How Limited Interaction Hinders Real Communication (and What It Means for Proof and Circuit Complexity)

Susanna F. de Rezende, Jakob Nordström and Marc Vinyals
School of Computer Science and Communication
KTH Royal Institute of Technology
Stockholm, Sweden
$\{$*sfdr,jakobn,vinyals*$\}$*@kth.se*

***Abstract*—We obtain the first true size-space trade-offs for the cutting planes proof system, where the upper bounds hold for size and total space for derivations with constant-size coefficients, and the lower bounds apply to length and formula space (i.e., number of inequalities in memory) even for derivations with exponentially large coefficients. These are also the first trade-offs to hold uniformly for resolution, polynomial calculus and cutting planes, thus capturing the main methods of reasoning used in current state-of-the-art SAT solvers.**

We prove our results by a reduction to communication lower bounds in a round-efficient version of the real communication model of [Krajíček '98], drawing on and extending techniques in [Raz and McKenzie '99] and [Göös et al. '15]. The communication lower bounds are in turn established by a reduction to trade-offs between cost and number of rounds in the game of [Dymond and Tompa '85] played on directed acyclic graphs.

As a by-product of the techniques developed to show these proof complexity trade-off results, we also obtain an exponential separation between monotone-AC^{i-1} and monotone-AC^{i}, improving exponentially over the superpolynomial separation in [Raz and McKenzie '99]. That is, we give an explicit Boolean function that can be computed by monotone Boolean circuits of depth $\log^i n$ and polynomial size, but for which circuits of depth $\mathrm{O}(\log^{i-1} n)$ require exponential size.

***Keywords*-proof complexity; communication complexity; circuit complexity; cutting planes; trade-offs; pebble games**

I. Introduction

Ever since the discovery of NP-completeness by Cook and Levin in [18], [39], the problem of how hard it is to decide satisfiability of formulas in propositional logic has played a leading role in theoretical computer science. Although the conventional wisdom is that SAT should be a very hard problem—to the extent that the *Exponential Time Hypothesis* [32] concerning its worst-case complexity is a standard assumption used in many other hardness results—essentially no non-trivial lower bounds on the time complexity of the SAT problem are known.

A less ambitious goal is to ask for lower bounds if not only the running time but also the memory usage of the algorithm is restricted. Yet it took until [25] to rule out a linear-time, logarithmic-space algorithm for SAT. Later research has shown that refuting unsatisfiable formulas on random-access machines cannot be done non-deterministically in simultaneous time $n^{4^{1/3}}$ and space $n^{o(1)}$ [22] and SAT cannot be decided deterministically in simultaneous time $n^{1.8}$ and space $n^{o(1)}$ [50]. On Turing machines, no non-deterministic algorithm solving SAT in time T and space s can achieve $T \cdot s = n^2/\log^3 n$ [47]. (See [49] for a good survey of the area with more details on this kind of results.)

For a problem that is believed to require exponential time, the results listed above might not seem very impressive. Yet they should not necessarily be viewed only as an illustration of the weaknesses of current techniques for proving lower bounds. It is important to realize that the adversary is formidable—applied research in the last 15–20 years has led to the development of amazingly efficient algorithms, so-called *SAT solvers*, that solve many real-world instances with millions of variables, and do so in linear time. Today, practitioners often think of SAT as an *easy problem to reduce to*, rather than a hard problem to reduce from (we refer the reader to [11] for more on this fascinating topic).

Virtually the only tool currently available for a rigorous analysis of the performance of such SAT solvers is *proof complexity* [19], where one studies the methods of reasoning used by the corresponding algorithms. The transcript of the computations made can be viewed as a formal proof applying the relevant method of reasoning, and proof complexity analyses the resources needed when all computational choices are made optimally (i.e., non-deterministically). Even though this is quite a challenging adversarial setting, proof complexity has nevertheless managed to give tight exponential lower bounds on the worst-case running time for many approaches for SAT used in practice by lower-bounding proof size.

The focus of this paper is on time-space trade-offs in computational models describing current state-of-the-art SAT solvers. This research is partly driven by SAT solver running time and memory usage—in practice, space consumption can be almost as much of a bottleneck as running time—but is also motivated by the fundamental importance of time and space complexity in computational complexity.

A. Previous Work on Proof Complexity Trade-offs

In *resolution* [12], which is arguably the most well-studied proof system in proof complexity, the input is an unsatisfiable formula in conjunctive normal form (CNF) and new disjunctive clauses are derived from this formula until an explicit contradiction is reached (in the form of the empty clause without literals). Resolution is also the method of reasoning underlying the currently most successful SAT solving paradigm based on so-called *conflict-driven clause learning (CDCL)* [3], [40], [41]. The question of time-space trade-offs for resolution was first raised by Ben-Sasson [7], who also obtained such trade-offs for the restricted subsystem of tree-like resolution. Size-space trade-offs for general, unrestricted resolution were later shown in [4], [6], [9], [43].

In contrast to the trade-off results for random-access and Turing machines reviewed above, in these more limited models of computation one can obtain exponential lower bounds on proof size (corresponding to running time) for proofs in sublinear but polynomial space [9], [43], and results in [4], [6] even exhibit trade-offs where size has to be superpolynomial and space has to be superlinear simultaneously. Another difference is that these results are *true trade-offs* in the sense that it is actually possible to refute the formulas both in small size and small space, only not simultaneously. A third nice feature of the trade-offs are that the upper bounds are on proof size and total space, whereas the (sometimes tightly matching) lower bounds are on *length* and *formula space*, meaning that one only charges one time unit for each derivation step regardless of its complexity, and only one space unit per "formula" (for resolution: per clause) regardless of how large it is. Thus, the upper bounds are algorithmically achievable, while the lower bounds hold in a significantly stronger model.

A stronger proof system than resolution is *polynomial calculus* [1], [17], where the clauses of a formula are translated to multilinear polynomials and calculations inside the ideal generated by these polynomials (basically corresponding to a Gröbner basis computation) establishes unsatisfiability. Among other things, polynomial calculus captures CDCL solvers extended with reasoning about systems of linear equations $\bmod 2$. The first size-space trade-offs for polynomial calculus—which were not true trade-offs in the sense discussed above, however—were obtained in [31], and these results were further improved in [6] to true trade-offs essentially matching the results cited above for resolution except for a small loss in parameters.

Another proof system that is also stronger than resolution and that has been the focus of much research is *cutting planes* [21], which formalizes the integer linear programming algorithm in [16], [27] and underlies so-called *pseudo-Boolean* SAT solvers. In cutting planes the clauses of a CNF formula are translated to linear inequalities, which are then manipulated to derive a contradiction. Thus, the question of Boolean satisfiability is reduced to the geometry of polytopes over the real numbers. Cutting planes is much more poorly understood than resolution and polynomial calculus, however, and size-space trade-offs have proven elusive. The results in [31] apply not only to resolution and polynomial calculus but also to cutting planes, and were improved further in [29] to hold for even stronger proof systems, but unfortunately are not true trade-offs in the sense discussed above.

The problem is that what is shown in [29], [31] is only that proofs in small space for certain formulas have to be very large, but it is not established that these formulas can be refuted space-efficiently. In fact, for resolution it can be shown using techniques from [8] that such small-space proofs provably do not exist, and for polynomial calculus there is circumstantial evidence for a similar claim. This turns out to be an inherent limitation of the technique used.

In a recent surprising paper [26], it was shown that cutting planes can refute any formula in *constant* space if we only count the number of lines or formulas. Plugging this result into [29], [31] yields a trade-off of sorts, since "small-space" proofs will always exist, but the catch is that such proofs will have exponentially large coefficients. This means that these trade-offs do not seem very "algorithmically relevant" in the sense that such proofs could hardly be found in practice, and saying that a proof with exponential-size coefficients has "constant space" somehow does not feel quite right.

B. Our Proof Complexity Contributions

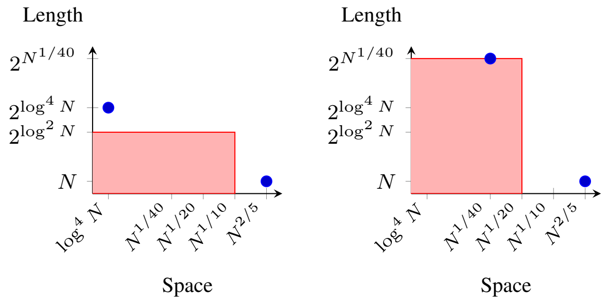
In this paper we report the first true, algorithmically realizable trade-offs for cutting planes, where the upper bounds hold for proof size and total space and the lower bounds apply to proof length and formula space (i.e., number of inequalities). The trade-offs also hold for resolution and polynomial calculus, making them the first trade-offs that hold for essentially all methods of reasoning used in the most successful SAT solvers to date.[1]

Below, we state two examples of the kind of trade-offs we obtain (referring the reader to Section II for the missing formal definitions). In the rest of this section we will focus on cutting planes, since this proof system is the main target of this work. However, all the lower bounds stated also hold for polynomial calculus (and for the strictly weaker proof system resolution), and since all our upper bounds are actually proven in resolution they transfer to both polynomial calculus and cutting planes.

The first result is a "robust trade-off" that holds all the way from polylogarithmic to polynomial space as stated next.

Theorem 1 (Informal). *There exists an explicitly constructible family of* 6*-CNF formulas* $\{F_N\}_{N=1}^{\infty}$ *of size* $\Theta(N)$ *such that:*

[1] We remark that this ignores the issue of formula *preprocessing techniques*, which are heavily used in most state-of-the-art SAT solvers, and some of which potentially require the full extended Frege proof system for a complete formal description (but can also sometimes cause a provable exponential *loss* in reasoning power). Since in practice SAT solvers fail to solve many of the combinatorial benchmark formulas that are hard for resolution, polynomial calculus, and cutting planes but easy for (even non-extended) Frege, however, and since in addition it is usually not hard to come up with formulas that foil any concrete preprocessing techniques actually used, this seems like a reasonable simplification.



(a) Robust trade-off

(b) Exponential trade-off

Figure 1. Pictorial illustrations of trade-offs in Theorems 1 and 2

1) *F_N can be refuted by cutting planes with constant-size coefficients in size $\mathrm{O}(N)$ and total space $\mathrm{O}\big(N^{2/5}\big)$.*
2) *F_N can be refuted by cutting planes with constant-size coefficients in total space $\mathrm{O}(\log^4 N)$ and size $2^{\mathrm{O}(\log^4 N)}$.*
3) *Any cutting planes refutation of F_N, even with coefficients of unbounded size, in formula space less than $N^{1/10-\epsilon}$ requires length greater than $2^{\Omega(\log^2 N)}$.*

The second trade-off holds over a smaller space range, but causes an exponential and not just superpolynomial blow-up in proof size.

Theorem 2 (Informal). *There exists an explicitly constructible family of* 6*-CNF formulas $\{F_N\}_{N=1}^{\infty}$ of size $\Theta(N)$ such that:*

1) *F_N can be refuted by cutting planes with constant-size coefficients in size $\mathrm{O}(N)$ and total space $\mathrm{O}\big(N^{2/5}\big)$.*
2) *F_N can be refuted by cutting planes with constant-size coefficients in total space $\mathrm{O}\big(N^{1/40}\big)$ and size $2^{\mathrm{O}(N^{1/40})}$.*
3) *Any cutting planes refutation of F_N, even with coefficients of unbounded size, in formula space less than $N^{1/20-\epsilon}$ requires length greater than $2^{\Omega(N^{1/40})}$.*

See Figure 1 for an illustration of these results, where blue dots denote provable upper bounds on time-space parameters of cutting planes refutations and the shaded red areas show ranges of parameters that are impossible to achieve.

C. Previous Work in Monotone Circuit Complexity

Since this paper also makes contributions to monotone circuit complexity, we next review some relevant background in this area. After superpolynomial lower bounds on the size of monotone circuits computing explicit functions were obtained in [2], [46], the first step towards the natural next goal of establishing a depth hierarchy for monotone circuits was taken in [34], proving that connectivity, which is in monotone-NC^2, requires depth $\Omega(\log^2 n)$ for monotone circuits with fan-in 2. This implies a separation between monotone-NC^1 and monotone-NC^2. The same approach was used in [45] to prove a separation between monotone-NC^{i-1} and monotone-NC^i for every i. This result can be rephrased as saying that there is a family of Boolean functions $\{f^i\}$ such that f^i can be computed by monotone circuits of depth $\log^i n$, fan-in 2, and polynomial size but cannot be computed by any monotone circuit of depth $\mathrm{o}(\log^i n)$ and fan-in 2. This result was later extended in [33] to circuits of semi-unbounded fan-in—i.e., with AND-gates of fan-in 2 and OR-gates of unbounded fan-in.

Going into more details, the function in [45] that witnesses the separation between monotone-NC^{i-1} and monotone-NC^i can be computed by a monotone circuit of depth $\log^{i-1} n$, polynomial fan-in, and polynomial size, and therefore the separation is between monotone-NC^{i-1} and monotone-AC^{i-1}. This immediately implies a separation between monotone-AC^{i-1} and monotone-AC^i as well, since monotone-AC^{i-1} is contained in monotone-NC^i. However, this separation only guarantees a superpolynomial circuit size lower bound. Furthermore, the function f^i only depends on $\log^{40i} n$ variables and so it can be computed by a monotone DNF of size $2^{\log^{40i} n}$, i.e., there is a quasipolynomial upper bound.

We remark that it is clearly not possible to prove a superpolynomial separation between monotone-NC^{i-1} and monotone-NC^i in view of the simple fact that circuits in these classes have fan-in 2, and hence it only makes sense to talk about superpolynomial versus exponential separations in the monotone-AC hierarchy. It should be noted that exponential separations between monotone circuits of bounded depth were previously known, but only for depth less than logarithmic. It was shown in [35] that the complete tree of depth k, arity $n^{1/k}$, and size $\Theta(n)$, with alternating levels of AND and OR, requires size $2^{\Omega(n^{1/k}/k)}$ to compute with circuits of depth $k-1$. This result was later reproven in [42] using the communication complexity of the pointer jumping function (see also [44]).

D. Our Monotone Circuit Complexity Contributions

In this paper, we establish an exponential separation in the monotone-AC hierarchy. More precisely, for each $i \in \mathbb{N}$ we exhibit a Boolean function f^i that can be computed by monotone circuits of depth $\log^i n$ but such that every monotone circuit of depth at most $\mathrm{O}(\log^{i-1} n)$ requires size $2^{n^{\Omega(1)}}$ (where the hidden constant in the lower bound depends inversely on that in the upper bound).

Theorem 3. *For every $i \in \mathbb{N}$ there is a Boolean function over n variables that can be computed by a monotone circuit of depth $\log^i n$, fan-in $n^{4/5}$, and size $\mathrm{O}(n)$, but for which every monotone circuit of depth $q \log^{i-1} n$ requires size $2^{\Omega(n^{1/(10+4\epsilon)q})}$.*

E. Discussion of Techniques

Let us now briefly discuss the techniques we use to establish the above results, focusing for concreteness on

Theorems 1 and 2. These theorems are proven by a careful chain of reductions as follows.

1) Our first step is to use the connection made explicit in [31], and also used in [29], that short and space-efficient proofs for a CNF formula F can be converted to efficient communication protocols for the *falsified clause search problem* for F. Going beyond [29], [31], however, we make the simple but absolutely crucial additional observation that protocols obtained in this way are also round-efficient. Furthermore, in contrast to [29], [31] we do not study randomized communication, but instead focus on the *real communication model* introduced by Krajíček [36] with the purpose of getting a tighter correspondence with cutting planes.
2) We next generalize the communication-to-decision-tree simulation theorem for *composed search problem* in the celebrated paper by Göös et al. [30] to the real communication model, and then extend it further to be able to handle rounds using the *parallel decision trees* introduced by Valiant [48]. This part is inspired by [13], where the simulation theorem in the precursor [45] of [30] was proven for real communication but without taking round efficiency into account.
3) To leverage this machinery we need a base CNF search problem, and just as in [6], [9], [29], [31] (and many other papers) the *pebbling formulas* Peb_G from [10] turn out to be handy here, provided that they are defined over appropriately chosen directed acyclic graphs G. These formulas are then *lifted* (corresponding to composition of search problems) as described in [5], though the parameters of the lifting are different (and unfortunately significantly worse than in [31]).
4) The following step is the relatively straightforward observation that efficient parallel decision trees for formulas Peb_G yield good strategies in the pebble game of Dymond and Tompa [23] played on the underlying graph G. At the same time, this is a somewhat unexpected twist, since in previous papers such as [6], [8], [9] size and space lower bounds for pebbling formulas always followed from the *black-white pebble game* [20] on G, but we cannot make use of that latter game here.
5) Since we have to use the Dymond–Tompa game rather than the black-white pebble game, as a consequence we also have to use different graphs than in [6], [9], [31]—in particular, modifying the construction of graphs with good black-white pebbling trade-offs in [38]—and as a concluding step we prove Dymond–Tompa trade-offs for these graphs.

Putting all these pieces together, we obtain a general theorem saying that graphs with Dymond–Tompa trade-offs yield explicit 6-CNF formulas with size-space trade-offs for cutting planes (and polynomial calculus and resolution). Theorem 3 follows by a similar chain of reductions.

F. Paper Outline

In Section II we give a more detailed overview of the steps in the proofs of our main theorems, introducing formal definitions of the concepts discussed above as need arises. Due to space constraints we can only sketch the proofs in this extended abstract, however, and therefore refer to the upcoming full-length version for all missing details. We conclude in Section III with a discussion of possible directions for future research.

II. Preliminaries and Proof Overview

In this section, we describe which components are needed for our results stated in Section I and how they fit together. Our goal is to give an accessible high-level outline of the proofs, but still make clear what are the main technical points in the arguments and also indicate some of the challenges that have to be overcome.

Let us start by reviewing the concepts we need from proof complexity. Throughout this paper all logarithms are to base 2 unless otherwise specified, and we write $[n]$ to denote the set $\{1, 2, \ldots, n\}$.

A. Proof Complexity Basics and Cutting Planes

For x a Boolean variable, a *literal over* x is either the variable x itself or its negation, denoted $\overline{x}$. It will also be convenient to use the notation $x^1 = x$ and $x^0 = \overline{x}$. A *clause* $C = a_1 \vee \cdots \vee a_k$ is a disjunction of literals and a *CNF formula* $F = C_1 \wedge \cdots \wedge C_m$ is a conjunction of clauses. We will think of clauses and CNF formulas as sets, so that the ordering is inconsequential and there are no repetitions. A *k-CNF formula* is a CNF formula consisting of clauses containing at most k literals.

We write α, β to denote truth value assignments, i.e., functions to $\{0, 1\}$, where we identify 0 with false and 1 with true (thus, x^b is the literal satisfied by setting $x = b$). We have the usual semantics that a clause is true under α, or *satisfied* by α, if at least one literal in it is true, and a CNF formula is satisfied if all clauses in it are satisfied. We write $\bot$ to denote the empty clause without literals, which is false under all truth value assignments.

Following [1], [24], we view a proof of unsatisfiability of a CNF formula F, or *refutation* of F, as a non-deterministic computation, with a special read-only input tape from which the clauses of the formula F being refuted (which we refer to as *axioms*) can be downloaded and a working memory where all derivation steps are made. In a *cutting planes (CP)* derivation, memory configurations are sets of linear inequalities $\sum_j a_j x_j \geq c$ with $a_j, c \in \mathbb{Z}$. We translate clauses C to linear inequalities $L(C)$ by identifying the clause $\bigvee_j x_j^{b_j}$ with the inequality $\sum_j (-1)^{1-b_j} x_j \geq 1 - \sum_j (1 - b_j)$. A CP refutation of F is a sequence of configurations $(\mathbb{L}_0, \ldots, \mathbb{L}_\tau)$ such that $\mathbb{L}_0 = \emptyset$, the inequality $0 \geq 1$ occurs in $\mathbb{L}_\tau$, and for $t \in [\tau]$ we obtain $\mathbb{L}_t$ from $\mathbb{L}_{t-1}$ by one of the following rules:

Axiom download: $\mathbb{L}_t = \mathbb{L}_{t-1} \cup \{L\}$ for L being either the encoding $L(C)$ of an *axiom clause* $C \in F$ or a *variable axiom* $x_j \geq 0$ or $-x_j \geq -1$ for any variable x_j.

Inference: $\mathbb{L}_t = \mathbb{L}_{t-1} \cup \{L\}$ for L inferred by *addition*

$$\frac{\sum_j a_j x_j \geq c \qquad \sum_j b_j x_j \geq d}{\sum_j (a_j + b_j) x_j \geq c + d} \ , \tag{1}$$

multiplication

$$\frac{\sum_j a_j x_j \geq c}{\sum_j k a_j x_j \geq kc} \ , \tag{2}$$

or *division*

$$\frac{\sum_j k a_j x_j \geq c}{\sum_j a_j x_j \geq \lceil c/k \rceil} \tag{3}$$

for $k \in \mathbb{N}^+$.

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

The *length* of a CP refutation is the number of linear inequalities L appearing in download and inference steps, counted with repetitions. We obtain the *size* of a refutation by also summing the sizes of the coefficients and constant terms in the inequalities, i.e., each inequality $\sum_j a_j x_j \geq c$ contributes $\log|c| + \sum_j \log|a_j|$. The *formula space* of a configuration $\mathbb{L} = \{\sum_j a_{i,j} x_{i,j} \geq c_i \mid i \in [s]\}$ is the number of inequalities s in it, and the *total space* of $\mathbb{L}$ is $\sum_{i \in [s]} \bigl(\log|c_i| + \sum_j \log|a_{i,j}|\bigr)$. We obtain the formula space or total space of a refutation by taking the maximum over all configurations in it. Finally, the length, size, formula space, and total space of refuting a formula F is obtained by taking the minimum over all CP refutations of the formula with respect to the corresponding complexity measure.

B. Composed Search Problems and Lifted CNF Formulas

Informally speaking, the idea behind *lifting*, or *composition*, is to take a relation over some domain and extend it to tuples from the same domain by combining it with a *selector* function that determines on which coordinates from the tuples the relation should be evaluated.

Let S be any relation on the Cartesian product $A \times Q$. We will think of S as a *search problem* with input domain A and output range Q, where given $a \in A$ the task is to find some $q \in Q$ such that $(a, q) \in S$ (assuming that S is such that there always exists at least one solution). Throughout this paper, we will have $A = \{0,1\}^m$ for some $m \in \mathbb{N}^+$, so for simplicity we fix A to be such a domain from now on.

For any $\ell \in \mathbb{N}^+$, we define the *lift of length ℓ of S* to be a new search problem $Lift_\ell(S) \subseteq \bigl([\ell]^m \times \{0,1\}^{m \cdot \ell}\bigr) \times Q$ with input domain $[\ell]^m \times \{0,1\}^{m \cdot \ell}$ and output range Q such that for any $x \in [\ell]^m$, any bit-vector $\{y_{i,j}\}_{i \in [m], j \in [\ell]}$, and any $q \in Q$, it holds that

$$\begin{aligned} (x, y, q) \in Lift_\ell(S) \quad & \text{if and only if} \\ & \bigl((y_{1,x_1}, y_{2,x_2}, \ldots, y_{m,x_m}), q\bigr) \in S \ . \end{aligned} \tag{4}$$

In what follows, we will refer to the coordinates of the x-vector as *selector variables* and those of the y-vector as *main variables*, and we will sometimes use the notation

$$\mathsf{select}(x_i, y_i) = y_{i,x_i} \tag{5}$$

to denote the bit in y_i selected by x_i. We extend this notation to vectors to write $\mathsf{select}(x, y) = y_x = (y_{1,x_1}, \ldots, y_{m,x_m})$.

As in [29], [31], we obtain our results by studying lifted search problems defined in terms of CNF formulas and proving communication lower bounds for such problems. Syntactically speaking, however, these objects are not themselves CNF formulas, which is what we use to feed to our proof system. Therefore, we need an additional step which translates the lifted search problems back to CNF as follows.

Definition 4 (Lifted formula [5]). Given $\ell \in \mathbb{N}^+$ and a CNF formula F over variables $u_1, \ldots, u_n$, the *lift of length ℓ of F*, denoted $Lift_\ell(F)$, is the formula over variables $\{x_{i,j}\}_{i \in [n], j \in [\ell]}$ (*selector variables*) and $\{y_{i,j}\}_{i \in [n], j \in [\ell]}$ (*main variables*) containing the following clauses:

- For every $i \in [n]$, an *auxiliary clause*
$$x_{i,1} \lor x_{i,2} \lor \cdots \lor x_{i,\ell} \ . \tag{6a}$$
- For every clause $C_i \in F$, where $C_i = u_{i_1}^{b_{i_1}} \lor \cdots \lor u_{i_s}^{b_{i_s}}$ for some $i_1, \ldots, i_s \in [n]$, and for every tuple $(j_1, \ldots, j_s) \in [\ell]^s$, a *main clause*
$$x^0_{i_1,j_1} \lor y^{b_{i_1}}_{i_1,j_1} \lor \cdots \lor x^0_{i_s,j_s} \lor y^{b_{i_s}}_{i_s,j_s} \ . \tag{6b}$$

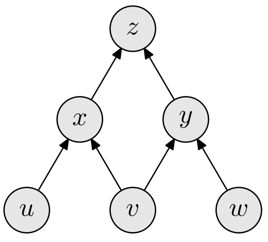
Intuitively, we can think of the selector variables as encoding the vector $x \in [\ell]^m$ in the lifted search problem (4). Since $\overline{x}_{i,j} \lor y_{i,j}$ is equivalent to the implication $x_{i,j} \rightarrow y_{i,j}$, we can rewrite (6b) as

$$\bigl(x_{i_1,j_1} \rightarrow y^{b_{i_1}}_{i_1,j_1}\bigr) \lor \cdots \lor \bigl(x_{i_s,j_s} \rightarrow y^{b_{i_s}}_{i_s,j_s}\bigr) \ , \tag{7}$$

from which we can see that for every clause C_i the auxiliary clauses encode that there is at least one choice of selector variables $x_{i,j}$ which are all true, and for this choice of selector variables the $y_{i,j}$-variables in the lifted main clause will play the role of the u_i-variables, giving us back the original clause C_i. It is easily verified that F is unsatisfiable if and only if $H = Lift_\ell(F)$ is unsatisfiable, and that if F is a k-CNF formula with m clauses, then H is a $\max(2k, \ell)$-CNF formula with at most $m\ell^k + n$ clauses. A small technical issue for us compared to [29], [31] is that $\ell \gg k$ will not be constant, but we can convert the wide clauses in (6a) to constant width using extension variables, and so we will just ignore this issue in our proof overview.

C. Pebbling Contradictions

An important role in many proof complexity trade-off results is played by so-called *pebbling contradictions*. For our purposes it suffices to say that they are defined in terms of directed acyclic graphs (DAGs) G, where for simplicity we assume that all vertices have indegree 0 or 2. We refer



(a) Pyramid graph Π_2 of height 2.

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

(b) Pebbling contradiction Peb_{Π_2}.

Figure 2. Example pebbling contradiction for the pyramid of height 2.

to vertices with indegree 0 as *sources* and assume that there is a unique *sink* vertex with outdegree 0. What the pebbling contradiction over G says is that the sources are true and that truth propagates from predecessors to successors, but that the sink is false.

Definition 5 (Pebbling contradiction [10]). Let G be a DAG with sources S and a unique sink z, and with all non-sources having indegree 2. Then the *pebbling contradiction* over G, denoted Peb_G, is the conjunction of the following clauses over variables $\{v \mid v \in V(G)\}$:

- for every source $s \in S$, a unit clause s (*source axioms*),
- For all non-sources w with immediate predecessors u, v, a clause $\overline{u} \vee \overline{v} \vee w$ (*pebbling axioms*),
- for the sink z, the unit clause $\overline{z}$ (*sink axiom*).

If G has n vertices, the formula Peb_G is an unsatisfiable 3-CNF formula with $n+1$ clauses over n variables. For an example of a pebbling contradiction, see the CNF formula in Figure 2b defined in terms of the graph in Figure 2a.

D. Real Communication and Clause Search Problems

For our communication complexity results we study a two-player communication model, referring to the players as *Alice* and *Bob* following tradition. We first briefly discuss the basic deterministic model, and then explain how we need to extend it, directing the reader to [37] for any omitted standard communication complexity facts.

In the communication problem of computing a function $f : X \times Y \to Q$, Alice is given an input $x \in X$, Bob is given an input $y \in Y$, and they are required to find $f(x, y)$ while minimizing the communication between them. A communication protocol is a binary tree where Alice and Bob start at the root, every node specifies who is going to speak, the value of the spoken bit is only a function of the node v and the input x if Alice speaks or y if Bob does, and leaves are labelled by correct values $f(x, y)$. Similarly, for any relation $S \subseteq X \times Y \times Q$, the communication problem for S is one in which Alice is given $x \in X$, Bob is given $y \in Y$, and they are required to communicate to find some q such that $(x, y, q) \in S$. The cost of a protocol is the maximum number of bits communicated on any input, and the number of rounds is the maximum number of alternations between Alice and Bob speaking.

In order to obtain trade-offs for cutting planes, we need to study the more general *real communication* model in [36], where Alice and Bob interact via a referee, and also introduce the concept of rounds in this model. It is convenient to describe the protocol as a (non-binary) tree, where at node v in the protocol tree Alice and Bob send k_v real numbers $\phi_{v,1}(x), \ldots, \phi_{v,k_v}(x)$ and $\psi_{v,1}(y), \ldots, \psi_{v,k_v}(y)$, respectively, to the referee. The referee announces the results of the comparisons $\phi_{v,i}(x) \leq \psi_{v,i}(y)$ for $i \in [k_v]$ as a k_v-bit binary string, after which the players move to the corresponding next node in the protocol tree. The number of rounds r of a protocol is the depth of the tree and the cost c is the total number of comparisons made by the referee for any input. It is easy to see that this model can simulate standard deterministic communication (for instance, if Alice wants to send a message, she sends the complement of that message to the referee and Bob sends a list of the same length with all entries $1/2$) and is in fact strictly stronger (since equality can be solved with just two bits of communication).

The communication problem that we are interested in is the *(falsified) clause search problem*. This is the problem of, given an unsatisfiable CNF formula F and a truth value assignment α, finding a clause $C \in F$ falsified by α. We denote this problem by $Search(F)$. In fact, from a communication complexity point of view we will be interested in lifts of this search problem $Lift(Search(F))$, while for our proof complexity trade-offs the perspective is slightly different in that we need to study the CNF formula $Lift(F)$ from Definition 4 and relate the hardness of this formula to the communication complexity of the falsified clause search problem $Search(Lift(F))$. Happily, this distinction does not really matter to us, since a good communication protocol for $Search(Lift(F))$ can also be used to solve $Lift(Search(F))$, and hence a lower bound for the latter communication problem applies also to the former.

Observation 6. *Suppose that F is an unsatisfiable CNF formula. Then any two-player real communication protocol for $Search(Lift_\ell(F))$ where all selector variables $x_{i,j}$ in the same block are given to the same player can be adapted to a protocol for $Lift_\ell(Search(F))$ with the same parameters.*

We refer to, e.g., [31] for the easy proof (which is independent of the concrete communication model under consideration). Thanks to this observation, we can freely switch perspectives between $Lift_\ell(Search(F))$ and $Search(Lift_\ell(F))$ when we want to prove lower bounds for the latter problem. The reason that such lower bounds are interesting, in turn, is that if a CNF formula H has a CP refutation in short length and small space, then such a proof can be used to construct a round- and communication-efficient protocol for $Search(H)$.

Lemma 7. *If a CNF formula H can be refuted by cutting planes in length L and formula space s, then for any partition of the variables of H between Alice and Bob there is a real communication protocol solving $Search(H)$ in $\lceil \log L \rceil$ rounds with communication cost at most $s \cdot \lceil \log L \rceil$.*

Sketching the proof very briefly, given a truth value assignment α Alice and Bob can do binary search over the refutation $(\mathbb{L}_0 = \emptyset, \mathbb{L}_1, \ldots, \mathbb{L}_L)$ of H until they find a $t \in [L]$ such that $\mathbb{L}_t$ evaluates to true under α but $\mathbb{L}_{t-1}$ evaluates to false. Then the derivation step at time t must be a download of an axiom $C \in H$ falsified by α. For the details we can reuse the proof from [31] verbatim, just adding the one simple but absolutely crucial observation that the protocol obtained in this way is also round-efficient, since all communication needed to evaluate a particular configuration $\mathbb{L}_t$ can be performed in parallel.

It is worth noting that although we state Lemma 7 for cutting planes here, there is nothing that really uses the syntactic properties of the cutting planes refutation. Thus, the proof works equally well for resolution, polynomial calculus, or any proof system for which configurations can be evaluated by round-efficient protocols where the communication scales as the space of the configuration.

E. Simulations of Protocols by Parallel Decision Trees

A *parallel decision tree* [48] for a search problem $S \subseteq \{0,1\}^m \times Q$ is a tree T such that each node v is labelled by a set of variables V_v and has exactly one outgoing edge for each of the $2^{|V_v|}$ possible assignments to these variables, and such that for every $\alpha \in \{0,1\}^m$ the path from the root of T defined by the edges consistent with α ends at a leaf labelled by some $q \in Q$ such that $(\alpha, q) \in S$ (where again the tacit assumption is that S is such that such a solution always exists). The *number of queries* of T is the maximal sum of set sizes $|V_v|$ along any path in T, and the *depth* of T is the length of a longest path.

Any decision tree T for a search problem S can be simulated by a communication protocol for the lifted problem $Lift(S)$ in a straightforward way, where if T wants to query the ith variable Alice and Bob can communicate to find y_{i,x_i} and then walk in T according to this value. Such a walk will end in a leaf labelled by a q such that $\big((y_{1,x_1}, y_{2,x_2}, \ldots, y_{m,x_m}), q\big) \in S$, i.e., a solution to the lifted search problem, and thus the query complexity of the original search problem provides an upper bound on the communication cost of the lifted problem. If in addition there is a parallel decision tree with small depth, then a protocol simulating such a tree will also be round-efficient. The main technical result of our paper is that simulating such a parallel decision tree is essentially the best any round-efficient protocol can do (provided that the lifting of the search problem is done with appropriate parameters).

Theorem 8 (Simulation theorem). *Let S be a relation with domain $\{0,1\}^m$ and let $\ell = m^{3+\epsilon}$ for some constant $\epsilon > 0$. If there is an r-round real communication protocol in cost c that solves $Lift_\ell(S)$, then there is a parallel decision tree in depth r solving S using $\mathrm{O}(c/\log \ell)$ queries.*

We remark that similar simulation theorems have previously been shown for both deterministic communication [30], [45] and real communication [13], but unfortunately they fail to take round efficiency into account. Our proof of Theorem 8 follows the approach in these papers to build a decision tree for the original problem that simulates the communication protocol for the lifted problem. In order to obtain an efficient simulation we have to maintain (in an amortized sense) that the decision tree queries a variable only when a noticeable amount of communication has taken place. To prove that the decision tree constructed in this way is correct, we need to show that at the end of the simulation there exists a pair of inputs to Alice and Bob that are compatible both with the transcript and with a lift of the original input. Towards this end, during the simulation we maintain a set of such compatible inputs, which must not be allowed to shrink too fast.

In order for the proof to work we need to be able to handle two kinds of events: *communication events*, where we simulate the players communicating; and *query events*, where the decision tree under construction queries some variable and gets its actual value. Both of these events force us to prune the set of compatible communication inputs. In the first case we want to choose a communication message that removes as few inputs as possible, whereas in the second case we have to restrict the communication inputs to a subset that is compatible with the value returned by the decision tree query. We make sure to query a variable only when the transcript "reveals too much information" about Alice's and Bob's lifted input related to that variable, and thanks to this we can argue that query events do not happen too often and that the amount of communication provides an upper bound on the total number of queries.

Extending these techniques to round-efficient protocols and simulations by parallel decision trees causes significant additional complications, however. Very briefly, one issue is that we cannot let the tree query an individual variable as soon as sufficient information has been "revealed" about it during the simulation, but have to wait until we can issue a whole set of queries corresponding to a single message of the protocol. This makes it challenging to maintain a set of compatible inputs for variables we have not yet been allowed to query. Another issue is that, in contrast to deterministic communication protocols, real protocols do not partition the input domain into combinatorial rectangles. While this is not a big problem for a single comparison by the referee, it becomes more challenging when we want to handle a round consisting of many simultaneous comparisons.

F. From Decision Trees to Dymond–Tompa Trade-offs

The *Dymond–Tompa game* [23][2] is played in rounds on a DAG G by two players *Pebbler* and *Challenger*. In the first round, Pebbler places pebbles on a non-empty subset of vertices of G including the unique sink z and Challenger picks some vertex in this set. In all subsequent rounds, Pebbler places pebbles on some non-empty subset of vertices not yet containing pebbles, and Challenger either challenges a vertex in this new set (*jumps*) or re-challenges the previously chosen vertex (*stays*). This repeats until at the end of a round Challenger is standing on a vertex with all immediate predecessors pebbled (or on a source, for which the condition vacuously holds), at which point the game ends. We say that Pebbler *wins the r-round Dymond–Tompa game on G in cost c* if there is a strategy such that Pebbler can always finish the game in at most r rounds placing a total of at most c pebbles regardless of how Challenger plays.

In order to obtain lower bounds on the query complexity of parallel decision trees of bounded depth, we use an adversary argument and describe strategies that give as unhelpful answers as possible for variables queried by the decision trees. If we specialize this to the clause search problem for pebbling contradictions Peb_G, such adversary strategies are equivalent to Challenger strategies in the Dymond–Tompa game on G. For standard binary decision trees and the Dymond–Tompa game with unlimited number of rounds this was proven in [14],[3] and we show that this equivalence extends also to our more general setting.

Lemma 9. *If there is a parallel decision tree for $Search\big(Peb_G\big)$ in depth r using at most c queries, then Pebbler has a winning strategy in the r-round Dymond–Tompa game on G in cost at most $c + 1$.*

It follows from this lemma that round-cost trade-offs for Dymond–Tompa pebbling implies depth-query trade-offs for parallel decision trees. To conclude the proof of the lower bound in our trade-off results, we need to find a family of graphs for which we can prove lower bounds for the cost of Pebbler strategies in the Dymond–Tompa game with bounded number of rounds. Towards this end, we establish that graphs that satisfy a certain connectivity property possess trade-offs between number of rounds and cost, and then exhibit such graphs. These graphs were inspired by graphs for which black-white pebbling trade-offs were shown in [38], but we need to make some modifications in order to obtain Dymond–Tompa trade-offs. To the best of our knowledge such round-cost trade-offs in the Dymond–Tompa game have not been studied before.

Lemma 10. *For any $n, r \in \mathbb{N}^+$ there exists an explicitly constructible DAG $G(n, r)$ with $\mathrm{O}(rn \log n)$ vertices such that the cost of the r-round Dymond–Tompa game on $G(n, r)$ is at least $\Omega(n)$.*

The graph $G(n, r)$ is structured in $r + 1$ layers and we obtain the lemma by showing that as long as Pebbler does not place too many pebbles Challenger can make sure that in the ith round the challenged pebble is above the ith layer. Hence, the game does not end within r rounds.

G. Proofs of Main Theorems

Combining all the components discussed above we can now prove the following trade-off lower bound.

Theorem 11. *Let G be a DAG over m vertices such that any winning strategy for Pebbler in the r-round Dymond–Tompa game has cost $\Omega(c)$ and let $\epsilon > 0$ and $\ell = m^{3+\epsilon}$. Then $Lift_\ell\big(Peb_G\big)$ is a 6-CNF formula over $\Theta(m^{4+\epsilon})$ variables and $N = \Theta(m^{10+3\epsilon})$ clauses such that any cutting planes refutation in formula space less than $\frac{c}{r} \log N$, even with coefficients of unbounded size, requires length at least $2^{\Omega(r)}$.*

Proof: Suppose for the sake of contradiction that there is a cutting planes refutation of $Lift_\ell\big(Peb_G\big)$ in length $2^{\mathrm{o}(r)}$ and formula space less than $\frac{c}{r} \log N$. By Lemma 7 this implies that there is a real communication protocol that solves $Lift_\ell\big(Search(Peb_G)\big)$ in $\mathrm{o}(r)$ rounds and total cost $\mathrm{o}(c \log N)$. Using Theorem 8 we obtain a parallel decision tree computing $Search(Peb_G)$ using $\mathrm{o}(c)$ queries and depth $\mathrm{o}(r)$. But if so, by Lemma 9 Pebbler wins the $\mathrm{o}(r)$-round Dymond–Tompa game on G in cost $\mathrm{o}(c)$, which contradicts the assumption of the theorem. ■

In order to attain our trade-off results we also need upper bounds on refutations of these formulas. Small-size upper bounds follow by essentially the same approach of lifting black pebbling upper bounds as in [9], [31], although more care is needed since our lifts are of non-constant length. For the small-space refutations, this technique does not work because the space loss due to the large lift length is larger than the upper bound we are aiming for. Luckily, we can instead prove upper bounds in the Dymond–Tompa game with unlimited rounds and then convert them into refutations in small space. Theorems 1 and 2 then follow from Theorem 11 applied to an appropriate family of graphs that exhibit Dymond–Tompa trade-offs as in Lemma 10.

The tools we have developed also allow us to prove the monotone circuit separation in Theorem 3. The function that witnesses the separation is inspired by the PYRAMID-GEN function of [45] adapted to the graphs in Lemma 10. Then we translate the Dymond–Tompa trade-off into a lower bound for deterministic communication protocols with few rounds, which we then transform into a lower bound for circuits of small depth via the Karchmer–Wigderson game [34].

III. Concluding Remarks

In this paper we report the first true size-space trade-offs for cutting planes, exhibiting CNF formulas which

[2] We give a slightly different, but essentially equivalent, description of the Dymond–Tompa game that is closer to recent papers such as [14], [15].

[3] This game on decision trees is called the *Raz–McKenzie game* in [14].

have small-size and small-space proofs with constant-size coefficients but for which any short proofs must use a lot of memory, even when using exponentially large coefficients and even when we measure just the number of lines (i.e., inequalities) rather than total size. Furthermore, these results also hold for resolution and polynomial calculus, and are thus the first trade-offs to uniformly capture the proof systems underlying the currently best SAT solvers.

The main technical component in our proof is a reduction to communication complexity as in [29], [31], but with the crucial difference that we reduce to round-efficient protocols in the real communication model of [36]. Extending the techniques in [13], [30], [45] to this more general setting, and combining them with new trade-off results for Dymond–Tompa pebbling [23], yields our results. Using the same approach we are also able to obtain an exponential separation between $\mathsf{monotone\text{-}AC}^{i-1}$ and $\mathsf{monotone\text{-}AC}^{i}$, improving on the superpolynomial separation in [45].

An interesting challenge would be to extend our reduction to stronger communication models such as two-party randomized or multi-party real communication, which would yield trade-offs for stronger proof systems. A recent result in this direction is [28], but unfortunately it seems hard to incorporate round-efficiency in this framework.

Another question concerns the size of the lifting gadgets we need to construct formulas exhibiting trade-offs. Our gadgets have large polynomial size, which incurs a substantial loss in the results. It would be nice to construct constant-size gadgets, which could lead to tighter trade-off results.

Many proof complexity trade-offs have been obtained by reducing to the *black-white pebble game* [20], but in this paper we use the Dymond–Tompa game. It would be desirable to obtain a better understanding of the role of these games and what kind of trade-offs can be obtained from them.

Finally, from a proof complexity perspective we have very few examples of formula families that exhibit size-space trade-offs. Apart from the pebbling formulas studied in this work, the only natural examples[4] are the Tseitin contradictions over long, narrow grids in [4], [6]. It would be interesting to prove size-space trade-offs for the latter formulas also in cutting planes, or to find other formulas with size-space trade-offs for this or other proof systems.

Acknowledgements

The authors wish to acknowledge Mladen Mikša, who participated in the initial stages of this work and has kept contributing helpful remarks throughout the project, and Massimo Lauria, with whom we have had many fruitful discussions on time-space trade-offs and other topics in proof complexity. We want to thank Siu Man Chan for introducing us to the wonderful world of Dymond–Tompa pebbling. Different subsets of the authors are grateful for detailed and very helpful discussions on communication complexity with Joshua Brody, Arkadev Chattopadhyay, Prahladh Harsha, Johan Håstad, Troy Lee, Jaikumar Radakrishnan, and Anup Rao. Finally, we are thankful to Dieter van Melkebeek and Ryan Williams for help with references for general SAT time-space trade-offs.

The authors were funded by the European Research Council under the European Union's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement no. 279611 as well as by the Swedish Research Council grant 621-2012-5645.

References

[1] M. Alekhnovich, E. Ben-Sasson, A. A. Razborov, and A. Wigderson, "Space complexity in propositional calculus," *SIAM Journal on Computing*, vol. 31, no. 4, pp. 1184–1211, 2002, preliminary version in *STOC '00*.

[2] A. E. Andreev, "On a method for obtaining lower bounds for the complexity of individual monotone functions," *Soviet Mathematics Doklady*, vol. 31, no. 3, pp. 530–534, 1985.

[3] R. J. Bayardo Jr. and R. Schrag, "Using CSP look-back techniques to solve real-world SAT instances," in *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI '97)*, Jul. 1997, pp. 203–208.

[4] P. Beame, C. Beck, and R. Impagliazzo, "Time-space trade-offs in resolution: Superpolynomial lower bounds for superlinear space," in *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC '12)*, May 2012, pp. 213–232.

[5] P. Beame, T. Huynh, and T. Pitassi, "Hardness amplification in proof complexity," in *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC '10)*, Jun. 2010, pp. 87–96.

[6] C. Beck, J. Nordström, and B. Tang, "Some trade-off results for polynomial calculus," in *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13)*, May 2013, pp. 813–822.

[7] E. Ben-Sasson, "Size-space tradeoffs for resolution," *SIAM Journal on Computing*, vol. 38, no. 6, pp. 2511–2525, May 2009, preliminary version in *STOC '02*.

[8] E. Ben-Sasson and J. Nordström, "Short proofs may be spacious: An optimal separation of space and length in resolution," in *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08)*, Oct. 2008, pp. 709–718.

[9] ——, "Understanding space in proof complexity: Separations and trade-offs via substitutions," in *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '11)*, Jan. 2011, pp. 401–416.

[10] E. Ben-Sasson and A. Wigderson, "Short proofs are narrow—resolution made simple," *Journal of the ACM*, vol. 48, no. 2, pp. 149–169, Mar. 2001, preliminary version in *STOC '99*.

[11] A. Biere, M. J. H. Heule, H. van Maaren, and T. Walsh, Eds., *Handbook of Satisfiability*, ser. Frontiers in Artificial Intelligence and Applications. IOS Press, Feb. 2009.

[12] A. Blake, "Canonical expressions in Boolean algebra," Ph.D. dissertation, University of Chicago, 1937.

[13] M. L. Bonet, J. L. Esteban, N. Galesi, and J. Johannsen, "On the relative complexity of resolution refinements and cutting planes proof systems," *SIAM Journal on Computing*, vol. 30, no. 5, pp. 1462–1484, 2000, preliminary version in *FOCS '98*.

[4] Ignoring trade-offs obtained in [43] by gluing together disjoint copies of unrelated formulas.

[14] S. M. Chan, "Just a pebble game," in *Proceedings of the 28th Annual IEEE Conference on Computational Complexity (CCC '13)*, Jun. 2013, pp. 133–143.

[15] S. M. Chan, M. Lauria, J. Nordström, and M. Vinyals, "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)*, Oct. 2015, pp. 466–485.

[16] V. Chvátal, "Edmond polytopes and a hierarchy of combinatorial problems," *Discrete Mathematics*, vol. 4, no. 1, pp. 305–337, 1973.

[17] M. Clegg, J. Edmonds, and R. Impagliazzo, "Using the Groebner basis algorithm to find proofs of unsatisfiability," in *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96)*, May 1996, pp. 174–183.

[18] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC '71)*, 1971, pp. 151–158.

[19] S. A. Cook and R. Reckhow, "The relative efficiency of propositional proof systems," *Journal of Symbolic Logic*, vol. 44, no. 1, pp. 36–50, Mar. 1979.

[20] S. A. Cook and R. Sethi, "Storage requirements for deterministic polynomial time recognizable languages," *Journal of Computer and System Sciences*, vol. 13, no. 1, pp. 25–37, 1976, preliminary version in *STOC '74*.

[21] W. Cook, C. R. Coullard, and G. Turán, "On the complexity of cutting-plane proofs," *Discrete Applied Mathematics*, vol. 18, no. 1, pp. 25–38, Nov. 1987.

[22] S. Diehl, D. van Melkebeek, and R. Williams, "An improved time-space lower bound for tautologies," *Journal of Combinatorial Optimization*, vol. 22, no. 3, pp. 325–338, Oct. 2011, preliminary version in *COCOON '09*.

[23] P. W. Dymond and M. Tompa, "Speedups of deterministic machines by synchronous parallel machines," *Journal of Computer and System Sciences*, vol. 30, no. 2, pp. 149–161, Apr. 1985, preliminary version in *STOC '83*.

[24] J. L. Esteban and J. Torán, "Space bounds for resolution," *Information and Computation*, vol. 171, no. 1, pp. 84–97, 2001, preliminary versions of these results appeared in *STACS '99* and *CSL '99*.

[25] L. Fortnow, "Time-space tradeoffs for satisfiability," *Journal of Computer and System Sciences*, vol. 60, no. 2, pp. 337–353, Apr. 2000, preliminary version in *CCC '97*.

[26] N. Galesi, P. Pudlák, and N. Thapen, "The space complexity of cutting planes refutations," in *Proceedings of the 30th Annual Computational Complexity Conference (CCC '15)*, ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 33, Jun. 2015, pp. 433–447.

[27] R. E. Gomory, "An algorithm for integer solutions of linear programs," in *Recent Advances in Mathematical Programming*, R. Graves and P. Wolfe, Eds. New York: McGraw-Hill, 1963, pp. 269–302.

[28] M. Göös, S. Lovett, R. Meka, T. Watson, and D. Zuckerman, "Rectangles are nonnegative juntas," in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, Jun. 2015, pp. 257–266.

[29] M. Göös and T. Pitassi, "Communication lower bounds via critical block sensitivity," in *Proceedings of the 46th Annual ACM Symposium on Theory of Computing (STOC '14)*, May 2014, pp. 847–856.

[30] M. Göös, T. Pitassi, and T. Watson, "Deterministic communication vs. partition number," in *Proceedings of the 56th Annual IEEE Symposium on Foundations of Computer Science (FOCS '15)*, Oct. 2015, pp. 1077–1088.

[31] T. Huynh and J. Nordström, "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)*, May 2012, pp. 233–248.

[32] R. Impagliazzo and R. Paturi, "On the complexity of k-SAT," *Journal of Computer and System Sciences*, vol. 62, no. 2, pp. 367–375, Mar. 2001, preliminary version in *CCC '99*.

[33] J. Johannsen, "Depth lower bounds for monotone semi-unbounded fan-in circuits," *RAIRO-Theoretical Informatics and Applications*, vol. 35, no. 3, pp. 277–286, 2001.

[34] M. Karchmer and A. Wigderson, "Monotone circuits for connectivity require super-logarithmic depth," *SIAM Journal on Discrete Mathematics*, vol. 3, no. 2, pp. 255–265, 1990, preliminary version in *STOC '88*.

[35] M. Klawe, W. J. Paul, N. Pippenger, and M. Yannakakis, "On monotone formulae with restricted depth," in *Proceedings of the 16th Annual ACM Symposium on Theory of Computing (STOC '84)*, 1984, pp. 480–487.

[36] J. Krajíček, "Interpolation by a game," *Mathematical Logic Quarterly*, vol. 44, pp. 450–458, 1998.

[37] E. Kushilevitz and N. Nisan, *Communication complexity*. Cambridge University Press, 1997.

[38] T. Lengauer and R. E. Tarjan, "Asymptotically tight bounds on time-space trade-offs in a pebble game," *Journal of the ACM*, vol. 29, no. 4, pp. 1087–1130, Oct. 1982, preliminary version in *STOC '79*.

[39] L. A. Levin, "Universal'nye zadachi perebora," *Problemy Peredachi Informatsii*, vol. 9, no. 3, pp. 115–116, 1973, in Russian.

[40] J. P. Marques-Silva and K. A. Sakallah, "GRASP: A search algorithm for propositional satisfiability," *IEEE Transactions on Computers*, vol. 48, no. 5, pp. 506–521, May 1999, preliminary version in *ICCAD '96*.

[41] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik, "Chaff: Engineering an efficient SAT solver," in *Proceedings of the 38th Design Automation Conference (DAC '01)*, Jun. 2001, pp. 530–535.

[42] N. Nisan and A. Wigderson, "Rounds in communication complexity revisited," *SIAM Journal on Computing*, vol. 22, no. 1, pp. 211–219, Feb. 1993, preliminary version in *STOC '91*.

[43] J. Nordström, "A simplified way of proving trade-off results for resolution," *Information Processing Letters*, vol. 109, no. 18, pp. 1030–1035, Aug. 2009.

[44] A. Rao and A. Yehudayoff, "Communication complexity," 2016, manuscript in preparation.

[45] R. Raz and P. McKenzie, "Separation of the monotone NC hierarchy," *Combinatorica*, vol. 19, no. 3, pp. 403–435, Mar. 1999, preliminary version in *FOCS '97*.

[46] A. A. Razborov, "Lower bounds for the monotone complexity of some Boolean functions," *Soviet Mathematics Doklady*, vol. 31, no. 2, pp. 354–357, 1985.

[47] R. Santhanam, "Lower bounds on the complexity of recognizing SAT by Turing machines," *Information Processing Letters*, vol. 79, no. 5, pp. 243–247, Sep. 2001.

[48] L. G. Valiant, "Parallelism in comparison problems," *SIAM Journal on Computing*, vol. 4, no. 3, pp. 348–355, Mar. 1975.

[49] D. van Melkebeek, "A survey of lower bounds for satisfiability and related problems," *Foundations and Trends in Theoretical Computer Science*, vol. 2, no. 3, pp. 197–303, Oct. 2007.

[50] R. Williams, "Time-space tradeoffs for counting NP solutions modulo integers," *Computational Complexity*, vol. 17, no. 2, pp. 179–219, May 2008, preliminary version in *CCC '07*.