Proving Unsatisfiability with Hitting Formulas

Yuval Filmus
Technion – Israel Institute of Technology, Haifa, Israel

Edward A. Hirsch
Department of Computer Science, Ariel University, Israel

Artur Riazanov
EPFL, Lausanne, Switzerland

Alexander Smal
Technion – Israel Institute of Technology, Haifa, Israel

Marc Vinyals
University of Auckland, New Zealand

Abstract

A hitting formula is a set of Boolean clauses such that any two of the clauses cannot be simultaneously falsified. Hitting formulas have been studied in many different contexts at least since [45] and, based on experimental evidence, Peitl and Szeider [53] conjectured that unsatisfiable hitting formulas are among the hardest for resolution. Using the fact that hitting formulas are easy to check for satisfiability we make them the foundation of a new static proof system Hitting: a refutation of a CNF in Hitting is an unsatisfiable hitting formula such that each of its clauses is a weakening of a clause of the refuted CNF. Comparing this system to resolution and other proof systems is equivalent to studying the hardness of hitting formulas.

Our first result is that Hitting is quasi-polynomially simulated by tree-like resolution, which means that hitting formulas cannot be exponentially hard for resolution and partially refutes the conjecture of Peitl and Szeider. We show that tree-like resolution and Hitting are quasi-polynomially separated, while for resolution, this question remains open. For a system that is only quasi-polynomially stronger than tree-like resolution, Hitting is surprisingly difficult to *polynomially* simulate in another proof system. Using the ideas of Raz–Shpilka's polynomial identity testing for noncommutative circuits [57] we show that Hitting is p-simulated by Extended Frege, but we conjecture that much more efficient simulations exist. As a byproduct, we show that a number of static (semi)algebraic systems are verifiable in deterministic polynomial time.

We consider multiple extensions of Hitting, and in particular a proof system Hitting($\oplus$) related to the Res($\oplus$) proof system for which no superpolynomial-size lower bounds are known. Hitting($\oplus$) p-simulates the tree-like version of Res($\oplus$) and is at least quasi-polynomially stronger. We show that formulas expressing the non-existence of perfect matchings in the graphs $K_{n,n+2}$ are exponentially hard for Hitting($\oplus$) via a reduction to the partition bound for communication complexity.

See the full version of the paper for the proofs. They are omitted in this Extended Abstract.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases hitting formulas, polynomial identity testing, query complexity

Digital Object Identifier 10.4230/LIPIcs.ITCS.2024.48

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/016

Funding This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 802020-ERC-HARMONIC.
Artur Riazanov: This work was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract number MB22.00026.

Acknowledgements We thank Jan Johannsen, Ilario Bonacina, Oliver Kullmann, and Stefan Szeider for introducing us to the topic; Zachary Chase, Susanna de Rezende, Mika Göös, Amir Shpilka, and Dmitry Sokolov for helpful discussions.

15th Innovations in Theoretical Computer Science Conference (ITCS 2024).
Editor: Venkatesan Guruswami; Article No. 48; pp. 48:1–48:20
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Propositional proof complexity is a well-established area with a number of mathematically rich results. A propositional proof system [19] is formally a deterministic polynomial-time algorithm that verifies candidate proofs of unsatisfiability of propositional formulas in conjunctive normal form. The existence of a proof system that has such polynomial-size refutations for all unsatisfiable formulas is equivalent to **NP** = **co-NP**, and (dis)proving it is out of reach of the currently available methods. Towards this goal, Cook and Reckhow's paper [19] started a program to develop new stronger proof systems that have short proofs for tautologies that are hard for known proof systems and to prove superpolynomial lower bounds for these new systems. The idea is that obtaining new results where our previous techniques fail helps in developing new techniques.

One of the oldest propositional proof systems is the propositional version of resolution (Res) [15, 23] that operates on Boolean clauses (disjunctions of literals treated as sets) and has only a single rule that allows introducing resolvents $\frac{\ell_1 \vee \cdots \vee \ell_k \vee x \quad \ell'_1 \vee \cdots \vee \ell'_m \vee \overline{x}}{\ell_1 \vee \cdots \vee \ell_k \vee \ell'_1 \vee \cdots \vee \ell'_m}$. Superpolynomial lower bounds on the size of a particular case of resolution proofs are known since [62], while exponential lower bounds on general Res proof size were proven by Haken [41] and Urquhart [63] for the pigeonhole principle and handshaking lemma, respectively. Furthermore Res encompasses CDCL algorithms for SAT [8, 54], that are the most successful SAT-solving algorithms to date.

Motivated by the quest of finding hard examples for modern SAT-solvers, Peitl and Szeider [53] experimentally investigated the hardness of *hitting formulas* for resolution. A hitting formula as a mathematical object has been studied under a number of names and in various contexts (a polynomial-time solvable SAT subclass, partitions of the Boolean cube viewed combinatorially, etc.) [45, 24, 48, 49, 38, 37, 50, 53, 29]. A formula $H = \bigwedge_i H_i$ in CNF with clauses H_i is a hitting formula if every pair of clauses cannot be falsified simultaneously (that is, there is a variable that appears in the two clauses with different signs). Equivalently, the sets S_i of truth assignments falsifying clauses H_i are disjoint, thus in an unsatisfiable hitting formula every assignment in $\{0,1\}^n$ is covered exactly once by S_i's. Peitl and Szeider conjectured that hitting formulas might be among the hardest formulas for resolution. Their conjecture was supported by experimental results for formulas with a small number of variables.

One of the reasons why hitting formulas received an abundance of attention is that they are one of the classes of CNFs that are polynomial-time tractable for satisfiability checking (along with e.g. Horn formulas and 2-CNFs) [45]. First, it is straightforward to check whether a CNF formula is hitting: simply enumerate all pairs of clauses and check that they contain some variable with opposite signs. Then the number of satisfying assignments of a hitting formula is $2^n - \sum_i 2^{n-|H_i|}$, where $|H_i|$ is the number of literals in H_i and n is the number of variables in H.

This nice property allows us to think about hitting not only as a class of formulas but as an algorithm to determine satisfiability. For the algorithm to apply to any kind of formulas we need to introduce nondeterminism, and this is best modelled with a proof system. Thus we define a new static proof system based on unsatisfiable hitting formulas. A refutation of an arbitrary CNF F in the Hitting proof system is an unsatisfiable hitting formula such that each of its clauses is a weakening of a clause in F (i.e. a clause of F with extra literals).

By thinking of hitting as a proof system we reinterpret the conjecture of Peitl and Szeider as the following question: is it possible to efficiently formalize the model counting argument above within the Res proof system? Then the question of the hardness of hitting formulas for Res can be phrased in terms of the relative strength of Res and Hitting: can Hitting be separated from Res? That is, can we find formulas that are easy to refute in Hitting and hard to refute in Res? More in general, by relating Hitting to other proof systems we can pinpoint both the hardness of hitting formulas and the ability to formalize Iwama's counting argument in those proof systems.

It turns out that Hitting is tightly connected to the tree-like version of Res (tl-Res), which is exponentially weaker than Res [14]. It encompasses all DPLL algorithms [23, 22], which form the base of multiple (exponential-time) upper bounds for SAT (see, e.g., [21] for a survey). A DPLL algorithm splits the input problem F into subproblems $F|_{x=0}$ and $F|_{x=1}$ for some variable x and applies easy simplification rules.

More precisely, tl-Res quasi-polynomially simulates Hitting (Theorem 3.1), and the simulation cannot be improved to a polynomial one (Theorem 3.8). This partially answers the question "How hard can hitting formulas be for resolution?" raised in [53] in the following way. Not only every hitting formula has proofs of quasi-polynomial size, their unsatisfiability can be decided in quasi-polynomial time by a DPLL algorithm. The simulation also entails that every exponential-size lower bound we already have for tl-Res holds for Hitting, which in particular allows for a separation of Res from Hitting.

Even though the very weak proof system tl-Res is enough to quasi-polynomially simulate Hitting, it is surprisingly difficult to push the upper bound all the way to a polynomial: even though we compare Hitting to a number of known proof systems with different strengths with the hope of obtaining a polynomial simulation, the only system where we can polynomially simulate Hitting is the very powerful Extended Frege (Corollary 4.3). As a byproduct of this result, we prove also that various static (semi)algebraic proof systems (Nullstellensatz, Sherali–Adams, Lovász–Schrijver, Sum-of-Squares) are indeed Cook–Reckhow (deterministically polynomial-time verifiable) proof systems even when we measure the proof size in a succinct way, ignoring the part enforcing Boolean variables. Such distinction can be safely ignored in lower bound results, but in principle ought to be accounted for when constructing upper bounds. Efficient deterministic formal proof verification becomes more and more important because of the increasing interest in algorithms based on sum-of-squares [6, 30].

In more detail, we study the relation between various versions of Hitting and known proof systems such as:

- Res($\oplus$), defined in [44] by analogy with the system Res(Lin) of [58] in the same vein as Krajíček's R($\ldots$) systems [47]. No superpolynomial-size lower bound is known for it, however, [44] proves an exponential bound for its tree-like version. Res($\oplus$) extends Res by allowing clauses to contain affine equations modulo two instead of just literals, and this is the weakest known bounded-depth Frege system with parity gates where we do not know a superpolynomial-size lower bound.

  We prove two separations showing that Hitting is incomparable with tl-Res($\oplus$) (Sect. 3.3, the separation is quasi-polynomial in one direction and exponential in the other direction).
- Nullstellensatz (NS), defined in [7] (where also an exponential-size lower bound was proved), along with its version NSR [27] that uses dual variables ($\overline{x} = 1 - x$ introduced in [2] for PC [18], which is a "dynamic" version of Hilbert's Nullstellensatz that allows generating elements of the ideal step-by-step). An exponential-size lower bound for NSR follows from [16] (see Corollary 5.4).
- Cutting Planes (CP), defined in [20], uses linear inequalities as its proof lines and has two rules: the rule introducing nonnegative linear combinations and the integer rounding rule ($\frac{\sum c_i x_i \geq c}{\sum c_i x_i \geq \lceil c \rceil}$ for integer c_i's).

- Frege, defined in [59, 19], can be thought of as any implicationally complete "textbook" derivation system for propositional logic. Proving superpolynomial lower bounds for it is a long-standing open problem that seems out of reach at the moment.
- Systems augmented by Tseitin's extension rule and its analogues, such as Extended Frege. This rule allows the introduction of new variables denoting some functions of already introduced variables.

Given that known proof systems do not obviously polynomially simulate Hitting, this leaves us with the following question: does augmenting SAT algorithms with the ability to reason about hitting formulas lead to any improvements? Or its counterpart about proof systems, how powerful are proof systems resulting from combining known proof systems with Hitting?

Recall that a DPLL algorithm splits the input problem F into subproblems $F|_{x=0}$ and $F|_{x=1}$. Algorithms that give upper bounds for SAT use more general splittings; in fact one can split over any tautology, that is, consider subproblems $F \wedge G_1, \ldots, F \wedge G_m$, where $G_1 \vee \cdots \vee G_k$ is a tautology. Put in another way, one can split over an unsatisfiable formula $\overline{G_1} \wedge \cdots \wedge \overline{G_k}$ – including unsatisfiable hitting formulas. We use this idea, although in a DAG-like context, to introduce the following generalization of Hitting.

Hitting Res merges Hitting with Res. It uses the weakening rule and also extends the main resolution rule to

$$\frac{C_1 \vee H_1, \ \ldots, \ C_k \vee H_k}{C_1 \vee \cdots \vee C_k}$$

for a hitting formula $H_1 \wedge \cdots \wedge H_k$. It is also p-simulated by Extended Frege (Corollary 4.4).

Other ways in which we can generalize Hitting while keeping with the spirit of the proof system are to allow some leeway in the requirement for the subcubes to form a partition, or in the type of objects that constitute the partition. While at first these may appear to be a mere mathematical curiosity, the connections to Nullstellensatz in the case of Odd Hitting and to the partition bound in the case of Hitting($\oplus$) show that these are natural proof systems.

Hitting[k] strengthens Hitting by allowing to cover a falsifying assignment with at most k sets. Such proofs can be efficiently verified and p-simulated in Extended Frege using the inclusion-exclusion formula and polynomial identity testing (PIT) (Theorem 4.7).

Odd Hitting strengthens Hitting by allowing to cover a falsifying assignment with an odd number of sets. Such proofs also can be efficiently verified and p-simulated in Extended Frege using PIT (Prop. 4.5). This system is equivalent to a certain version of Nullstellensatz, which we discuss in Sect. 5. We prove a lower bound for Odd Hitting (Corollary 5.4) that allows us to separate it from Res.

Hitting($\oplus$) strengthens Hitting by allowing the complements of affine subspaces instead of clauses, that is, a clause can now contain affine equations instead of just literals, and S_i is thus an affine subspace. Such proofs can be verified similarly to Hitting using Gaussian elimination. We prove an exponential-size lower bound for Hitting($\oplus$) (Theorem 6.1) which, additionally, separates it from CP.

A summary of our simulations and separations is depicted in Figure 1, and more precise bounds are stated in Table 1. Now we turn to a more detailed discussion.

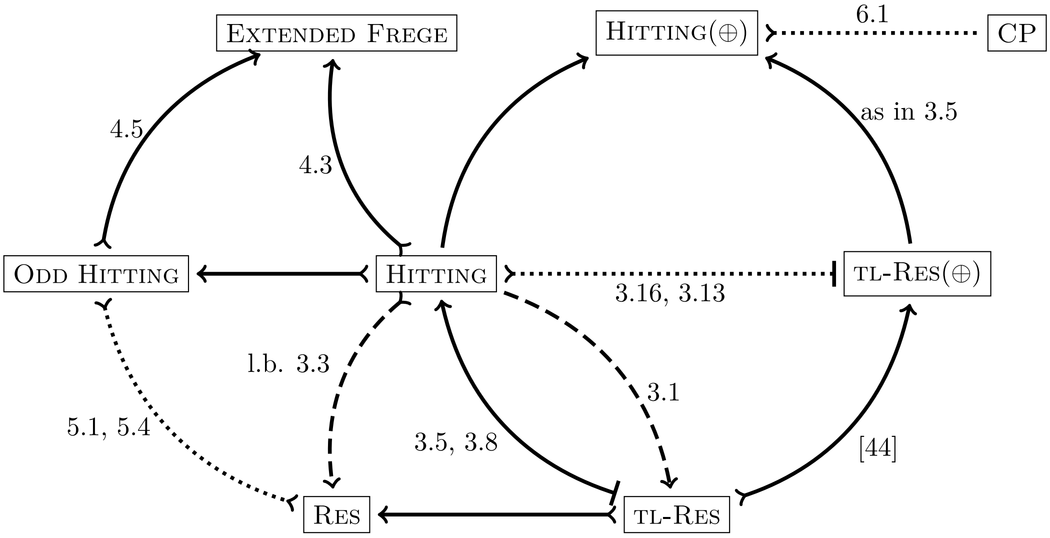

Figure 1 Arrow $\boxed{\text{A}}\longrightarrow\boxed{\text{B}}$ means that B p-simulates A, a dashed arrow $\boxed{\text{A}}\dashrightarrow\boxed{\text{B}}$ means B quasi-polynomially simulates A. $\boxed{\text{A}}\vdash\cdots\cdots\boxed{\text{B}}$ means a quasi-polynomial separation (a lower bound is for the system A). An arrowhead in the tail $\boxed{\text{A}}\succ\cdots\cdots\boxed{\text{B}}$ means that A is exponentially separated from B. A dotted line $\boxed{\text{A}}\cdots\cdots\boxed{\text{B}}$ means that we do not know any simulations between A and B. Known simulations involving CP and Extended Frege are not shown.

Table 1 Precise bounds in our separations. Upper bounds are black and lower bounds are purple.

Thm	Hitting	Hitting(⊕)	Odd Hitting	tl-Res	tl-Res(⊕)	Res	CP
3.8, 3.17	$2^{\tilde{O}(m)}$			$2^{\tilde{\Omega}(m^{2-\varepsilon})}$		$2^{\tilde{O}(m)}$	
3.16	$2^{\tilde{O}(m)}$				$2^{\tilde{\Omega}(m^{2-\varepsilon})}$		
5.4			$2^{\tilde{\Omega}(n)}$			poly	
3.13, [44]	$2^{n^{\Omega(1)}}$				poly	$2^{\Omega(n)}$	
6.1		$2^{\Omega(n)}$					poly

1.1 Our results and methods

1.1.1 Simulations of Hitting-based systems and proof verification using PIT

Proof verification is not straightforward in static (semi)algebraic proof systems that use either dual variables $\bar{x} = 1 - x$ or do not open the parentheses in $(1 - x)$ for the negation of a variable x (such as static Lovász–Schrijver or Sherali–Adams proofs or NS proofs with dual variables). A similar situation occurs with the verification of Hitting proofs which, contrary to most (or all?) known proof systems, is based on model counting. Such reasoning is not expressed naturally in propositional logic, and it makes it difficult to simulate Hitting proofs in other proof systems. We observe that Hitting proofs can be expressed similarly to NS proofs with dual variables without explicitly mentioning the side polynomials for $x^2 - x$ and $x + \bar{x} - 1$ (in particular, we notice that over GF(2), such proofs, which we call *succinct* NSR proofs, are equivalent to Odd Hitting proofs, and that over any field they p-simulate Hitting proofs in a straightforward manner). We show that the two problems have the same cure: we provide an efficient polynomial identity testing procedure for multilinear polynomials modulo $x + \bar{x} - 1$ that can also be formalized in Extended Frege.

Our approach uses the main idea of the Raz–Shpilka polynomial identity testing for noncommutative circuits [57]. We introduce new variables for quadratic polynomials; crucially it suffices to do so for a basis instead of the potentially exponential number of polynomials. This serves as an inductive step cutting the degrees. Namely, at the first step we consider two variables x_1 and x_2 and quadratic polynomials (potentially, $(1-x_1)(1-x_2)$, $(1-x_1)x_2$, $x_1(1-x_2)$, x_1x_2, $1-x_1$, x_1, $1-x_2$, and x_2) appearing in the monomials m_i as $\bar{x}_1\bar{x}_2$, $\bar{x}_1x_2$, etc., and replace them using linear combinations of new variables $y_i^{1,2}$, thus decreasing the degree by one. At the next step we treat all the variables $y_i^{1,2}$ as a single "layer" (note that they are *not* multiplied by each other). We merge this layer of $y_i^{1,2}$ with x_3, getting a layer of variables $y_j^{1,2,3}$, and so on, until we reach a linear equation, which is easy to verify.

In order to implement this strategy we prove a lemma allowing us to merge two layers of variables by ensuring that after the merge the equivalence of polynomials still holds.

By using this polynomial identity testing we get not only an efficient algorithm for checking static proofs (including succinct NSR proofs), but also a polynomial simulation in the Extended Polynomial Calculus (Ext-PC) system that has been recently used in [3], where an exponential-size lower bound has been proved. It is not difficult to see that Ext-PC over GF(2) is equivalent to Extended Frege, so we obtain p-simulations of Hitting, Hitting Res, Odd Hitting and Hitting[k] in Extended Frege.

1.1.2 Separations of Hitting from classical systems

A polynomial simulation of tl-Res in Hitting (Theorem 3.5) can be easily shown by converting tl-Res to a decision tree, then the assignments in the leaves provide a disjoint partition of the Boolean cube. We show a quasi-polynomial simulation in the other direction through careful analysis of a recursive argument (Theorem 3.1). The main idea is that an unsatisfiable formula containing m clauses must necessarily contain a clause of width $w \leq \log_2 m$, and in a hitting formula this clause must contain a variable that occurs with the opposite sign in at least $(m-1)/w$ clauses. Making a decision over this variable thus removes a lot of clauses in one of the two branches. We also employ a generalization of this idea to show that Hitting[k] proofs can be quasi-polynomially simulated in Hitting (Prop. 3.2) and hence in tl-Res.

A polynomial simulation in the other direction is impossible because of a superpolynomial separation. To show this result (Theorem 3.8) we use query complexity, and in particular, the result of [4] separating unambiguous query complexity from randomized query complexity. We lift it using xorification to obtain the desired separation.

We then obtain a two-way separation between Hitting and tl-Res($\oplus$) (Sect. 3.4). On the one hand Tseitin formulas are hard for Res [63] and hence for Hitting. On the other hand, [44] shows that they have polynomial-size tl-Res($\oplus$) (and thus also Hitting($\oplus$)) proofs. In the other direction, similarly to the separation between Hitting and tl-Res, we again use the separation of [4] between unambiguous certificate complexity and randomized query complexity as our starting point. However, since for tl-Res($\oplus$) we are unable to use decision trees, we need to go through randomized communication complexity arguments, using a randomized query-to-communication lifting theorem [40].

Eventually, we discuss separations of Hitting from Res and NS. While the relevant lower bounds for Hitting follow directly from known lower bounds for tl-Res, the other direction seems much more difficult, if possible at all. One natural candidate for a separation result could be the formulas that we used to separate Hitting from tl-Res, but this cannot work because they turn out to have Res proofs of polynomial size (Theorem 3.17). We show this fact using dag-like query complexity [31], the analogue of resolution width in query complexity, which stems from a game characterization of Res [56, 5]. We need to reprove the result of [4] accordingly, improving it to a separation between unambiguous dag-like query complexity and randomized query complexity. This immediately yields an upper bound on the Res width. Concerning NS, it is a simple observation that Hitting is simulated by NS with respect to width vs degree. Furthermore, as we discussed above, succinct NSR proofs (over any field) simulate Hitting with respect to size, therefore separating Hitting from Res would amount to separating explicit vs succinct NSR size.

1.1.3 A lower bound for Odd Hitting

As mentioned above Odd Hitting is polynomially equivalent to succinct NSR proofs over GF(2), and we explain this in more detail in the beginning of Sect. 5. It is easy to see that Odd Hitting has short proofs of Tseitin formulas and thus it is exponentially separated from Res. The opposite direction (Cor. 5.4) requires slightly more effort. It is known that Res width can be separated from NS degree [16]. We use this result to get our size separation using xorification and the random restriction technique of Aleknhovich and Razborov (see [13]).

1.1.4 A lower bound for Hitting($\oplus$)

Our lower bound for Hitting($\oplus$) (Theorem 6.1) uses a communication complexity argument. Communication complexity reductions have a long history of applications in proof complexity [11, 42, 35, 44, 25]. The first step in these reductions is a simulation theorem, which shows that a refutation of an arbitrary CNF ϕ in the proof system of interest can be used to obtain a low-cost communication protocol solving the communication problem Search(ϕ): given an assignment to the variables of ϕ, find a clause of ϕ falsified by this assignment. The second step is reducing a known hard communication problem (usually set disjointness) to Search(ϕ) for a carefully chosen CNF ϕ.

Until recently the applications of these reductions were limited to either proving a lower bound for a tree-like version of the system or proving a size-space tradeoff, neither of which applies to our result. However, over the last few years, the list of applications of the communication approach in proof complexity has grown significantly. A major breakthrough came in [61, 31] with a dag-like lifting theorem from resolution to monotone circuits and cutting plane refutations. Another novel idea was introduced in [39], where the authors derived a lower bound for Nullstellensatz via a communication-like reduction from the $\Omega(\sqrt{n})$ lower bound on the approximate polynomial degree of AND_n [52].

We use yet another twist on this idea: we apply a communication reduction to the *partition bound* [46], a generalization of randomized communication protocols which simulates Hitting($\oplus$). To the best of our knowledge this is the first application of the partition bound in a proof complexity context. We then adapt a communication reduction from set disjointness in [43] so that it works for the partition bound and use the fact that set disjointness is still hard for the partition bound to get our lower bound (Theorem 6.1). The choice of the reduction of [43] is not particularly important, and we believe that reductions from [11, 35, 44] should also work. A nice feature of the reduction we use is that we get a lower bound for a natural combinatorial principle: a formula encoding the non-existence of a perfect matching in a complete bipartite graph $K_{n,n+2}$. Because this formula is known to have short CP proofs, we obtain a separation between Hitting($\oplus$) and CP as an immediate corollary.

1.2 Further research

Relation between HITTING and RES. Although we have gained a lot of understanding of the hardness of hitting formulas for resolution, the initial question of Peitl and Szeider is not fully answered. In particular, we do not know whether hitting formulas can be superpolynomially hard for Res. The negative answer implies a simulation of Hitting by Res. To show the positive answer it is sufficient to separate two query complexity measures: dag-like query complexity (w) and unambiguous certificate complexity (UC). The dag-like query complexity of the falsified clause search problem for a formula F corresponds to the resolution width of F. The unambiguous certificate complexity for this problem corresponds to the width of Hitting refutations of F. Note that unambiguous certificate complexity only makes sense for functions, while dag-like query complexity is defined for (total) relations. Unfortunately, separating even regular certificate complexity (C) and w is an open problem for *functions* (without the uniqueness requirement the certificate complexity can only decrease, so it might be easier to separate w from C than from UC). It turns out that w is resistant to known lower bound techniques in the field of query complexity, so tackling it will likely lead to finding new techniques there. Notice that we know how to separate w and C for *relations*: every lower bound on the resolution width for an $O(1)$-CNF formula constitutes a separation for the corresponding falsified clause search problem. Such a separation (constant vs. polynomial) is unachievable for functions (we cannot hope for better than quadratic separation for functions as $\mathsf{w}(R) \leq \mathsf{C}(R)^2$). Can we use ideas from resolution lower bounds to separate w and UC (or at least C)?

Separate HITTING and HITTING[2]. With xorification this problem can be shown to be equivalent to a simple (if only in the statement!) question in query complexity: separate unambiguous certificate complexity and 2-unambiguous certificate complexity (where every input has one or two certificates). It is known how to separate one-sided versions of these query models [33], but similarly to the question of Hitting vs Res it is unclear how to extend this to the two-sided case.

Is it possible to separate HITTING($\oplus$) and TL-RES($\oplus$)? In the full version of this paper we give evidence that a simulation of Hitting($\oplus$) by tl-Res($\oplus$) along the lines of Theorem 3.1 is not possible. That, however, does not rule out the existence of such a simulation. [60, Conjecture 5.1.3] conjectures that every affine subspace partition can be refined to one corresponding to a parity decision tree with a quasi-polynomial blow-up. With some caveats[1], the statement of this conjecture is equivalent to the existence of quasi-polynomial simulation of Hitting($\oplus$) by tl-Res($\oplus$). So, is there an exponential separation between these two systems? It seems that communication-based lower bounds for tl-Res($\oplus$) can be transferred to Hitting($\oplus$) as it is done in Section 6. There are several other techniques that yield tl-Res($\oplus$) lower bounds such as prover-delayer games [44, 36], reduction to polynomial calculus degree [32], and the recent lifting from decision tree depth to parity decision tree size directly [17, 10]. None of those seem to work for Hitting($\oplus$), so it is reasonable to think that some of the yielded formulas may have an upper bound in Hitting($\oplus$). The most promising technique seems to be lifting since it yields a wide family of formulas hard for tl-Res($\oplus$) with the source of hardness inherent to the tree-like structure of refutations.

[1] The refinement might be non-constructive, but its mere existence does not imply the simulation. The simulation might produce parity decision trees that are not refinements of the initial Hitting($\oplus$) refutation but nevertheless, solve the relation Search(ϕ).

Better upper bound on HITTING. One intriguing matter is that although a very weak proof system such as TL-RES is enough to quasi-polynomially simulate HITTING, we need to go all the way to the very strong proof system EXTENDED FREGE for the simulation to become polynomial. A natural question is then what is the weakest proof system that is enough to polynomially simulate HITTING.

It is consistent with our findings that a fairly weak proof system such as NSR is already enough to simulate HITTING; indeed this would be the case if NSR and succinct NSR were equivalent. Hence we ask the same question regarding succinct (semi)algebraic proof systems: what is the weakest proof system that polynomially simulates succinct NSR or succinct SA? And in particular, is succinct NSR equivalent to NSR and is succinct SA equivalent to SA? One way to answer all these questions would be to formalize the PIT of Theorem 4.2 in a weaker proof system.

The situation with HITTING($\oplus$) is even worse. We have shown how to p-simulate most of the generalizations of HITTING that we defined, including ODD HITTING and HITTING[k], in EXT-PC, but the argument does not work as is for HITTING($\oplus$) since we are relying on a noncommutative PIT. Therefore we do not know even an EXTENDED FREGE simulation of HITTING($\oplus$) (though it is of course quite expected).

Non-automatability of HITTING. It follows from Theorem 3.1 and quasi-polynomial automatability of TL-RES [9] that HITTING is also quasi-polynomially automatable. Can we show that it is impossible to do better? We think that it is possible to adapt the similar result of de Rezende [26] for TL-RES.

2 Basic definitions

2.1 Basic notation

For a function $f : \mathbb{N} \to \mathbb{R}$, $\tilde{O}(f)$ and $\tilde{\Omega}(f)$ denote O and Ω up to logarithmic factors, that is, $g = \tilde{O}(f)$ and $h = \tilde{\Omega}(f)$ if $g = O(f \log^C f)$ and $h = O(f / \log^C f)$ respectively for a constant C. For example, $2^n n^2 = \tilde{O}(2^n)$ and $n/\log n = \tilde{\Omega}(n)$.

Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. The *deterministic query complexity of* f, denoted by $\mathsf{D}(f)$, is the minimal number of (adaptive) queries to the input variables that is enough to compute $f(x)$ for any input x. The *randomized query complexity of* f, $\mathsf{R}(f)$, is the minimum number of queries needed by a randomized algorithm that outputs $f(x)$ for any input x with probability at least $2/3$. A partial assignment $\alpha \in \{0,1,*\}^n$ is a *certificate* for f if for any two assignments $x, y \in \{0,1\}^n$ agreeing with α, $f(x) = f(y)$. The *size* of a certificate is the number of non-star entries. The *certificate complexity of* f *on an input* x, denoted $\mathsf{C}(f,x)$, is size of the smallest certificate α such that x agrees with α. For $b \in \{0,1\}$, the *(one-sided)* b*-certificate complexity of* f is defined as $\mathsf{C}_b(f) = \max_{x : f(x) = b} \mathsf{C}(f,x)$. The *(two-sided) certificate complexity of* f is the maximum of 0- and 1-certificate complexities, $\mathsf{C}(f) = \max\{\mathsf{C}_0(f), \mathsf{C}_1(f)\}$. We say that a family of certificates $A \subset \{0,1,*\}^n$ is *unambiguous* if any two distinct certificates $\alpha, \beta \in A$ conflict, i.e., there is no assignment that agrees with both α and β. For $b \in \{0,1\}$, the *(one-sided) unambiguous* b*-certificate complexity of* f, $\mathsf{UC}_b(f)$, is the minimum number w such that there is an unambiguous family of certificates A such that A contains only certificates of size at most w and every $x \in f^{-1}(b)$ agrees with some certificate in A. The *(two-sided) unambiguous certificate complexity of* f is defined as $\mathsf{UC}(f) = \max\{\mathsf{UC}_0(f), \mathsf{UC}_1(f)\}$.

We use the following notation for widely known proof systems: Res for Resolution, tl-Res for tree-like Resolution, Res($\oplus$) for Resolution over XORs of [44], tl-Res($\oplus$) for its tree-like version, CP for Cutting Planes, NS for Nullstellensatz, PC for Polynomial Calculus, Frege for Frege and Extended Frege for Extended Frege.

Deterministic communication complexity of a search problem defined by a ternary relation R is the minimal amount of communication (number of bits) that is enough to solve the following communication problem for two players on any input: Alice is given x, Bob is given y, and their goal is to find some z such that $(x, y, z) \in R$. Alice and Bob can exchange information by sending bit messages to each other. At the end of the game both players must know z. *(Public coin) ε-error randomized communication complexity* of a search problem is the minimal amount of communication that is enough for players to win the communication game with probability at least $1 - \varepsilon$ if the players have access to a public source of random bits. If ε is not explicitly specified then we assume $\varepsilon = 1/3$. More information on the standard definitions of communication complexity can be found in [51].

2.2 Hitting formulas and proof system

Iwama [45] started to study hitting formulas as a polynomial-time tractable subclass of satisfiability problems (see also [48]).

▶ **Definition 2.1** (Hitting formula). *A* hitting formula *is a formula $F = C_1 \wedge \dots \wedge C_m$ in conjunctive normal form such that every two of its clauses C_i and C_j contain contrary literals, that is, there is some literal ℓ such that $\ell \in C_i$ and $\bar{\ell} \in C_j$; in other words, $C_i \vee C_j$ is a tautology.*

Sometimes the notion is defined for formulas in disjunctive normal form. We call them a different name to avoid misunderstanding.

▶ **Definition 2.2** (Unambiguous DNF). *An* unambiguous DNF *is the negation of a hitting formula, that is, every two its terms (conjunctions) contradict each other.*

▶ **Definition 2.3** (Hitting proof system). *A refutation of a CNF F in* Hitting *is an unsatisfiable hitting formula H such that every clause C in H has a strengthening $C' \subseteq C$ in F.*

Hitting refutations can be verified in polynomial time: the unsatisfiability of H can be easily checked by counting the number of falsifying assignments, as implicitly noticed by Iwama [45] (note that the sets of falsifying assignments for any two clauses of H are disjoint), and matching clauses to their strengthening is done simply by considering all pairs $C \in H$, $C' \in F$.

The soundness of Hitting is trivial, the completeness is given by the "complete" hitting formula consisting of all possible clauses containing all the variables of F: the unique assignment falsifying such a clause C must also falsify some clause C' of (unsatisfiable) F, which is then the required strengthening of C.

2.3 Other Hitting-based proof systems

2.3.1 Hitting Res

Hitting is a "static" proof system with no real derivation procedure. We add more power to it by incorporating such steps into a Res refutation. Indeed, a resolution step can be generalized to resolve over any contradiction, not just $x \wedge \bar{x}$. In Hitting Res we resolve by hitting formulas:

▶ **Definition 2.4** (Hitting Res). *This proof system embraces both* Hitting *and* Res. *One derivation step uses an unsatisfiable hitting formula $H_1 \wedge \cdots \wedge H_k$:*

$$\frac{C_1 \vee H_1,\ \ldots,\ C_k \vee H_k}{C_1 \vee \cdots \vee C_k}.$$

We also allow weakening steps:

$$\frac{C}{C \vee D}.$$

▶ **Proposition 2.5.** Hitting Res *p-simulates both* Hitting *and* Res.

2.3.2 Odd Hitting

While a hitting formula covers every falsifying assignment exactly once, that is, it satisfies exactly one clause, an odd hitting formula does this an odd number of times.

▶ **Definition 2.6** (Odd hitting formula). *An* odd hitting formula *is a formula $F = C_1 \wedge \cdots \wedge C_m$ in conjunctive normal form such that every falsifying assignment falsifies an odd number of its clauses.*

▶ **Definition 2.7** (Odd Hitting proof system). *A refutation of a CNF F in* Odd Hitting *is an unsatisfiable odd hitting formula H such that every clause C in H has a strengthening $C' \subseteq C$ in F.*

It is not straightforward how to verify that a (not necessarily unsatisfiable) formula is an odd hitting formula, and how to verify that a formula is an unsatisfiable odd hitting formula (thus verifying Odd Hitting proofs). We show it in Prop. 4.6 and Prop. 4.5.

2.3.3 Hitting[k]

One can generalize hitting formulas by allowing a falsifying assignment to falsify a limited number of clauses (and not just a single clause) [49].

▶ **Definition 2.8** (Hitting-k formula). *A* hitting-k formula *is a formula F in conjunctive normal form such that every assignment falsifying F falsifies at most k clauses of F.*

▶ **Definition 2.9** (Hitting[k]). *A refutation of a CNF F in* Hitting[k] *is an unsatisfiable hitting-k formula H such that every clause C in H has a strengthening $C' \subseteq C$ in F.*

We show in Theorem 4.7 that Hitting[k] refutations can be verified in polynomial time.

2.3.4 Hitting($\oplus$)

Hitting($\oplus$) stands to Hitting the same way as Res($\oplus$) stands to Res, where Res($\oplus$) is the system defined in [44] that allows clauses to contain affine equations modulo two instead of just literals. It resembles the system Res(Lin) of [58] and falls under the concept of Krajíček's R($\ldots$) systems [47].

▶ **Definition 2.10** (Hitting($\oplus$) formula). *A* hitting($\oplus$) formula *decomposes $\{0,1\}^n$ into disjoint affine subspaces over* GF(2). *Namely, it is a conjunction of $\oplus$-clauses of the form $\bigvee_k c_k \oplus \bigoplus_{i \in I_k} x_i$, where $c_k \in \{0,1\}$ is a constant, x_i's are variables, and any two its $\oplus$-clauses do not share a common falsifying assignment.*

Note that we can check that two affine subspaces are disjoint using Gaussian elimination, and this gives an efficient way of checking whether a given formula is hitting($\oplus$).

$\oplus$-clauses can be thought of as sets of linear (affine) equations similarly to clauses that we usually think of as sets of literals.

▶ **Definition 2.11** (Hitting($\oplus$) proof system). *A refutation of a CNF F in* Hitting($\oplus$) *is an unsatisfiable hitting($\oplus$) formula H such that every $\oplus$-clause C in H has a strengthening $C' \subseteq C$ in F.*

Note that Hitting($\oplus$) can be thought of also as a *proof system for sets of affine subspaces covering* $\{0,1\}^n$, that is, unsatisfiable systems of disjunctions of linear (affine) equations.

3 Hitting vs tl-Res and other classical systems

We prove that while Hitting p-simulates tl-Res, in the other direction tl-Res simulates Hitting only quasi-polynomially. Moreover, tl-Res is quasi-polynomially weaker than Hitting. We also relate Hitting to other proof systems: the tree-like version of Res($\oplus$) (they are incomparable), certain versions of NS, and Res.

3.1 tl-Res quasi-polynomially simulates Hitting

We use a construction of small decision trees from DNF covers of Boolean functions to quasi-polynomially simulate Hitting in tl-Res [28]. We adapt the argument of Ehrenfeucht–Haussler to prove the following theorem. Intuitively, every hitting formula defines a subcube partition of the Boolean cube $\{0,1\}^n$. The structure of this partition can be used to greedily construct a decision tree (tl-Res refutation) that always queries the most conflicting variable in the narrowest clause.

▶ **Theorem 3.1.** *If a CNF formula F has a* Hitting *refutation of size m, then F has a* tl-Res *refutation of size at most $O(2^{2\log^3 m})$.*

The argument can be extended to Hitting[k] simulation by Hitting (hence, by tl-Res).

▶ **Proposition 3.2.** Hitting *quasi-polynomially simulates* Hitting[k] *up to $k = (\log m)^{O(1)}$.*

Later in Theorem 3.8 we show that the simulation of Hitting by tl-Res cannot be polynomial; however, we do not know whether it can be improved to $m^{O(\log m)}$.

▶ **Corollary 3.3.** *There are formulas that have polynomial-size* Res *proofs but require exponential-size* Hitting *proofs.*

▶ Remark 3.4. Similarly to Theorem 3.1, Hitting Res can be quasi-polynomially simulated in Res (every hitting resolution step can be simulated using Theorem 3.1), and thus an exponential-size lower bound for it also follows from exponential-size lower bounds for Res (e.g., [63]).

3.2 Hitting is quasi-polynomially stronger than tl-Res

The simulation is not difficult to see.

▶ **Theorem 3.5.** Hitting *p-simulates* tl-Res.

We use $\oplus$-lifting to prove the separation result. We employ the following separation of randomized query complexity (deterministic is enough for our purpose) and unambiguous certificate complexity from [4].

▶ **Definition 3.6** ([1, 4]). *Let $f\colon \{0,1\}^N \to \{0,1\}$ be a function, $c = 10\log N$ and $m = c \cdot \mathsf{C}(f)\log N = 10\mathsf{C}(f)\log^2 N$. Then the cheat sheet version of f, denoted f_{CS}, is a total function $f_{\mathrm{CS}}\colon (\{0,1\}^N)^c \times (\{0,1\}^m)^{2^c} \to \{0,1\}$.*

Let the input be written as $(x^1, x^2, \ldots, x^c, Y_1, Y_2, \ldots, Y_{2^c})$, where for all $i \in [c]$, $x^i \in \{0,1\}^N$, and for all $j \in [2^c]$, $Y_j \in \{0,1\}^m$. Let $\ell_i = f(x^i)$ and $\ell \in [2^c]$ be the positive integer corresponding to the binary string $\ell_1, \ldots, \ell_c$. Then we define the value of f_{CS} to be 1 if and only if Y_ℓ contains certificates for $f(x^i) = \ell_i$ for all $i \in [c]$.

At first glance, the definition of f_{CS} might look nonconstructive due to the usage of $\mathsf{C}(f)$. However, the theorem of [4] uses an appropriate upper bound on $\mathsf{C}(f)$, which is proved along with the interactive construction of the function.

▶ **Theorem 3.7** ([4, Theorem 5.1]). *Let $f_0 = \mathrm{AND}_n$ and f_k be defined inductively as $f_k := \mathrm{AND}_n \circ (\mathrm{OR}_n \circ f_{k-1})_{\mathrm{CS}}$, where f_k has $O(n^{25^k})$ inputs. Then $\mathsf{R}(f_k) = \tilde{\Omega}(n^{2k+1})$ and $\mathsf{UC}(f_k) = \tilde{O}(n^{k+1})$.*

The unambiguous DNFs from this theorem, lifted by $(\oplus_2)^m$, yield the upper bound.

▶ **Theorem 3.8.** *For every $\varepsilon > 0$, there exists a sequence of unsatifiable hitting formulas G_m containing $2^{\tilde{O}(m)}$ clauses of width at most $\tilde{O}(m)$ such that G_m requires* TL-RES *proof size $2^{\tilde{\Omega}(m^{2-\varepsilon})}$.*

3.3 HITTING and TL-RES(⊕) are incomparable

3.3.1 A hard formula for HITTING

We show that there exist formulas that are easy for TL-RES(⊕) and exponentially hard for HITTING. We recall the separation of TL-RES(⊕) from RES for Tseitin formulas [44].

▶ **Definition 3.9** (Tseitin formulas $T_{G,c}$). *For a constant-degree graph $G = (V, E)$ and a 0/1 vector c of "charges" for the vertices, consider the following linear system in the variables x_e for $e \in E$:*

$$\bigwedge_{v\in V}\left(\bigoplus_{e\ni v} x_e = c_v\right),$$

where $\bigoplus_{v\in V} c_v = 1$. In the corresponding Tseitin formula $T_{G,c}$ in CNF each vertex constraint $\bigoplus_{e\ni v} x_e = c_v$ expands into $2^{\deg v - 1}$ clauses of width $\deg v$.

▶ **Theorem 3.10** ([63]). *There exists a family of constant-degree graphs G_n with n nodes and a family of charge vectors c_n such that Ts_{G_n,c_n} requires* RES *refutation of size $2^{\Omega(n)}$.*

▶ **Theorem 3.11** ([44]). *For any graph G and charges c the Tseitin formula $\mathrm{Ts}_{G,c}$ has a tree-like Res(⊕) refutation of size linear in the size of the CNF.*

Given the quasi-polynomial simulation of Theorem 3.1 and the following generalization of Theorem 3.5, we can separate HITTING from TL-RES(⊕) and HITTING(⊕).

▶ **Proposition 3.12.** *If F has a tree-like* RES(⊕) *refutation of size s, then it has a* HITTING(⊕) *refutation of size s.*

▶ **Corollary 3.13.** *There exists a family of CNF formulas F_n such that F_n requires Resolution refutation of size $2^{\Omega(n)}$, Hitting refutation of size $2^{n^{\Omega(1)}}$ and admits polynomial-size tree-like Res(⊕) refutation (and, consequently, polynomial-size* HITTING(⊕) *refutation).*

3.3.2 A hard formula for tl-Res($\oplus$)

In addition to separating Hitting from tl-Res, we can follow the same plan to separate it from a stronger tl-Res($\oplus$) proof system, that is, to lift a separation between unambiguous certificate complexity and query complexity. We cannot use decision tree size to bound tl-Res($\oplus$) size, but rather the stronger randomized communication complexity measure.

▶ **Theorem 3.14** ([44, Theorem 3.11]). *Let F be an unsatisfiable CNF that has tree-like Res($\oplus$) refutation of size t then the randomized communication complexity of the falsified clause search problem for F is $O(\log t)$.*

The function $\textsc{Indexing}_m \colon [m] \times \{0,1\}^m \to \{0,1\}$ is defined as $\textsc{Indexing}_m(i,x) = x_i$, i.e. it accepts an index and a vector and returns the element of the vector with the given index. Observe that $\textsc{Indexing}_m$ has a decision tree of depth $\lceil \log_2 m \rceil + 1$: we first query the index and then query a single bit of the vector.

▶ **Theorem 3.15** ([40]). *If a function $f \colon \{0,1\}^n \to \{0,1\}$ requires a randomized decision tree of depth t, then the function $f \circ (\textsc{Indexing}_m)^n$ where $m = n^{256}$ requires randomized communication cost $\Omega(t \log n)$.*

By lifting formulas from Theorem 3.7 with $\textsc{Indexing}_{M^{256}}$, we prove the separation result.

▶ **Theorem 3.16.** *For every $\epsilon > 0$, there is a sequence of unsatisfiable hitting formulas G_m containing $2^{\tilde{O}(m)}$ clauses of width $\tilde{O}(m)$ that requires* tl-Res($\oplus$) *proof size $2^{\tilde{\Omega}(m^{2-\epsilon})}$.*

3.4 Relation to Res and NS

As we discussed in Section 3.3.1, a corollary of Theorem 3.1, which shows that tl-Res quasi-polynomially simulates Hitting, is that if a proof system $\mathcal{P}$ is exponentially separated from tl-Res then $\mathcal{P}$ is also exponentially separated from Hitting. Since this is the case with Res and NS – which have short proofs of the ordering principle and the bijective pigeonhole principle [12] respectively, while tl-Res requires exponentially long proofs of both – we conclude that Res and NS are exponentially separated from Hitting.

In this section we explore whether a simulation or separation in the other direction exists. We show that the formula that we used for the quasi-polynomial separation of Hitting from tl-Res has short Res refutations, and therefore cannot be used for showing a separation from Res. We also show that in a sense NS simulates Hitting.

3.4.1 Upper bound in Res

To construct a Res refutation we first reprove the upper bound part of Theorem 3.7 (separating unambiguous certificate complexity from randomized query complexity) strengthening it to an upper bound for unambiguous dag-like query complexity in place of unambiguous certificate complexity. We need to make a few minor changes arising from the fact that $\mathsf{w}(\mathrm{AND}_n) = n$ while $\mathsf{C}_0(\mathrm{AND}_n) = 1$, but using the fact that $\mathsf{w}(\mathrm{OR}_n \circ \mathrm{AND}_n) = O(n)$ and not $\Theta(n^2)$ is enough for our purposes.

▶ **Theorem 3.17.** *The formula G_m of Theorem 3.8 has a* Res *refutation of size $2^{\tilde{O}(m)}$.*

3.4.2 Upper bound in NS

Given a clause $C = \bigvee_{i \in P} x_i \vee \bigvee_{i \in N} \overline{x_i}$, let $p_C = \prod_{i \in P}(1 - x_i) \cdot \prod_{i \in N} x_i$ be the polynomial whose roots are the satisfying assignments of C. Recall that a NS certificate that polynomials $\{p_i\}$ have no common roots is a set of polynomials $\{q_i\}$ such that $\sum p_i q_i \equiv 1$, and the degree of a certificate is $\max_i \deg(p_i q_i)$. A NS refutation of a CNF F is a NS certificate for $\{p_C \mid C \in F\} \cup \{x_i^2 - x_i\}$. It turns out that NS simulates Hitting with respect to degree.

▶ **Proposition 3.18.** NS *degree is at most* HITTING *width.*

When measuring the size of a NS refutation it is more appropriate to consider a definition that allows us to introduce dual variables $\bar{x} = 1 - x$ [2] resulting in a new system NSR [27], since otherwise a formula containing a wide clause with many positive literals would already require exponential size when translated to polynomials. We discuss this system in Sect. 4. Moreover, we discuss *succinct* NSR proofs that contain only side polynomials for the input axioms and not for $x^2 - x = 0$ or $x + \bar{x} - 1 = 0$. In fact, Prop. 3.18 already shows that succinct NSR polynomially simulates HITTING with respect to size.

4 PIT helps to simulate HITTING, and more

4.1 EXTENDED FREGE p-simulates HITTING

We prove that HITTING can be p-simulated at least in the most powerful logical propositional proof system, EXTENDED FREGE. The obstacle is that the soundness of HITTING is based on the counting argument that involves the number of assignments falsified by a clause, and it is not easy to express this argument in propositional logic.

Our strategy is to p-simulate HITTING in a strong algebraic system that is p-equivalent to EXTENDED FREGE in the case of GF(2).

There are several proof systems extending the power of PC by allowing to express polynomials in a more compact way than linear combinations of monomials. Grigoriev and Hirsch [34] introduced $\mathcal{F}$-PC that allows to express polynomials as algebraic formulas without opening the parentheses. Of course, this needs usual associativity–commutativity–distributivity rules to transform these formulas. The next powerful system is EXT-PC considered by Alekseev [3]. This is simply PC with Tseitin's extension rule generalized so that variables can be introduced for arbitrary polynomials. It can be viewed as a way to express PC proofs where polynomials can be represented as algebraic circuits (but transformations of these circuits must be justified using the definitions of extension variables that denote gates values). Eventually, Grochow and Pitassi [55, 35] suggested to generalize proof systems to allow the randomized verification of the proofs, and in these proof systems, one can switch for free between different circuit representations of a polynomial.

A FREGE system [19, §2] is defined as any implicationally complete inference system that uses sound constant-size rule schemata for Boolean formulas (a schema means that the formulas in the rules are represented by meta-variables, for example, F and G in the modus ponens rule $\frac{F;\ F \supset G}{G}$ can be any formulas). An EXTENDED FREGE system additionally allows us to introduce new variables using the axiom schema $x \Leftrightarrow A$ for *any* formula A, where x is a new variable.

Grigoriev and Hirsch [34, Theorem 3] prove that $\mathcal{F}$-PC (over any field), a system that allows us to represent polynomials using arbitrary algebraic formulas and to transform them using the ring rules, p-simulates FREGE (and also a similar statement for constant-depth $\mathcal{F}$-PC over finite fields versus FREGE with modular gates). They also state that FREGE p-simulates $\mathcal{F}$-PC over GF(2) [34, Remark 5]. We include a formal proof of this statement in the full version for completeness. Namely, we prove that $\mathcal{F}$-PC over GF(2) is a FREGE system itself (and it is known that all sound and implicationally complete FREGE systems over all possible sets of Boolean connectives are equivalent [59, Theorem 5.3.1.4.i]).

▶ **Definition 4.1** ([3]). *An* EXT-PC *refutation over a ring R of a set of polynomials $P \subset R[x_1, \ldots, x_n]$ is a* PC *refutation over R of a set of polynomials $P \cup Q$, where $Q := \{y_1 - q_1(x_1, \ldots, x_n), \ldots, y_m - q_m(x_1, \ldots, x_n, y_1, \ldots, y_{m-1})\}$ consists of polynomials defining new variables y_i for arbitary polynomials $q_i \in R[x_1, \ldots, x_n, y_1, \ldots, y_{i-1}]$.*

While [3] defines Ext-PC over arbitrary fields and even rings, we use it over finite fields only. It is not difficult to see that Ext-PC over GF(2) is an Extended Frege system (and it is known that all Extended Frege systems are p-equivalent [59, Theorem 5.3.2.a]).

The main theorem of this section is

▶ **Theorem 4.2.** Ext-PC *over a finite field p-simulates* Hitting*.*

▶ **Corollary 4.3.** Extended Frege *p-simulates* Hitting*.*

▶ **Corollary 4.4.** Extended Frege *p-simulates* Hitting Res*.*

The proof of Theorem 4.2 (which can be found in the full version of this paper) can be used for proving in Ext-PC similar statements about multilinear polynomials that use dual variables. In particular, it can be used for simulating Odd Hitting and Hitting$[k]$.

▶ **Proposition 4.5.** Odd Hitting *proofs can be verified in deterministic polynomial time.* Ext-PC *over* GF(2) *p-simulates* Odd Hitting*. In particular,* Extended Frege *p-simulates* Odd Hitting*.*

This argument allows us to verify *unsatisfiable* odd hitting formulas. However, a similar technique also makes it possible to check arbitrary formulas for being odd hitting.

▶ **Proposition 4.6.** *Given a formula in CNF, it can be checked in deterministic polynomial time whether F is an odd hitting formula.*

▶ **Theorem 4.7.** Hitting$[k]$ *proofs can be verified in deterministic polynomial time.* Ext-PC *over a finite field p-simulates* Hitting$[k]$*. In particular,* Extended Frege *p-simulates* Hitting$[k]$*.*

5 Odd Hitting

Like NS over GF(2), Odd Hitting can efficiently refute Tseitin formulas modulo 2 (see Def. 3.9), which require exponential-size resolution proofs [63].

▶ **Proposition 5.1.** *For any constant-degree graph $G = (V, E)$ and 0/1-vector c,* Odd Hitting *has a polynomial-size refutation of $T_{G,c}$.*

A separation between Odd Hitting and NS without dual variables follows immediately from the separation between NSR and NS of de Rezende et al [27].

In the opposite direction, there are formulas that require exponentially larger proofs in Odd Hitting than in Res. Dmitry Sokolov [private communication] suggested that the well-known technique of xorification can produce an exponential separation between the size of Res and NSR proofs from the bounds of [16]:

▶ **Theorem 5.2** ([16])**.** *There exists a family of formulas that have* Res *proofs of constant width and require* NS *degree $\Omega(n/\log n)$.*

We notice that this technique is still viable for succinct NSR proofs, and hence Odd Hitting. In the following lemma we apply xorification and the random restriction technique of Alekhnovich and Razborov (see [13]) to get the separation.

▶ **Lemma 5.3.** *Let F be a CNF formula that requires degree d to refute in* NS *over a field $\mathbb{F}$. Then $F \circ (\oplus_2)^n$ requires size $2^{\Omega(d)}$ to refute in succinct* NSR *over $\mathbb{F}$.*

Combining xorification with a lower bound on the degree of pebbling formulas we obtain a separation between Odd Hitting and Res.

▶ **Corollary 5.4.** *There exists a family of formulas that have* Res *proofs of polynomial size and require* Odd Hitting *proofs of size $2^{\Omega(n/\log n)}$.*

6 Hitting(⊕)

Hitting(⊕) extends Hitting to formulas that work with linear equations modulo two. We know from Cor. 3.13 that Tseitin formulas separate Hitting(⊕) from Hitting and Res.

We show that perfect matching formulas (that have polynomial-size CP proofs) require exponential-size Hitting(⊕) refutations. In order to do this, we lift them using (binary) xorification and then reduce the question to the known communication complexity lower bound for set disjointness.

▶ **Theorem 6.1.** *Any* Hitting(⊕) *refutation of* PM_G *for the complete bipartite graph* $K_{40n+1,40n+3}$ *contains* $2^{\Omega(n)}$ *many subspaces.*

▶ Remark 6.2. Note that the PM_G formulas for $K_{i,j}$ for $i \neq j$ have polynomial-size CP proofs: it can be easily derived from the 2-clauses that the number of edges around a vertex is at most 1; then take the sum of such inequalities around all vertices in the smaller part, and take the sum of the other input inequalities in the larger part.

References

1 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *STOC-2016*, pages 863–876, 2016. doi:10.1145/2897518.2897644.

2 Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM Journal on Computing*, 31(4):1184–1211, 2002. doi:10.1137/S0097539700366735.

3 Yaroslav Alekseev. A lower bound for polynomial calculus with extension rule. In *36th Computational Complexity Conference*, volume 200 of *LIPIcs. Leibniz Int. Proc. Inform.* Schloss Dagstuhl. Leibniz-Zent. Inform., 2021. Art. 21:18. doi:10.4230/LIPIcs.CCC.2021.21.

4 Andris Ambainis, Martins Kokainis, and Robin Kothari. Nearly optimal separations between communication (or query) complexity and partitions. In *31st Conference on Computational Complexity*, volume 50 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 4, 14. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2016. doi:10.4230/LIPIcs.CCC.2016.4.

5 Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *J. Comput. System Sci.*, 74(3):323–334, 2008. doi:10.1016/j.jcss.2007.06.025.

6 Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *ECCC*, TR14-059, 2014. URL: https://eccc.weizmann.ac.il/report/2014/059, arXiv:TR14-059.

7 P. Beame, R. Impagliazzo, J. Krajíček, T. Pitassi, and P. Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. *Proc. London Math. Soc.*, 73(3):1–26, 1996. doi:10.1112/plms/s3-73.1.1.

8 P. Beame, H. A. Kautz, and A. Sabharwal. Towards understanding and harnessing the potential of clause learning. *J. Artif. Intell. Res.*, 22:319–351, 2004. doi:10.1613/jair.1410.

9 P. Beame and T. Pitassi. Simplified and improved resolution lower bounds. In *37th Conference on Foundations of Computer Science*, pages 274–282, 1996. doi:10.1109/SFCS.1996.548486.

10 Paul Beame and Sajin Koroth. On disperser/lifting properties of the index and inner-product functions, 2022. arXiv:2211.17211.

11 Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász-Schrijver systems and beyond follow from multiparty communication complexity. In *Automata, Languages and Programming*, volume 3580 of *LNCS*, pages 1176–1188. Springer, 2005.

12 Paul Beame and Søren Riis. More on the relative strength of counting principles. In *Proof Complexity and Feasible Arithmetics*, volume 39 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 13–35, 1996. doi:10.1090/dimacs/039/02.

13 Eli Ben-Sasson. Size-space tradeoffs for resolution. *SIAM Journal on Computing*, 38(6):2511–2525, 2009. doi:10.1137/080723880.

14 Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of tree-like and general resolution. *Combinatorica*, 24(4):585–603, 2004. doi:10.1007/s00493-004-0036-5.

15 Archie Blake. *Canonical expressions in Boolean algebra*. PhD thesis, University of Chicago, 1937.

16 Josh Buresh-Oppenheim, Matthew Clegg, Russell Impagliazzo, and Toniann Pitassi. Homogenization and the polynomial calculus. *Comput. Complex.*, 11(3-4):91–108, 2002. doi:10.1007/s00037-002-0171-6.

17 Arkadev Chattopadhyay, Nikhil S. Mande, Swagato Sanyal, and Suhail Sherif. Lifting to parity decision trees via stifling, 2022. arXiv:2211.17214.

18 Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *28th Annual ACM Symposium on the Theory of Computing*, pages 174–183, New York, 1996. ACM. doi:10.1145/237814.237860.

19 Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional proof systems. *The Journal of Symbolic Logic*, 44(1):36–50, 1979. doi:10.2307/2273702.

20 W. Cook, C. R. Coullard, and G. Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18(1):25–38, 1987.

21 E. Dantsin and E. A. Hirsch. Worst-case upper bounds. In A. Biere, M. Heule, H. van Maaren, and T. Walsh, editors, *Handbook of Satisfiability, 2nd Ed.*, volume 336 of *Frontiers in Artificial Intelligence and Applications*, pages 669–692. IOS Press, 2021. doi:10.3233/FAIA200999.

22 M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962.

23 M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.

24 G. Davydov and I. Davydova. Dividing formulas and polynomial classes for satisfiability. In *SAT'98, 2nd Workshop on the Satisfiability Problem*, pages 12–21, 1998.

25 S.F. de Rezende, J. Nordström, and M. Vinyals. How limited interaction hinders real communication (and what it means for proof and circuit complexity). In *57th Annual Symposium on Foundations of Computer Science*, pages 295–304, 2016. doi:10.1109/FOCS.2016.40.

26 Susanna F. de Rezende. Automating tree-like resolution in time $n^{o(\log n)}$ is ETH-hard. *Procedia Computer Science*, 195:152–162, 2021. Proceedings of the XI Latin and American Algorithms, Graphs and Optimization Symposium. doi:10.1016/j.procs.2021.11.021.

27 Susanna F. de Rezende, Massimo Lauria, Jakob Nordström, and Dmitry Sokolov. The power of negative reasoning. In *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 40:1–40:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.40.

28 Andrzej Ehrenfeucht and David Haussler. Learning decision trees from random examples. *Inform. and Comput.*, 82(3):231–246, 1989. doi:10.1016/0890-5401(89)90001-1.

29 Y. Filmus, E. A. Hirsch, S. Kurz, F. Ihringer, A. Ryazanov, A. V. Smal, and M. Vinyals. Irreducible subcube partitions. *Electron. J. Comb.*, 30(3), 2023. doi:10.37236/11862.

30 N. Fleming, P. Kothari, and T. Pitassi. Semialgebraic proofs and efficient algorithm design. *Found. Trends Theor. Comput. Sci.*, 14(1-2):1–221, 2019. doi:10.1561/0400000086.

31 Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. *Theory Comput.*, 16(13):1–30, 2020. doi:10.4086/toc.2020.v016a013.

32 Michal Garlík and Leszek Aleksander Kołodziejczyk. Some subsystems of constant-depth Frege with parity. *ACM Trans. Comput. Logic*, 19(4), November 2018. doi:10.1145/3243126.

33 Mika Göös, Stefan Kiefer, and Weiqiang Yuan. Lower Bounds for Unambiguous Automata via Communication Complexity. In *49th International Colloquium on Automata, Languages, and Programming*, volume 229 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages 126:1–126:13. Schloss Dagstuhl. Leibniz-Zent. Inform., 2022. doi:10.4230/LIPIcs.ICALP.2022.126.

34 Dima Grigoriev and Edward A. Hirsch. Algebraic proof systems over formulas. *Theoret. Comput. Sci.*, 303(1):83–102, 2003. Logic and complexity in computer science (Créteil, 2001). doi:10.1016/S0304-3975(02)00446-2.

35 Joshua A. Grochow and Toniann Pitassi. Circuit complexity, proof complexity, and polynomial identity testing: The ideal proof system. *J. ACM*, 65(6):37:1–37:59, 2018. doi:10.1145/3230742.

36 Svyatoslav Gryaznov. Notes on resolution over linear equations. In *CSR-2019*, volume 11532 of *LNCS*, pages 168–179. Springer, 2019. doi:10.1007/978-3-030-19955-5_15.

37 Matthew Gwynne. *Hierarchies for efficient clausal entailment checking: With applications to satisfiability and knowledge compilation.* PhD thesis, Swansea University, 2014.

38 Matthew Gwynne and Oliver Kullmann. Towards a theory of good SAT representations. *CoRR*, abs/1302.4421, 2013. URL: http://arxiv.org/abs/1302.4421, arXiv:1302.4421.

39 Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Separations in proof complexity and TFNP, 2022. arXiv:2205.02168.

40 Mika Göös, Toniann Pitassi, and Thomas Watson. Query-to-communication lifting for bpp. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 132–143, 2017. doi:10.1109/FOCS.2017.21.

41 Armin Haken. The intractability of resolution. *Theor. Comput. Sci.*, 39:297–308, 1985.

42 Trinh Huynh and Jakob Nordström. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *44th Annual ACM Symposium on Theory of Computing*, pages 233–248, 2012. doi:10.1145/2213977.2214000.

43 Dmitry Itsykson and Artur Riazanov. Proof Complexity of Natural Formulas via Communication Arguments. In *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages 3:1–3:34, Dagstuhl, Germany, 2021. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern. doi:10.4230/LIPIcs.CCC.2021.3.

44 Dmitry Itsykson and Dmitry Sokolov. Resolution over linear equations modulo two. *Ann. Pure Appl. Logic*, 171(1):102722, 31, 2020. doi:10.1016/j.apal.2019.102722.

45 Kazuo Iwama. CNF-satisfiability test by counting and polynomial average time. *SIAM J. Comput.*, 18(2):385–391, 1989. doi:10.1137/0218026.

46 Rahul Jain and Hartmut Klauck. The partition bound for classical communication complexity and query complexity. In *25th Annual IEEE Conference on Computational Complexity*, pages 247–258, 2010. doi:10.1109/CCC.2010.31.

47 Jan Krajíček. Discretely ordered modules as a first-order extension of the cutting planes proof system. *Journal of Symbolic Logic*, 63(4):1582–1596, 1998. doi:10.2307/2586668.

48 Oliver Kullmann. The combinatorics of conflicts between clauses. In Enrico Giunchiglia and Armando Tacchella, editors, *Theory and Applications of Satisfiability Testing (SAT 2003)*, volume 2919 of *LNCS*, pages 426–440. Springer, 2004. doi:10.1007/978-3-540-24605-3_32.

49 Oliver Kullmann. Constraint satisfaction problems in clausal form II: minimal unsatisfiability and conflict structure. *Fundam. Informaticae*, 109(1):83–119, 2011. doi:10.3233/FI-2011-429.

50 Oliver Kullmann and Xishun Zhao. On Davis-Putnam reductions for minimally unsatisfiable clause-sets. *Theoret. Comput. Sci.*, 492:70–87, 2013. doi:10.1016/j.tcs.2013.04.020.

51 Eyal Kushilevitz and Noam Nisan. *Communication complexity.* CUP, 1997.

52 Noam Nisan and Mario Szegedy. On the degree of boolean functions as real polynomials. *Computational Complexity*, 4, 1995. doi:10.1145/129712.129757.

53 Tomás Peitl and Stefan Szeider. Are hitting formulas hard for resolution? *CoRR*, abs/2206.15225, 2022. doi:10.48550/arXiv.2206.15225.

54 Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning SAT solvers as resolution engines. *Artif. Intell.*, 175(2):512–525, 2011. doi:10.1016/j.artint.2010.10.002.

55 Toniann Pitassi. Algebraic propositional proof systems. In *Descriptive Complexity and Finite Models*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 215–244. DIMACS/AMS, 1996. doi:10.1090/dimacs/031/07.

56 Pavel Pudlák. Proofs as games. *Am. Math. Mon.*, 107(6):541–550, 2000. URL: http://www.jstor.org/stable/2589349.

57 Ran Raz and Amir Shpilka. Deterministic polynomial identity testing in non-commutative models. *Comput. Complexity*, 14(1):1–19, 2005. doi:10.1007/s00037-005-0188-8.

58 Ran Raz and Iddo Tzameret. Resolution over linear equations and multilinear proofs. *Ann. Pure Appl. Logic*, 155(3):194–224, 2008. doi:10.1016/j.apal.2008.04.001.

59 R. A. Reckhow. *On the Lengths of Proofs in the Propositional Calculus*. PhD thesis, University of Toronto, 1976. URL: https://www.cs.toronto.edu/~sacook/homepage/reckhow_thesis.pdf.

60 Suhail Sherif. *Communication Complexity and Quantum Optimization Lower Bounds via Query Complexity*. PhD thesis, Tata Institute of Fundamental Research, Mumbai, 2021.

61 Dmitry Sokolov. Dag-like communication and its applications. In *CSR-2017*, volume 10304 of *LNCS*, pages 294–307. Springer, 2017. doi:10.1007/978-3-319-58747-9_26.

62 G. S. Tseitin. On the complexity of derivation in the propositional calculus. *Zapiski nauchnykh seminarov LOMI*, 8:234–259, 1968. Translation: Consultants Bureau, N.Y., 1970, pp. 115–125.

63 Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987. doi:10.1145/7531.8928.