Space Characterizations of Complexity Measures and Size-Space Trade-Offs in Propositional Proof Systems

Theodoros Papamakarios
Department of Computer Science, University of Chicago, IL, USA

Alexander Razborov
University of Chicago, IL, USA
Steklov Mathematical Institute, Moscow, Russia

Abstract
We identify two new big clusters of proof complexity measures equivalent up to polynomial and \( \log n \) factors. The first cluster contains, among others, the logarithm of tree-like resolution size, regularized (that is, multiplied by the logarithm of proof length) clause and monomial space, and clause space, both ordinary and regularized, in regular and tree-like resolution. As a consequence, separating clause or monomial space from the (logarithm of) tree-like resolution size is the same as showing a strong trade-off between clause or monomial space and proof length, and is the same as showing a super-critical trade-off between clause space and depth. The second cluster contains width, \( \Sigma_2 \) space (a generalization of clause space to depth 2 Frege systems), both ordinary and regularized, as well as the logarithm of tree-like size in the system \( R(\log) \). As an application of some of these simulations, we improve a known size-space trade-off for polynomial calculus with resolution. In terms of lower bounds, we show a quadratic lower bound on tree-like resolution size for formulas refutable in clause space 4. We introduce on our way yet another proof complexity measure intermediate between depth and the logarithm of tree-like size that might be of independent interest.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof Complexity, Resolution, Size-Space Trade-offs

Digital Object Identifier 10.4230/LIPIcs.ICALP.2022.100

Category Track A: Algorithms, Complexity and Games


1 Introduction

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 [13], where its fundamental role in complexity theory was identified.

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 propositional 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). To aid further discussion, let us review those measures and previous results that are most pertinent to this work.
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 [8], who, building on the earlier works of Clegg et al. [12] and Impagliazzo et al. [23] showing an analogous result for polynomial calculus, showed that a short resolution proof can be transformed into a narrow one. Namely, we have

\[ W(F ⊢ ⊥) ≤ \log S_T(F ⊢ ⊥) + W(F), \]

\[ W(F ⊢ ⊥) ≤ O(\sqrt{n} \log S_R(F ⊢ ⊥)) + W(F). \]

Here \( W(F ⊢ ⊥), S_T(F ⊢ ⊥) \) and \( S_R(F ⊢ ⊥) \) stand for the minimum width, tree-like size and DAG-like size respectively of refuting an unsatisfiable CNF \( F \) in resolution; similar notation is employed throughout the paper. \( W(F) \) is the maximum width of a clause in \( F \).

Space complexity for propositional proofs was introduced in [16, 1]. Esteban and Torán [16] showed that a short tree-like resolution proof can be transformed into a resolution proof of small clause space:

\[ \text{CSpace}(F ⊢ ⊥) ≤ \log S_T(F ⊢ ⊥). \]  

Atserias and Dalmau [2] 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:

\[ W(F ⊢ ⊥) ≤ \text{CSpace}(F ⊢ ⊥) + W(F). \]  

Constructive versions of their result were given by Fihmns et al. [18] and Razborov (unpublished), see also Krajíček [26, Theorem 5.5.5]. It is worth noting that (3) and (4) taken together provide a refinement of (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 in this paper.

More recently, Bonacina [9] showed that for total space in resolution (measured as the sum of widths of clauses in a configuration) we have

\[ W(F ⊢ ⊥) ≤ O(\sqrt{T_{\text{Space}}(F ⊢ ⊥)}) + W(F), \]

and Galesi et al. [19] showed a weakened version of (4), but for the analogue of clause space in stronger proof systems operating with polynomials (or in fact even arbitrary Boolean functions of monomials):

\[ W(F ⊢ ⊥) ≤ O((\text{MSpace}(F ⊢ ⊥))^2) + W(F). \]

Regularized\(^1\) versions \( \mu^* \) of space complexity measures are defined by multiplying the measure in question \( \mu \) by the logarithm of the proof length; these were considered e.g. by Ben-Sasson [4] and Razborov [37]. The latter paper also contains a suggestion that the “right” level of precision when comparing measures of this kind are up to polynomial and \( \log n \) factors;\(^2\) we will henceforth call two measures equivalent if they simulate each other.

---

\(^1\) The paper [37] used the word “amortized” but Sam Buss pointed out to us that it is somewhat misleading in this context.

\(^2\) Note that the size/length measures appear in this set-up under a logarithm. Hence this corresponds to quasi-polynomial simulations in the Cook-Reckhow framework.
in this sense. The paper [37] identified a big cluster of ordinary and regularized space complexity measures, including total and variable space, that are all equivalent to proof depth in resolution. One notable measure that defied this classification was (regularized) clause space.

Our contributions

In this paper we identify two other big clusters of equivalent complexity measures not covered by the results in [37]. The cumulative picture combining both previously known and new results is summarized in Figure 1. There, arrows are to be interpreted as inequalities, and ≈ as equality, both up to polynomial and log n factors. A solid arrow from μ₁ to μ₂ indicates that a separation between μ₁ and μ₂ is known, that is, it additionally indicates that there exists a sequence \( \{F_n\} \) of unsatisfiable CNFs such that μ₂(\( F_n \vdash \bot \)) ≥ (μ₁(\( F_n \vdash \bot \)) + log n)ω(1).

To improve readability, we have omitted from Figure 1 the argument \( F \vdash \bot \).

Let us briefly explain this picture. The first new cluster is centered around the logarithm of tree-like resolution size. Given the proof method of the simulation (3) in [16], it can be obviously strengthened in two directions: by replacing the left-hand side with clause space in tree-like resolution or by replacing it with regularized clause space. Tree-like clause space in resolution was shown to be equivalent to the logarithm of tree-like size in the same paper [16, Corollary 5.1]; in other words, after this replacement in the left-hand side, the bound (3) becomes tight, within the precision we are tolerating.

---

3 A technical remark: [37, Theorem 3.2] does not apply to clause space as it is not bounded from below by the number of variables.
We show that the second variant, that is regularized clause space, is also equivalent to the logarithm of tree-like resolution size, and this result extends to also include regularized monomial space to the same cluster. Given that [16, Corollary 5.1] also holds for (ordinary) clause space in regular resolution [16, Corollary 4.2], this means that all these space measures turn out to be equivalent to each other and to the log of tree-like resolution size. We also remark (given the results above, this readily follows from definitions) that regularized versions of the clause space in tree-like or regular resolution are also in this cluster.

The question of whether (ordinary) clause space also belongs here is what we consider to be a major, and most likely very difficult, open problem. But since it has turned out to be closely related to several other threads in proof complexity, we prefer to keep the momentum and defer further discussion to the concluding Section 5.

Our second cluster is presided by resolution width. First, we introduce a natural analogue of clause space in DNF resolution that we call $\Sigma_2$ space. This can be seen as an extension of clause space to depth 2 Frege systems; indeed, the restriction of $\Sigma_2$ space to depth 1 Frege is precisely clause space, and its restriction to $k$-DNF resolution, for constant $k$, coincides, up to a constant factor, with the concept of space that has been studied before for such systems (see e.g. [15, 7]). In our model, configurations are arbitrary sets of DNFs, and we charge $k$ for every individual $k$-DNF in the memory. Clearly, $\Sigma_2$ Space $\leq$ CS成败 and $\Sigma_2$ Space$^* \leq$ CS成败*. Then we strengthen the Atserias-Dalmau bound (4) by replacing CS成败 with $\Sigma_2$ Space and continue to show that both ordinary and regularized versions of $\Sigma_2$ space are actually equivalent to resolution width.

Thus, remarkably, the difficult open question on whether we have a strong trade-off between space and length for clause space gets a relatively easy negative solution for a stronger proof system. 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). 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 but it is implicit in Lauria [27] and, with a bit of effort, can be traced back as far as Krajíček [24].

It is worth noting that some of the simulations in this cluster work only in the syntactical setting. This comes in contrast with what happens with the other two clusters: all simulations involving clause, monomial, variable and total space, also work in a purely semantic setting. For example, in case of monomial space we can allow arbitrary Boolean functions of monomials involving clause, monomial, variable and total space, also work in a purely semantic setting.

We use (some of) these simulations to prove:

1. There are unsatisfiable CNFs $F$ of size $O(n)$ with $S(F \vdash \bot) \leq O(n)$, $W(F \vdash \bot) \leq O(1)$ and $\text{MSpace}^*(F \vdash \bot) \geq \Omega(n/\log n)$ (Theorem 4.1). This is an improvement on the previously known bounds $\text{MSpace}^*(F \vdash \bot) \geq \Omega(n^{2/11})$ [3], $\text{MSpace}^*(F \vdash \bot) \geq \Omega(n^{1/4})$ [22] and $\text{MSpace}^*(F \vdash \bot) \geq n^{1/2}/(\log n)^{O(1)}$ [21]. Unlike these previous results, our proof is remarkably simple.

2. There are unsatisfiable CNFs $F$ of size $O(n)$ with $\text{CSpace}(F \vdash \bot) = 4$ and $S_T(F \vdash \bot) \geq \Omega(n^2/\log n)$ (Theorem 4.7). This is a first, admittedly modest, step toward separating clause space and, say, tree-like size; as we already said, we will discuss this question in more details in Section 5. It is for this proof that we need the last unexplained entry $D_P$ on Figure 1: it stands for positive depth, and it is a one-sided version of depth. We also remark that the space bound in this result is optimal. More precisely, we make a relatively simple observation (Theorem 4.2) that $\text{CSpace}(F \vdash \bot) \leq 3$ if and only if $F$ is “essentially Horn” in which case it will possess a linear size tree-like resolution refutation.
Finally, let us briefly summarize what is known (to the best of our knowledge) in terms of separating the measures in Figure 1. Let us start with “true” separations, i.e., separations that work modulo polynomial overheads and log \( n \) factors. From now on, for proof complexity measures \( \mu_1, \mu_2 \) we will use the notation \( \mu_1 \preceq \mu_2 \) to stand for \( \mu_1(F \vdash \bot) \leq (\mu_2(F \vdash \bot))^{O(1)} \) for any CNF \( F \) in \( n \) variables; \( \mu_1 \approx \mu_2 \) is the same as \( \mu_1 \preceq \mu_2 \wedge \mu_2 \preceq \mu_1 \). Clearly \( \preceq \) is transitive, and this implies that \( \approx \) is an equivalence relation and \( \preceq \) imposes a partial order on its equivalence classes.

Bonet and Galesi [10] proved that \( W \not\preceq \log S_R \). More precisely, there are constant width formulas \( F \) of size \( O(n^3) \) such that \( S_R(F \vdash \bot) \leq O(n^3) \) and \( W(F \vdash \bot) \geq \Omega(n) \). Ben-Sasson [4] proved that \( \text{VSpace} \not\preceq \text{CSpace} \), and after negating the variables in his formulas, this works two more levels up on Figure 1. Namely, there are constant-width formulas \( F \) of size \( O(n) \) such that \( \text{VSpace}(F \vdash \bot) \geq \Omega(n/\log n) \) while \( D_P(F \vdash \bot) \leq O(1) \). This also provides a separation between \( D_P \) and \( D \) that, though, is much easier to prove directly [38, Theorem 4.6]. Without negating the variables, it is easy to see that Ben-Sasson’s proof actually gives \( D_P(F \vdash \bot) \geq \Omega(n/\log n) \), thus separating \( D_P \) from \( \log S_T \) and hence from the whole middle cluster. Again, it is also easy to see this directly. Ben-Sasson, Håstad and Nordström [31, 6] separated clause space from width; while it is believed that their formulas should also have large monomial space complexity, the questions of separating clause space from monomial space, as well as monomial space from width are widely open.

Separating space complexity measures from their own regularized versions appear to be a very daunting task in general. As follows from Figure 1, for variable space this is equivalent to separating it from depth [38]. A quadratic separation between \( \text{VSpace} \) and \( \text{VSpace}^* \) was proved in [37, Section 6], with a disappointingly elaborate proof. Nothing is known in terms of separating \( \text{CSpace} \) from (the cluster of) \( \text{CSpace}^* \): Theorem 4.7 makes a progress in that direction, but it is admittedly rather modest. Nothing seems to be known for \( \text{CSpace} \) vs. \( \text{MSpace} \), and our structural picture provides a good heuristic explanation of the difficulty of this question: it would also separate \( \text{MSpace} \) from \( \text{MSpace}^* \). Finally, in [32] a quadratic separation between width and monomial space has been established using methods very different from those in [6].

The paper is organized as follows. After giving the necessary definitions in Section 2, in Section 3 we refine (many simulations do not actually involve a polynomial overhead or extra log \( n \) factors) and prove the relations of Figure 1. In Section 4 we prove items 1 and 2 above. The paper is concluded with a few remarks and open problems in Section 5.

## 2 Preliminaries

A literal is a propositional variable \( x \) or its negation \( \overline{x} \). We let \( \overline{x} \overset{\text{def}}{=} x \). A clause is a disjunction (possibly empty) of literals over distinct variables, and a term is a conjunction (possibly empty) of such literals. For a clause \( C = \ell_1 \lor \cdots \lor \ell_w \), we define the term \( C \overset{\text{def}}{=} \overline{\ell}_1 \land \cdots \land \overline{\ell}_w \); similarly for a term \( t = \ell_1 \land \cdots \land \ell_w \), \( t \overset{\text{def}}{=} \overline{\ell}_1 \lor \cdots \lor \overline{\ell}_w \). The width of a clause or a term is the number of literals it contains. A CNF formula is a conjunction of clauses, and a DNF formula is a disjunction of terms. The width, \( W(F) \), of a CNF or DNF formula \( F \) is the width of the largest clause or term it contains. A CNF or DNF formula of width at most \( w \) is called \( w \)-CNF or \( w \)-DNF respectively. Clauses may be alternatively viewed as 1-DNFs, but the latter class is slightly larger than tautological 1-DNFs like \( x \lor \overline{x} \) are allowed.

A partial (true) assignment (often called restriction) is a mapping from a subset \( V \) of all propositional variables to \( \{0, 1\} \); it is naturally extended to the negations of the variables in \( V \) by \( \alpha(\overline{x}) \overset{\text{def}}{=} \overline{\alpha(x)} \). The result of applying a partial assignment \( \alpha \) to a CNF formula \( F \) is
another CNF formula $F|_\alpha$, obtained by deleting from $F$ all literals $\ell$ such that $\alpha(\ell) = 0$ and deleting all clauses containing a literal $\ell$ such that $\alpha(\ell) = 1$. Similarly for DNF formulas, $F|_\alpha$ is called the restriction of $F$ to $\alpha$. For a formula $F$, we write $\alpha \models F$ if every total extension of $\alpha$ satisfies $F$ or, in other words, if $F|_\alpha$ is semantically equal to 1. For a set of formulas $S$, $\alpha \models S$ means $\alpha \models \bigwedge_{F \in S} F$, 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 $\alpha_C$ the minimal partial assignment such that $\alpha_C \models C$.

Resolution is a proof system operating with clauses. Its inference rules are:

\[
\frac{C \lor D}{C \lor D}, \quad \frac{C \lor x}{D \lor \overline{x}}, \quad \frac{D \lor \overline{x}}{C \lor D}.
\]  

The leftmost one is called the weakening rule; the rightmost one is called the resolution rule. We refer to the variable $x$ in an application of the resolution rule as the variable being resolved. One of the reasons to include the (redundant) weakening rule is that it makes resolution proofs closed under restricting by a partial assignment.

The width $W(\pi)$ of a resolution proof $\pi$ is defined as the maximum width of a clause in it, and the width $W(F \vdash \bot)$ is usually defined as the minimum width $W(\pi)$ of a resolution refutation $\pi$ of $F$. This definition, however, is ill-suited for those CNFs that themselves have large width, like the pigeonhole principle. We have found it way more natural and convenient to work with its slightly modified version used in [20] that we will denote by $W(\vdash_F \bot)$. It is defined as follows.

Instead of just allowing the clauses $C$ of $F$ as axioms, we allow them to participate in the form of the following more general $F$-cut rule:

\[
\frac{D \lor t_1 \ldots D \lor t_r}{{D}},
\]

where $t_1 \lor \ldots \lor t_r$ is a clause of $F$. In case some $D \lor t_j$ contains contradictory literals, it is removed from the premises. In particular, when $D = C$, the list of premises becomes empty so the clauses of $F$ are still available as axioms.

It is easy to see that $W(\vdash_F \bot) \leq W(F \vdash \bot) \leq W(\vdash_F \bot) + W(F) - 1$, hence the difference between the standard definition and ours becomes immaterial when $W(F)$ is small, and it does not have any noticeable impact on the size of a refutation.

One immediate advantage of this definition is that if we replace $W(F \vdash \bot)$ with $W(\vdash_F \bot)$ in (1), (2), (4), (5) or (6), we need not keep the annoying terms $W(F)$ any more, they just disappear. Simulations on Figure 1 will work without any restrictions on the width of the refuted CNF. More advantages of a similar flavor will become clear later, see Theorems 3.4 and 4.2 in particular.

Let us also remark that resolution with the $F$-cut rule is nothing else but Gentzen’s sequent calculus with only atomic cuts, restricted to proving sequents of the form $C_1, \ldots, C_m \Rightarrow$, where $C_1, \ldots, C_m$ are clauses (see [32]).

DNF resolution, or depth 2 Frege, is the straightforward extension of resolution where we allow, apart from variables in the resolution rule, also formulas of depth 1\(^4\) to be resolved. DNF resolution operates with DNF formulas. Its axioms and inference rules are:

\[
\frac{G}{{x} \lor \overline{x}}, \quad \frac{G \lor t_1}{G \lor H}, \quad \frac{G \lor t_2}{G \lor H \lor (t_1 \land t_2)}, \quad \frac{G \lor t}{G \lor H}, \quad \frac{H \lor \overline{t}}{G \lor H},
\]

\(^{4}\) For this reason, some authors use the term “depth 1 Frege” for DNF resolution; we prefer to stick to the convention under which depth refers to lines in a Hilbert-style proof.
where $G$ and $H$ are DNF formulas and $t, t_1, t_2$ and $t_1 \land t_2$ are terms. The leftmost rule is the weakening rule in this context, and the rightmost rule is called the cut rule. The remaining rule allows us to deal with $\land$ connectives, and is called $\land$-introduction.

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 $f(s)$. $R(k)$ for $k$ a constant is usually denoted by $\text{Res}(k)$ (thus, resolution is $\text{Res}(1)$). DNF resolution and $R(f)$ were first introduced in [25].

Next, we would like to consider systems for manipulating terms. The syntactic details of such systems will not matter for our results, but for concreteness, let us present a prominent system of algebraic flavor originally introduced in [12]. We will actually use an extension, proposed in [1], called polynomial calculus with resolution and abbreviated as PCR. PCR works with a fixed field $\mathbb{F}$. Clauses/terms are represented as monomials. The syntactic objects PCR operates with are monomials in $\mathbb{F}[x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n]$, represented as linear combinations over $\mathbb{F}$ of monomials, and a proof line $P$ is to be interpreted as asserting that $P = 0$. The axioms and inference rules of the system are:

$$
\frac{P + Q}{\ell P}, \frac{P}{\alpha P + \beta Q}, \frac{P}{\ell P}
$$

where $\ell \in \{x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}$, $P, Q \in \mathbb{F}[x_1, \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n]$ and $\alpha, \beta \in \mathbb{F}$.

In each of the above systems, non-logical axioms are given as a set of clauses $S$, viewed as a CNF formula $F$ (in PCR, a clause $C = \ell_1 \lor \cdots \lor \ell_k \in S$ is represented as the monomial $\ell_1 \cdots \ell_k$). A proof of the unsatisfiability of $F$, or a refutation of $F$, is a derivation of a syntactic contradiction, denoted by $\bot$, from the clauses of $F$. In resolution and DNF resolution $\bot$ is the empty clause; in PCR, it is the polynomial $1$.

We can view proofs as DAGs, by drawing edges from premises to conclusions in applications of the inference rules. If a proof DAG is a tree, that is every formula or polynomial in it is used as a premise at most once, then we say that the proof is tree-like. The size of a tree-like proof is the number of its leaves, and its depth is the length of its longest root-to-leaf path. We will also consider a one-sided version of depth, which we call positive depth. (The analogue of this notion in the context of computational complexity was recently defined in [29].) The positive depth of a tree-like resolution proof is the maximum number of negative literals introduced along a root-to-leaf path. We denote tree-like size, depth and positive depth by $S_T, D$ and $D_P$ respectively.

To define space complexity measures, we need to consider a different topology, namely view a proof as a sequence of memory configurations [16, 1]. A memory configuration will be a set of clauses in resolution, a set of DNF formulas in DNF resolution, or a set of polynomials in PCR. In a proof from a CNF $F$ then, to go from a memory configuration to the next we may do one of the following:

**Axiom Download**: add a clause of the formula $F$, or a logical axiom of the system we are working with;

**Erasure**: delete a clause/DNF formula/polynomial, or

**Inference**: add the result of applying an inference rule to formulas in the current configuration.

A proof in configurational form 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.

The clause space of a configuration in 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 $\Sigma_2$ space of a configuration. The $\Sigma_2$ space of a configuration $M = \{G_1, \ldots, G_s\}$ is defined as the sum of widths: $\Sigma_2 \text{Space}(M) \equiv W(G_1) + \cdots + W(G_s)$. For PCR, we will consider the monomial space, which is the number of distinct monomials in a configuration.
For a space measure $\mu$ on configurations and a proof $\pi = \mathcal{M}_1, \ldots, \mathcal{M}_t$, we naturally let $\mu(\pi) \equiv \max \{ \mu(M_i) \mid 1 \leq i \leq t \}$. As in [37], we will also consider regularized versions $\mu^*$ defined as $\mu^*(\pi) \equiv \mu(\pi) \cdot \log |\pi|$, where $|\pi| \equiv t$ is the length of $\pi$. All logarithms in this paper have base 2.

Finally, for a complexity measure $\mu$ on proofs, we write $\mu(F \vdash G)$ for the minimum value of $\mu(\pi)$, taken over all proofs of $G$ from $F$; if such a proof does not exist, we set $\mu(F \vdash G)$ to be $\infty$. In most cases, the measure $\mu$ clearly suggests what the underlying proof system should be. For example, $W(F \vdash \bot)$ is the minimum width of a resolution refutation of $F$, and $\text{MSpace}^*(F \vdash \bot)$ is the minimum regularized monomial space of a PCR refutation (in configurational form) of $F$. $S_T(F \vdash \bot)$ shall mean the minimum size of a tree-like resolution refutation of $F$. We shall use the notation $S_{T,R}(f)(F \vdash \bot)$ to mean the minimum size of a tree-like $R(f)$-refutation of $F$. TCSpace($F \vdash \bot$) is the minimum clause space taken over all tree-like configurational refutations of $F$ in resolution. Likewise, RCSpace($F \vdash \bot$) stands for the clause space in regular resolution, i.e. the subsystem of resolution where we require that a variable cannot be resolved more than once on any path in (the DAG resulting from the expansion of) the configurational proof $\pi$.

3 Simulations

3.1 Tree-like resolution size and regularized monomial space

First we show that $\log S_T$ in resolution, TCSpace, RCSpace, CSpace$^*$ and MSpace$^*$, are all equivalent. Our main new contribution is the following simulation.

**Theorem 3.1.** For any unsatisfiable CNF formula $F$ over $n$ variables,

\[
\log S_T(F \vdash \bot) \leq 2\text{MSpace}^*(F \vdash \bot)\log(n + 1),
\]

\[
\text{TCSpace}(F \vdash \bot) \leq 2(\text{MSpace}^*(F \vdash \bot) + 1).
\]

**Proof.** The proof is analogous to the construction in [37] 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 $\alpha_D \models \mathcal{M}_i$ and $\alpha_D \models \neg \mathcal{M}_j$, it holds that $S_T(F \vdash D) \leq (n + 1)^{d}s$ and, moreover, the assumed tree-like resolution proof can be carried out in clause space at most $ds + 2$. The theorem follows by taking $[i..j] := [1..t]$, $d := \lceil \log t \rceil$ and $D := \bot$.

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 $\alpha_D \models \mathcal{M}_i$ and $\alpha_D \models \neg \mathcal{M}_j$. Then we necessarily must have $\alpha_D \models \neg C$, 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 $\alpha_D \models \mathcal{M}_i$ and $\alpha_D \models \neg \mathcal{M}_j$. Set $k := i + \lfloor (j - i)/2 \rfloor$, 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 following derivation of $A$:
would still be in this cluster. This allows us to re-cast the main open problem of whether weaker regularization multiplying by the logarithm of depth, and the resulting measure

The first inequality follows from the observation that every tree-like refutation can be pruned

The argument is a refinement of the argument in [16] showing that tree-like clause

\[ A \lor \ell_1 \lor \ldots \lor \ell_r \]

\[ A \lor \ell_1 \lor \ldots \lor \ell_{r-1} \]

\[ A \lor \ell_1 \]

\[ A \lor \ell_2 \]

\[ A \lor \ell_2 \lor \ell_1 \]

\[ A \lor \ell_r \lor \ldots \lor \ell_2 \]

\[ A \lor \ell_r \lor \ldots \lor \ell_1 \]

\[ A \lor \ell_r \]

Call this derivation tree \( T_{A,m} \). For the required tree-like resolution proof of \( D \), start with \( T_{D,m_1} \). To every leaf of \( T_{D,m_1} \) labelled by a clause \( D' \), append the tree \( T_{D',m_2} \). 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 contract at its parent. The resulting tree \( T \) has at most \((n + 1)^s \) leaves, and each of its leaves is labelled by a clause \( E \) such that \( \alpha_E \models M_k \) or \( \alpha_E \models \neg M_k \). 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 \( T \) 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 \( s + (d - 1)s + 2 = ds + 2 \).

For the rest of the relations, we claim that for an unstatifiable CNF \( F \) in \( n \) variables,

\[ \text{RCSpace}(F \vdash \bot) \leq \text{TCSpace}(F \vdash \bot) \leq \log S_T(F \vdash \bot) + 2 \]

\[ \leq 2M\text{Space}^*(F \vdash \bot) \log(n + 1) + 2 \leq 2\text{CSpace}^*(F \vdash \bot) \log(n + 1) + 2 \]

\[ \leq 2\text{RCSpace}^*(F \vdash \bot) \log(n + 1) + 2 \leq 2(\text{RCSpace}(F \vdash \bot))^2 \log(n + 1) \log(2n) + 2. \]

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 [16, Theorem 2.1], and the third is Theorem 3.1. The fourth and the fifth inequalities are obvious. Finally, the last inequality follows from [16, Corollary 4.2].

As a byproduct, we get that \( \text{TCSpace} \approx \text{RCSpace} \). This comes in sharp contrast with the situation for size, where there is an exponential separation between tree-like and regular resolution [5].

We also see from [16, Corollary 4.2] 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 re-cast the main open problem of whether \( \text{CSpace} \approx \text{CSpace}^* \) in terms of the existence of a super-critical (in the sense of [36]) trade-off between clause space and depth.

The remaining (non-trivial) simulation on Figure 1 involving this cluster is:

\[ \textbf{Theorem 3.2}. \text{For any unsatisfiable CNF formula } F, \text{TCSpace}(F \vdash \bot) \leq D_P(F \vdash \bot) + 2. \]

\textbf{Proof}. The argument is a refinement of the argument in [16] showing that tree-like clause space is bounded by depth. We show, by induction on \( T \), that if \( T \) 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 $T$ has size at most 2, then $d \leq 1$, and $\text{TCSpace}(F \vdash \perp) \leq 3$. Otherwise, let $T_1$ and $T_2$ be the subproofs of $T$ proving 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 $T_1$ and $T_2$, say $T_1$, must have positive depth at most $d - 1$. From the induction hypothesis, there is a tree-like proof $\pi_1$ of $E_1$ of clause space at most $d + 1$, and a tree-like proof $\pi_2$ of $E_2$ of clause space at most $d + 2$. Deriving first $E_2$ using $\pi_2$, and then, keeping $E_2$ in memory, deriving $E_1$ using $\pi_1$, we get a proof of $E$ of clause space at most $d + 2$. ▶

### 3.2 Resolution width and $\Sigma_2$ space

The simulations for our second cluster will depend upon the following “locality” property of DNF resolution.

**Lemma 3.3.** Let $\alpha$ be a partial assignment. For each of the inference rules of DNF resolution, if both premises contain a term satisfied by $\alpha$, then $\alpha$ 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. As a historical remark, an extension of the Atserias-Dalmau bound (4) for the case of Res$(k)$ is sketched in [18], and, although it is not stated explicitly, it is also apparent in [15].

**Theorem 3.4.** For any unsatisfiable CNF formula $F$,

$$\frac{1}{5} \Sigma_2 \text{Space}(F \vdash \perp) \leq W(\vdash_F \perp) \leq \Sigma_2 \text{Space}(F \vdash \perp).$$

**Proof.** Let $M_1, \ldots, M_t$ be a DNF resolution refutation of $F$, of $\Sigma_2$ space $s$. We will construct a sequence $T_1, \ldots, T_t$ of derivations in the system “resolution plus the F-cut rule (8)”. The property we are going to maintain is that for every clause $D$ labelling a leaf of $T_i$, either $D$ is a weakening of a clause $C$ in $F$ (call such a leaf an axiom leaf) or the following hold:

1. for every $G \in M_i$, $\alpha_D$ satisfies some term of $G$;
2. $W(D) \leq \Sigma_2 \text{Space}(M_i)$.

$T_1$ has one vertex labelled by the empty clause. Now suppose we have constructed $T_{i-1}$ 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 $M_i = M_{i-1} \cup \{C\}$, where $C = \ell_1 \lor \cdots \lor \ell_r$ is either a clause of $F$ (viewed as a 1-DNF) or a logical axiom $x \lor \pi$. If $C$ and $D$ contain conflicting literals, then item 1 is automatically satisfied and we do nothing at this leaf. Next, $C \subseteq D$ would have implied that $C$ is a clause of $F$ which is impossible since we have assumed that the leaf is non-axiom. Thus, there exists at least one $j \in [r]$ such that $\ell_j \notin D$, and for any such $j$ we add to $v$ a child labelled by $D \lor \ell_j$. This will be an application of the $F$-cut rule if $C$ is a clause or of the resolution rule if $C$ is $x \lor \pi$.

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

The case of an *inference* is immediately taken care of by Lemma 3.3. $D$ does not change. Since $\perp \notin M_i$, $T_i$ may not contain any non-axiom leaves and hence defines a refutation. Also, it is clear from the construction and property 2 above that any clause $D$ appearing in it must satisfy $W(D) \leq \max_{1 \leq i \leq t} \Sigma_2 \text{Space}(M_i) = s$. Hence $W(\vdash_F \perp) \leq s$. 


For the converse inequality, suppose that \( C_1, \ldots, C_t \) is a refutation in the system “resolution plus the \( F \)-cut rule”, of width \( w \). For every \( i \in [t] \), set \( G_i := \bigvee_{j=1}^{t} C_j \). Each \( G_i \) is a \( w \)-DNF. For our small space refutation, we will first derive \( G_i \) and \( G_{i-1} \lor C_i \), then cutting on \( C_i \) derive from these formulas \( G_{i-1} \), then derive \( G_{i-2} \lor C_{i-1} \), and continue this way until we get the empty clause. Notice that \( G_{i-1} \lor C_i \) is either a tautology with an obvious derivation in DNF resolution, or \( C_i \) is a clause of \( F \). In the latter case, we can immediately derive \( G_{i-1} \lor C_i \). Otherwise, \( C_i \) will be the result of applying either the resolution rule or the weakening rule or \( F \)-cut rule to some clauses among \( C_1, \ldots, C_{i-1} \). In either case, it can be checked that \( G_{i-1} \lor C_i \) has a tree-like proof of \( \Sigma_2 \) space at most \( 3w \), and therefore the proof above can be carried in \( \Sigma_2 \) space at most \( 5w \).

\[ \Sigma \]

Remark 3.5. The second part of this proof implies that \emph{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 \emph{a posteriori}, that we can balance our definition of \( \Sigma_2 \) space replacing in it \( W(G_1) + \ldots + W(G_s) \) with \( s \cdot \max(W(G_1), \ldots, W(G_s)) \), and the resulting measure will still be equivalent to \( \Sigma_2 \) space. We are not aware of a direct proof of this simulation by-passing width.

We get from Theorem 3.4 that strong length-space trade-offs conjectured for variable, clause and monomial space, are ruled out for DNF resolution. In particular, we get:

Corollary 3.6. For any unsatisfiable CNF formula \( F \) with \( n \) variables,

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

Proof. Let \( s := \Sigma_2 \text{Space}(F \vdash \bot) \). By the first part of Theorem 3.4, \( F \) has a width \( O(s) \) resolution refutation with the additional \( F \)-cut rule. We apply to this refutation the construction from the second part of Theorem 3.4 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,

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

Corollary 3.7. If \( F \) has a constant \( \Sigma_2 \) space refutation, then it has a refutation of constant \( \Sigma_2 \) space and polynomial length.

Proof. The refutation constructed in the proof of Corollary 3.6 will in our case also have constant \( \Sigma_2 \) space.

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

Theorem 3.8. Let \( F \) be an unsatisfiable CNF formula over \( n \) variables. Then

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

Proof. For the upper bound, let \( \pi \) 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 3.4 once again. By inspection (cf. the proof of Corollary 3.6), 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 [16] showing (3). Namely, by pebbling, a tree-like proof $T$ of size $s > 1$ can be turned into a proof in configurational form, where each configuration contains at most $\log s$ formulas occurring in $T$. If $T$ is a refutation in $R(\log)$, then all terms occurring in $T$ have width at most $\log s$, so the resulting refutation has $\Sigma_2$ space $(\log s)^2$.

Remark 3.9. For the more conventional system $\text{Res}(\log n)$, the subsystem of DNF resolution where each DNF in a refutation of $F$ is required to have width $O(\log n)$, $n$ the number of variables of $F$, the second inequality in Theorem 3.8 is false (see Figure 2). This follows from an easy adaptation of the proof of [15, Corollary 14].

4 Size-space trade-offs and tree-like size lower bounds

4.1 A lower bound on regularized monomial space

One application of the results of the previous section is that they easily allow us to show trade-offs between regularized clause or monomial space and size.

It is known [22, 21] that there are formulas $F$ of size $\Theta(n)$ that have a resolution refutation of size $O(n)$ (and thus a $O(n)$ refutation in the stronger system PCR), but $\text{MSpace}^*(F \vdash \bot) \geq n^{1/2}/(\log n)^{O(1)}$. Theorem 3.1, combined with the lower bounds of [5] and [17] on $\log ST$ and $\text{TCSpace}$ immediately gives the following improvement.

Theorem 4.1. 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 $\text{MSpace}^*(F \vdash \bot) \geq \Omega(n/\log n)$.

Proof. [5] demonstrates the existence of an $O(1)$-CNF $F$ that has resolution refutations of size $O(n)$, width $O(1)$, and such that $\log ST(F \vdash \bot) \geq \Omega(n/\log n)$. In fact, [5] 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 [35] played on $F$. Now, it is proved in [17] that this number of points is precisely equal to $\text{TCSpace}(F \vdash \bot)$ and then the result immediately follows from the second inequality in Theorem 3.1.

4.2 Trade-offs between positive depth and tree-like size for Horn formulas and tree-like size lower bounds

We would like next to focus on tree-like size lower bounds for resolution attained for formulas with small clause space. We will show that a tree-like resolution refutation of a Horn formula actually describes a pebbling strategy, the space and time of the strategy being the positive depth and size respectively of the proof. This gives a more transparent version of the result of [5] used in the proof of Theorem 4.1, which moreover has a natural generalization allowing us to prove some tree-like lower bounds for formulas of small clause space.

---

5 We would like to stress that, following the (perhaps, unfortunate) convention established in the previous papers, we mean potential trade-offs. In other words, we prove lower bounds on the regularized space and we only know that our method fails to extend them to the ordinary monomial space. As we explained in Section 1 and will further elaborate in Section 5, proving actual trade-offs in this setting is a major and difficult open problem.
4.2.1 Horn formulas – basics

Horn formulas, that include pebbling formulas, have seen a plethora of applications in proof complexity over the past two decades, including separating resolution size from tree-like resolution size [5], separating width from variable space and clause space [4, 6, 7], separating depth from tree-like clause space [38], and giving trade-offs [4, 7, 22, 3], to name a few.

A CNF formula is called Horn if every clause in it has at most one non-negated variable. Equivalently, a Horn formula is a set of implications involving variables, with at most one variable at the right hand side of the implication. An implication of the form \( x_1, \ldots, x_k \rightarrow y \) is asserting that if all the \( x_i \)'s are true, then \( y \) is true; \( x_1, \ldots, x_k \rightarrow \) is asserting that one of the \( x_i \)'s is false, \( \rightarrow y \) is asserting that \( y \) is true, and \( \rightarrow \) is a contradiction.

The following result states that Horn formulas make up, in a certain sense, the easiest class of formulas for proof complexity. For its purposes, it is convenient to define a slightly modified version \( \text{CSpace}(\vdash_F \bot) \) of the clause space, in the same vein we defined \( W(\vdash_F \bot) \) above. Namely, we replace the three standard rules with the following

Three-in-one rule: from a configuration \( M \), infer any configuration \( M^* \subseteq M \cup F \cup \{C\} \), where \( C \) is obtained from clauses in \( M, F \) via the resolution or weakening rule.

\[ \text{Theorem 4.2.} \quad \text{Let } F \text{ be a CNF formula. The following are equivalent:} \]

1. \( F \) contains an unsatisfiable CNF sub-formula resulting from a Horn formula by negating some of its variables;
2. \( \text{CSpace}(\vdash \bot) \leq 3; \)
3. \( \text{CSpace}(\vdash_F \bot) \leq 1; \)
4. \( W(\vdash_F \bot) \leq 1. \)

\[ \text{Proof.} \quad \text{For } 1 \impliedby 2, \text{ we can w.l.o.g. assume that } F \text{ itself is an unsatisfiable Horn formula. We show, by induction on the number of variables } n, \text{ that it can be refuted in clause space } 3. \text{ The base case is trivial. Now, suppose that } n > 0, \text{ and let } y \text{ be a variable such that } F \text{ contains the clause } \rightarrow y. \text{ Such a clause must exist, for if every clause contained a negated variable, then we could satisfy } F \text{ by setting every variable to false. Setting } y := 1, \text{ we get an unsatisfiable Horn formula } F|_{y=1} \text{ with } n - 1 \text{ variables. From the induction hypothesis, there is a clause space } 3 \text{ refutation of } F|_{y=1}. \text{ Weakening every clause in it by } \neg y \text{ gives us a space } 3 \text{ proof of } \neg F \text{ from } F. \text{ Now we only have to download } y \text{ and infer } \bot. \]

For \( 2 \implies 3, \) let \( M_1, \ldots, M_t \) be a space 3 refutation of the formula \( F; \) we can assume w.l.o.g. that it does not contain weakening rules. Consider a path in the corresponding refutation tree of maximum possible length, say \( C_i \in M_t, (0 \leq i \leq h) \) are such that \( t_0 < \ldots < t_h = t \), \( C_0 \) is an axiom, \( C_h = \bot \) and for \( i \geq 1, C_i \) is obtained by resolving \( C_{i-1} \) with some \( D_{i-1} \in M_{i-1} \). It remains to show that \( D_{i-1} \) is actually an axiom for any \( i \geq 1 \). For \( i = 1 \) this follows from the maximality of the chosen path. For \( i \geq 2, \) we have \( M_{i-1} = \{C_{i-2}, D_{i-2}, C_{i-1}\} \). Therefore \( C_{i-1} \) is consistent (and hence not resolvable) with the two other clauses in \( M_{i-1} \). All clauses that may have been inferred in \( M_{i-1} \), \( M_{i-2}, \ldots, M_{i-1} \), must have \( C_{i-1} \) as one of their premises and, as a consequence, are also not resolvable with \( C_{i-1} \). Hence the only clauses in those configurations that may be resolvable with \( C_{i-1} \) (in particular, \( D_{i-2} \)) are the axioms.

The implication \( 3 \implies 4 \) is proven by an argument similar to the first part of the proof of Theorem 3.4. Namely, a space 1 refutation of minimum length in the three-in-one model must necessarily be a sequence \( \{C_1\}, \ldots, \{C_t\} \), where \( C_{i+1} \) is obtained by resolving \( C_i \) with a clause in \( F \). The non-axiom leaves of the tree \( T_i \) will simply be all those literals among \( \ell_{i,1}, \ldots, \ell_{i,r_i} \), where \( C_i = \ell_{i,1} \lor \ldots \lor \ell_{i,r_i} \), that are not axioms of \( F \). It can routinely be
checked that, as in the proof of Theorem 3.4, \( T \) will be a resolution derivation using only the \( F \)-cut rule (notice that to keep the width 1, the weakening rule has to be incorporated into the \( F \)-cut rule).

Finally, for \( 4 \implies 1 \), 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 refutation of \( F \), forces \( F \) to have a one literal clause (since the refutation must start somewhere), say \( \ell \). Setting \( \ell := 1 \), we get a width 1 refutation of \( F \mid \ell := 1 \). From the induction hypothesis, a sub-formula \( G \) of \( F \mid \ell := 1 \) is unsatisfiable Horn up to negating some variables. Let \( \hat{G} \) be the corresponding sub-formula of \( F \); \( \hat{G} \) is obtained from \( G \) by restoring \( \ell \) to some of its clauses. Then \( \hat{G} \land \ell \) is an unsatisfiable Horn sub-formula of \( F \).

### 4.2.2 Tree-like resolution proofs as pebbling strategies

The paper [4] shows that a configurational resolution refutation \( \pi \) of the so-called pebbling contradiction \( \text{Peb}_T \) on a graph \( G \) defines a pebbling strategy on \( G \), of time at most \( |\pi| \) and space equal to the variable space \( \text{VSpace}(\pi) \). These are strategies in the so-called black-white game of [14]. We shall show that a tree-like resolution proof \( T \) of any Horn formula \( H \) defines a pebbling strategy of time equal to the size of \( T \) and space essentially equal to the positive depth of \( T \). These are strategies in the more basic black-only pebbling game that in the case \( H = \text{Peb}_G \) corresponds to the black-only pebbling game on \( G \). Urquhart [38] showed how to relate them to ordinary depth. In a sense, our Proposition 4.3 below can be viewed as a (far-reaching) refinement of his result.

The rules of the black-only pebbling game, played on a Horn formula \( H \), are as follows. There is a limited amount of pebbles. Pebbles are placed on the variables of \( H \) according to the rules:

1. A pebble can be placed on a variable \( y \) if \( x_1, \ldots, x_k \rightarrow y \) 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 \( \rightarrow y \) 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 the variables of \( H \). A pebbling strategy is a sequence of configurations, each resulting from the previous one by one of the rules above. We say that a pebbling strategy refutes \( H \) if it ends with a configuration where for some clause \( x_1, \ldots, x_k \rightarrow y \) of \( H \), all variables \( x_1, \ldots, x_k \) are pebbled. Note that if \( H \) is unsatisfiable, then such a clause must exist.

\[ \begin{align*}
\text{Proposition 4.3.} & \quad \text{Let } H \text{ be an unsatisfiable Horn formula. A tree-like resolution refutation } T \text{ of } H \text{ of size } s \text{ and positive depth } d \text{ can be converted into a pebbling strategy that, starting with the empty configuration, refutes } H \text{ in at most } s \text{ steps and using at most } d + 1 \text{ pebbles.} \\
\end{align*} \]

\[ \text{Proof.} \] We begin with a slight modification of our refutation. Namely, viewing \( T \) as a decision tree, its nodes naturally correspond to partial assignments, and for the clause \( C \) sitting at the node \( \alpha \), we have \( \alpha \models \neg C \). Let us replace \( C \) with the maximal clause satisfying this property. This will give us a refutation, of the same size and positive depth, in which the resolution rule (7) is reduced to

\[ \begin{align*}
C \lor x & \quad C \lor \overline{\pi} \\
\hline
C & \quad \text{(9)}
\end{align*} \]

and leaves are labelled by weakenings of axioms in \( H \).

This refutation need not necessarily consist of Horn formulas even if the original one did so. Nonetheless we will still represent clauses in the sequential form \( S \rightarrow T \), where \( S, T \) are disjoint sets of variables, like at the beginning of Section 4.2.1. Note that \( |S| \leq d \) for any clause \( S \rightarrow T \) appearing in \( T \).
We shall now show by induction that every subtree of $T$ deriving a clause $S \rightarrow T$, leads to a pebbling strategy that, starting with pebbles on all variables of $S$ and using at most $d + 1$ pebbles, either refutes $H$, or ends with a configuration which has pebbles on all variables of $S$ and on one variable of $T$. Thus, if $T$ is empty then the former must occur and, in particular, the strategy corresponding to the empty sequent $\vdash$ will start with no pebbles on the variables of $H$ and will refute $H$.

Suppose that $S \rightarrow T$ is at a leaf. If there are variables $x_1, \ldots, x_k$ in $S$ such that $x_1, \ldots, x_k \rightarrow$ is a clause of $H$, then that leaf describes a strategy that, starting with pebbles on all variables in $S$, immediately refutes $H$. Otherwise, there must be variables $x_1, \ldots, x_k$ in $S$ and a variable $y$ in $T$ such that $x_1, \ldots, x_k \rightarrow y$ is a clause of $H$. Then the strategy of that leaf is to put a pebble on $y$. Since $|S| \leq d$, the number of pebbles used is at most $d + 1$, as required.

If $S \rightarrow T$ is not at a leaf, then consider its left and right subtrees $T_1$ and $T_2$ proving $S, x \rightarrow T$ and $S \rightarrow T, x$ respectively (cf. (9)). The strategy corresponding to $S \rightarrow T$ is defined as follows. First follow $T_2$’s strategy. If that strategy either refutes $H$ or places a pebble on one of $T$’s variables, then we are done. Otherwise, when the strategy of $T_2$ is concluded, there are pebbles on $S$ and $x$. Remove all other pebbles and follow the strategy of $T_1$. The bound $d + 1$ on the number of pebbles used at any moment follows from the same bound for $T_1$ and $T_2$.

Clearly, the number of steps of the pebbling strategy corresponding to $\vdash$ is at most the size of $T$, and the required bound on the number of pebbles was already noticed.

**Remark 4.4.** The proof of Proposition 4.3 relies on an intuitionistic interpretation of the resolution rule. In the intuitionistic tradition, the denotational view of assigning truth values is, philosophically, nonsense. A proposition is “true” if it is provable, and a proof of e.g. a formula $S \rightarrow T$ is a construction that given proofs of all the elements of $S$ produces a proof of some element in $T$. What Proposition 4.3 says is that a tree-like resolution derivation of $S \rightarrow T$ precisely describes such a construction, assuming that proofs of all the clauses of $H$ are known. Moreover this construction will be economical in the number of steps and memory if the size and the positive depth respectively of the proof are small. Let us further notice, that although Proposition 4.3 is stated for Horn formulas, it really is general; it could be stated, with minimal modifications, for arbitrary CNFs.

### 4.2.3 Tree-like size lower bounds

The following theorem turns pebbling time-space trade-offs for a Horn formula $H$ into tree-like size lower bounds for its substituted version $H[\lor_2]$. We formulate it in a somewhat general form, to account for various kinds of pebbling trade-offs in the literature. The substituted version $F[\lor_2]$ of a CNF $F(x_1, \ldots, x_n)$ is defined by replacing $x_i$ with $y_i \lor z_i$ for mutually distinct variables $\{y_1, z_1, \ldots, y_n, z_n\}$, followed by converting the result back to the CNF form in the straightforward way.

**Theorem 4.5.** 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 $(d - 1) \cdot f(s) \geq g(n)$ for non-decreasing positive functions $f, g$. Then $f(t) \log t \geq g(n)$, where $t \equiv S_T(H[\lor_2] \vdash 0)$.

**Proof.** Create a probability space of partial assignments by choosing independently for every variable $x$ of $H$, which was substituted by $y \lor z$, one of $y$ and $z$ with probability $1/2$ and setting it to zero. Note that for any $\alpha$ from this space, $H[\lor_2]|_\alpha$ is identical to $H$ up to
renaming its variables and hence \( T|_\alpha \) is a refutation of \( H \), again up to renaming variables. Let \( T \) be an arbitrary tree-like resolution refutation of \( H[\lor_2] \) of size \( t \) represented as in the proof of Proposition 4.3. That is weakenings are omitted from the resolution rule, and appear at the leaves only. Let \( D_1, \ldots, D_k \) be all clauses of positive depth \( g(n)/f(t) \) occurring in \( T \).

We have that

\[
P \left[ \bigvee_{i=1}^k (D_i|_\alpha \neq 1) \right] \leq \sum_{i=1}^k P[D_i|_\alpha \neq 1] \leq k2^{-g(n)/f(t)} \leq 12^{-g(n)/f(t)}.
\]

If \( f(t) \log t < g(n) \), then the above probability is smaller than 1, which means that there is a point \( \alpha \) in our sample space such that \( T|_\alpha \) is a tree-like resolution refutation of size at most \( t \) and positive depth \( \leq g(n)/f(t) \). This, from Proposition 4.3, gives a pebbling strategy that refutes \( H \) in \( t \) steps using \( d \) pebbles, where \( (d-1) \cdot f(t) < g(n) \).

Recall that for a DAG \( G \), the pebbling contradiction \( \text{Peb}_G \) is defined as the Horn formula consisting of all clauses \( S \rightarrow x \), where \( x \in V(G) \) and \( S \) is the set of all its immediate predecessors, as well as the clauses \( x \rightarrow \) for any sink \( x \). Plugging into Theorem 4.5 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.

\begin{corollary}[5] There are formulas of size \( O(n) \) having DAG-like resolution refutations of size \( O(n) \), every tree-like resolution refutation of which requires size \( \exp(\Omega(n/\log n)) \).
\end{corollary}

\textbf{Proof.} This is by setting \( f := 1 \) in Theorem 4.5, and using the graphs of [34] having constant in-degree and requiring \( \Omega(n/\log n) \) pebbles to pebble. \( \triangleright \)

The next result was promised in the introduction. It should be compared with Theorem 4.2.

\begin{theorem} There are formulas of size \( O(n) \) having tree-like resolution refutations of clause space 4, every tree-like resolution refutation of which has size \( \Omega(n^2/\log n) \).
\end{theorem}

\textbf{Proof sketch.} This is by setting \( f(t) := t \) in Theorem 4.5, and using the graphs of [28, Theorem 2.3.2] having linear size and exhibiting a \( dt \geq \Omega(n^2) \) trade-off. These graphs can be pebbled using 3 pebbles, and that immediately gives that \( \text{CSpace}(\text{Peb}_{G_i}[\lor_2] \vdash \bot) \leq O(1) \). By being more careful, it is possible to bring the space down to the minimum possible value, namely 4, for which a super-linear lower bound on tree-like resolution size is possible. \( \triangleright \)

By using the construction from [28, Theorem 4.2.6], Theorem 4.7 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 4.7 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 [35], which also gives a lower bound for the clause space of tree-like resolution refutations (cf. Theorem 4.1).

\section{5 Concluding remarks}

We showed that \( \log S_T \), \( \text{CSpace}^* \) and \( \text{MSpace}^* \) are equivalent up to polynomial and \( \log n \) factors, demonstrating a picture perfectly analogous to the picture involving \( D \), \( \text{VSpace}^* \) and \( \text{TSpace}^* \) in [37]. The most important question remains (widely) open:

\begin{problem} Is it true that \( \text{CSpace} \approx \log S_T \) or \( \text{MSpace} \approx \log S_T \)? Recall for comparison that \( \log S_T \approx \text{CSpace}^* \approx \text{MSpace}^* \).
\end{problem}
Equivalently, do there exist strong trade-offs between clause (or monomial) space and length? It should be contrasted with trade-offs results in e.g. [7, 3], and it is a perfect analogue of Urquhart’s question [38] about variable space vs. depth studied in [37, Section 6]. Let us make a few more remarks about this problem.

Firstly, for very small space essentially this question was already asked in the literature before. Namely (see e.g. [30, 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:

▶ **Problem 5.2 (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 5.1, 5.2 and Nordström’s question. In fact, we would like to cautiously conjecture that there are formulas $F$ with $\text{CSpace}(F \vdash \bot) \leq 5$ and $\text{CSpace}^*(F \vdash \bot) \geq \exp\{n^{O(1)}\}$. But the only result we were able to prove in that direction is the rather weak Theorem 4.7.

Secondly, as suggested by Figure 1, any strong separation between monomial space and clause space would immediately solve Problem 5.1 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:

▶ **Problem 5.3.** Is it true that $\text{CSpace} \approx \text{MSpace}$? Is it true that $\text{MSpace} \approx W$?

We note that by the result from [31, 6], at least one of these must be false. A quadratic separation between width and monomial space has been recently proved by the first author (manuscript in preparation). For a discussion on related topics, see also [11, Section 7.5.5].

Finally, while all these conjectured trade-offs are very strong, they are still not super-critical in the sense of [36] (the required lower bound on length never exceeds $2^n$). However, as we pointed out in Section 3.1 in all these questions refutation length can be replaced with depth. Since the depth, as a stand-alone measure, is always bounded by $n$, these actually are questions about the existence of a *super-critical* trade-off between clause space and depth.

We have (somewhat surprisingly) proved that DNF resolution behaves very differently from resolution with respect to space. Intermediate systems based on $\text{Res}(k)$ 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 $\text{Res}(k)$-refutations, the definition of space from [15, 7] (formula space) coincides with ours up to a factor of $k$ so we need not distinguish between the two. Then Theorem 3.1 readily generalizes to this regime and gives $\log S_{T,\text{Res}(k)} \approx \text{Res}(k)\text{Space}^*$, extending the bottom half of Figure 1 as shown in Figure 2. The proof of Corollary 3.6, however, fails for a constant $k$ as badly as it fails for $k = 1$. Hence we have one more question to ask:

▶ **Problem 5.4 (Res(k)-variant).** Is there a constant $k > 0$ such that $\log S_{T,\text{Res}(k)} \approx \text{Res}(k)\text{Space}$ or at least $\log S_{T,\text{Res}(k)} \preceq \text{CSpace}$?

Let us also mention that as $k$ increases, both hierarchies, $\log S_{T,\text{Res}(k)}$ (and, hence, also $\text{Res}(k)\text{Space}^*$) and $\text{Res}(k)\text{Space}$ are proper ([15] and [7] respectively). This excludes the dual version of Remark 3.5: 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 [4]). Let us re-iterate the problem posed e.g. in [37]:

**Problem 5.5.** 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.

---

**References**


