Towards an Optimal Separation of Space and Length in Resolution

[Extended Abstract] [*]

Jakob Nordström[†]
Royal Institute of Technology (KTH)
SE-100 44 Stockholm, Sweden
jakobn@kth.se

Johan Håstad
Royal Institute of Technology (KTH)
SE-100 44 Stockholm, Sweden
johanh@kth.se

ABSTRACT

Most state-of-the-art satisfiability algorithms today are variants of the DPLL procedure augmented with clause learning. The main bottleneck for such algorithms, other than the obvious one of time, is the amount of memory used. In the field of proof complexity, the resources of time and memory correspond to the length and space of resolution proofs. There has been a long line of research trying to understand these proof complexity measures, as well as relating them to the width of proofs, i.e., the size of the largest clause in the proof, which has been shown to be intimately connected with both length and space. While strong results have been proven for length and width, our understanding of space is still quite poor. For instance, it has remained open whether the fact that a formula is provable in short length implies that it is also provable in small space (which is the case for length versus width), or whether on the contrary these measures are completely unrelated in the sense that short proofs can be arbitrarily complex with respect to space.

In this paper, we present some evidence that the true answer should be that the latter case holds and provide a possible roadmap for how such an optimal separation result could be obtained. We do this by proving a tight bound of $\Theta(\sqrt{n})$ on the space needed for so-called pebbling contradictions over pyramid graphs of size n.

Also, continuing the line of research initiated by (Ben-Sasson 2002) into trade-offs between different proof complexity measures, we present a simplified proof of the recent length-space trade-off result in (Hertel and Pitassi 2007), and show how our ideas can be used to prove a couple of other exponential trade-offs in resolution.

[*]The full version is available as arXiv.org Technical Report 0803.0661 on `http://arxiv.org/abs/0803.0661`.

[†]Research supported in part by grants from the foundations *Johan och Jakob Söderbergs stiftelse* and *Sven och Dagmar Saléns stiftelse*.

Categories and Subject Descriptors: F.4.1[Mathematical Logic and Formal Languages]: Mathematical Logic — *proof theory*; F.1.3[Computation by Abstract Devices]: Complexity Measures and Classes —*Relations among complexity measures*; I.2.3[Artificial Intelligence]: Deduction and Theorem Proving

General Terms: Theory

Keywords: Proof complexity, resolution, length, space, separation, lower bound, pebbling

1. INTRODUCTION

Ever since the ground-breaking NP-completeness result of Cook [13], the problem of deciding whether a given propositional logic formula in conjunctive normal form (CNF) is satisfiable or not has been on center stage in Theoretical Computer Science. In more recent years, SATISFIABILITY has gone from a problem of mainly theoretical interest to a practical approach for solving applied problems. Although all known Boolean satisfiability solvers (SAT-solvers) have exponential running time in the worst case, enormous progress in performance has led to satisfiability algorithms becoming a standard tool for solving a large number of real-world problems such as hardware and software verification, experiment design, circuit diagnosis, and scheduling.

A somewhat surprising aspect of this development is that the most successful SAT-solvers to date are still variants of the resolution-based so-called DPLL procedure [16, 17] augmented with *clause learning*. For instance, the great majority of the best algorithms at the 2007 round of the international SAT competitions [30] fit this description. DPLL procedures perform a recursive backtrack search in the space of partial truth value assignments. The idea behind clause learning, or *conflict-driven learning*, is that at each failure (backtrack) point in the search tree, the system derives a reason for the inconsistency in the form of a new clause and then adds this clause to the original CNF formula ("learning" the clause). This can save a lot of work later on in the search, when some other partial truth value assignment fails for similar reasons. The main bottleneck for this approach, other than the obvious one of time, is the amount of memory used by the algorithms. Thus, understanding time and memory requirements for clause learning algorithms, and how these requirements are related to one another, is a question of great practical importance. We refer to, e.g., [4, 24] for a more detailed discussion of clause learning (and SAT-solving in general) with examples of applications.

The study of proof complexity originated with the seminal paper of Cook and Reckhow [14]. In its most general form, a proof system for a language L is a predicate $P(x, \pi)$, computable in time polynomial in $|x|$ and $|\pi|$, such that for all $x \in L$ there is a string π (a *proof*) for which $P(x, \pi) = 1$, whereas for any $x \notin L$ it holds for all strings π that $P(x, \pi) = 0$. A proof system is said to be polynomially bounded if for every $x \in L$ there is a proof π_x of size at most polynomial in $|x|$. A *propositional proof system* is a proof system for the language of tautologies in propositional logic.

From a theoretical point of view, one important motivation for proof complexity is the intimate connection with the fundamental question of P versus NP. Since NP is exactly the set of languages with polynomially bounded proof systems, and since TAUTOLOGY can be seen to be the dual problem of SATISFIABILITY, we have the famous theorem of [14] that $\mathsf{NP} = \mathsf{co\text{-}NP}$ if and only if there exists a polynomially bounded propositional proof system. Thus, if it could be shown that there are no polynomially bounded propositional proof systems, $\mathsf{P} \neq \mathsf{NP}$ would follow as a corollary since P is closed under complement. One way of approaching this distant goal is to study stronger and stronger proof systems and try to prove superpolynomial lower bounds on proof size. However, although great progress has been made in the last couple of decades for a variety of proof systems, it seems that we are still very far from fully understanding the reasoning power of even quite simple ones.

A second important motivation is that, as was mentioned above, designing efficient algorithms for proving tautologies (or, equivalently, testing satisfiability), is a very important problem not only in the theory of computation but also in applied research and industry. All automated theorem provers, regardless of whether they actually produce a written proof, explicitly or implicitly define a system in which proofs are searched for and rules which determine what proofs in this system look like. Proof complexity analyzes what it takes to simply write down and verify the proofs that such an automated theorem-prover might find, ignoring the computational effort needed to actually find them. Thus a lower bound for a proof system tells us that any algorithm, even an optimal (non-deterministic) one making all the right choices, must necessarily use at least the amount of a certain resource specified by this bound. In the other direction, theoretical upper bounds on some proof complexity measure give us hope of finding good proof search algorithms with respect to this measure, provided that we can design algorithms that search for proofs in the system in an efficient manner. For DPLL procedures with clause learning, the time and memory resources used are measured by the *length* and *space* of proofs in the resolution proof system.

The field of proof complexity also has rich connections to cryptography, artificial intelligence and mathematical logic. Two good surveys providing more details are [3, 31].

1.1 Previous Work

Any formula in propositional logic can be converted to a CNF formula that is only linearly larger and is unsatisfiable if and only if the original formula is a tautology. Therefore, any sound and complete system for refuting CNF formulas can be considered as a general propositional proof system.

Perhaps the single most studied proof system for propositional logic, *resolution*, is such a system that produces proofs of the unsatisfiability of CNF formulas. Resolution appeared in [10] and began to be investigated in connection with automated theorem proving in the 1960s [16, 17, 28]. Because of its simplicity—there is only one derivation rule—and because all lines in a proof are clauses, resolution readily lends itself to proof search algorithms.

Being so simple and fundamental, resolution was also a natural target to attack when developing methods for proving lower bounds in proof complexity. In this context, it is most straightforward to prove bounds on the *length* of refutations, i.e., the number of clauses, rather than on the total size. The length and size measures are easily seen to be polynomially related. In 1968, Tseitin [34] presented a superpolynomial lower bound on length for a restricted form of resolution, called *regular* resolution, but it was not until almost 20 years later that Haken [21] proved the first superpolynomial lower bound for general resolution. This weakly exponential bound of Haken has later been followed by many other strong results, among others truly exponential lower bound on resolution refutation length for different formula families in, for instance, [9, 12, 35].

A second complexity measure for resolution is the *width*, i.e., the maximal size of a clause in the refutation. Ben-Sasson and Wigderson [9] showed that the minimal width $W(F \vdash 0)$ of any refutation of a k-CNF formula F is bounded from above by the minimal refutation length $L(F \vdash 0)$ by

$$W(F \vdash 0) = \mathrm{O}\bigl(\sqrt{n \log L(F \vdash 0)}\bigr) \ , \qquad (1)$$

where n is the number of variables in F. Since it is also easy to see that refutations of polynomial-size formulas in small width must necessarily be short (for the reason that $(2 \cdot \#\text{variables})^w$ is an upper bound on the total number of distinct clauses of width w), the result in [9] can be interpreted as saying roughly that there exists a short refutation of F if and only if there exists a (reasonably) narrow refutation of F. This gives rise to a natural proof search heuristic: to find a short refutation, search for refutations in small width. It was shown in [8] that there are formula families for which this heuristic exponentially outperforms any DPLL procedure regardless of branching function.

The formal study of *space* in resolution was initiated by Esteban and Torán [18, 32]. Intuitively, the space $Sp(\pi)$ of a refutation π is the maximal number of clauses one needs to keep in memory while verifying the refutation, and the space $Sp(F \vdash 0)$ of refuting F is defined as the minimal space of any refutation of F. A number of upper and lower bounds for refutation space in resolution and other proof systems were subsequently presented in, for example, [1, 7, 19]. Just as for width, the minimum space of refuting a formula can be upper-bounded by the size of the formula. Somewhat unexpectedly, however, it also turned out that the lower bounds on resolution refutation space for several formula families exactly matched previously known lower bounds on refutation width. Atserias and Dalmau [2] showed that this was not a coincidence, but that the inequality

$$W(F \vdash 0) \leq Sp(F \vdash 0) + \mathrm{O}(1) \qquad (2)$$

holds for any k-CNF formula F, where the (small) constant term depends on k. In [26], the first author proved that the inequality (2) is asymptotically strict by exhibiting a k-CNF formula family of size $\mathrm{O}(n)$ refutable in width $W(F_n \vdash 0) = \mathrm{O}(1)$ but requiring space $Sp(F_n \vdash 0) = \Theta(\log n)$.

The space measure discussed above is known as *clause space*. A less well-studied space measure, introduced by

Alekhnovich et al. [1], is *variable space*, which counts the maximal number of variable occurrences that must be kept in memory simultaneously. Ben-Sasson [5] used this measure to obtain a trade-off result for clause space versus width in resolution, proving that there are k-CNF formulas F_n that can be refuted in constant clause space and constant width, but for which any refutation π_n must have $Sp(\pi_n) \cdot W(\pi_n) = \Omega(n/\log n)$. More recently, Hertel and Pitassi [22] showed that there are CNF formulas F_n for which any refutation in minimal variable space $VarSp(F_n \vdash 0)$ must have exponential length, but by adding just 3 extra units of storage one can instead get a resolution refutation in linear length.

1.2 Questions Left Open by Previous Research

Despite all the research that has gone into understanding the resolution proof system, a number of fundamental questions still remain unsolved. We touch briefly on two such questions below, and then discuss a third one, which is the main focus of this paper, in somewhat more detail.

Equation (1) says that short refutation length implies narrow refutation width. Combining Equation (2) with the observation above that narrow refutations are trivially short, we get a similar statement that small refutation clause space implies short refutation length. Note, however, that this does *not* mean that there is a refutation that is both short and narrow, or that any small-space refutation must also be short. The reason is that the resolution refutations on the left- and right-hand sides of (1) and (2) need not (and in general will not) be the same one.

In view of the minimum-width proof search heuristic mentioned above, an important question is whether short refutation length of a formula entails that there is a refutation that is both short and narrow. Also, it would be interesting to know if small space of a refutation implies that it is short. It is not known whether there are such connections or whether on the contrary there exist some kind of trade-off phenomena here similar to the one for space and width in [5].

A third, even more interesting problem is to clarify the relation between length and clause space. For width, rewriting the bound in (1) in terms of the number of clauses $|F_n|$ instead of the number of variables we get that that if the width of refuting F_n is $\omega\bigl(\sqrt{|F_n| \log|F_n|}\bigr)$, then the length of refuting F_n must be superpolynomial in $|F_n|$. This is known to be almost tight, since [11] shows that there is a k-CNF formula family $\{F_n\}_{n=1}^{\infty}$ with $W(F_n \vdash 0) = \Omega\bigl(\sqrt[3]{|F_n|}\bigr)$ but $L(F_n \vdash 0) = \mathrm{O}(|F_n|)$. Hence, formula families refutable in polynomial length can have somewhat wide minimum-width refutations, but not arbitrarily wide ones.

What does the corresponding relation between space and length look like? The inequality (2) tells us that any correlation between length and clause space cannot be tighter than the correlation between length and width, so in particular we get from the previous paragraph that k-CNF formulas refutable in polynomial length may have at least "somewhat spacious" minimum-space refutations. At the other end of the spectrum, given any resolution refutation π of F in length L it can be proven using results from [18, 23] that $Sp(\pi) = \mathrm{O}(L/\log L)$. This gives an upper bound on any possible separation of the two measures. But is there a Ben-Sasson–Wigderson kind of upper bound on clause space in terms of length similar to (1)? Or are length and space on the contrary unrelated in the sense that there exist k-CNF formulas F_n with short refutations but maximal possible refutation space $Sp(F_n \vdash 0) = \Omega\bigl(L(F_n \vdash 0)/\log L(F_n \vdash 0)\bigr)$ in terms of length?

We note that for the restricted case of so-called tree-like resolution, [18] showed that there is a tight correspondence between length and clause space, exactly as for length versus width. The case for general resolution has been discussed in, for instance, [5, 19, 33], but there seems to have been no consensus on what the right answer should be. However, these papers identify a plausible formula family for answering the question, namely so-called *pebbling contradictions* defined in terms of pebble games over directed acyclic graphs.

1.3 Our Contribution

The main result in this paper provides some evidence that the true answer to the question about the relationship between clause space and length is more likely to be at the latter extreme, i.e., that the two measures can be separated in the strongest sense possible. More specifically, as a step towards this goal we prove an asymptotically tight bound on the space of refuting pebbling contradictions over pyramids.

THEOREM 1. *The clause space of refuting pebbling contradictions over pyramid graphs of height h in resolution grows as $\Theta(h)$, provided that the number of variables per vertex in the pebbling contradictions is at least* 2.

This yields the first result separating clause space and length that is not a consequence of a corresponding lower bound on width, as well as an exponential improvement of the separation of clause space and width in [26].

COROLLARY 2. *For all $k \geq 4$, there is a family $\{F_n\}_{n=1}^{\infty}$ of k-CNF formulas of size $\Theta(n)$ that can be refuted in resolution in length $L(F_n \vdash 0) = \mathrm{O}(n)$ and width $W(F_n \vdash 0) = \mathrm{O}(1)$ but require clause space $Sp(F_n \vdash 0) = \Theta(\sqrt{n})$.*

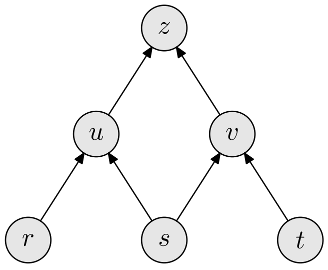
In addition to our main result, we also make the the observation that the proof of the recent trade-off result in [22] can be greatly simplified, and the parameters slightly improved. Using similar ideas, we can also prove exponential trade-offs for length with respect to clause space and width. Namely, we show that there are k-CNF formulas such that if we insist on finding the resolution refutation in smallest clause space or smallest width, respectively, then we have to pay with an exponential increase in length. We state the theorem only for length versus clause space.

THEOREM 3. *There is a family $\{F_n\}_{n=1}^{\infty}$ of k-CNF formulas of size $\Theta(n)$ such that:*

- *The minimal clause space of refuting F_n in resolution is $Sp(F_n \vdash 0) = \Theta\bigl(\sqrt[3]{n}\bigr)$.*
- *Any resolution refutation $\pi : F_n \vdash 0$ in minimal clause space must have length $L(\pi) = \exp\bigl(\Omega\bigl(\sqrt[3]{n}\bigr)\bigr)$.*
- *There are refutations $\pi' : F_n \vdash 0$ in asymptotically minimal clause space $Sp(\pi') = \mathrm{O}\bigl(Sp(F_n \vdash 0)\bigr)$ and length $L(\pi') = \mathrm{O}(n)$, i.e., linear in the formula size.*

A theorem of exactly the same form can be proven for length versus width as well.

The outline of this paper is as follows. We give a brief, informal review of the preliminaries in Section 2. In Section 3, we sketch the proofs of our results. Most of the proofs are fairly involved, however, and due to space considerations we refer to [27] for the technical details. We conclude in Section 4 by giving suggestions for further research.



$$
\begin{aligned}
& (x(r)_1 \lor x(r)_2) \\
\land\; & (x(s)_1 \lor x(s)_2) \\
\land\; & (x(t)_1 \lor x(t)_2) \\
\land\; & (\overline{x(r)}_1 \lor \overline{x(s)}_1 \lor x(u)_1 \lor x(u)_2) \\
\land\; & (\overline{x(r)}_1 \lor \overline{x(s)}_2 \lor x(u)_1 \lor x(u)_2) \\
\land\; & (\overline{x(r)}_2 \lor \overline{x(s)}_1 \lor x(u)_1 \lor x(u)_2) \\
\land\; & (\overline{x(r)}_2 \lor \overline{x(s)}_2 \lor x(u)_1 \lor x(u)_2) \\
\land\; & (\overline{x(s)}_1 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2) \\
\land\; & (\overline{x(s)}_1 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2) \\
\land\; & (\overline{x(s)}_2 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2) \\
\land\; & (\overline{x(s)}_2 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2) \\
\land\; & (\overline{x(u)}_1 \lor \overline{x(v)}_1 \lor x(z)_1 \lor x(z)_2) \\
\land\; & (\overline{x(u)}_1 \lor \overline{x(v)}_2 \lor x(z)_1 \lor x(z)_2) \\
\land\; & (\overline{x(u)}_2 \lor \overline{x(v)}_1 \lor x(z)_1 \lor x(z)_2) \\
\land\; & (\overline{x(u)}_2 \lor \overline{x(v)}_2 \lor x(z)_1 \lor x(z)_2) \\
\land\; & \overline{x(z)}_1 \\
\land\; & \overline{x(z)}_2
\end{aligned}
$$

Figure 1: The pebbling contradiction $Peb^2_{\Pi_2}$ for the pyramid graph Π_2 of height 2.

2. PRELIMINARIES

A *resolution refutation* of a CNF formula F can be viewed as a sequence of derivation steps on a blackboard. In each step we may write a clause from F on the blackboard (an *axiom* clause), erase a clause from the blackboard or derive some new clause implied by the clauses currently written on the blackboard. The refutation ends when we reach the contradictory empty clause. The *length* of a resolution refutation is the number of distinct clauses in the refutation, the *width* is the size of the largest clause in the refutation, and the *clause space* is the maximum number of clauses on the blackboard simultaneously. We write $L(F \vdash 0)$, $W(F \vdash 0)$ and $Sp(F \vdash 0)$ to denote the minimum length, width and clause space, respectively, of any resolution refutation of F.

The *pebble game* played on a directed acyclic graph (DAG) G models the calculation described by G, where the sources contain the input and non-source vertices specify operations on the values of the predecessors. Placing a pebble on a vertex v corresponds to storing in memory the partial result of the calculation described by the subgraph rooted at v. Removing a pebble from v corresponds to deleting the partial result of v from memory. A *pebbling* of G is a sequence of moves starting with the graph empty and ending with all vertices empty except for a pebble on the (unique) sink vertex. The *cost* of a pebbling is the maximal number of pebbles used simultaneously at any point in time during the pebbling. The *pebbling price* of G is the minimum cost of any pebbling, i.e., the minimum number of registers required to perform the complete calculation described by G.

The pebble game on a DAG G can be encoded as an unsatisfiable CNF formula Peb^d_G, a so-called *pebbling contradiction* of degree d, as follows (see Figure 1 for an example):

- Associate d variables $x(v)_1, \ldots, x(v)_d$ with each vertex v (in Figure 1 we have $d = 2$).
- Specify that all sources have at least one true variable, for example, the clause $x(r)_1 \lor x(r)_2$ for the vertex r.
- Add clauses propagating the truth from predecessors to successors (e.g. clauses 4–7 in Figure 1 say that $(x(r)_1 \lor x(r)_2) \land (x(s)_1 \lor x(s)_2) \rightarrow (x(u)_1 \lor x(u)_2)$.
- To get a contradiction, conclude with $\overline{x(z)}_1 \land \cdots \land \overline{x(z)}_d$ where z is the sink of the DAG.

We will need the observation from [8] that a pebbling contradiction of degree d over a graph with n vertices can be refuted by resolution in length $\mathrm{O}(d^2 \cdot n)$ and width $\mathrm{O}(d)$.

3. OVERVIEW OF PROOFS

Pebble games have been used extensively as a tool to prove time and space lower bounds and trade-offs for computation. Loosely put, a lower bound for the pebbling price of a graph says that although the computation that the graph describes can be performed quickly, it requires large space. Our hope is that when we encode pebble games in terms of CNF formulas, these formulas inherit the same properties as the underlying graphs. That is, if we pick a DAG G with high pebbling price, since the corresponding pebbling contradiction encodes a calculation which needs a lot of memory we would like to try to argue that any resolution refutation of this formula should require large space. Then a separation result would follow since we already know from [8] that the formula can be refuted in short length.

3.1 Proof Idea for Space Bound

More specifically, what we would like to do is to establish a connection between resolution refutations of pebbling contradictions on the one hand, and the so-called *black-white pebble game* [15] modelling the non-deterministic computations described by the underlying graphs on the other. Our belief is that the resolution proof system should have to conform to the combinatorics of the pebble game in the sense that from any refutation of a pebbling contradiction Peb^d_G we should be able to extract a pebbling of the DAG G.

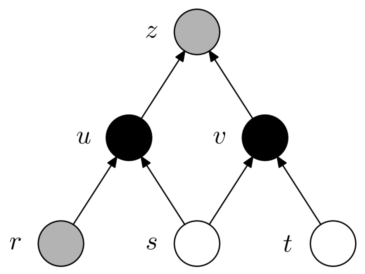
Ideally, we would like to give a proof of a lower bound on the resolution refutation space of pebbling contradictions along the following lines:

1. First, find a natural interpretation of sets of clauses currently "on the blackboard" in a refutation of the formula Peb^d_G in terms of black and white pebbles on the vertices of the DAG G.
2. Then, prove that this interpretation captures the pebble game in the following sense: for any resolution refutation of Peb^d_G, looking at consecutive sets of clauses on the blackboard and considering the corresponding sets of pebbles we get a black-white pebbling of G.
3. Finally, show that the interpretation captures clause space in the sense that if the content of the blackboard induces N pebbles on the graph, then there must be at least N clauses on the blackboard.

Combining the above with known lower bounds on the pebbling price of G, this would imply a lower bound on the

$$\begin{bmatrix} x(u)_1 \lor x(u)_2 \\ \overline{x(s)}_1 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_1 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(v)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(v)_2 \end{bmatrix}$$

(a) Clauses on blackboard.



(b) Corresponding pebbles in the graph.

Figure 2: Example of intuitive correspondence between sets of clauses and pebbles.

refutation space of pebbling contradictions. The separation from length and width would then follow from the fact that pebbling contradictions are known to be refutable in linear length and constant width if d is fixed.

Unfortunately, this idea does not quite work. In the next subsection, we describe the modifications that we are forced to make, and show how we can make the bits and pieces of our construction fit together to yield Theorem 1 and Corollary 2 for the special case of pyramid graphs.

3.2 Overview of Formal Proof of Space Bound

The black-white pebble game played on a DAG G can be viewed as a way of proving the end result of the calculation described by G. Black pebbles denote proven partial results of the computation. White pebbles denote assumptions about partial results which have been used to derive other partial results (i.e., black pebbles), but these assumptions will have to be verified for the calculation to be complete. The final goal is a black pebble on the sink z and no other pebbles in the graph, corresponding to an unconditional proof of the end result of the calculation.

Translating this to pebbling contradictions, it turns out that a fruitful way to think of a black pebble on v is that it should correspond to truth of the disjunction $\bigvee_{i=1}^{d} x(v)_i$ of all positive literals over v, or to "truth of v". With this correspondence it is straightforward to translate a pebbling of G using only black pebbles into refutation of the pebbling contradiction Peb_G^d. The only observation needed is that if we have derived the clauses $\bigvee_{i=1}^{d} x(s)_i$ and $\bigvee_{i=1}^{d} x(t)_i$ for the two predecessors s and t of v, then by downloading the axioms saying that truth propagates from s and t to v we can derive $\bigvee_{i=1}^{d} x(v)_i$. The correspondence here is quite close in that the space used by the refutation is at most an additive constant larger than the number of black pebbles used.

When we look at pebblings involving also white pebbles the translation gets slightly more complicated. White pebbles enable us to place black pebbles "in the middle" of the DAG without first having to pebble bottom-up from the sources. For instance, if we white-pebble u and v in Figure 1, we can then place a black pebble on their common successor z. Next, the white pebble on, say, v can be eliminated by placing white pebbles on the predecessors s and t, allowing the pebble on v to be removed. A resolution derivation can mimic these pebbling moves by writing all axioms $\overline{x(u)}_i \lor \overline{x(v)}_j \lor \bigvee_{n=1}^{d} x(z)_n$ and $\overline{x(s)}_i \lor \overline{x(r)}_j \lor \bigvee_{n=1}^{d} x(v)_n$ on the blackboard and then using all these clauses to derive $\overline{x(u)}_i \lor \overline{x(s)}_j \lor \overline{x(r)}_l \lor \bigvee_{n=1}^{d} x(z)_n$ for $i, j, l \in [d]$. As it happens, it is possible to translate any black-white pebbling to a refutation in this way (modulo some technical details), but the reduction is not as tight as in the case of a black-pebbles-only pebbling. As we can see from the example above, this naive translation can transform N white pebbles into a set of clauses of size d^N.

The key to our argument, however, is a translation in the other direction. We want to start with a resolution refutation and produce a black-white pebbling and then use the existing lower bound machinery for the black-white pebble game to get a lower bound on clause space. Let us first try to give the intuition behind our translation, and then discuss some technical complications that arise and how we adapt our construction to cope with these problems.

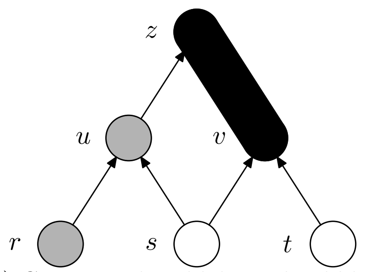
For black pebbles, we can reuse the ideas above for transforming pebblings into refutations. If the clause $\bigvee_{i=1}^{d} x(v)_i$ is implied by the current content of the blackboard, we will let this correspond to a black pebble on v. A white pebble in a pebbling is a "debt" that has to be paid. It is difficult to see how any clause could be a liability in the same way and therefore no set of clauses corresponds naturally to isolated white pebbles. But if we think of white pebbles as assumptions that allow us to place black pebbles higher up in the DAG, it makes sense to say that if the content of the blackboard conditionally implies $\bigvee_{i=1}^{d} x(v)_i$ given that we also assume the set of clauses $\left\{\bigvee_{i=1}^{d} x(w)_i \middle| w \in W\right\}$ for some vertex set W, then this could be interpreted as a black pebble on v and white pebbles on the vertices in W.

Using this correspondence, we can translate sets of clauses into black and white pebbles in a way that fits nicely with the resolution derivations sketched above. To give a concrete example, the clauses in Figure 2(a) correspond to the pebbles in Figure 2(b). To see this, note that if we assume $x(s)_1 \lor x(s)_2$ and $x(t)_1 \lor x(t)_2$, this assumption together with the clauses on the blackboard imply $x(v)_1 \lor x(v)_2$, so v is black-pebbled and s and t are white-pebbled in Figure 2(b). The vertex u is also black since $x(u)_1 \lor x(u)_2$ certainly is implied by the blackboard.

The problem is that refutations can derive clauses that cannot be translated, at least not naturally, to pebbles in the way indicated above. A particularly dangerous sitation is when clauses are derived that are the disjunction of positive literals from different vertices. Such clauses do not appear to be very useful, but nevertheless we have to model them in some way. To see why, consider the following example. Starting from the blackboard in Figure 2(a), a refutation could add the axioms $\overline{x(u)}_i \lor \overline{x(v)}_2 \lor x(z)_1 \lor x(z)_2$ for $i = 1, 2$, derive the clauses $\overline{x(s)}_i \lor \overline{x(t)}_j \lor x(v)_1 \lor x(z)_1 \lor x(z)_2$ for $i, j = 1, 2$, and then erase $x(u)_1 \lor x(u)_2$ to save space, resulting in

$$\begin{bmatrix} \overline{x(s)}_1 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(z)_1 \lor x(z)_2 \\ \overline{x(s)}_1 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(z)_1 \lor x(z)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_1 \lor x(v)_1 \lor x(z)_1 \lor x(z)_2 \\ \overline{x(s)}_2 \lor \overline{x(t)}_2 \lor x(v)_1 \lor x(z)_1 \lor x(z)_2 \end{bmatrix}$$

(a) New set of clauses on blackboard.



(b) Corresponding blobs and pebbles.

Figure 3: Intepreting sets of clauses as black blobs and white pebbles.

the blackboard in Figure 3(a). This blackboard does not correspond to any pebbles under our tentative translation. However, the clauses can easily be used to derive something that does. For instance, writing down all axioms $\overline{x(u)}_i \lor \overline{x(v)}_j \lor x(z)_1 \lor x(z)_2$, $i, j = 1, 2$, we get that the truth of s, t, and u implies the truth of z. We have decided to interpret such a set of clauses as a black pebble on z and white pebbles on s, t, and u, but this pebble configuration cannot arise out of nothing in an empty DAG.

Although it is perhaps hard to see from such a small example, this turns out to be a serious problem. There appears to be no way that we can interpret sets of clauses as in Figure 3(a) in terms of black and white pebbles without making some component in the proof idea in Section 3.1 break down. Instead, what we do is to invent a new pebble game, with white pebbles just as before, but with black *blobs* that can cover multiple vertices instead of single-vertex black pebbles. A blob on a vertex set V can be thought of as truth of some vertex $v \in V$. The clauses in Figure 3(a) are consequently translated into white pebbles on s and t, as before, and a black blob covering both v and z in Figure 3(b).

We use this blob-pebble game to build a lower bound proof as outlined in Section 3.1. First, we establish that for a fairly general class of graphs, any refutation of a pebbling contradiction can be interpreted as a blob-pebbling on the DAG in terms of which this pebbling contradiction is defined.

THEOREM 4. *Let Peb^d_G denote any pebbling contradiction over a layered DAG G. Then there is a translation from sets of clauses derived from Peb^d_G into sets of black blobs and white pebbles in G such that any refutation π of Peb^d_G corresponds to a blob-pebbling $\mathcal{P}_\pi$ of G under this translation.*

PROOF OUTLINE. If there are vertex sets B and W with $B \cap W = \emptyset$ and a subset $\mathbb{C}$ of blackboard clauses such that

$$\mathbb{C} \cup \left\{ \bigvee_{i=1}^d x(w)_i \middle| w \in W \right\} \vDash \bigvee_{v \in B} \bigvee_{i=1}^d x(v)_i \qquad (3)$$

but this implication does not hold for any subset of B, W, or $\mathbb{C}$, the blackboard *induces* a black blob on B *supported* by white pebbles on W. This *subconfiguration* of pebbles is denoted $[B]\langle W \rangle$. The pebble configuration corresponding to the blackboard is the set $\mathbb{S}$ of all induced subconfigurations.

When an axiom clause $\overline{x(s)}_i \lor \overline{x(t)}_j \lor \bigvee_{l=1}^d x(v)_l$ is written on the blackboard, we match this (if needed) by a black blob on v and white pebbles on s and t, i.e., by the subconfiguration $[v]\langle s, t \rangle$. This is an *introduction* move, and it corresponds to the rules for black and white pebble placement in the standard pebble game. See Figure 4(b) for an example. If v is a source, we get the subconfiguration $[v]\langle \emptyset \rangle$.

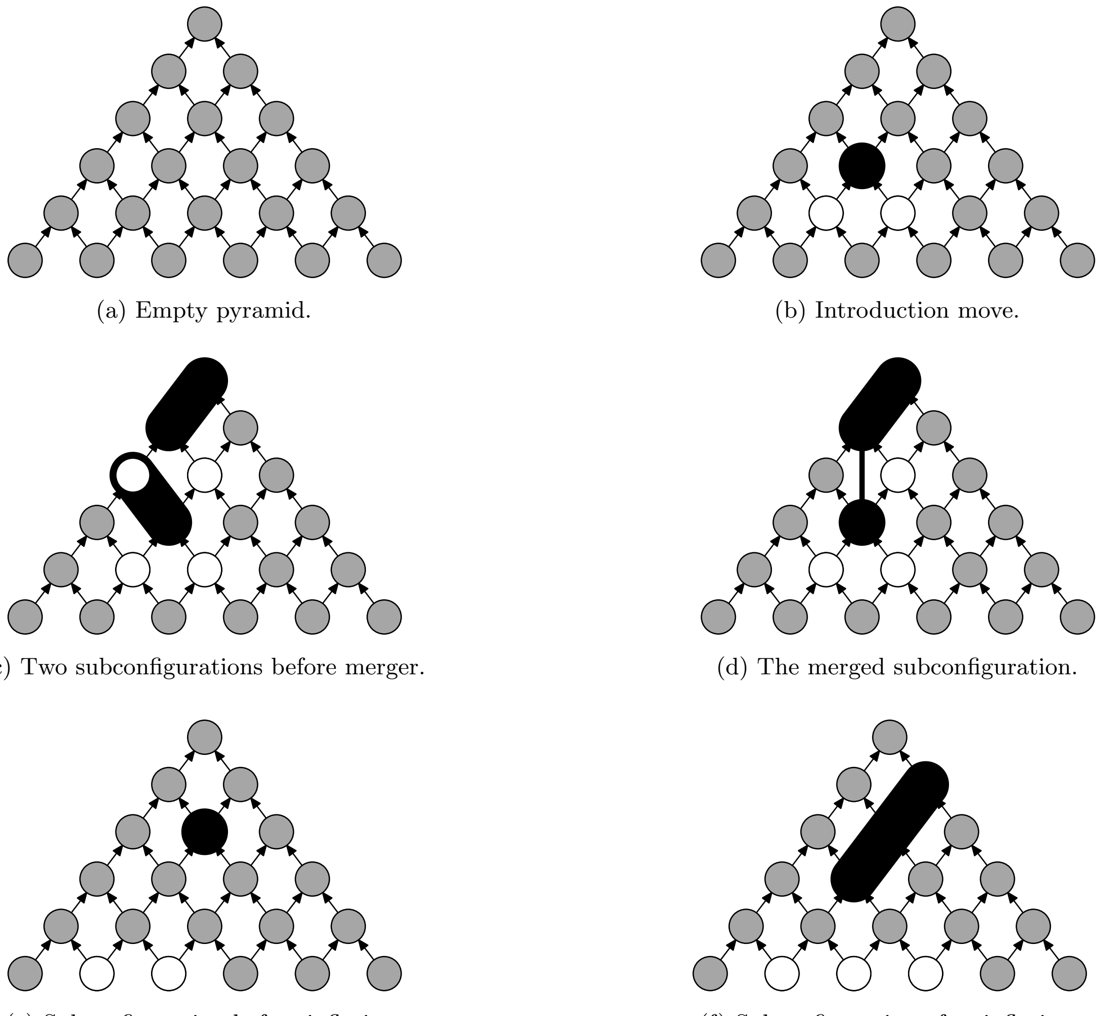
Because of the new axiom, there can also appear other blobs and pebbles. We show that they can all be explained in terms of *mergers* of existing subconfigurations $[B_1]\langle W_1 \rangle$ and $[B_2]\langle W_2 \rangle$ such that $B_1 \cap W_2 = \emptyset$ and $B_2 \cap W_1 = \{v^*\}$ for some vertex v^* into $\left[(B_1 \cup B_2) \setminus \{v^*\}\right]\left\langle (W_1 \cup W_2) \setminus \{v^*\} \right\rangle$ as in the transition from Figure 4(c) to Figure 4(d). To see why the merger rule is defined the way it is, note that if the blackboard plus truth of all $w \in W_i$ implies the truth of some $v \in B_i$ for $i = 1, 2$, then certainly the blackboard plus truth of all $w \in (W_1 \cup W_2) \setminus \{v^*\}$ implies the truth of some vertex either in B_1 (if v^* is true) or $B_2 \setminus \{v^*\}$ (if v^* is false).

When new clauses are derived, we expect nothing to happen since these clauses are implied by what is already on the blackboard. It can be the case, though, that clauses are derived that are in some sense "weaker" than what is implied by the blackboard, but if so we can make *inflation* moves that inflate blobs to cover more vertices and/or add white pebbles. See Figures 4(e) and 4(f) for an example.

Finally, blobs and pebbles can disappear when clauses are erased from the blackboard. A problem here is that we can get erasures of white pebbles, which is not acceptable in the black-white pebble game. However, by associating white pebbles with black blobs in subconfigurations $[B]\langle W \rangle$, we can allow erasures of white pebbles W as long as the blobs B that they support are erased as well. Thus, one cannot erase individual pebbles but only entire subconfigurations. In this way (sweeping the technical details under the rug) we can associate a blob-pebbling $\mathcal{P}_\pi$ with any refutation π. $\square$

The next step is to design a cost function for black blobs and white pebbles so that the cost of the blob-pebbling $\mathcal{P}_\pi$ in Theorem 4 is related to the space of the resolution refutation π. Consider first two special cases. If a clause set induces N disjoint blobs without any supporting white pebbles, it is not hard to prove that the size of this set is at least N. This is clearly tight, so the cost of a single blob can never exceed one. And if $\mathbb{C} \cup \left\{ \bigvee_{i=1}^d x(w)_i \middle| w \in W \right\}$ implies $\bigvee_{v \in B} \bigvee_{i=1}^d x(v)_i$ with the vertex set W chosen minimal so that this implication still holds, it can be shown that $|\mathbb{C}| > (d-1)|W|$ (so here we need $d > 1$). This follows from the fact that a minimally unsatisfiable CNF formula over N variables must contain strictly more than N clauses.

In general, matters will be more complicated. Distinct blobs will not be disjoint, and therefore cannot always all count towards the cost. Also, black blobs and white pebbles from different subconfigurations can intersect in tricky ways. We do not have the space to elaborate on why this works, but if we only allow blobs B that are chains (i.e., where all $v \in B$ are ordered topologically), look at the lowest vertex


(a) Empty pyramid.

(b) Introduction move.

(c) Two subconfigurations before merger.

(d) The merged subconfiguration.

(e) Subconfiguration before inflation.

(f) Subconfiguration after inflation.

Figure 4: Examples of moves in the blob-pebble game.

in each blob and count the number of distinct such vertices, and also only charge for white pebbles in $[B]\langle W\rangle$ that are located below the bottom vertex of B, we get the required result. Once we have defined the cost function in this way, the proof that blob-pebbling cost yields a lower bound on clause space is similar to the proof of Theorem 19 in [26].

THEOREM 5. *If π is a refutation of a pebbling contradiction of degree $d > 1$, the cost of the associated blob-pebbling $\mathcal{P}_\pi$ is bounded by the space of π by $\mathit{cost}(\mathcal{P}_\pi) \leq Sp(\pi) + \mathrm{O}(1)$.*

Finally, we need lower bounds on blob-pebbling price. Because of the inflation rule in combination with the peculiar cost function, the blob-pebble game seems to behave somewhat differently from the standard black-white pebble game, and therefore we cannot appeal directly to known lower bounds on black-white pebbling price. Luckily, the lower bound construction in [25] can be generalized to the blob-pebble game giving the following theorem.

THEOREM 6. *Pyramids Π_h have blob-pebbling price $\Theta(h)$.*

PROOF OUTLINE. The key idea (adapted from [25]) is to define a *potential* measure for the set of subconfigurations $\mathbb{S} = \{[B_i]\langle W_i\rangle \mid i = 1, \ldots, m\}$ currently in the graph as an indicator of "how good" this set is and prove two facts from which the desired bound immediately follows:

1. The potential of the current pebble configuration $\mathbb{S}_t$ is upper-bounded, up to a fixed multiplicative constant, by the maximum cost of any configuration $\mathbb{S}_{t'}$, $t' \leq t$.
2. The final pebble configuration $\mathbb{S}_\tau = \big\{[z]\langle\emptyset\rangle\big\}$ consisting of a single black blob on the sink has potential $\Theta(h)$.

More precisely, let $U\{\succeq j\}$ denote the vertices in U on or above level j (sources are on level 0 and the sink z is on level h) and define $m(U) = \max\big\{j + 2|U\{\succeq j\}| : U\{\succeq j\} \neq \emptyset\big\}$ to be the *measure* of U. U *blocks* $[B]\langle W\rangle$ if $U \cup W$ intersects every path P from a source vertex such that $B \subseteq P$, and U blocks $\mathbb{S}$ if it blocks every $[B]\langle W\rangle \in \mathbb{S}$. The *potential* of $\mathbb{S}$ is $\mathrm{pot}(\mathbb{S}) = \min\{m(U) : U \text{ blocks } \mathbb{S}\}$.

Fact 2 of the proof now follows easily, since it can be shown that the set U with smallest measure blocking $\mathbb{S}_\tau = \big\{[z]\langle\emptyset\rangle\big\}$ is $U = \{z\}$ with $m(U) = h + 2$ (this is a useful exercise if one wants to get some feeling for how the potential works).

Fact 1 is proven by induction. Suppose that U_t blocks $\mathbb{S}_t$ and that $\mathrm{pot}(\mathbb{S}_t) = m(U_t)$. By the inductive hypothesis, we have $\mathrm{pot}(\mathbb{S}_t) \leq C \cdot \max_{t' \leq t}\{\mathit{cost}(\mathbb{S}_{t'})\}$. We want to show $\mathrm{pot}(\mathbb{S}_{t+1}) \leq C \cdot \max_{t' \leq t+1}\{\mathit{cost}(\mathbb{S}_{t'})\}$, which clearly holds if

$$\mathrm{pot}(\mathbb{S}_{t+1}) \leq \max\{\mathrm{pot}(\mathbb{S}_t), C \cdot \mathit{cost}(\mathbb{S}_{t+1})\} \quad . \tag{4}$$

To establish this inequality, we note that if the pebbling move at time $t+1$ is an inflation, $\mathbb{S}_{t+1}$ is blocked by U_t (e.g., if U_t blocks the subconfiguration in Figure 4(e), then it blocks Figure 4(f)). Hence, $\mathrm{pot}(\mathbb{S}_{t+1}) \leq m(U_t) = \mathrm{pot}(\mathbb{S}_t)$ in this case. In the same way it can be verified that if U_t blocks two subconfigurations being merged, it must also block the result of the merger (compare Figures 4(c) and 4(d)). And if we make an introduction move on a non-source vertex, the white pebbles on the predecessors block the black blob no matter what U_t looks like (see Figure 4(b)).

Thus, the potential can only increase when an introduction of $[v]\langle\emptyset\rangle$ is performed on a source v. It turns out that what we need to prove (4) in this case is that pyramid graphs have the following property: *There exists a constant C' such that for any configuration $\mathbb{S}$ there is a blocking vertex set U with $\mathrm{pot}(\mathbb{S}) = m(U)$ and $|U| \leq C' \cdot \mathit{cost}(\mathbb{S})$.*

This far the construction closely parallels that in [25], but showing that we can choose blocking sets that achieve the minimum measure and at the same time have limited cardinality requires new tools, as well as using the proof in [25] as a subroutine. We refer to [27] for the details. $\square$

We remark that the proof of Theorem 6 applies in (almost) the same generality as in [25]. It works for all layered DAGs that are also "spreading" in the sense that (loosely speaking) for every vertex v on any level L and every $K \leq L$, there are at least $K+1$ vertices located exactly K levels below v from which v is reachable. This class of graphs includes among others complete binary trees and pyramid graphs.

It is an intriguing open question to determine the exact relation between the blob-pebble game and the black-white pebble game. On the one hand, to prove Theorem 6 we use additional techniques and get worse constants compared to the construction in [25]. On the other hand, we do not know of a single example where the possibility to use blobs reduces the cost of the cheapest pebbling.

Returning to our main path of reasoning and putting all of this together, we can now prove our main theorem.

THEOREM 1 (RESTATED). *Let $\mathit{Peb}^d_{\Pi_h}$ denote the pebbling contradiction of degree $d > 1$ defined over the pyramid graph of height h. Then the clause space of refuting $\mathit{Peb}^d_{\Pi_h}$ by resolution is $\mathit{Sp}(\mathit{Peb}^d_{\Pi_h} \vdash 0) = \Theta(h)$.*

PROOF. The upper bound $\mathit{Sp}(\mathit{Peb}^d_{\Pi_h} \vdash 0) = \mathrm{O}(h)$ is easy, so the interesting part is the lower bound. Let π be any resolution refutation of $\mathit{Peb}^d_{\Pi_h}$ and consider the associated blob-pebbling $\mathcal{P}_\pi$ provided by Theorem 4. On the one hand, we know that $\mathit{cost}(\mathcal{P}_\pi) = \mathrm{O}(\mathit{Sp}(\pi))$ by Theorem 5, provided that $d > 1$. On the other hand, Theorem 6 tells us that the cost of any blob-pebbling of Π_h is $\Omega(h)$, so in particular we must have $\mathit{cost}(\mathcal{P}_\pi) = \Omega(h)$. Combining these two bounds on $\mathit{cost}(\mathcal{P}_\pi)$, we see that $\mathit{Sp}(\pi) = \Omega(h)$. $\square$

The pebbling contradiction Peb^d_G is a $(2+d)$-CNF formula and for constant d the size of the formula is linear in the number of vertices of G (compare Figure 1). Hence, for pyramid graphs Π_h the corresponding pebbling contradictions $\mathit{Peb}^d_{\Pi_h}$ have size quadratic in the height h. Also, when d is fixed the upper bounds mentioned at the end of Section 2 become $L(\mathit{Peb}^d_G \vdash 0) = \mathrm{O}(n)$ and $W(\mathit{Peb}^d_G \vdash 0) = \mathrm{O}(1)$. Corollary 2 now follows if we set $F_n = \mathit{Peb}^d_{\Pi_h}$ for $d = k-2$ and $h = \lfloor\sqrt{n}\rfloor$ and use Theorem 1.

COROLLARY 2 (RESTATED). *For every $k \geq 4$, there is a family of k-CNF formulas $\{F_n\}_{n=1}^\infty$ of size $\Theta(n)$ that can be refuted in length $L(F_n \vdash 0) = \mathrm{O}(n)$ and width $W(F_n \vdash 0) = \mathrm{O}(1)$ but require clause space $\mathit{Sp}(F_n \vdash 0) = \Theta(\sqrt{n})$.*

3.3 Overview of Trade-off Results

Let us also quickly sketch the ideas (or tricks, really) used to prove our trade-off theorems for resolution.

We show the following version of the length-variable space trade-off theorem of Hertel and Pitassi [22], with somewhat improved parameters and a very much simpler proof.

THEOREM 7. *There is a family of CNF formulas $\{F_n\}_{n=1}^\infty$ of size $\Theta(n)$ such that:*

- *The minimal variable space of refuting F_n in resolution is $\mathit{VarSp}(F_n \vdash 0) = \Theta(n)$.*
- *Any resolution refutation $\pi : F_n \vdash 0$ in minimal variable space has length $\exp(\Omega(\sqrt{n}))$.*
- *Adding at most 2 extra units of storage, one can obtain a refutation in space $\mathit{VarSp}(F_n \vdash 0) + 3 = \Theta(n)$ and length $\mathrm{O}(n)$, i.e., linear in the formula size.*

The idea behind our proof is as follows. Take formulas G_n that are really hard for resolution and formulas H_m which have short refutations but require linear variable space, and set $F_n = G_n \wedge H_m$ for m chosen so that $\mathit{VarSp}\big(H_m \vdash 0\big)$ is only just larger than $\mathit{VarSp}\big(G_n \vdash 0\big)$. Then refutations in minimal variable space will have to take care of G_n, which requires exponential length, but adding one or two literals to the memory we can attack H_m instead in linear length.

The trade-off result in Theorem 3 for length versus clause space and its twin theorem for length versus width are shown using similar ideas. Again, the details can be found in [27].

4. CONCLUSION AND OPEN PROBLEMS

We have proven an asymptotically tight bound on the refutation clause space in resolution of pebbling contradictions over pyramid graphs. This yields the currently best known separation of length and clause space in resolution. Also, in contrast to previous polynomial lower bounds on clause space, our result does not not follow from corresponding lower bounds on width for the same formulas. Instead, a corollary of our result is an exponential improvement of the separation of width and space in [26]. This is a first step towards answering the question of the relationship between length and space posed in, for instance, [5, 19, 33].

More technically, we have established that for all graphs G in the class of "layered spreading DAGs" (including binary trees and pyramids) the height h of G, which coincides with the black-white pebbling price, is an asymptotical lower bound for the refutation clause space $\mathit{Sp}\big(\mathit{Peb}^d_G \vdash 0\big)$ of pebbling contradictions Peb^d_G provided that $d \geq 2$. Plugging in pyramids we get an $\Omega(\sqrt{n})$ bound on space, which is the best one can get for any spreading graph.

An obvious question is whether this lower bound on clause space in terms of black-white pebbling price is true for arbitrary DAGs. In particular, does it hold for the family of DAGs $\{G_n\}_{n=1}^\infty$ in [20] of size $\mathrm{O}(n)$ that have maximal black-white pebbling price $\mathit{BW\text{-}Peb}(G_n) = \Omega(n/\log n)$ in terms of size? If it could be proven for pebbling contradictions over such graphs that pebbling price bounds clause space from below, this would immediately imply that there are k-CNF formulas refutable in small length that can be maximally complex with respect to clause space.

Open Problem 1. *Is there a family of k-CNF formulas $\{F_n\}_{n=1}^{\infty}$ of size $\mathrm{O}(n)$ such that $L(F_n \vdash 0) = \mathrm{O}(n)$ and $W(F_n \vdash 0) = \mathrm{O}(1)$ but $Sp(F_n \vdash 0) = \Omega(n/\log n)$?*

A second question, more related to Theorem 3 and our other trade-off results, is as follows. We know from (1) that short resolution refutations imply the existence of narrow refutations, and in view of this an appealing proof search heuristic is to search exhaustively for refutations in minimal width. One serious drawback of this approach is that there is no guarantee that the short and narrow refutations are the same one. On the contrary, the narrow refutation π' resulting from the proof in [9] is potentially exponentially longer than the short proof π that we start with. However, we have no examples of formulas where the refutation in minimum width is actually known to be substantially longer than the minimum-length refutation. Therefore, it would be valuable to know whether this increase in length is necessary. That is, is there a formula family which exhibits a length-width trade-off in the sense that there are short refutations and narrow refutations, but all narrow refutations have a length blow-up (polynomial or superpolynomial)? Or is the exponential blow-up in [9] just an artifact of the proof?

Open Problem 2. *If F is a k-CNF formula over n variables refutable in length L, is it true that there is always a refutation π of F in width $W(\pi) = \mathrm{O}\big(\sqrt{n \log L}\big)$ with length no more than, say, $L(\pi) = \mathrm{O}(L)$ or at most $\mathrm{poly}(L)$?*

A similar trade-off question can be posed for clause space. Given a refutation in small space, we can prove using (2) that there must exist a refutation in short length. But again, the short refutation resulting from the proof is not the same as that with which we started. For concreteness, let us fix the space to be constant. If a polynomial-size k-CNF formula has a refutation in constant space, we know that it must be refutable in polynomial length. But can we get a refutation in both short length and small space simultaneously?

Open Problem 3. *Suppose that $\{F_n\}_{n=1}^{\infty}$ is a family of polynomial-size k-CNF formulas with refutation clause space $Sp(F_n \vdash 0) = \mathrm{O}(1)$. Does this imply that there are refutations $\pi_n : F_n \vdash 0$ simultaneously in length $L(\pi_n) = \mathrm{poly}(n)$ and clause space $Sp(\pi_n) = \mathrm{O}(1)$?*

Or can it be that restricting the clause space, we sometimes have to end up with really long refutations? We would like to know what holds in this case, and how it relates to the trade-off results for variable space in [22].

Finally, we note that all bounds on clause space proven so far is in the regime where the clause space $Sp(\pi)$ is less than the number of clauses $|F|$ in F. This is quite natural, since the size of the formula can be shown to be an upper bound on the minimal clause space needed [18].

Such lower bounds on space might not seem too relevant to clause learning algorithms, since the size of the cache in practical applications usually will be very much larger than the size of the formula. For this reason, it seems to be a highly interesting problem to determine what can be said if we allow extra clause space. Assume that we have a CNF formula F of size roughly n refutable in length $L(F \vdash 0) = L$ for L suitably large (say, $L = \mathrm{poly}(n)$ or $L = n^{\log n}$ or so). Suppose that we allow clause space more than the minimum $n+\mathrm{O}(1)$, but less than the trivial upper bound $L/\log L$. Can we then find a refutation using at most that much space and achieving at most a polynomial increase in length compared to the minimum?

Open Problem 4 ([6]). *Let F be any CNF formula with $|F| = n$ clauses (or $|Vars(F)| = n$ variables). Suppose that $L(F \vdash 0) = L$. Does this imply that there is a resolution refutation $\pi : F \vdash 0$ in clause space $Sp(\pi) = \mathrm{O}(n)$ and length $L(\pi) = \mathrm{poly}(L)$?*

If so, this could be interpreted as saying that a smart enough clause learning algorithm can potentially find any short resolution refutation in reasonable space (and for formulas that cannot be refuted in short length we cannot hope to find refutations efficiently anyway).

We conclude with a couple of comments on clause space versus clause learning.

Firstly, we note that it is unclear whether one should expect any fast progress on Open Problem 4, at least if if our experience from the case where $Sp(\pi) \leq |F|$ is anything to go by. Proving lower bounds on space in this "low-end regime" for formulas easy with respect to length has been (and still is) very challenging. However, it certainly cannot be excluded that problems in the range $Sp(\pi) > |F|$ might be approached with different and more successful techniques.

Secondly, we would like to raise the question of whether, in spite of what was just said before Open Problem 4, lower bounds on space can nevertheless give indications as to which formulas might be hard for clause learning algorithms and why. Suppose that we know for some CNF formula F that $Sp(F \vdash 0)$ is large. What this tells us is that any algorithm, even a non-deterministic one making optimal choices concerning which clauses to save or throw away, will have to keep a fairly large number of "active" clauses in memory in order to carry out the refutation. Since this is so, a real-life deterministic proof search algorithm, which has no sure-fire way of knowing which clauses are the right ones to concentrate on at any given moment, might have to keep working on a lot of extra clauses in order to be sure that the fairly large critical set of clauses needed to find a refutation will be among the "active" clauses.

Intriguingly enough, pebbling contradictions over pyramids might in fact be an example of this. We know that these formulas are very easy with respect to length and width, having constant-width refutations that are essentially as short as the formulas themselves. But in [29], it was shown that state-of-the-art clause learning algorithms can have serious problems with even moderately large pebbling contradictions. (Their "grid pebbling formulas" are exactly our pebbling contradictions of degree $d = 2$ over pyramids.) Although we are certainly not arguing that this is the whole story—it was also shown in [29] that the branching order is a critical factor, and that given some extra structural information the algorithm can achieve an exponential speed-up—we wonder whether the high lower bound on space can nevertheless be part of the explanation. It should be pointed out that pebbling contradictions are the only formulas we know of that are really easy with respect to length and width but hard for clause space. And if there is empirical data showing that for these very formulas clause learning algorithms can have great difficulties finding refutations, it might be worth investigating whether this is just a coincidence or a sign of some deeper connection.

5. ACKNOWLEDGEMENTS

We are grateful to Per Austrin and Mikael Goldmann for generous feedback during various stages of this work, and to Gunnar Kreitz for quickly spotting some bugs in a preliminary version of the blob-pebble game. Also, we would like to thank Paul Beame, Maria Klawe, Philipp Hertel, and Toniann Pitassi for valuable correspondence concerning their work, Nathan Segerlind for comments and pointers regarding clause learning, and Eli Ben-Sasson for stimulating discussions about proof complexity in general and the problems in Section 4 in particular.

6. REFERENCES

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson. Space complexity in propositional calculus. *SIAM J. Comput.*, 31(4):1184–1211, 2002.

[2] A. Atserias and V. Dalmau. A combinatorical characterization of resolution width. In *Proc. 18th IEEE Annual Conference on Computational Complexity (CCC '03)*, pages 239–247, 2003.

[3] P. Beame. Proof complexity. In S. Rudich and A. Wigderson, editors, *Computational Complexity Theory*, volume 10 of *IAS/Park City Mathematics Series*, pages 199–246. AMS, 2004.

[4] P. Beame, H. Kautz, and A. Sabharwal. Understanding the power of clause learning. In *Proc. 18th International Joint Conference in Artificial Intelligence (IJCAI '03)*, pages 94–99, 2003.

[5] E. Ben-Sasson. Size space tradeoffs for resolution. In *Proc. 34th Annual ACM Symposium on Theory of Computing (STOC '02)*, pages 457–464, 2002.

[6] E. Ben-Sasson. Personal communication, 2007.

[7] E. Ben-Sasson and N. Galesi. Space complexity of random formulae in resolution. *Rand. Struct. Algorithms*, 23(1):92–109, 2003.

[8] E. Ben-Sasson, R. Impagliazzo, and A. Wigderson. Near optimal separation of treelike and general resolution. *Combinatorica*, 24(4):585–603, 2004.

[9] E. Ben-Sasson and A. Wigderson. Short proofs are narrow—resolution made simple. *J. ACM*, 48(2):149–169, 2001.

[10] A. Blake. *Canonical Expressions in Boolean Algebra*. PhD thesis, University of Chicago, 1937.

[11] M. L. Bonet and N. Galesi. Optimality of size-width tradeoffs for resolution. *Comput. Complexity*, 10(4):261–276, 2001.

[12] V. Chvátal and E. Szemerédi. Many hard examples for resolution. *J. ACM*, 35(4):759–768, 1988.

[13] S. A. Cook. The complexity of theorem-proving procedures. In *Proc. 3rd Annual ACM Symposium on Theory of Computing (STOC '71)*, pages 151–158, 1971.

[14] S. A. Cook and R. Reckhow. The relative efficiency of propositional proof systems. *J. Symbolic Logic*, 44(1):36–50, 1979.

[15] S. A. Cook and R. Sethi. Storage requirements for deterministic polynomial time recognizable languages. *J. Comput. System Sci.*, 13(1):25–37, 1976.

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

[17] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215, 1960.

[18] J. L. Esteban and J. Torán. Space bounds for resolution. *Inform. and Comput.*, 171(1):84–97, 2001.

[19] J. L. Esteban and J. Torán. A combinatorial characterization of treelike resolution space. *Inform. Process. Lett.*, 87(6):295–300, 2003.

[20] J. R. Gilbert and R. E. Tarjan. Variations of a pebble game on graphs. Technical Report STAN-CS-78-661, Stanford University, 1978.

[21] A. Haken. The intractability of resolution. *Theoret. Comput. Sci.*, 39(2-3):297–308, 1985.

[22] P. Hertel and T. Pitassi. Exponential time/space speedups for resolution and the PSPACE-completeness of black-white pebbling. In *Proc. 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS '07)*, pages 137–149, 2007.

[23] J. Hopcroft, W. Paul, and L. Valiant. On time versus space. *J. ACM*, 24(2):332–337, 1977.

[24] H. Kautz and B. Selman. The state of SAT. *Discr. Appl. Math.*, 155(12):1514–1524, 2007.

[25] M. M. Klawe. A tight bound for black and white pebbles on the pyramid. *J. ACM*, 32(1):218–228, 1985.

[26] J. Nordström. Narrow proofs may be spacious: Separating space and width in resolution. In *Proc. 38th Annual ACM Symposium on Theory of Computing (STOC '06)*, pages 507–516, 2006.

[27] J. Nordström and J. Håstad. Towards an optimal separation of space and length in resolution. Technical Report 0803.0661, arXiv.org, 2008. Available at `http://arxiv.org/abs/0803.0661`.

[28] J. A. Robinson. A machine-oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.

[29] A. Sabharwal, P. Beame, and H. Kautz. Using problem structure for efficient clause learning. In *6th International Conference on Theory and Applications of Satisfiability Testing (SAT '03)*, volume 2919 of *LNCS*, pages 242–256. Springer, 2004.

[30] The international SAT Competitions web page. `http://www.satcompetition.org`.

[31] N. Segerlind. The complexity of propositional proofs. *Bull. Symbolic Logic*, 13(4):482–537, 2007.

[32] J. Torán. Lower bounds for space in resolution. In *Proc. 13th International Workshop on Computer Science Logic (CSL '99)*, volume 1683 of *LNCS*, pages 362–373. Springer, 1999.

[33] J. Torán. Space and width in propositional resolution. *Bull. EATCS*, 83:86–104, 2004.

[34] G. Tseitin. On the complexity of derivation in propositional calculus. In A. O. Silenko, editor, *Structures in Constructive Mathematics and Mathematical Logic, Part II*, pages 115–125. Consultants Bureau, New York-London, 1968.

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