Simplified and Improved Separations Between Regular and General Resolution by Lifting

Marc Vinyals[1], Jan Elffers[2,4], Jan Johannsen[3], and Jakob Nordström[4,2]

[1] Technion, Haifa, Israel – marcviny@cs.technion.ac.il
[2] Lund University, Sweden – jan.elffers@cs.lth.se
[3] Ludwig-Maximilians-Universität München, Germany – jan.johannsen@ifi.lmu.de
[4] University of Copenhagen, Denmark – jn@di.ku.dk

Abstract. We give a significantly simplified proof of the exponential separation between regular and general resolution of Alekhnovich et al. (2007) as a consequence of a general theorem lifting proof depth to regular proof length in resolution. This simpler proof then allows us to strengthen the separation further, and to construct families of theoretically very easy benchmarks that are surprisingly hard for SAT solvers in practice.

1 Introduction

In the *resolution proof system* [17] the unsatisfiability of a formula in conjunctive normal form (CNF) is shown by iteratively deriving new disjunctive clauses until contradiction is reached (in the form of the empty clause). A resolution proof is said to be *regular* [59] if along the path of derivation steps from any input clause to contradiction every variable is eliminated, or *resolved*, at most once. This condition appears quite natural, since it essentially means that intermediate results should not be proven in a form stronger than what will later be used in the derivation, and indeed DPLL-style algorithms [26,27] can be seen to search for regular proofs. In view of this, it is natural to ask whether regularity can be assumed without loss of proof power, but this was ruled out in [40]. General resolution was shown to be superpolynomially stronger than regular resolution in [31], a separation that was improved to exponential in [2,61]. Regular resolution is in turn known to be exponentially stronger than *tree-like* resolution [11,19], where no intermediate clause can be used for further derivations more than once.

There is an interesting connection here to the quest for a better understanding of state-of-the-art SAT solvers based on *conflict-driven clause learning (CDCL)* [47,48].[5] Tree-like resolution corresponds to solvers without any clause learning, whereas CDCL solvers have the potential to be as strong as general resolution [3,51]. The proofs of the latter result crucially use, among other assumptions, that solvers make frequent restarts, but it has remained open whether this is strictly needed, or whether "smarter" CDCL solvers without restarts could be equally powerful. To model CDCL without restarts, proof systems such as *pool resolution* [62] and different variants of *resolution trees with lemmas (RTL)* [20]

[5] A similar idea in the context of CSPs was independently developed in [5].

have been introduced, which sit between regular and general resolution. Therefore, if one wants to prove that restarts increase the reasoning power of CDCL solvers, then formulas that could show this would, in particular, have to separate regular from general resolution. However, all known formulas witnessing this separation [2,61] have also been shown to have short pool resolution proofs [18,21]. It is therefore interesting to develop methods to find new formula families separating regular and general resolution. This brings us to our next topic of *lifting*.

In one sentence, a *lifting theorem* takes a weak complexity lower bound and amplifies it to a much stronger lower bound by simple syntactic manipulations. Focusing for concreteness on Boolean functions, one can take some moderately hard function $f : \{0,1\}^n \to \{0,1\}$ and compose it with a *gadget* $g : \{0,1\}^m \to \{0,1\}$ to obtain the new *lifted function* $f \circ g^n \colon \{0,1\}^{mn} \to \{0,1\}$ defined as $f(g(\boldsymbol{y}_1), g(\boldsymbol{y}_2), \ldots, g(\boldsymbol{y}_n))$, where $\boldsymbol{y}_j \in \{0,1\}^m$ for $j \in [n]$. If the gadget g is carefully chosen, one can show that there is essentially no better way of evaluating $f \circ g^n$ than first computing $g(\boldsymbol{y}_j)$ for all $j \in [n]$ and then applying f to the outputs. From this it follows that $f \circ g^n$ is a much harder function than f or g in isolation.

A seminal early paper implementing this paradigm is [54], and the rediscovery and strengthening of this work has led to dramatic progress on many long-standing open problems in communication complexity [33,34,35,37,38]. Other successful examples of the lifting paradigm include lower bounds in monotone complexity [52,53,58], extension complexity [32,43,45], and data structures [24]. Lifting has also been a very productive approach in proof complexity. Interestingly, many of the relevant papers [6,8,9,12,13,19,41,49,50] predate the "lifting revolution" and were not thought of as lifting papers at the time, but in later works such as [29,36,57] the connection is more explicit.

As described above, in the lifting construction different copies of the gadget g are evaluated on disjoint sets of variables. In [55] it was instead proposed to let the variable domains for different gadgets overlap as specified by well-connected so-called *expander graphs*. This idea of recycling variables between gadgets has turned out to be very powerful, and an ingredient in a number of strong trade-off results between different complexity measures [15,16,56].

Our Contributions The starting point of our work is the simple but crucial observation that the *stone formulas* in [2] can be viewed as lifted versions of *pebbling formulas* [14] with maximal overlap, namely as specified by complete bipartite graphs. This raises the question whether there is a lifting theorem waiting to be discovered here, and indeed we prove that the separation in [2] can be proven more cleanly as the statement that strong enough lower bounds on proof *depth* can be lifted to exponential lower bounds on proof *length* in regular resolution. This in turn implies that if one can find formulas that have short resolution proofs with only small clauses, but that require large depth, then lifting with overlap yields formulas that separate regular and general resolution.

This simpler, more modular proof of [2] is the main conceptual contribution of our paper, but this simplicity also opens up a path to further improvements. Originally, lifting with overlap was defined in [55] for low-degree expander graphs,

and we show that our new lifting theorem can be extended to this setting also. Intuitively, this yields "sparse" versions of stone formulas that are essentially as hard as the original ones but much smaller. We use this finding for two purposes.

Firstly, we slightly improve the separation between regular and general resolution. It was known that there are formulas having general resolution proofs of length L that require regular proofs of length $\exp\big(\Omega\big(L/((\log L)^7 \log\log L)\big)\big)$ [61]. We improve the lower bound to $\exp\big(\Omega\big(L/((\log L)^3(\log\log L)^5)\big)\big)$.

Secondly, and perhaps more interestingly from an applied perspective, sparse stone formulas provide the first benchmarks separating regular and general resolution that are sufficiently small to allow meaningful experiments with CDCL solvers. Original stone formulas have the problem that they grow very big very fast. The so-called *guarded* formulas in [2,61] do not suffer from this problem, but the guarding literals ensuring the hardness in regular resolution are immediately removed during standard preprocessing, making these formulas very easy in practice. In contrast, sparse stone formulas exhibit quite interesting phenomena. Depending on the exact parameter settings they are either very dependent on frequent restarts, or very hard even with frequent restarts. This is so even though short proofs without restarts exist, which also seem to be possible to find algorithmically if the decision heuristic of the solver is carefully hand-coded.

Outline of This Paper After reviewing some preliminaries in Section 2, we present our proof of [2] as a lifting result in Section 3. We extend the lower bound to sparse stone formulas in Section 4. We conclude with brief discussions of some experimental results in Section 5 and directions for future research in Section 6.

2 Preliminaries

Resolution Throughout this paper 0 denotes false and 1 denotes true. A literal a is either a variable x or its negation $\overline{x}$. A clause C is a disjunction $a_1 \vee \cdots \vee a_k$ of literals; the *width* of C is k. A CNF formula $F = C_1 \wedge \cdots \wedge C_m$ is a conjunction of clauses, the *size* (or *length*) of which is m. We view clauses and CNF formulas as sets, so order is irrelevant and there are no repetitions.

A *resolution proof* for (the unsatisfiability of) F, also referred to as a *resolution refutation* of F, is a sequence of clauses, ending with the empty clause $\bot$ containing no literals, such that each clause either belongs to F or is obtained by applying the resolution rule $C \vee x,\ D \vee \overline{x} \vdash C \vee D$ to two previous clauses. If we lay out the proof as a graph the result is a directed acyclic graph (DAG) where each node is labelled with a clause, where without loss of generality there is a single source labelled $\bot$, where each sink is a clause in F, and each intermediate node can be written on the form $C \vee D$ with edges to the children $C \vee x$ and $D \vee \overline{x}$. The length of a refutation is the number of clauses, its width is the maximal width of a clause in it, and its depth is the longest path in the refutation DAG. The resolution length, width and depth of a formula are the minimum over all resolution refutations of it.

A restriction ρ is a partial assignment of truth values to variables. We write $\rho(x) = *$ to denote that variable x is unassigned. We obtain the restricted

clause $C\restriction_\rho$ from C by removing literals falsified by ρ, and the restricted formula $F\restriction_\rho$ from F by removing clauses satisfied by ρ and replacing other clauses C by $C\restriction_\rho$.

If a formula F has a resolution refutation π, then for every restriction ρ the restricted formula $F\restriction_\rho$ has a refutation π'—denoted by $\pi\restriction_\rho$—the length, width and depth of which are bounded by the length, width and depth of π, respectively. If π is regular, then so is $\pi\restriction_\rho$. We will need the following straightforward property of resolution depth.

Lemma 1 ([60]). *If F requires resolution depth D, then for every variable x in F it holds for some $b \in \{0,1\}$ that $F\restriction_{\{x:=b\}}$ requires resolution depth $D-1$.*

Branching Programs In the *falsified clause search problem* for an unsatisfiable CNF formula F, the input is some (total) assignment α and a valid output is any clause of F that α falsifies.

From a resolution refutation of F we can build a *branching program* for the falsified clause search problem with the same underlying graph, where every non-source node queries a variable x and has outgoing edges 0 and 1, and where any assignment α leads to a sink labelled by a clause that is a valid solution to the search problem for F. We maintain the invariant that an assignment α can reach a node labelled by C if and only if α falsifies C—in what follows, we will be slightly sloppy and identify a node and the clause labelling it. In order to maintain the invariant, if a node $C \vee D$ has children $C \vee x$ and $D \vee \overline{x}$, we query variable x at that node, move to the child with the new literal falsified by the assignment to x, and forget the value of any variable not in this child. A proof is regular if and only if it yields a *read-once* branching program, where any variable is queried at most once along any path, and it is tree-like if it yields a search tree.

Pebbling Formulas Given a DAG H of indegree 2 with a single sink, the pebbling formula over H [14], denoted Peb_H, has one variable per vertex, a clause u for each source u, a clause $\overline{u} \vee \overline{v} \vee w$ for each non-source w with predecessors u and v, and a clause $\overline{z}$ for the sink z.

Pebbling formulas over n-vertex DAGs H have short, small-width refutations, of length $\mathrm{O}(n)$ and width 3, but may require large depth. More precisely, the required depth coincides with the so-called *reversible pebbling number* of H [22], and there exist graphs with pebbling number $\Theta(n/\log n)$ [30]. We will also need that so-called *pyramid graphs* have pebbling number $\Theta(\sqrt{n})$ [23,25].

Lifting We proceed to define *lifting with overlap* inspired by [55]. Let F be a formula with n original variables x_i. We have m new main variables r_j, which we often refer to as stone variables. Let G be a bipartite graph of left degree d and right degree d' with original variables on the left side and main variables on the right side. We have dn new selector variables $s_{i,j}$, one for each edge (i,j) in G.

For convenience, let us write $y^1 = y$ and $y^0 = \overline{y}$ for the positive and negative literals over a variable y. Then the lifting of x_i^b for $b \in \{0,1\}$ is the conjunction of d clauses $\mathcal{L}^G(x_i^b) = \bigwedge_{j \in N(i)} \overline{s}_{i,j} \vee r_j^b$. The lifting of a clause C of width w is the expression $\mathcal{L}^G(C) = \bigvee_{x_i^b \in C} \mathcal{L}^G(x_i^b)$, expanded into a CNF formula of

width $2w$ and size d^w. The lifting of a CNF formula F is the formula $\mathcal{L}^G(F) = \bigwedge_{C \in F} \mathcal{L}^G(C) \wedge \bigwedge_{i \in [n]} \bigvee_{j \in N(i)} s_{i,j}$ of size at most $d^w|F| + n$. We will omit the graph G from the notation when it is clear from context.

If G is a disjoint union of stars, then we obtain the usual lifting defined in [7], and if G is a complete bipartite graph with $m \geq 2n$ and F is a pebbling formula, then we obtain a stone formula [2]. We will need the fact, implicit in [13], that formulas with short, small-width refutations remain easy after lifting.

Lemma 2. *Let π be a resolution refutation of F of length L and width w, and let G be a bipartite graph of left degree d. Then there is a resolution refutation of $\mathcal{L}^G(F)$ of length $\mathrm{O}(d^{w+1}L)$.*

For the particular case of pebbling formulas, where there is a refutation where each derived clause is of width at most 2 even if some axioms are of width 3, the upper bound can be improved to $\mathrm{O}\big(d^3 L\big)$.

Graphs In Section 3, we use complete bipartite graphs to reprove the known lower bounds on stone formulas. In Section 4, we consider bipartite random graphs sampled from the $\mathcal{G}(n, m, d)$ distribution, where the left and right sides U and V have n and m vertices respectively, and d right neighbours are chosen at random for each left vertex.

A bipartite graph is an (r, κ)-*expander* if every left subset of vertices $U' \subseteq U$ of size $|U'| \leq r$ has at least $\kappa|U'|$ neighbours. It is well-known (see for instance [39]) that random graphs are good expanders.

Lemma 3. *With high probability a graph $G \sim \mathcal{G}(n, m, d)$ with $d = \Theta(\log(n/m))$ is an (r, κ)-expander with $\kappa = \Theta(d)$, $r = \Theta(m/\kappa)$, and right degree $d' \leq 2dn/m$.*

The following lemma, as well as its proof, is essentially the same as Lemmas 5 and 6 in [1] but adapted to vertex expansion.

Lemma 4. *If G is an (r, κ)-expander, then for every set $V' \subseteq V$ of size at most $\kappa r/4$ there exists a set $U' \subseteq U$ of size at most $r/2$ such that the graph $G \setminus (U' \cup N(U') \cup V')$ obtained from G by removing U', $N(U')$, and V' is an $(r/2, \kappa/2)$-expander.*

Matchings and the Matching Game A matching μ in a bipartite graph is a set of vertex-disjoint edges. We write $\mu(u) = v$ if the edge (u, v) is in μ. The *matching game* [10] on a bipartite graph is played between two players Prover and Disprover, with r fingers each numbered $1, \ldots, r$. In each round:

- either Prover places an unused finger i on a free vertex $u \in U$, in which case Disprover has to place his i-th finger on a vertex $v \in N(u)$ not currently occupied by other fingers;
- or Prover removes one finger i from a vertex, in which case Disprover removes his i-th finger.

Prover wins if at some point Disprover cannot answer one of his moves, and Disprover wins if the game can continue forever.

Theorem 5 ([10, Theorem 4.2]). *If a graph is an $(r, 1+\delta)$-bipartite expander, then Prover needs at least $\delta r/(2+\delta)$ fingers to win the matching game.*

3 Lower Bound for Stone Formulas as a Lifting Theorem

We reprove the result in [2] by reinterpreting it as a lifting theorem.

Theorem 6. *If F has resolution depth D and $m \geq 2D$, then $\mathcal{L}^G(F)$ for G the complete bipartite graph $K_{n,m}$ has regular resolution length* $\exp(\Omega(D^2/n))$.

When we choose as F the pebbling formula of a graph of pebbling number $\Omega(n/\log n)$ [30] we reprove the result in [2], slightly improving the lower bound from $\exp(\Omega(n/\log^3 n))$ to $\exp(\Omega(n/\log^2 n))$.

Corollary 7. *There are formulas that have general resolution refutations of length* $\mathrm{O}(n^4)$ *but require regular resolution length* $\exp(\Omega(n/\log^2 n))$.

We start with an overview and a few definitions common to this and the next section. The proof at a high level follows a common pattern in proof complexity: given some complexity measure on clauses, we apply a restriction to the resolution refutation that removes all complex clauses from a short enough proof. In a separate argument, we show that the restricted formula always requires complex clauses, contradicting our assumption of a short refutation.

To build a restriction we use the following concepts. Let $\mu\colon I \to J$ be a partial matching from original to stone variables. A matching μ induces an assignment ρ to selector variables as follows.

$$\rho(s_{i,j}) = \begin{cases} 1 & \text{if } \mu(i) = j, \\ 0 & \text{if } i \in \operatorname{dom}(\mu) \text{ or } j \in \operatorname{im}(\mu) \text{ but } \mu(i) \neq j, \\ * & \text{otherwise.} \end{cases}$$

We say that an assignment ρ whose restriction to selector variables is of this form *respects the lifting* because $\mathcal{L}^G(F)\restriction_\rho = \mathcal{L}^{G'}(F\restriction_\sigma)$, where G' is the induced subgraph $G[(I \setminus \operatorname{dom}\mu) \cup (J \setminus \operatorname{im}\mu)]$ and σ is the induced assignment to original variables $\sigma(x_i) = \rho(r_{\mu(i)})$ if $i \in \operatorname{dom}(\mu)$, and $\sigma(x_i) = *$ otherwise. An assignment that respects the lifting is *uninformative* if it induces an empty assignment to original variables, that is $\rho(r_j) = *$ whenever $j \in \operatorname{im}\mu$. Given an uninformative assignment ρ and an assignment to original variables σ, we can extend the former to agree with the latter as $\rho(r_j) = \sigma(x_{\mu^{-1}(j)})$ if $j \in \operatorname{im}\mu$ and $\rho(r_j) = *$ otherwise. The size of an assignment is the maximum of the size of the matching and the number of assigned stone variables.

A helpful complexity measure is the width of a clause; we use a complexity measure from [2] that enforces an additional structure with respect to the lifting.

Definition 8. *A clause C is (c, z)-complex if either*

1. *C contains at least c stone variables,*
2. *there is a matching μ of size c such that C contains the literal $\overline{s}_{i,j}$ for each $(i, j) \in \mu$, or*
3. *there is a set W of size c where C contains at least z literals $s_{i,j}$ for each $i \in W$.*

In this section we only use (c, c)-complex clauses, which we refer to as c-complex. Note that c can range from 1 to m. We also need the following lemma, which can be established by a straightforward calculation.

Lemma 9. *Consider a set of s clauses $\mathcal{C}$ and a set of n possibly dependent literals $\mathcal{L}$ such that after setting $\ln(s)\, n/p$ literals in $\mathcal{L}$ (plus any dependencies), for each clause $C \in \mathcal{C}$ there is a subset $\mathcal{L}_C \subseteq \mathcal{L}$ of at least p literals, each of which satisfies C. Then there is a set of $\ln(s)\, n/p$ literals that satisfies $\mathcal{C}$.*

From now on we assume that G is the complete bipartite graph $K_{n,m}$. The first step is to show that we can remove all complex clauses from a short proof.

Lemma 10. *There exists $\epsilon > 0$ such that if π is a resolution refutation of $\mathcal{L}(F)$ of size $s = \exp(\epsilon c^3/mn)$, then there exists an uninformative restriction ρ of size $c/2$ such that $\pi\restriction_\rho$ has no c-complex clauses.*

Proof. We build a restriction greedily. First we choose a matching μ so that after setting the corresponding selector variables with the restriction ρ induced by μ we satisfy all c-complex clauses of type 2 and 3 in Definition 8. There are mn positive selector literals s_{ij}. A clause of type 2 is satisfied if we set one of c variables $s_{i,j} = 0$, and that happens if we assign a literal $s_{ij'} = 1$ with $j' \neq j$, for a total of $c(m-1) \geq c^2$ choices. A clause of type 3 is satisfied if we set one of c^2 literals $s_{i,j} = 1$. After picking k pairs to be matched there are still at least $(c-k)(m-k-1) \geq (c-k)^2$ literals available to satisfy clauses of type 2, and $(c-k)^2$ literals available to satisfy clauses of type 3, so we can apply Lemma 9 and obtain that setting $q \leq \ln(s)\, mn/(c^2/4)$ literals is enough to satisfy all such clauses. Note that we used that $k \leq \ln(s)\, mn/(c^2/4) \leq c/2$.

Next we extend ρ to ρ' by setting some stone variables that are untouched by μ so that we satisfy all clauses of type 1. There are $m - q$ such variables, hence at most $2m$ literals, and a clause is satisfied when one of c variables is picked with the appropriate polarity. After picking k literals there are at least $c - q - k \geq c/2 - k$ choices left for each clause, so we can apply Lemma 9 and get that setting $q' = \log s\, m/(c/8)$ variables is enough to satisfy all clauses of type 1. Note that we used that $k \leq \ln(s)\, m/(c/8) \leq c/4$, which follows from $\ln(s) \leq c^2/16m \leq c^3/16mn$.

The size of the restriction ρ' is then at most $c/2$. □

Next we show that *regular* resolution proofs always contain a complex clause.

Lemma 11. *If F requires depth D, then any regular resolution refutation of $\mathcal{L}(F)$ with $m < D$ has an $m/4$-complex clause.*

Proof. We build a path through the read-once branching program corresponding to the proof, using a decision tree T for F of depth D to give the answers to some queries. We also keep a matching μ, with the invariant that there is an edge (i, j) in the matching if and only if $s_{ij} = 1$ or there are $m/4$ stones $j' \neq j$ such that $s_{ij'} = 0$. We can do so using the following strategy as long as at most $m/4$ stones are assigned and at most $m/4$ stones are matched.

– If the adversary queries s_{ij} then if neither i nor j are matched we answer 1 and add (i, j) to the matching, if $\mu(i) = j$ we answer 1, and otherwise we answer 0. If more than $m/4$ variables $s_{ij'}$ are 0 (for i fixed and $j' \in [m]$) we choose one of the $m/4$ stones j'' that are not assigned, nor matched, nor have $s_{ij''} = 0$ and add (i, j'') to the matching.
– If the adversary queries r_j and j is matched to i, we answer b so that the depth of T only shrinks by 1 when original variable x_i is set to b, as given by Lemma 1. Otherwise we answer arbitrarily.
– If the adversary forgets a variable and there is an edge in the matching that does not respect the invariant, we remove it.

Assume for the sake of contradiction that we never reach an $m/4$-complex clause. Then we can maintain the invariant until we reach a leaf of the branching program, and that leaf never falsifies a clause of the form $\bigvee_{j\in[m]} s_{i,j}$. It follows that the path ends at a clause from $\mathcal{L}(D)$, at which point the depth of T reduced to 0. Observe that the depth of T only decreases by 1 when a stone variable is queried and that, since the branching program is read-once, these queries must be to D different stones, but only $m < D$ stones are available. □

We use these lemmas to complete the plan outlined at the beginning of this section and prove our lifting theorem.

Proof (of Theorem 6). Assume for the sake of contradiction that π is a refutation of $\mathcal{L}(F)$ of length less than $\exp(\delta D^2/n)$, where $\delta = \epsilon/1024$ for the ϵ of Lemma 10.

We invoke Lemma 10 with $c = D/8$ to obtain that there is an uninformative restriction ρ of size $D/16$ such that $\pi\restriction_\rho$ has no $D/8$-complex clauses. By Lemma 1 we can assign values to the matched stones in a way that the induced assignment to original variables σ yields a formula of depth $15D/16$. We additionally assign all but the first $15D/16 - 1$ stones arbitrarily and set all selector variables that point to an assigned stone to 0. Let ρ' be the new restriction.

The formula $F' = \mathcal{L}(F)\restriction_{\rho'}$ is the lifted version of a formula $F\restriction_\sigma$ of depth $D' = 15D/16$ with $m' = D' - 1$ stones, hence by Lemma 11 any refutation of F' has an $m'/4$-complex clause. But since $m'/4 \geq 15D/64 - 1 > D/8$, this contradicts the fact that the refutation $\pi\restriction_{\rho'}$ has no $D/8$-complex clauses. □

4 Lower Bound for Sparsely Lifted Formulas

We now generalize the lifting to sparse graphs. The first step is again to show that we can remove all complex clauses from a short proof, but this becomes a harder task so let us begin with an informal overview. Say that we start with a lifted formula whose selector variable graph is an expander and, as in Lemma 10, we want to make a few stones be assigned and a few stones be matched. After we remove these stone vertices from the graph, it will likely stop being an expander (e.g. because we will likely remove all the neighbours of some vertex).

Fortunately by Lemma 4 given a subset V' of right vertices to remove there is a subset U' of left vertices such that removing V', U', and $N(U')$ from the graph

yields an expander, but this is still not enough because removing U' forces us to a matching that may interfere with our plans. Maybe there is some vertex $i \in U'$ corresponding to an original variable that we want to assign to 0 but all of its neighbours are assigned to 1, or maybe there is some original variable $i \in U'$ all of whose neighbours are already matched to other original variables.

Our solution is to add one backup vertex for each stone vertex j, so that we can delay the expansion restoring step. Of course we cannot decide beforehand which vertices are primary and which are backup, otherwise it might be that all complex clauses would talk only about backup vertices and our assignment would not affect them, so we have to treat primary and backup vertices equally. But still we make sure that if a vertex j is assigned 1, then its backup is assigned 0 and viceversa, taking care of the first problem; and that if a stone vertex j is matched to some original variable i then its backup is still free and viceversa, taking care of the second problem.

To make the concept of backup vertices formal, we say that a bipartite graph G of the form $U \cup (V_0 \cup V_1)$ is a mirror if the subgraphs $G_0(U \cup V_0)$ and $G_1(U \cup V_1)$ are isomorphic, which we also refer to as the two halves of G.

We can state our sparse lifting theorem using the concept of mirror graphs.

Theorem 12. *If F has resolution depth D, and G is a mirror graph with $G_0 \sim \mathcal{G}(n, D/2, d)$, where $d = \Theta(\log(n/D))$, then with high probability $\mathcal{L}^G(F)$ has regular resolution length $\exp(\Omega(D^3/d^2n^2))$.*

As before, if we choose for F the pebbling formula of a graph of pebbling number $\Theta(n/\log n)$, then we get the following improved separation of regular and general resolution.

Corollary 13. *There are formulas that have general resolution refutations of length $\mathrm{O}(n \log\log^3 n)$ but require regular resolution length $\exp(\Omega(n/\log^3 n \log\log^2 n))$.*

Let us establish some notation. After fixing an isomorphism $\Psi\colon G_0 \to G_1$ we name the vertices in pairs $j0$ and $j1$ so that $j1 = \Psi(j0)$. If $jb \in V_b$ and $a \in \{0,1\}$, we let $jb + a$ denote the vertex $j(a + b \mod 2) \in V_{a+b \mod 2}$. Let $m = |V_0|$ so there are $2m$ right vertices in G. In this section c-complex stands for $(c,1)$-complex and we assume that $d = \Theta(\log(n/m))$.

Lemma 14. *If G is a mirror (r,κ)-expander with $\kappa > 2$, where $\kappa r = \Theta(m)$, and π is a resolution proof of $\mathcal{L}^G(F)$ of size $s = \exp(\mathrm{O}(c^2m/d^2n^2))$, where $c = \Theta(m)$, then there is a restriction ρ such that $\pi{\restriction_\rho}$ is a proof of $\mathcal{L}^{G'}(F')$ that has no c-complex clauses, where F' has resolution depth at least $D - r/2 - \kappa r/8$ and G' is an $(r/2, \kappa/2)$-expander.*

Proof. We show that such a restriction exists using a hybrid between a random and a greedy restriction. We randomly partition the stone vertices in V_0 into free, assigned, and matched stones, and mirror the partition in V_1. Of the assigned stones, a set A_0^- of $\kappa r/16$ stones are set to 0, and a set A_0^+ of $\kappa r/16$ stones are set to 1, while the stones in the corresponding sets $A_1^- = \psi(A_0^-)$ and $A_1^+ = \psi(A_0^+)$ are set to 1 and 0 respectively. We plan to use the sets M_0 and $M_1 = \psi(M_0)$ of $\kappa r/8$

matched stones each to greedily build a matching. The remaining $2(m - \kappa r/4)$ stone vertices are tentatively left untouched.

First we claim that, with high probability, all clauses of type 1 are satisfied. To show this we note that a clause C of type 1 contains at least $c/4$ literals of the same polarity and referring to the same half of the graph. Assume without loss of generality that C contains $c/4$ positive literals referring to stones in V_0 and let $C_0^+ = \{j0 \in V_0 : r_{j0} \in C\}$ be these stones.

The probability that no positive stone literal is satisfied is

$$\Pr[C_0^+ \cap A_0^+ = \emptyset] \leq \frac{\binom{|V_0 \setminus C_0^+|}{|A_0^+|}}{\binom{|V_0|}{|A_0^+|}} \leq (1 - c/4m)^{\kappa r/16} = \exp(-\Omega(\kappa r)) = \exp(-\Omega(m))$$

and since $\ln s = \mathrm{O}\big(c^2 m/d^2 n^2\big) = \mathrm{O}\big(m(c^2/d^2 n^2)\big) = \mathrm{o}(m)$ the claim follows from a union bound over all clauses of type 1.

Next we greedily build a matching μ with the goal of satisfying all clauses of types 2 and 3. We ensure that overlaying both halves of the matching would also result in a matching; in other words if a vertex jb is matched then we ensure that $jb + 1$ is not. For each edge (i, jb) in the matching we set $s_{i,jb} = 1$, we set $s_{i',jb} = 0$ and $s_{i,j'b'} = 0$ for all $i' \neq i$, $j' \neq j$, and $b' \in \{0, 1\}$, and we leave $s_{i,jb+1}$ tentatively unset for all i. Before we actually build the matching we need to prove that, with high probability, each of these clauses can be satisfied by choosing one of $c\kappa r/32m$ edges (i, jb) with $j \in M_b$ to be in the matching.

For a clause C of type 3 we assume without loss of generality that $c/2$ literals refer to stones in V_0. We can express the number of edges that satisfy C as the random variable $x_C = \sum_{j0 \in V_0} x_{C,j0}$ where $x_{C,j0}$ takes the value $t_{C,j0} = |\{(i, j0) \in E : s_{i,j0} \in C\}|$ if $j0 \in M_0$ and 0 otherwise. We have that

$$\mathrm{E}_C = \mathrm{E}[x_C] = \sum_{j0 \in V_0} \mathrm{E}[x_{C,j0}] = \sum_{j0 \in V_0} t_{C,j0} \cdot \Pr[j0 \in M_0] =$$
$$= \frac{|M_0|}{m} \sum_{j0 \in V_0} t_{C,j0} \geq \frac{\kappa r}{8m} \cdot \frac{c}{2} = \frac{c\kappa r}{16m} = \Theta(c)$$

and each of $x_{C,j}$ is bounded by the right degree $d' \leq 2dn/m$, therefore by Hoeffding's inequality for sampling without replacement we obtain that

$$\Pr[x_C < \mathrm{E}_C/2] \leq \exp\left(-2\frac{(\mathrm{E}_C - \mathrm{E}_C/2)^2}{\sum_{j0 \in V_0} t_{C,j0}^2}\right) = \exp(-\Omega(c^2/d'c)) = \exp(-\Omega(cm/dn))$$

and the claim follows from a union bound over all clauses of type 3.

For clauses of type 2, for each literal $\overline{s_{i,j0}} \in C$ it is enough to choose as an edge one of the $(d - 1)$ edges $(i, j'0)$ with $j' \neq j$. Hence the number of available choices is the random variable x_C defined as before except that $t_{C,j0} = |\{(i, j0) \in E_0 : \exists j' \in V_0 \setminus \{j\}\ ,\ \overline{s}_{i,j'} \in C\}|$. We have $\mathrm{E}_C = \mathrm{E}[x_C] \geq (d-1)c\kappa r/16m$ therefore $\Pr[x_C < \mathrm{E}_C] = \exp(-\Omega(cm/n))$ and the claim follows from a union bound.

Let us finish this step of the proof by building the matching. Observe that choosing an edge makes up to $d + d'$ incident edges ineligible, as well as up to

$d + d'$ edges in the other half, for a total of $2(d + d') \le 5d'$, hence after making ℓ choices there are still $e(\ell) = c\kappa r/32 - 5\ell d'$ choices available for each clause. By averaging, there is an edge that satisfies at least an $e(\ell)/dn$ fraction of the clauses of types 2 and 3. Hence after picking

$$k = e^{-1}(c\kappa r/64m) = \frac{c\kappa r/64m}{5d'} \le \frac{c\kappa r}{320dn}$$

edges the remaining fraction of clauses is at most

$$\prod_{\ell=1}^{k}\left(1 - \frac{e(\ell)}{dn}\right) \le \left(1 - \frac{e(k)}{dn}\right)^k = \left(1 - \frac{c\kappa r}{64mdn}\right)^{\frac{c\kappa r}{320dn}} = \exp\left(-\Omega\left(\frac{c^2 m}{d^2 n^2}\right)\right) .$$

The last step is to ensure that after removing $V_0' = A_0 \cup M_0$ from G_0 we still have a good expander. By Lemma 4 there is a set U' of size $r/2$ such that $G_0 \setminus U' \cup N(U') \cup V_0'$ is an $(r/2, \kappa/2)$-expander. Let $U'' = U' \setminus \operatorname{dom}\mu$. Let $\nu\colon U'' \to V_0$ be an injective mapping from indices to stones, which exists by Hall's theorem, and let $\sigma\colon U'' \to \{0,1\}$ be an assignment to U'' such that the depth of $F{\restriction_\sigma}$ reduces by at most $|\sigma|$.

We match each vertex $i \in U''$ to a stone as follows. If $\nu(i) \in A_0^-$ then $r_{\nu(i)+\sigma(i)} = \sigma(i)$ so we set $s_{i,\nu(i)+\sigma(i)} = 1$, while if $\nu(i) \in A_0^+$ then $r_{\nu(i)+\sigma(i)} = 1 - \sigma(i)$ so we set $s_{i,\nu(i)+\sigma(i)+1} = 1$. If $\nu(i) \in M_0$ then note that by construction of the matching μ at least one of $\nu(i)$ and $\nu(i)+1$ is not matched; we let jb be that stone and set $s_{i,jb} = 1$ and $r_{jb} = \sigma(i)$. Otherwise we add $\nu(i)$ to M_0 and $\nu(i)+1$ to M_1, and do as in the previous case.

We also assign values to matched stones. Let $\operatorname{dom}\mu$ be the matched original variables and let $\tau\colon \operatorname{dom}\mu \to \{0,1\}$ be an assignment to $\operatorname{dom}\mu$ such that the depth of $F{\restriction_{\sigma\cup\tau}}$ reduces by at most $|\tau|$. For each vertex $i \in \operatorname{dom}\mu$ we set $r_{\mu(i)} = \tau(i)$. To obtain our final graph we set to 0 any variable $s_{i,j}$ with $i \in U' \cup \operatorname{dom}\mu$ or $j \in V_0' \cup N(U') \cup V_1$ that remains unassigned.

Let us recap and show that $\mathcal{L}^G(F){\restriction_\rho} = \mathcal{L}^{G'}(F')$ where G' is an expander and F' has large depth as we claimed. G' is the subgraph of G induced by $U \setminus (U' \cup \operatorname{dom}\mu)$ and $V_0 \setminus (V_0' \cup N(U'))$, since we did not assign any selector variable corresponding to an edge between these two sets, but we did assign every other selector variable. The graph induced by $U \setminus U'$ and $V_0 \setminus (V_0' \cup N(U'))$ is an $(r/2, \kappa/2)$-expander by Lemma 4, and since removing left vertices does not affect expansion, so is G'. Regarding F', for every variable $s_{i,j} = 1$ we have that $r_j = (\sigma \cup \tau)(i)$, so $F' = F{\restriction_{\sigma\cup\tau}}$ which has depth at least $D - r/2 - \kappa r/8$. □

To prove an equivalent of Lemma 11 we use the *extended matching game*, where we allow the following additional move:

- Prover places an unused finger i on a free vertex $v \in V$, in which case Disprover places his i-th finger on v and optionally moves Prover's finger to a free vertex $u \in N(v)$.

Lemma 15. *If Prover needs p fingers to win the matching game on a graph of right degree d', then it needs $p - d'$ fingers to win the extended matching game.*

The proof can be found in the forthcoming full version.

Finally we are ready to prove our last lemma and complete the proof.

Lemma 16. *If F has resolution depth D, and G is a bipartite graph whose right hand side is of size $m < D$, duch that G requires r fingers in the extended matching game, then any regular resolution refutation of $\mathcal{L}^G(F)$ has an $r/3$-complex clause.*

Proof. At a high level we proceed as in the proof of Lemma 11, except that now keeping a matching is a more delicate task, and hence we use the extended matching game for it. We want to match any index i for which we have information about, this is the value of a variable $s_{i,j}$ is remembered.

- If the adversary queries r_j and $\mu(i) = j$ for some i, then we answer so that the depth of the decision tree only shrinks by 1.
- If the adversary queries r_j where j is not in the matching, then we play j in the matching game. If we receive an answer i we add (i, j) to the matching and answer so that the depth of the decision tree only shrinks by 1. If instead we receive the answer j, we answer arbitrarily.
- If the adversary queries $s_{i,j}$ where either i or j are in the matching then we answer 1 if (i, j) is in the matching and 0 otherwise.
- If the adversary queries $s_{i,j}$ where neither i nor j are in the matching then we play i in the matching game and receive an answer j'. We add (i, j') to the matching and answer 1 if $j = j'$ and 0 otherwise.
- If after the adversary forgets a variable there is an index i such that $\mu(i) = j$ but none of $s_{i,j'}$ and r_j are assigned, we forget i in the matching game.

Assume for the sake of contradiction that Prover does not win the matching game. It follows that the branching program ends at a clause in $\mathcal{L}(D)$ for $D \in F$, at which point the depth of T reduced to 0. Observe that the depth of T only decreases by 1 when a stone variable is queried and that, since the branching program is read-once, these queries must be to D different stones. However, only $m < D$ stones are available.

It follows that Prover eventually uses r fingers in the matching game, at which point we claim that we are at an $r/3$-complex clause. Let us see why. For each finger i in the matching game we remember either a selector literal $s_{i,j} = 1$, a selector literal $s_{i,j} = 0$, or a stone variable r_j, hence we remember at least $r/3$ variables of either type. In the first case we are at a clause of type 2, in the second at a clause of type 3, and in the third at a clause of type 1. □

Proof (of Theorem 12). By Lemma 3, with high probability G_0 is an (r, κ)-expander for $r = \Theta(m/d)$ and $\kappa = \Theta(d)$, and has right degree at most $2dn/m$. Assume for the sake of contradiction that π is a refutation of $\mathcal{L}^G(F)$ of length less than $\exp(\epsilon D^3/d^2n^2)$.

Let ρ be the restriction given by Lemma 14 so that $\pi\restriction_\rho$ is a regular resolution proof with no c-complex clauses with $c = \kappa r/75 = \Theta(m)$. The formula $\mathcal{L}(F)\restriction_\rho$ is the lifted version $\mathcal{L}^{G'}(F')$ of a formula F' of depth at least $D - r/2 - \kappa r/8$, and the graph G' is an $(r/2, \kappa/2)$-expander with $m' \leq m - \kappa r/8 \leq D - r/2 - \kappa r/8$.

Since for each set U of size at most $\kappa r/8$ and subset $U' \subseteq U$ of size $|U| \cdot 4/\kappa \leq r/2$ it holds that $|N(U)| \geq |N(U')| \geq \kappa/2|U'| = 2|U|$, G' is also a $(\kappa r/8, 2)$-expander, hence by Theorem 5 and Lemma 15 G' requires $\kappa r/24 - d' \geq \kappa r/25$ fingers in the extended matching game. By Lemma 11 any regular resolution proof of $\mathcal{L}^{G'}(F')$ has a $\kappa r/75$-complex clause. But this contradicts that the proof $\pi\restriction_\rho$ has no $\kappa r/75$-complex clauses. □

It would also be interesting to prove a lower bound with plain random graphs, not relying on the additional mirror structure. Unfortunately, without backup vertices, the expansion restoring step would make $r/2$ right vertices ineligible to be matched, and that can prevent us from satisfying clauses of type 3 of complexity up to $d'r/2 \gg m$.

5 Experiments

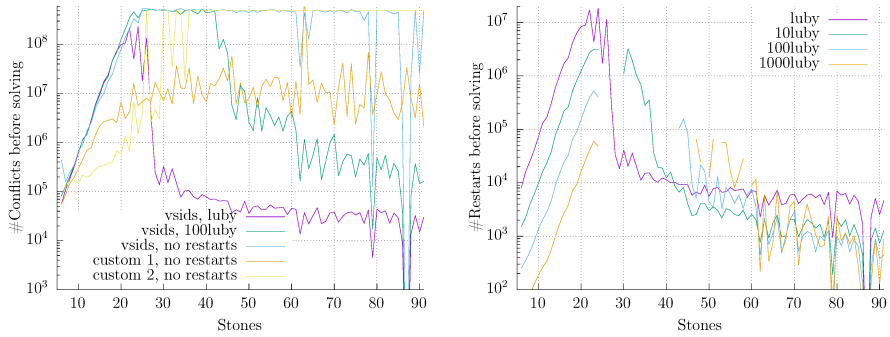
We have run some experiments to investigate how hard sparse stone formulas are in practice and how restarts influence solvers running on this particular family.

As base formulas we use pebbling formulas over Gilbert–Tarjan graphs with butterflies [30,42], which require depth $\Theta(n/\log^2 n)$, and over pyramid graphs, which require depth $\Theta(\sqrt{n})$. Note that lifting the first type of formulas yields benchmarks that are provably hard for regular resolution, whereas for the second type of formulas we are not able to give any theoretical guarantees. Our experimental results are very similar, however, and so below we only discuss formulas obtained from pyramids, for which more benchmarks can be generated.

We used an instrumented version [28] of the solver Glucose [4] to make it possible to experiment with different heuristics. The results reported here are for the settings that worked best, namely VSIDS decision heuristic and preprocessing switched on. To vary the restart frequency we used Luby restarts with factors 1, 10, 100, and 1000 plus a setting with no restarts. The time-out limit was 24 hours. For the record, we also ran some preliminary experiments for standard Glucose (with adaptive restarts) and Lingeling [46], but since the results were similar to those for Luby restarts with a factor 100 we did not run full-scale experiments with these configurations.

We illustrate our findings in Figure 1 by plotting results from experiments using the pebbling formula over a pyramid graph of height 12 as the base formula and varying the number of stones. We used random graphs of left degree 6 as selector variable graphs. Note that once the pebbling DAG for the base formula has been fixed, changing the number of stones does not change the size of the formula too much. For the particular pebbling DAG in Figure 1, the number of variables is in the interval from 550 to 650.

Empirically, the formulas are hardest when the number of stones is close to the proof depth for the base formula, which is also the scenario where the calculations in Section 4 yield the strongest bound. We expect the hardness to increase as the number of stones approaches from below the proof depth of the base formula, but as the number of stones grow further the formulas should get



(a) # Conflicts (b) # Restarts

Fig. 1: Solving stone formulas over a pyramid of height 12.

easier again. This is so since the fact that the selector graph left degree is kept constant means that the overlap decreases and ultimately vanishes, and pebbling formulas lifted without overlap are easy for regular resolution.

Interestingly, the solver behaviour is very different on either side of this hardness peak. As we can see on the left in Figure 1a, in the beginning the number of conflicts (and hence the running time) grows exponentially in the number of stones, independently of the number of restarts. With more stones, however, restarts become critical. The number of restarts used to solve a particular instance remains similar among all solver configurations, as shown on the right in Figure 1b. Therefore, if the solver restarts more frequently it reaches this number of restarts faster and solves the formula faster, as shown by the conflict counts on the right in Figure 1a.

To make CDCL solvers run as fast as possible, we crafted a custom decision order tailored to stone formulas over pyramids. With this decision order, no restarts, and very limited clause erasures, the solver decided dense stone formulas over pyramids of height h with h stones in a number of conflicts proportional to $h^{7.28}$ (where we note that these formulas have $\mathrm{O}(h^3)$ variables and $\mathrm{O}(h^5)$ clauses). For sparse stone formulas, we found one decision order (custom 1 in Figure 1a) that worked reasonably well for small pyramids but failed for larger ones. A second attempt (custom 2) performed well for all pyramid sizes as long as the number of stones was below the hardness peak, but failed for more stones (when the formulas become easy for VSIDS with frequent restarts).

Summing up, even though stone formulas always possess short resolution refutations, and even though CDCL solvers can sometimes be guided to decide the formulas quickly even without restarts, these formulas can be surprisingly hard in practice for state-of-the-art solvers with default heuristics. The frequency of restarts seems to play a crucial role—which is an interesting empirical parallel of the theoretical results in [3,51]—but for some settings of stone formula parameters even frequent restarts cannot help the solver to perform well.

6 Concluding Remarks

In this work we employ lifting, a technique that has led to numerous breakthroughs in computational complexity theory in the last few years, to give a significantly simplified proof of the result in [2] that general resolution is exponentially more powerful than regular resolution. We obtain this separation as a corollary of a generic lifting theorem amplifying lower bounds on proof depth to lower bounds on regular proof length in resolution. Thanks to this new perspective we are also able to extend the result further, so that we obtain smaller benchmark formulas that slightly strengthen the parameters of the previously strongest separation between regular and general resolution in [61].

Furthermore, these new formulas are also small enough to make it possible to run experiments with CDCL solvers to see how the running time scales as the formula size grows. Our results show that although these formulas are theoretically very easy, and have resolution proofs that seem possible to find for CDCL solvers without restarts if they are given guidance about which variable decisions to make, in practice the performance depends heavily on settings such as frequent restarts, and is sometimes very poor even for very frequent restarts.

Our main result implies that if we can find CNF formulas that have resolution proofs in small width but require sufficiently large depth, then lifted versions of such formulas separate regular and general resolution. (This is so since proof width can only increase by a constant factor after lifting, and small-width proofs have to be short in general resolution by a simple counting argument.) Unfortunately, the only such formulas that are currently known are pebbling formulas. It would be very interesting to find other formulas with the same property.

Also, it would be desirable to improve the parameters of our lifting theorem. A popular family of pebbling graphs are pyramids, but the proof depth for pebbling formulas based on such graphs is right below the threshold where the lower bound amplification kicks in. Could the analysis in the proof of the lifting theorem be tightened to work also for, e.g., pebbling formulas over pyramids?

On the applied side, it is intriguing that sparse stone formulas can be so hard in practice. One natural question is whether one could find some tailor-made decision heuristic that always makes CDCL solvers run fast on such formulas, with or even without restarts. An even more relevant question is whether some improvement in standard CDCL heuristics could make state-of-the-art solvers run fast on these formulas (while maintaining performance on other formulas).

Acknowledgements We are most grateful to Robert Robere for the interesting discussions that served as the starting point for this project. We also acknowledge the important role played by the Dagstuhl seminar 18051 "Proof Complexity," where some of this work was performed. Our computational experiments were run on resources provided by the Swedish National Infrastructure for Computing (SNIC). Our benchmarks were generated using the tool `CNFgen` [44].

The first author was supported by the Prof. R Narasimhan post-doctoral award. The second and fourth authors were funded by the Swedish Research Council (VR) grant 2016-00782. The fourth author was also supported by the the Independent Research Fund Denmark (DFF) grant 9040-00389B.

References

1. Alekhnovich, M., Hirsch, E.A., Itsykson, D.: Exponential lower bounds for the running time of DPLL algorithms on satisfiable formulas. Journal of Automated Reasoning **35**(1–3), 51–72 (Oct 2005), preliminary version in *ICALP '04*
2. Alekhnovich, M., Johannsen, J., Pitassi, T., Urquhart, A.: An exponential separation between regular and general resolution. Theory of Computing **3**(5), 81–102 (May 2007), preliminary version in *STOC '02*
3. Atserias, A., Fichte, J.K., Thurley, M.: Clause-learning algorithms with many restarts and bounded-width resolution. Journal of Artificial Intelligence Research **40**, 353–373 (Jan 2011), preliminary version in *SAT '09*
4. Audemard, G., Simon, L.: Predicting learnt clauses quality in modern SAT solvers. In: Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI '09). pp. 399–404 (Jul 2009)
5. Bayardo Jr., R.J., Schrag, R.: Using CSP look-back techniques to solve real-world SAT instances. In: Proceedings of the 14th National Conference on Artificial Intelligence (AAAI '97). pp. 203–208 (Jul 1997)
6. Beame, P., Beck, C., Impagliazzo, R.: Time-space tradeoffs in resolution: Superpolynomial lower bounds for superlinear space. SIAM Journal on Computing **45**(4), 1612–1645 (Aug 2016), preliminary version in *STOC '12*
7. Beame, P., Huynh, T., Pitassi, T.: Hardness amplification in proof complexity. In: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC '10). pp. 87–96 (Jun 2010)
8. Beame, P., Pitassi, T., Segerlind, N.: Lower bounds for Lovász–Schrijver systems and beyond follow from multiparty communication complexity. SIAM Journal on Computing **37**(3), 845–869 (2007), preliminary version in *ICALP '05*
9. Beck, C., Nordström, J., Tang, B.: Some trade-off results for polynomial calculus. In: Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13). pp. 813–822 (May 2013)
10. Ben-Sasson, E., Galesi, N.: Space complexity of random formulae in resolution. Random Structures and Algorithms **23**(1), 92–109 (Aug 2003), preliminary version in *CCC '01*
11. Ben-Sasson, E., Impagliazzo, R., Wigderson, A.: Near optimal separation of tree-like and general resolution. Combinatorica **24**(4), 585–603 (Sep 2004)
12. Ben-Sasson, E., Nordström, J.: Short proofs may be spacious: An optimal separation of space and length in resolution. In: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08). pp. 709–718 (Oct 2008)
13. Ben-Sasson, E., Nordström, J.: Understanding space in proof complexity: Separations and trade-offs via substitutions. In: Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11). pp. 401–416 (Jan 2011)
14. Ben-Sasson, E., Wigderson, A.: Short proofs are narrow—resolution made simple. Journal of the ACM **48**(2), 149–169 (Mar 2001), preliminary version in *STOC '99*
15. Berkholz, C., Nordström, J.: Near-optimal lower bounds on quantifier depth and Weisfeiler-Leman refinement steps. In: Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS '16). pp. 267–276 (Jul 2016)
16. Berkholz, C., Nordström, J.: Supercritical space-width trade-offs for resolution. SIAM Journal on Computing **49**(1), 98–118 (Feb 2020), preliminary version in *ICALP '16*

17. Blake, A.: Canonical Expressions in Boolean Algebra. Ph.D. thesis, University of Chicago (1937)
18. Bonet, M.L., Buss, S., Johannsen, J.: Improved separations of regular resolution from clause learning proof systems. Journal of Artificial Intelligence Research **49**, 669–703 (Apr 2014)
19. Bonet, M.L., Esteban, J.L., Galesi, N., Johannsen, J.: On the relative complexity of resolution refinements and cutting planes proof systems. SIAM Journal on Computing **30**(5), 1462–1484 (2000), preliminary version in *FOCS '98*
20. Buss, S.R., Hoffmann, J., Johannsen, J.: Resolution trees with lemmas: Resolution refinements that characterize DLL-algorithms with clause learning. Logical Methods in Computer Science **4**(4:13) (Dec 2008)
21. Buss, S.R., Kołodziejczyk, L.: Small stone in pool. Logical Methods in Computer Science **10**(2), 16:1–16:22 (Jun 2014)
22. Chan, S.M.: Just a pebble game. In: Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13). pp. 133–143 (Jun 2013)
23. Chan, S.M., Lauria, M., Nordström, J., Vinyals, M.: Hardness of approximation in PSPACE and separation results for pebble games (Extended abstract). In: Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15). pp. 466–485 (Oct 2015)
24. Chattopadhyay, A., Koucky, M., Loff, B., Mukhopadhyay, S.: Simulation beats richness: New data-structure lower bounds. In: Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18). pp. 1013–1020 (Jun 2018)
25. Cook, S.A.: An observation on time-storage trade off. Journal of Computer and System Sciences **9**(3), 308–316 (1974), preliminary version in *STOC '73*
26. Davis, M., Logemann, G., Loveland, D.: A machine program for theorem proving. Communications of the ACM **5**(7), 394–397 (Jul 1962)
27. Davis, M., Putnam, H.: A computing procedure for quantification theory. Journal of the ACM **7**(3), 201–215 (1960)
28. Elffers, J., Giráldez-Cru, J., Gocht, S., Nordström, J., Simon, L.: Seeking practical CDCL insights from theoretical SAT benchmarks. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI '18). pp. 1300–1308 (Jul 2018)
29. Garg, A., Göös, M., Kamath, P., Sokolov, D.: Monotone circuit lower bounds from resolution. In: Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18). pp. 902–911 (Jun 2018)
30. Gilbert, J.R., Tarjan, R.E.: Variations of a pebble game on graphs. Technical Report STAN-CS-78-661, Stanford University (1978), available at http://infolab.stanford.edu/TR/CS-TR-78-661.html
31. Goerdt, A.: Regular resolution versus unrestricted resolution. SIAM Journal on Computing **22**(4), 661–683 (Aug 1993)
32. Göös, M., Jain, R., Watson, T.: Extension complexity of independent set polytopes. SIAM Journal on Computing **47**(1), 241–269 (Feb 2018)
33. Göös, M., Jayram, T.S., Pitassi, T., Watson, T.: Randomized communication vs. partition number. In: Proceedings of the 44th International Colloquium on Automata, Languages and Programming (ICALP '17). Leibniz International Proceedings in Informatics (LIPIcs), vol. 80, pp. 52:1–52:15 (Jul 2017)
34. Göös, M., Kamath, P., Pitassi, T., Watson, T.: Query-to-communication lifting for P^NP. In: Proceedings of the 32nd Annual Computational Complexity Conference (CCC '17). Leibniz International Proceedings in Informatics (LIPIcs), vol. 79, pp. 12:1–12:16 (Jul 2017)

35. Göös, M., Lovett, S., Meka, R., Watson, T., Zuckerman, D.: Rectangles are nonnegative juntas. In: Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15). pp. 257–266 (Jun 2015)
36. Göös, M., Pitassi, T.: Communication lower bounds via critical block sensitivity. SIAM Journal on Computing **47**(5), 1778–1806 (Oct 2018), preliminary version in *STOC '14*
37. Göös, M., Pitassi, T., Watson, T.: Deterministic communication vs. partition number. In: Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15). pp. 1077–1088 (Oct 2015)
38. Göös, M., Pitassi, T., Watson, T.: The landscape of communication complexity classes. Computational Complexity **27**(2), 245–304 (Jun 2018), preliminary version in *ICALP '16*
39. Hoory, S., Linial, N., Wigderson, A.: Expander graphs and their applications. Bulletin of the American Mathematical Society **43**(4), 439–561 (Oct 2006)
40. Huang, W., Yu, X.: A DNF without regular shortest consensus path. SIAM Journal on Computing **16**(5), 836–840 (Oct 1987)
41. Huynh, T., Nordström, J.: On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity (Extended abstract). In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12). pp. 233–248 (May 2012)
42. Järvisalo, M., Matsliah, A., Nordström, J., Živný, S.: Relating proof complexity measures and practical hardness of SAT. In: Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming (CP '12). Lecture Notes in Computer Science, vol. 7514, pp. 316–331. Springer (Oct 2012)
43. Kothari, P.K., Meka, R., Raghavendra, P.: Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs. In: Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC '17). pp. 590–603 (Jun 2017)
44. Lauria, M., Elffers, J., Nordström, J., Vinyals, M.: CNFgen: A generator of crafted benchmarks. In: Proceedings of the 20th International Conference on Theory and Applications of Satisfiability Testing (SAT '17). Lecture Notes in Computer Science, vol. 10491, pp. 464–473. Springer (Aug 2017)
45. Lee, J.R., Raghavendra, P., Steurer, D.: Lower bounds on the size of semidefinite programming relaxations. In: Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15). pp. 567–576 (Jun 2015)
46. Lingeling, Plingeling and Treengeling. http://fmv.jku.at/lingeling/
47. Marques-Silva, J.P., Sakallah, K.A.: GRASP: A search algorithm for propositional satisfiability. IEEE Transactions on Computers **48**(5), 506–521 (May 1999), preliminary version in *ICCAD '96*
48. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Proceedings of the 38th Design Automation Conference (DAC '01). pp. 530–535 (Jun 2001)
49. Nordström, J.: Narrow proofs may be spacious: Separating space and width in resolution. SIAM Journal on Computing **39**(1), 59–121 (May 2009), preliminary version in *STOC '06*
50. Nordström, J., Håstad, J.: Towards an optimal separation of space and length in resolution. Theory of Computing **9**, 471–557 (May 2013), preliminary version in *STOC '08*
51. Pipatsrisawat, K., Darwiche, A.: On the power of clause-learning SAT solvers as resolution engines. Artificial Intelligence **175**(2), 512–525 (Feb 2011), preliminary version in *CP '09*

52. Pitassi, T., Robere, R.: Strongly exponential lower bounds for monotone computation. In: Proceedings of the 49th Annual ACM Symposium on Theory of Computing (STOC '17). pp. 1246–1255 (Jun 2017)
53. Pitassi, T., Robere, R.: Lifting Nullstellensatz to monotone span programs over any field. In: Proceedings of the 50th Annual ACM Symposium on Theory of Computing (STOC '18). pp. 1207–1219 (Jun 2018)
54. Raz, R., McKenzie, P.: Separation of the monotone NC hierarchy. Combinatorica **19**(3), 403–435 (Mar 1999), preliminary version in *FOCS '97*
55. Razborov, A.A.: A new kind of tradeoffs in propositional proof complexity. Journal of the ACM **63**(2), 16:1–16:14 (Apr 2016)
56. Razborov, A.A.: On space and depth in resolution. Computational Complexity **27**(3), 511–559 (Sep 2018)
57. de Rezende, S.F., Nordström, J., Vinyals, M.: How limited interaction hinders real communication (and what it means for proof and circuit complexity). In: Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16). pp. 295–304 (Oct 2016)
58. Robere, R., Pitassi, T., Rossman, B., Cook, S.A.: Exponential lower bounds for monotone span programs. In: Proceedings of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS '16). pp. 406–415 (Oct 2016)
59. Tseitin, G.: On the complexity of derivation in propositional calculus. In: Silenko, A.O. (ed.) Structures in Constructive Mathematics and Mathematical Logic, Part II, pp. 115–125. Consultants Bureau, New York-London (1968)
60. Urquhart, A.: The depth of resolution proofs. Studia Logica **99**(1-3), 349–364 (2011)
61. Urquhart, A.: A near-optimal separation of regular and general resolution. SIAM Journal on Computing **40**(1), 107–121 (2011), preliminary version in *SAT '08*
62. Van Gelder, A.: Pool resolution and its relation to regular resolution and DPLL with clause learning. In: Proceedings of the 12th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR '05). Lecture Notes in Computer Science, vol. 3835, pp. 580–594. Springer (2005)