refutation, hence, by Lemma 5.4.1 and the union bound, the probability that either there is a depth-1 subformula A with $h(A|_\rho) > 3h$ or $\mathrm{RRef}_{d,N}(F, s)|_\rho$ does not contain $\mathrm{RRef}_{d-1,N}(F, s)$ is $o(1)$. Therefore, for large n, there must be a restriction ρ_1' such that $\mathrm{RRef}_{d,N}(F, s)|_{\rho_1'}$ contains $\mathrm{RRef}_{d-1,N}(F, s)$ and all depth-1 subformulas of all $G_i|_{\rho_1'}$ are disjunctions or conjunctions of at most $3h$ literals. Let ρ_1 be a restriction extending ρ_1' such that $\mathrm{RRef}_{d,N}(F, s)|_{\rho_1}$ is exactly $\mathrm{RRef}_{d-1,N}(F, s)$.

We continue by applying Lemma 5.4.2 with $k = 3h$ and $p = N^{-1/2}$ to a $3h$-CNF or $3h$-DNF depth-2 subformula B of $G_i|_{\rho_1}$ to get

$$P\left[h(B|_\rho) > (3h)^2\right] = o(N^{-h}).$$

Since $G_i|_{\rho_1}$ has at most N^h depth-2 subformulas, there is a restriction ρ_2 such that $\mathrm{RRef}_{d,N}(F,s)|_{\rho_1\rho_2}$ becomes $\mathrm{RRef}_{d-2,N}(F,s)$ and all depth-2 subformulas of all $G_i|_{\rho_1}$ can be represented by decision trees of height at most $(3h)^2$. A formula representable by a decision tree of height at most $(3h)^2$ can be written as both a $(3h)^2$-CNF and a $(3h)^2$-DNF, so for all $i \in [t]$, $G_i|_{\rho_1\rho_2} \in \Sigma_{d-1}^{N^h,(3h)^2}$.

Repeating the same argument $d-1$ times, applying Lemma 5.4.2 at the j-th time to depth-2 subformulas of $\Sigma_{d-j+1}^{N^h,(3h)^j}$-formulas equivalent to $G_i|_{\rho_1\ldots\rho_{d-1}}$, we get restrictions $\rho_1,\ldots,\rho_{d-1}$ such that $\mathrm{RRef}_{d,N}(F,s)|_{\rho_1\ldots\rho_{d-1}}$ becomes $\mathrm{RRef}_{1,N}(F,s)$ and for all $i \in [t]$, $G_i|_{\rho_1\ldots\rho_{d-1}} \in \Sigma_2^{N^h,(3h)^{d-1}}$.

We are now ready to apply Lemma 5.4.3. First notice that $\mathrm{RRef}_{1,N}(F,s)$ contains $\mathrm{R}^\ell\mathrm{Ref}(F,s)$ for large n, where $\ell := (3h)^{d-1}$. For ρ selected randomly as specified in Lemma 5.4.3 for this ℓ, we get that the expected number of active indices is $s/2^\ell$, hence $\mathrm{RRef}_{1,N}(F,s)|_\rho$ contains $\mathrm{Ref}(F,s')$, where $s' := s/2^{\ell+1}$, with high probability. Furthermore, Lemma 5.4.3 gives

$$P\left[\hbar(C|_\rho) > n^{(3h)^{d-1}}\right] \le 2^{-\Omega(n)},$$

where C is a $(3h)^{d-1}$-DNF formula equivalent to some $G_i|_{\rho_1\ldots\rho_{d-1}}$. Therefore there must be a restriction ρ_d such that $\mathrm{RRef}_{d,N}|_{\rho_1\ldots\rho_d}$ becomes $\mathrm{Ref}(F,s')$ and for every $i \in [t]$, $\hbar(G_i|_{\rho_1\ldots\rho_{d-1}}) \le n^{(3h)^{d-1}}$. Applying now the construction of Lemma 5.1.1[1] to $G_1|_{\rho_1\ldots\rho_{d-1}}$, $\ldots, G_t|_{\rho_1\ldots\rho_{d-1}}$ gives a resolution refutation of $\mathrm{Ref}(F,s')$ of index-width at most $3n^{(3h)^{d-1}} = 3s/n^2$, contradicting Theorem 5.4.1 for large n. □

1. Lemma 5.1.1 is stated for height and width, but it is not hard to see that the same construction yields the lemma with index-height and index-width instead of height and width respectively.

5.5 Non-automatability of bounded-depth Frege systems

Theorem 5.5.1. *If* $\mathsf{P} \neq \mathsf{NP}$*, then depth-d Frege systems are not automatable.*

Proof. Suppose there is an algorithm $\mathbf{A}$ which, given an unsatisfiable CNF formula G, returns a depth-d refutation of G in time polynomial in $S(G)+S$, where $S(G)$ is the size of G and S the size of the smallest depth-d refutation of G. Let $c, n_0 \geq 1$ be integers such that for every G with $|G| \geq n_0$, $\mathbf{A}$ runs in time at most $(S(G)+S)^c$. We will use $\mathbf{A}$ to decide in polynomial time whether 3-SAT is satisfiable. Given a 3-CNF formula F with n variables (and thus of size $O(n^3)$), we construct the formula $G \coloneqq \mathrm{RRef}_{d,N}(F, s)$, where $s \coloneqq n^{(3h)^{d-1}+2}$, $N \coloneqq s$ and h is an integer such that

$$\left((3h)^{d-1}+2\right)h > c\left(\left((3h)^{d-1}+2\right)(2(d+3)) + 8\left((3h)^{d-1}+3\right)+1\right).$$

Notice that the left hand side of the above inequality is a polynomial of degree d in h and the right hand side a polynomial of degree $d-1$, hence such an h must exist. Since N and s are polynomials in n, the size of G is polynomial in n, hence its construction takes polynomial time. Let S be the size of the smallest depth-d refutation of G and let $n_1 \geq n_0$ be an integer such that for all $n \geq n_1$,

$$\begin{aligned} F \text{ satisfiable} &\implies S + S(G) \leq n^{\left((3h)^{d-1}+2\right)(2(d+3))+8\left((3h)^{d-1}+3\right)+1}; \\ F \text{ not satisfiable} &\implies S \geq n^{((3h)^{d-1}+2)h}. \end{aligned}$$

Here we use the bounds given by Proposition 5.3.1 and Theorem 5.4.2. To decide whether F is satisfiable, if $n < n_1$, then we check all possible assignments to its variables to see if there is a satisfying one. Otherwise, we run $\mathbf{A}$ on G for

$$n^{c\left(\left((3h)^{d-1}+2\right)(2(d+3))+8\left((3h)^{d-1}+3\right)+1\right)}$$

steps. If **A** stops, then we can assert that F is satisfiable; otherwise we can assert that F is unsatisfiable. □

CHAPTER 6

CONCLUSION

We end with a summary, focusing on some open problems.

6.1 On strong size-space trade-offs

We showed that $\log S_{TR}$, CSpace* and MSpace* are equivalent up to polynomial and $\log n$ factors, demonstrating a picture perfectly analogous to the picture involving D, VSpace* and TSpace* in [60]. The most important question remains open:

Open Problem. Is it true that CSpace $\approx \log S_{TR}$ or MSpace $\approx \log S_{TR}$? Recall for comparison that $\log S_{TR} \approx$ CSpace$^* \approx$ MSpace*.

Equivalently, do there exist strong trade-offs between clause (or monomial) space and length? Strong trade-offs here means that restricting, let's say clause space, must necessarily give rise to refutations π of F of length $|\pi| \gg \exp(\text{CSpace}(F \vdash \bot) \log n)$. Trade-offs of this kind should be contrasted with trade-off results in e.g. [13, 8]. Moreover, the existence of such trade-offs is a perfect analogue of Urquhart's question [68] about variable space vs. depth studied in [60, Section 6]. Let us make a few more remarks about this problem.

First, for very small space essentially this question was already asked in the literature before. Namely (see e.g. [50, Open Problem 16]), are there formulas having constant clause space refutations and with the property that any such refutation must necessarily have super-polynomial length? Suitably adjusting it to our framework:

Open Problem (small space variant). Are there formulas that have $(\log n)^{O(1)}$ clause or monomial space refutations and with the property that any such refutation must be of super-quasi-polynomial length $\exp((\log n)^{\omega(1)})$? Equivalently, any tree-like resolution refutation must have super-quasi-polynomial length.

In terms of the perceived difficulty, we do not discern too much of a difference between Problems 6.1, 6.1 and Nordström's question [50, Open Problem 16]. In fact, we would like to cautiously conjecture that there are formulas F with $\mathrm{CSpace}(F \vdash \bot) \leq 5$ and $\mathrm{CSpace}^*(F \vdash \bot) \geq \exp\left(n^{\Omega(1)}\right)$. But the only result we were able to prove in that direction is the rather weak Theorem 3.3.4.

Secondly, as suggested by Figure 1.1, any strong separation between monomial space and clause space would immediately solve Problem 6.1 for monomial space. As we consider the latter to be most likely very difficult, we take it as a good heuristic explanation of why we have not seen any progress on the former problem as well. But let us ask this, and one obviously relevant question, explicitly anyway:

Open Problem. Is it true that $\mathrm{CSpace} \approx \mathrm{MSpace}$? Is it true that $\mathrm{MSpace} \approx W$?

We note that by the result from [51, 12], at least one of these must be false. In Chapter 4, we show a quadratic separation between width and monomial space, but this is too weak for our framework of simulations. For a discussion on related topics, see also [22, Section 7.5.5].

Finally, while all these conjectured trade-offs are very strong, they are still not super-critical in the sense of [59], at least not according to our framework. A super-critical trade-off is a trade-off between two complexity measures in which restricting one measure causes an increase in the second measure that goes well beyond a worst case upper bound for it. Here, a worst case upper bound on proof length is 2^n, and every proof of clause space s can be assumed to have length $2^{O(sn)}$, as this is an upper bound on the total number of different configurations of clause space at most s. This still leaves us with a potential gap between 2^n and 2^{n^2} and, in fact, a result along these lines is known [7]. But from the perspective we are trying to advocate here, their *logarithms* are polynomially related.

However, as we pointed out in Chapter 3, in all these questions refutation length can be replaced with depth. Since the depth, as a stand-alone measure, is always bounded by n, the question e.g. of whether $\mathrm{CSpace} \approx \mathrm{CSpace}^*$ is actually the same as the question of whether

there exists a *super-critical* trade-off between clause space and *depth*.

We have surprisingly proved that DNF resolution (or depth-2 Frege) behaves very differently from resolution with respect to space (Theorems 3.2.3 and 3.2.4 and Corollaries 3.2.1 and 3.2.2). Intermediate systems based on Res(k) for a constant k were studied in a similar context before, and it is very natural to wonder what is the situation for those systems.

Let us first remark that for Res(k)-refutations, the definition of space from [28, 13] (formula space) coincides with ours up to a factor of k so we need not distinguish between the two. Then Theorem 3.2.1 readily generalizes to this regime and gives $\log S_{T,\mathrm{Res}(k)} \approx \mathrm{Res}(k)\mathrm{Space}^*$, extending the bottom half of Figure 1.1 as shown in Figure 6.1. The proof of

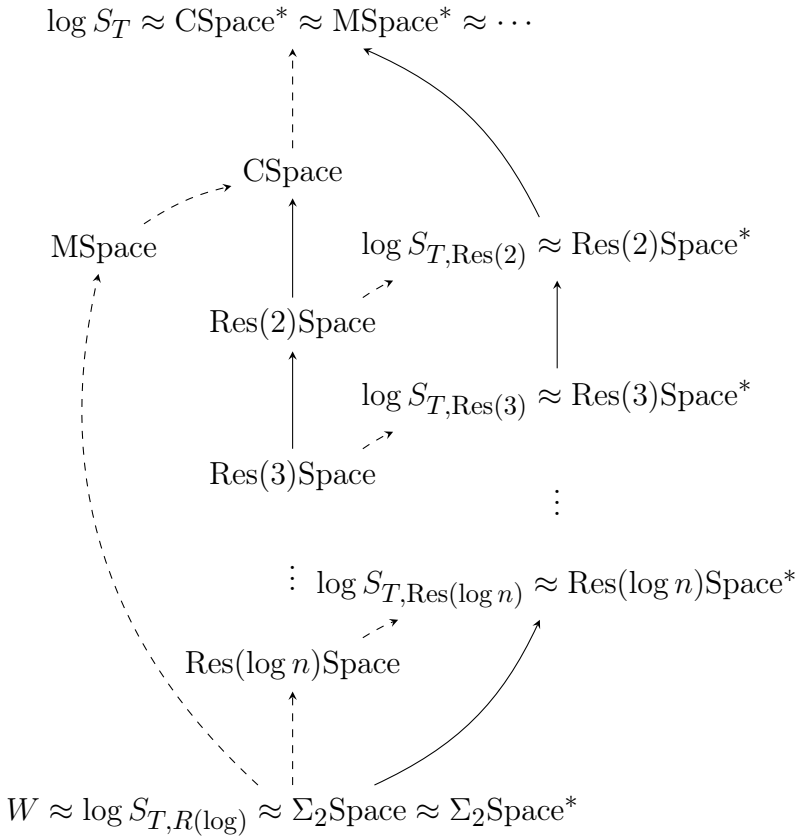


Figure 6.1: Σ_2 space and tree-like size for subsystems of DNF resolution

Corollary 3.2.1, however, fails for a constant k as badly as it fails for $k = 1$. Hence we have one more question to ask:

Open Problem (Res(k)-variant). Is there a constant $k > 0$ such that $\log S_{T,\mathrm{Res}(k)} \approx \mathrm{Res}(k)\mathrm{Space}$ or at least $\log S_{T,\mathrm{Res}(k)} \preceq \mathrm{CSpace}$?

Let us also mention that as k increases, both hierarchies, $\log S_{T,\mathrm{Res}(k)}$ (and, hence, also Res(k)Space*) and Res(k)Space are proper ([28] and [13] respectively). This excludes the dual version of Remark 3.2.3: while the formula space of DNF resolution refutations can be reduced to constant, this is not true for the widths of individual formulas in the memory.

The relation between VSpace and CSpace is also unknown in one direction (the opposite one is taken care of by [10]). Let us re-iterate the problem posed e.g. in [60]:

Open Problem. Is it true that CSpace $\preceq$ VSpace?

Just as with the questions of similar nature discussed above, a negative answer would also imply a separation between VSpace and VSpace*, hence we can expect it to be very difficult.

6.2 On resolution and cut-free LK

We showed a quadratic gap between resolution and cut-free sequent calculus width. In terms of the sequent calculus, this says that atomic cuts can shorten the width of proofs. It is well known that cuts can make proofs exponentially shorter. Allowing arbitrary cuts we get a system polynomially equivalent with any Frege system. These are very powerful; proving non-trivial lower bounds for them is completely out of reach of current methods. But even allowing cuts of depth $d + 1$ in an LK system that has cuts of depth d for any constant $d \geq 0$, gives exponentially shorter proofs [45]. And this goes lower: For any constant $k \geq 0$, allowing as cut formulas conjunctions and disjunctions of arity $k + 1$ in an LK system that has as cuts conjunctions and disjunctions of arity at most k, again gives exponentially shorter proofs [63]. We show in this paper that even allowing propositional variables as cuts, gives super-polynomially shorter proofs.

Cut-free sequent width for refuting CNF formulas naturally compares to well studied complexity measures related to resolution: it sits between resolution width and clause space. Our quadratic gap in particular, provides a separation between resolution width and clause space. Stronger such separations are known [12, 13]. Nontheless, our basic construction extends to provide a quadratic gap between resolution width and monomial space. This is to be seen in conjunction with the relation

$$W_R(F \vdash \bot) \leq O\left((\text{MSpace}(F \vdash \bot))^2\right) + W(F) \tag{6.1}$$

of Galesi et al. [35] showing that monomial space provides an upper bound to resolution width.

Several questions remain open:

1. Can cut-free sequent calculus width for refuting CNF formulas be bounded in terms of resolution width? Given the similarity between the two measures, the combination of Lemmas 4.2.1 and 4.2.2 giving a quadratic separation might come as a surprise. Can this separation be improved? A strong separation in particular, would give an exponential separation between resolution and cut-free sequent calculus.

2. Our super-polynomial separation of resolution and cut-free sequent calculus on the one hand applies only when clauses are seen as disjunctions of unbounded arity. On the other hand, it involves formulas whose size grows exponentially. Can there be a separation independent of the representation of clauses? Can there be a separation for formulas whose size grows polynomially?

3. Cut-free sequent calculus width is bounded by clause space. Can it be bounded in terms of monomial space in a relation similar to (6.1)? This is a good point to also mention that whether (6.1) can be improved to a linear inequality or there are examples where it is tight is unknown as well, and there do not seem to be strong indications

for which case is true.

4. We show that resolution width and monomial space cannot coincide. Whether they coincide up to polynomial factors however remains open, although it is speculated (cf. [42]) that this is not the case, and moreover, as it is the case for resolution width and clause space [12, 13], there is an $O(1)$ vs $\Omega(n/\log n)$ separation.

6.3 On the automatability of bounded-depth Frege systems

We showed the non-automatability of bounded-depth Frege system assuming $\mathsf{P} \neq \mathsf{NP}$. We do this, following [6], by constructing, given a CNF formula F, a formula $\mathrm{RRef}_{d,N}(F, s)$, and exhibiting a gap between the size of the shortest depth-d Frege refutations of $\mathrm{RRef}_{d,N}(F, s)$ when F is satisfiable and the size of the shortest depth-d Frege refutations of $\mathrm{RRef}_{d,N}(F, s)$ when F is not satisfiable.

To show the lower bound for depth-d Frege refutations of $\mathrm{RRef}_{d,N}(F, s)$ in the case F is not satisfiable, we employ the Furst-Saxe-Sipser switching lemma [34]. While sufficient for the purpose of showing non-automatability assuming $\mathsf{P} \neq \mathsf{NP}$, this can only give lower bounds of the form n^h, where h is a barely superconstant function of n. It would be nice to have an exponential lower bound. In particular, as in [6], an exponential lower bound would rule out the automatability of bounded-depth Frege systems in quasipolynomial time unless NP problems can be solved in quasipolynomial time, and their automatability in subexponential time unless NP problems can be solved in subexponential time.

$\mathrm{RRef}_{d,N}(F, s)$ consists of formulas of depth d. In particular, this does not preclude the possibility of bounded-depth Frege systems being automatable on refuting, say CNF formulas. A natural question is whether we could use CNFs, or at least formulas of constant depth, not depending on d, instead. Let us mention here that whether there is a constant depth formula exponentially separating depth-d from depth-$(d+1)$ Frege is open as well; currently, only a super-polynomial separation is known [43] (see also [47, Section 14.5]).

Moreover, the formulas $\mathrm{RRef}_{d,N}(F,s)$ are ad hoc and rather artificial. It would be nice if one could establish a lower bound for formulas $\mathrm{Ref}_d(F,s)$ for an unsatisfiable formula F, encoding the fact that there are depth-d refutations of F of size s (see Problem 2 in [57]), showing that proving lower bounds for a depth-d Frege system is hard within the system. The latter problem for a proof system is considered by Pudlák [57] to be a more important question than the question of whether the system is automatable. Note that a CNF encoding of $\mathrm{Ref}_d(F,s)$ is a candidate formula for the question of whether bounded-depth Frege systems for refuting CNFs are automatable, and a CNF encoding of the reflection principle $\mathrm{Sat}(F,v) \wedge \mathrm{Ref}_d(F,s)$, where $\mathrm{Sat}(F,v)$ encodes that v is an assignment satisfying F, is a candidate formula for the depth-d vs depth-$(d+1)$ Frege problem (see [57]).

Finally, the non-automatability result of [6] has been shown for cutting planes [39], Res(k) [37], and various algebraic proof systems [27]. As far as we know, two remaining open cases are the sum of squares and Sherali-Adams[1] proof systems.

1. The non-automatability result for Sherali-Adams reported in [27] has been retracted.

CHAPTER 7

REFERENCES

[1] Michael Alekhnovich, Eli Ben-Sasson, Alexander Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal of Computing*, 31:1184–1211, 2002.

[2] Michael Alekhnovich and Alexander Razborov. Resolution is not automatizable unless W[P] is tractable. *SIAM Journal of Computing*, 38:1347–1363, 2008.

[3] Noriko Arai. Relative efficency of propositional proof systems: resolution vs. cut-free LK. *Annals of Pure and Applied Logic*, 104:3–16, 2000.

[4] Noriko Arai, Toniann Pitassi, and Alasdair Urquhart. The complexity of analytic tableaux. *Journal of Symbolic Logic*, 71:777–790, 2006.

[5] Albert Atserias and Victor Dalmau. A combinatorial characterization of resolution width. *Journal of Computer and System Sciences*, 74:323–334, 2008.

[6] Albert Atserias and Moritz Müller. Automating resolution is NP-hard. *Journal of the ACM*, 67:31:1–31:17, 2020.

[7] Paul Beame, Chris Beck, and Russell Impagliazzo. Time-space trade-offs in resolution: Superpolynomial lower bounds for superlinear space. *SIAM Journal of Computing*, 45:1612–1645, 2016.

[8] Chris Beck, Jakob Nordström, and Bangsheng Tang. Some trade-off results for polynomial calculus: extended abstract. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 813–822, 2013.

[9] Arnold Beckmann and Samuel Buss. Separation results for the size of constant-depth propositional proofs. *Annals of Pure and Applied Logic*, 136:30–55, 2005.

[10] Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM Journal of Computing*, 38, 2009.

[11] Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of tree-like and general resolution. *Combinatorica*, 24:585–603, 2004.

[12] 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*, pages 709–718, 2008.

[13] 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*, pages 401–416, 2011.

[14] Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow—resolution made simple. *Journal of the ACM*, 48:149–169, 2001.

[15] Patrick Bennett, Ilario Bonacina, Nicola Galesi, Tony Huynh, Mike Molloy, and Paul Wollan. Space proof complexity for random 3-cnfs. *Information and Computation*, 255:165–176, 2017.

[16] Ilario Bonacina. Total space in resolution is at least width squared. In *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming*, volume 55, pages 56:1–56:13, 2016.

[17] Ilario Bonacina and Nicola Galesi. Pseudo-partitions, transversality and locality: a combinatorial characterization for the space measure in algebraic proof systems. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, pages 455–472, 2013.

[18] Ilario Bonacina and Nicola Galesi. A framework for space complexity in algebraic proof systems. *Journal of the ACM*, 62:23:1–23:20, 2015.

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

[20] Maria Luisa Bonet and Nicola Galesi. Optimality of size-width tradeoffs for resolution. *Computational Complexity*, 10:261–276, 2002.

[21] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz. On interpolation and automatization for frege systems. *SIAM Journal of Computing*, 29:1939–1967, 2000.

[22] Samuel Buss and Jakob Nordström. Proof complexity and SAT solving. In Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, chapter 7, pages 233–350. IOS Press, 2nd edition, 2021.

[23] 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*, pages 174–183, 1996.

[24] Stephen Cook and Robert Reckhow. On the lengths of proofs in the propositional calculus (preliminary version). In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing*, pages 135–148, 1974.

[25] Stephen Cook and Robert Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44:36–50, 1979.

[26] Stefan Dantchev and Søren Riis. On relativisation and complexity gap. In *Proceedings of the 12th Annual Conference of the EACSL*, pages 142–154, 2003.

[27] Susanna de Rezende, Mika Göös, Jakob Nordström, Toniann Pitassi, Robert Robere, and Dmitry Sokolov. Automating algebraic proof systems is NP-hard. In *Proccedings of the 53rd Annual ACM Symposium on Theory of Computing*, pages 209–222, 2021.

[28] Juan Luis Esteban, Nicola Galesi, and Jochen Messner. On the complexity of resolution with bounded conjunctions. *Theoretical Computer Science*, 321:347–370, 2004.

[29] Juan Luis Esteban and Jacobo Torán. Space bounds for resolution. *Information and Computation*, 171:84–97, 2001.

[30] Juan Luis Esteban and Jacobo Torán. A combinatorial characterization of treelike resolution space. *Information Processing Letters*, 87:295–300, 2003.

[31] Yuval Filmus, Massimo Lauria, Mladen Miksa, Jakob Nordström, and Marc Vinyals. Towards an understanding of polynomial calculus: New separations and lower bounds (extended abstract). In *Proceedings of the 40th International Colloquium on Automata, Languages, and Programming*, volume 7965, pages 437–448, 2013.

[32] Melvin Fitting. *Intuitionistic Logic Model Theory and Forcing*. North-Holland Publishing Company, 1969.

[33] Noah Fleming, Toniann Pitassi, and Robert Robere. Extremely deep proofs. In *13th Innovations in Theoretical Computer Science Conference*, volume 215, pages 70:1–70:23, 2022.

[34] Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17:13–27, 1984.

[35] Nicola Galesi, Leszek Kołodziejczyk, and Neil Thapen. Polynomial calculus space and resolution width. In *Proceedings of the 60th Annual IEEE Symposium on Foundations of Computer Science*, pages 1325–1337, 2019.

[36] Michal Garlík. Resolution lower bounds for refutation statements. In *Proccedings of the 44th International Symposium on Mathematical Foundations of Computer Science*, volume 138, pages 37:1–37:13, 2019.

[37] Michal Garlík. Failure of feasible disjunction property for k-DNF resolution and NP-hardness of automating it. *Electronic Colloqium on Computational Complexity*, 2020.

[38] Gerhard Gentzen. Untersuchungen über das logische schließen. I. *Mathematische Zeitschrift*, 39:176–210, 1934.

[39] Mika Göös, Sajin Koroth, Ian Mertz, and Toniann Pitassi. Automating cutting planes is NP-hard. In *Proccedings of the 52nd Annual ACM Symposium on Theory of Computing*, pages 68–77, 2020.

[40] Mika Göös and Toniann Pitassi. Communication lower bounds via critical block sensitivity. *SIAM Journal of Computing*, 47:1778–1806, 2018.

[41] Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.

[42] 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 Conference*, pages 233–248, 2012.

[43] Russell Impagliazzo and Jan Krajícek. A note on conservativity relations among bounded arithmetic theories. *Mathematical Logic Quarterly*, 48:375–377, 2002.

[44] Russell Impagliazzo, Pavel Pudlák, and Jirí Sgall. Lower bounds for the polynomial calculus and the gröbner basis algorithm. *Computational Complexity*, 8:127–144, 1999.

[45] Jan Krajíček. Lower bounds to the size of constant-depth propositional proofs. *Journal of Symbolic Logic*, 59:73–86, 1994.

[46] Jan Krajíček. *Bounded Arithmetic, Propositional Logic, and Complexity Theory*. Cambridge University Press, 1995.

[47] Jan Krajíček. *Proof Complexity*. Cambridge University Press, 2019.

[48] Massimo Lauria. A note about k-DNF resolution. *Information Processing Letters*, 137:33–39, 2018.

[49] Thomas Lengauer and Robert Tarjan. Asymptotically tight bounds on time-space trade-offs in a pebble game. *Journal of the ACM*, 29:1087–1130, 1982.

[50] Jakob Nordström. Pebble games, proof complexity, and time-space trade-offs. *Logical Methods in Computer Science*, 9, 2013.

[51] Jakob Nordström and Johan Håstad. Towards an optimal separation of space and length in resolution. *Theory of Computing*, 9:471–557, 2013.

[52] Theodoros Papamakarios. A super-polynomial separation between resolution and cut-free sequent calculus. In *Proceedings of the 48th International Symposium on Mathematical Foundations of Computer Science*, volume 272, pages 74:1–74:15, 2023.

[53] Theodoros Papamakarios. Depth-d frege systems are not automatable unless P=NP. In *Proceedings of the 39th Computational Complexity Conference*, 2024. to appear.

[54] Theodoros Papamakarios and Alexander Razborov. Space characterizations of complexity measures and size-space trade-offs in propositional proof systems. *Journal of Computer and System Sciences*, 137:20–36, 2023.

[55] Wolfgang Paul, Robert Tarjan, and James Celoni. Space bounds for a game on graphs. *Mathematical Systems Theory*, 10:239–251, 1977.

[56] Pavel Pudlák. On reducibility and symmetry of disjoint NP pairs. *Theoretical Computer Science*, 295:323–339, 2003.

[57] Pavel Pudlák. Reflection principles, propositional proof systems, and theories, 2020. arXiv:2007.14835.

[58] Pavel Pudlák and Russell Impagliazzo. A lower bound for DLL algorithms for k-SAT. In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 128–136, 2000.

[59] Alexander Razborov. A new kind of tradeoffs in propositional proof complexity. *Journal of the ACM*, 63:16:1–16:14, 2016.

[60] Alexander Razborov. On space and depth in resolution. *Computational Complexity*, 27:511–559, 2018.

[61] Robert Reckhow. *On the lengths of proofs in the propositional calculus*. PhD thesis, Department of Computer Science, University of Toronto, 1975.

[62] Søren Riis. A complexity gap for tree resolution. *Computational Complexity*, 10:179–209, 2001.

[63] Nathan Segerlind, Samuel Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for k-DNF resolution. *SIAM Journal of Computing*, 33:1171–1200, 2004.

[64] Michael Sipser. Borel sets and circuit complexity. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 61–69, 1983.

[65] Raymond Smullyan. *First-order Logic*. Dover, 1995.

[66] Grigori Tseitin. On the complexity of derivation in propositional calculus. *Studies in constructive mathematics and mathematical logic, part 2*, pages 115–125, 1968.

[67] Alasdair Urquhart. The complexity of propositional proofs. *Bulletin of Symbolic Logic*, 1:425–467, 1995.

[68] Alasdair Urquhart. The depth of resolution proofs. *Studia Logica*, 99:349–364, 2011.

[69] Arild Waaler and Lincoln Wallen. Tableaux for intuitionistic logics. In Marcello D'Agostino, Dov M. Gabbay, Reiner Hähnle, and Joachim Posegga, editors, *Handbook of Tableau Methods*, pages 255–296. Springer, 1999.

INDEX