# Satisfiability Allows No Nontrivial Sparsification unless the Polynomial-Time Hierarchy Collapses

HOLGER DELL, LIAFA, Université Paris Diderot, France
DIETER VAN MELKEBEEK, University of Wisconsin-Madison

Consider the following two-player communication process to decide a language $L$: The first player holds the entire input $x$ but is polynomially bounded; the second player is computationally unbounded but does not know any part of $x$; their goal is to decide cooperatively whether $x$ belongs to $L$ at small cost, where the cost measure is the number of bits of communication from the first player to the second player.

For any integer $d \geq 3$ and positive real $\epsilon$, we show that, if satisfiability for $n$-variable $d$-CNF formulas has a protocol of cost $O(n^{d-\epsilon})$, then coNP is in NP/poly, which implies that the polynomial-time hierarchy collapses to its third level. The result even holds when the first player is conondeterministic, and is tight as there exists a trivial protocol for $\epsilon = 0$. Under the hypothesis that coNP is not in NP/poly, our result implies tight lower bounds for parameters of interest in several areas, namely sparsification, kernelization in parameterized complexity, lossy compression, and probabilistically checkable proofs.

By reduction, similar results hold for other NP-complete problems. For the vertex cover problem on $n$-vertex $d$-uniform hypergraphs, this statement holds for any integer $d \geq 2$. The case $d = 2$ implies that no NP-hard vertex deletion problem based on a graph property that is inherited by subgraphs can have kernels consisting of $O(k^{2-\epsilon})$ edges unless coNP is in NP/poly, where $k$ denotes the size of the deletion set. Kernels consisting of $O(k^2)$ edges are known for several problems in the class, including vertex cover, feedback vertex set, and bounded-degree deletion.

Categories and Subject Descriptors: F.1.3 [**Computation by Abstract Devices**]: Complexity Measures and Classes—*Relations among complexity classes*

General Terms: Algorithms, Theory

Additional Key Words and Phrases: Sparsification, kernelization, parameterized complexity, probabilistically checkable proofs, satisfiability, vertex cover, feedback vertex set, hereditary graph properties, vertex deletion problems, arithmetic progression free sets

## 1. INTRODUCTION

Satisfiability of Boolean formulas constitutes one of the most central problems in computer science. It has attracted a lot of applied and theoretical research because of its immediate relevance in areas like AI and verification and as the seminal NP-complete

problem. Of particular interest is $d$-SAT, the satisfiability problem for $d$-CNF formulas, which is NP-complete for any integer $d \geq 3$ [Cook 1971; Karp 1972; Levin 1973].

In this article, we investigate the complexity of $d$-SAT and other NP-complete problems in a communication setting that captures several transformations studied in the theory of computing. Assuming the polynomial-time hierarchy does not collapse, we show that a trivial communication protocol is essentially optimal for $d$-SAT. Under the same hypothesis, the result implies tight lower bounds for parameters of interest in several areas. We first discuss those areas and then state our result for $d$-SAT.

*Sparsification.* The satisfiability of $d$-CNF formulas chosen by uniformly at random picking $m$ clauses out of all possible clauses on $n$ variables seems to exhibit a phase transition as a function of the ratio $m/n$. We know that the probability of satisfiability jumps from almost zero to almost one when the ratio $m/n$ crosses a very narrow region around $2^d \ln 2$, and the existence of a single threshold point is conjectured [Achlioptas and Moore 2007; Achlioptas and Peres 2004; Friedgut and Bourgain 1999]. Experiments also suggest that known SAT solvers have the hardest time on randomly generated instances when the ratio $m/n$ lies around the threshold, and in some cases, rigorous analyses corroborate the experiments.

Nevertheless, from a complexity-theoretic perspective these results fall short of establishing sparse formulas as the hardest instances. This is because formulas that express problems like breaking random RSA instances exhibit a lot of structure and therefore have a negligible contribution to the uniform distribution. An interesting complexity-theoretic formalization would be a reduction from arbitrary formulas to formulas on the same number of variables that are sparse. Impagliazzo et al. [2001] developed such reductions but they run in subexponential time. In polynomial time, we can trivially reduce a $d$-CNF formula to one with $m = O(n^d)$ clauses. Since there are only $2^d \cdot \binom{n}{d} = O(n^d)$ distinct $d$-clauses on $n$ variables, it suffices to remove duplicate clauses. Is there a polynomial-time reduction that maps a $d$-CNF formula on $n$ variables to one on $n$ variables and $m = O(n^{d-\epsilon})$ clauses for some positive constant $\epsilon$?

*Kernelization.* Parameterized complexity investigates the computational difficulty of problems as a function of the input size and an additional natural parameter, $k$, which often only takes small values in instances of practical interest. A good example – and one we will return to soon – is deciding whether a given graph has a vertex cover of size at most $k$. The holy grail in parameterized complexity are algorithms with running times of the form $O(f(k) \cdot s^c)$ on instances of size $s$ and parameter $k$, where $f$ denotes an arbitrary computable function and $c$ a constant. Kernelization constitutes an important technique for realizing such running times: Reduce in time polynomial in $s$ to an instance of size bounded by some computable function $g$ of the parameter $k$ only, and then run a brute-force algorithm on the reduced instance; the resulting algorithm has a running time of the form $O(s^c + f(k))$. In order to obtain good parameterized algorithms, the functions $f$ and $g$ should not behave too badly, which justifies the quest for kernels of polynomial or smaller size $g(k)$.

The number of variables $n$ forms a natural parameter for satisfiability. In the case of $d$-CNF formulas, $n$ is effectively polynomially related to the size of the input, which makes the existence of kernels of polynomial size trivial. Nevertheless, the quest for a small kernel is a relaxation of the quest for sparsification in polynomial time. Eliminating duplicate clauses yields a kernel of bitlength $O(n^d \log n)$. Does satisfiability of $n$-variable $d$-CNF formulas have kernels of size $O(n^{d-\epsilon})$?

*Lossy Compression.* Harnik and Naor [2010] introduced a notion of compression with the goal of succinctly storing instances of computational problems for resolution in the

future, where there may be more time and more computational power available. The compressed version need not be an instance of the original problem, and the original instance need not be recoverable from the compressed version. The only requirement is that the solution be preserved. In the case of decision problems, this simply means the yes/no answer. In an analogy to image compression, one can think of the Harnik-Naor notion of compression as a "lossy compression", where the only aspect of the scenery that is guaranteed not to be lost is the solution to the problem.

Harnik and Naor [2010] applied their notion to languages in NP and showed the relevance to problems in cryptography when the compression is measured as a function of the bitlength of the underlying witnesses. In the case of satisfiability, the latter coincides with the number of variables of the formula. This way lossy compression becomes a relaxation of the notion of kernelization – we now want a polynomial-time mapping reduction to *any* problem, rather than to the original problem, such that the reduced instances have small bitlength as a function of $n$. For $d$-CNF formulas, bitlength $O(n^d)$ is trivially achievable – simply map to the characteristic vector that for each possible $d$-clause on $n$ variables indicates whether it is present in the given formula. Can we lossily compress to instances of bitlength $O(n^{d-\epsilon})$?

*Probabilistically Checkable Proofs.* A somewhat different question deals with the size of probabilistically checkable proofs (PCPs). A PCP for a language $L$ is a randomized proof system in which the verifier only needs to read a constant number of bits of the proof in order to verify that a given input $x$ belongs to $L$. Completeness requires that, for every input $x$ in $L$, there exists a proof which the verifier accepts with probability one. Soundness requires that, for any input $x$ outside of $L$, no proof can be accepted with probability above some constant threshold less than one. For satisfiability of Boolean formulas, Dinur [2007] constructed PCPs of bitlength $O(s \cdot \text{poly} \log s)$, where $s$ denotes the size of the formula. For $d$-CNF formulas on $n$ variables, Dinur's construction yields PCPs of bitlength $O(n^d \cdot \text{poly} \log n)$. On the other hand, standard proofs only contain $n$ bits. Do $n$-variable $d$-CNF formulas have PCPs of bitlength $O(n^{d-\epsilon})$?

## 1.1. Our Results for Satisfiability

We give evidence that the answer to all four of these questions is negative: If any answer is positive, then coNP is in NP/poly. The latter is considered unlikely as it means the existence of a nonuniform polynomial-time proof system for tautologies, or equivalently, that coNP has polynomial-size nondeterministic circuits, and implies that the polynomial-time hierarchy collapses to its third level [Yap 1983].

We obtain those statements as corollaries to a more general result, in which we consider the following communication process to decide a language $L$.

*Definition* 1 (*Oracle Communication Protocol*). An *oracle communication protocol* for a language $L$ is a communication protocol between two players. The first player is given the input $x$ and has to run in time polynomial in the length of the input; the second player is computationally unbounded but is not given any part of $x$. At the end of the protocol, the first player should be able to decide whether $x \in L$. The cost of the protocol is the number of bits of communication from the first player to the second player.

We often refer to the second player as the oracle. Note that the bits sent by the oracle do not contribute towards the cost. By default the players in an oracle communication protocol are deterministic, but one can consider variants in which one or both players are randomized, nondeterministic, etc.

Satisfiability of $n$-variable $d$-CNF formulas has a trivial protocol of cost $O(n^d)$. The following result implies that there is no protocol of cost $O(n^{d-\epsilon})$ unless the polynomial-time hierarchy collapses. In fact, the result even holds when the first player is conondeterministic, that is, when the first player can have multiple valid moves to choose from in any given step, possibly leading to different conclusions about the satisfiability of a given input formula $\varphi$, but such that (i) if $\varphi$ is satisfiable then every valid execution comes to that conclusion, and (ii) if $\varphi$ is not satisfiable then at least one valid execution comes to that conclusion.

THEOREM 1. *Let $d \geq 3$ be an integer and $\epsilon$ a positive real. If* coNP $\not\subseteq$ NP/poly, *there is no protocol of cost $O(n^{d-\epsilon})$ to decide whether an $n$-variable $d$-CNF formula is satisfiable, even when the first player is conondeterministic.*

The corollaries about sparsification, kernelization, and lossy compression follow by considering deterministic single-round protocols in which the polynomial-time player acts as a mapping reduction, sends the reduced instance to the computationally unbounded player, and the latter answers this query as a membership oracle. The corollary about probabilistically checkable proofs follows by considering a similar single-round protocol in which the first player is conondeterministic. Note that Theorem 1 can handle more general reductions, in which multiple queries are made to the oracle over multiple rounds. These corollaries can be strengthened correspondingly. In fact, Theorem 1 is even more general as it allows the oracle to play a more active role that goes beyond answering queries from the polynomial-time player. We will discuss this potential further in the article.

## 1.2. Our Results for Other NP-Complete Problems

By reducibility the lower bounds from Theorem 1 carry over to other parameterized NP-complete problems, where the tightness depends on how the reduction affects the parameterization. In fact, we derive Theorem 1 from a similar result for the vertex cover problem on $d$-uniform hypergraphs.

THEOREM 2. *Let $d \geq 2$ be an integer and $\epsilon$ a positive real. If* coNP $\not\subseteq$ NP/poly, *there is no protocol of cost $O(n^{d-\epsilon})$ to decide whether a $d$-uniform hypergraph on $n$ vertices has a vertex cover of at most $k$ vertices, even when the first player is conondeterministic.*

The cases of Theorem 2 with $d \geq 3$ are equivalent to the corresponding cases of Theorem 1. Note, though, that Theorem 2 also holds for $d = 2$, that is, for standard graphs.

Similar to Theorem 1, Theorem 2 can be interpreted in terms of (graph) sparsification, kernelization, lossy compression, and probabilistically checkable proofs. Regarding kernelization, Theorem 2 has an interesting implication for the vertex cover problem parameterized by the size of the vertex cover – one of the prime examples of a parameterized problem that is NP-hard but fixed-parameter tractable. Kernelizations for this problem have received considerable attention. For standard graphs, S. Buss [Buss and Goldsmith 1993] came up with a kernelization before the term was coined. He observed that any vertex of degree larger than $k$ must be contained in any vertex cover of size $k$, should it exist. This gives rise to a kernelization with $O(k^2)$ vertices and $O(k^2)$ edges. Subsequently, several researchers tried to reduce the size of the kernel. Various approaches based on matching, linear programming, and crown reductions (see Guo and Niedermeier [2007] for a survey) led to kernels with $O(k)$ vertices, but the resulting kernels are all dense. It remains open to find kernels with $O(k^{2-\epsilon})$ edges. Since $k \leq n$, the case $d = 2$ of Theorem 2 implies that such kernels do not exist unless the polynomial-time hierarchy collapses.

In fact, a similar result holds for a wide class of problems known as vertex deletion problems. For a fixed graph property $\Pi$, the corresponding vertex deletion problem asks whether removing at most $k$ vertices from a given graph $G$ can yield a graph that satisfies $\Pi$. A host of well-studied specific problems can be cast as the vertex deletion problem corresponding to some graph property $\Pi$ that is inherited by subgraphs. Examples besides the vertex cover problem include the feedback vertex set problem and the bounded-degree deletion problem (see Section 5 for the definitions of these problems and for more examples).

If only finitely many graphs satisfy $\Pi$ or if all graphs satisfy $\Pi$, the vertex deletion problem is trivially decidable in polynomial time. For all other graph properties $\Pi$ that are inherited by subgraphs, Lewis and Yannakakis [1980] showed that the problem is NP-hard.[1] They did so by constructing a mapping reduction from the vertex cover problem. By improving their reduction such that it preserves the size of the deletion set up to a constant factor, we obtain the following result.

THEOREM 3. *Let $\Pi$ be a graph property that is inherited by subgraphs, and is satisfied by infinitely many but not all graphs. Let $\epsilon$ be a positive real. If* coNP $\not\subseteq$ NP/poly, *there is no protocol of cost $O(k^{2-\epsilon})$ for deciding whether a graph satisfying $\Pi$ can be obtained from a given graph by removing at most $k$ vertices, even when the first player is conondeterministic.*

Theorem 3 implies that problems like feedback vertex set and bounded-degree deletion do not have kernels consisting of $O(k^{2-\epsilon})$ edges unless the polynomial-time hierarchy collapses. For both problems, the result is tight in the sense that kernels with $O(k^2)$ edges exist. For feedback vertex set, we argue that a recent kernelization by Thomassé [2010] does the job; for bounded-degree deletion, kernels with $O(k^2)$ edges were known to exist [Fellows et al. 2011].

## 1.3. Techniques and Related Work

At a high level, our approach refines the framework developed by Bodlaender et al. [2009a] to show that certain parameterized NP-hard problems are unlikely to have kernels of polynomial size. Harnik and Naor [2010] realized the connection between their notion of lossy compression and kernelization and PCPs for satisfiability of general Boolean formulas, and Fortnow and Santhanam [2011] proved the connection with the hypothesis coNP $\not\subseteq$ NP/poly in the superpolynomial setting. Several authors subsequently applied the framework in that setting [Binkele-Raible et al. 2012; Bodlaender et al. 2009b; Chen et al. 2007; Dom et al. 2009; Kratsch and Wahlström 2010, 2013].

We develop the first application of the framework in the polynomial setting, that is, to problems that *do* have kernels of polynomial size, or more generally, oracle communication protocols of polynomial cost. Under the same hypothesis, we show that problems like $d$-SAT and vertex cover do not have protocols of polynomial cost of degree less than the best known. The following lemma captures the refinement of the framework we use in order to do so. The statement refers to the OR operator on languages. For a given language $L$, the language OR($L$) consists of all tuples $(x_1, \ldots, x_t)$ such that $x_i \in L$ for at least one $i \in [t]$.

LEMMA 1. *Suppose that a parameterized problem $\Pi$ has the following property for some constant c: For some NP-complete language L, there exists a polynomial-time mapping reduction from* OR($L$) *to $\Pi$ that maps an instance $(x_1, \ldots, x_t)$ of* OR($L$) *in*

---

[1]In fact, Lewis and Yannakakis [1980] showed this to be the case even for graph properties that are inherited by *induced* subgraphs only.

*which each $x_i$ has size at most $s$ to an instance of $\Pi$ with parameter $k \leq t^{1/c+o(1)} \cdot \mathrm{poly}(s)$. Then, $\Pi$ does not have a communication protocol of cost $O(k^{c-\epsilon})$ for any constant $\epsilon > 0$ unless* coNP $\subseteq$ NP/poly, *even when the first player is conondeterministic.*

In particular, the conclusion of Lemma 1 implies that $\Pi$ does not have kernels of size $O(k^{c-\epsilon})$ for any constant $\epsilon > 0$ unless coNP $\subseteq$ NP/poly.

In order to obtain tight results, we need to construct the reductions in the hypothesis of Lemma 1 for optimal values of $c$. Our main result, Theorem 2, deals with the vertex cover problem on $d$-uniform hypergraphs, or equivalently, with the clique problem on such graphs, parameterized by $k = n$, the number of vertices. Since the clique problem on $d$-uniform hypergraphs is NP-complete for any integer $d \geq 2$, without loss of generality we can take $L$ in Lemma 1 to be this language. Thus, we need to reduce the question whether at least one of $t$ given graphs has a clique of a given size into a single instance of the clique problem on a $d$-uniform hypergraph with few vertices $n$.

As observed by Harnik and Naor [2010], the disjoint union of the given hypergraphs provides such a reduction. However, the number of vertices is $n = s \cdot t$, which does not allow us to handle any value $c > 1$. As a critical piece in our proof, we present a reduction that only needs $n = s \cdot t^{1/d+o(1)}$ vertices, where $d$ is the size of the hyperedges in the given hypergraphs. This way, we can handle any $c$ up to $c = d$.

Our reduction hinges on a graph packing that is based on high-density subsets of the integers without nontrivial arithmetic progressions of length three. After we developed our construction, we have learned about other applications of those sets in the theory of computing, including three-party communication protocols [Chandra et al. 1983], the asymptotically fastest algorithms for matrix multiplication [Coppersmith and Winograd 1990; Davie and Stothers 2013; Stothers 2010; Vassilevska Williams 2012], the soundness analysis of graph tests for linearity [Håstad and Wigderson 2003], and lower bounds for property testing [Alon 2002; Alon et al. 2000, 2008; Alon and Shapira 2004, 2005, 2006]. The latter two applications as well as ours implicitly or explicitly rely on a connection due to Ruzsa and Szemerédi [1978] between these subsets and dense three-partite graphs whose edges partition into triangles and that contain no other triangles. The graph packing we develop is most akin to a construction by Alon and Shapira [2005] in the context of property testing. We refer to Section 4 for a more detailed discussion of the relationships.

The core argument in the proof of Lemma 1 shows that whenever OR($L$) has a cheap protocol, the complement of $L$ has short witnesses that can be verified efficiently with the help of a polynomial-size advice string. We refer to the formal statement as the Complementary Witness Lemma. Its proof involves a refined analysis and generalization of a result by Fortnow and Santhanam [2011] that establishes the case where the protocol implements a mapping reduction to instances of bitlength bounded by some fixed polynomial in $s$. We analyze what happens for mapping reductions without the latter restriction. We also observe that the argument generalizes to our oracle communication protocol setting. Our applications of Theorem 1 only use oracle communication protocols that implement mapping or general reductions. However, the setting of oracle communication protocols is more natural and allows us to prove known results in a simpler way. We refer to Section 6 for more details. In recent subsequent work, the more general setting was critically used to establish a tight kernel lower bound [Kratsch et al. 2014].

Another aspect of our techniques (and Lemma 1 in particular) that could be exploited further, is the fact that the first player in the protocols can be conondeterministic. We exploit it to derive our tight lower bounds for the size of probabilistically checkable proofs. Subsequently it has been used to show kernel lower bounds for a Ramsey-type problem [Kratsch 2012].

Also subsequent to our work but using different techniques and under a seemingly slightly weaker hypothesis than coNP $\not\subseteq$ NP/poly, Drucker [2012] was able to extend our lower bounds for NP-hard problems to randomized compression algorithms with two-sided error. Furthermore, his results also apply to compressive reductions from AND($L$) instead of OR($L$). Lower bounds for general randomized oracle communication protocols with two-sided error remain open.

## 1.4. Organization

We review some preliminaries in Section 2. Section 3 contains the proof of the main result (Theorem 2) following the approach outlined above, modulo the packing construction and the Complementary Witness Lemma. We develop the former in Section 4, and the latter in Section 6. Section 5 expands on the implications for satisfiability (Theorem 1 and its corollaries) and for vertex deletion problems (Theorem 3). We conclude with some open problems in Section 7.

## 2. PRELIMINARIES

Most of our notation is standard (see Arora and Barak [2009] and Goldreich [2008] for general and Downey and Fellows [1999], Flum and Grohe [2006], and Niedermeier [2006] for parameterized complexity). We suffice with a review of some particular notions and notation we use.

*Problems.* By a problem, we usually mean a decision problem, that is, deciding membership to a language $L \subseteq \{0, 1\}^*$. Apart from their bitlength $|x|$, instances $x \in \{0, 1\}^*$ often have another natural complexity parameter $k(x)$, such as the number of vertices in the case of graph problems, or the witness length in the case of NP-problems. The function $k : \{0, 1\}^* \to \mathbb{N}$ is called *parameterization* and a *parameterized problem* is a pair $(L, k)$. We often write $L$ for both the parameterized and unparameterized problem, for example, when saying that a parameterized problem is NP-complete.

We denote the *complement* of $L$ by $\overline{L}$. The OR *of a language L* is the language OR($L$) that consists of all tuples $(x_1, \ldots, x_t)$ for which there is an $i \in [t]$ with $x_i \in L$.

*Satisfiability.* A $d$-CNF formula on the variables $x_1, \ldots, x_n$ is a conjunction of clauses where a clause is a disjunction of exactly $d$ literals, that is, the variables $x_i$ and their negations $\overline{x_i}$. We denote by $d$-SAT the problem of deciding whether a given $d$-CNF formula has at least one satisfying assignment, that is, a truth assignment to its variables that makes the formula evaluate to true.

*Hypergraph Problems.* A hypergraph $G = (V(G), E(G))$ consists of a finite set $V(G)$ of vertices and a set $E(G)$ of subsets of $V(G)$, the (hyper)edges. A hypergraph is $d$-uniform if every edge has size exactly $d$. A *vertex cover of G* is a set $S \subseteq V(G)$ that contains at least one vertex from every edge of $G$, and $d$-VERTEX COVER is the problem of deciding whether, for a given $d$-uniform hypergraph $G$ and integer $k$, there exists a vertex cover of $G$ of size at most $k$. Similarly, a *clique of G* is a set $S \subseteq V(G)$ all of whose subsets of size $d$ are edges of $G$, and $d$-CLIQUE is the problem of deciding whether, for given $(G, k)$, there exists a clique of $G$ of size at least $k$. The two problems are dual to each other, in the sense that $\overline{G}$, the $d$-uniform hypergraph obtained from $G$ by flipping the presence of all edges of size $d$, has a clique of size $k$ if and only if $G$ has a vertex cover of size $n-k$. Note that this transformation preserves the number of vertices.

*Reductions.* Unless stated otherwise, the reductions we consider are computable in time polynomial in the bitlength of the input. We indicate this by a superscript $p$ in the notation $\leq^p$ for reducibility. We consider both general reductions (also known as Turing reductions) as well as mapping reductions (also known as many-one reductions). A

*mapping reduction*, or $\leq_m^p$-*reduction*, from $L$ to $L'$ is a mapping $R$ from $\{0,1\}^*$ to $\{0,1\}^*$ such that $R(x) \in L'$ if and only if $x \in L$.

A *kernelization* of a parameterized problem $(L, k)$ is a $\leq_m^p$-reduction from $L$ to itself that maps instances with parameter $k$ to instances of bitlength at most $g(k)$ for some function $g$ independent of the input size. Note that any parameterized NP-problem that has a kernelization is fixed-parameter tractable, that is, it can be solved in deterministic time $f(k) \cdot \text{poly}(n)$ for some computable function $f$: The reduced instance has size at most $g(k)$ and can be solved in some time $f(k)$ by exhaustively testing all possible NP-witnesses.

*Complexity Classes.* The polynomial-time hierarchy PH is the union $\cup_{i \geq 0} \Sigma_i^p$, where $\Sigma_0^p = \text{P}$, and $\Sigma_{i+1}^p = \text{NP}^{\Sigma_i^p}$ for $i \geq 0$. We say that the polynomial-time hierarchy collapses to its $i$th level if $\text{PH} = \Sigma_i^p$. It is widely conjectured that the polynomial-time hierarchy does not collapse to any level.

Given a class $\mathcal{C}$ of languages, we denote by co$\mathcal{C}$ the class $\{\overline{L} \,|\, L \in \mathcal{C}\}$. Apart from the first few levels of the polynomial-time hierarchy and their co-classes, we make use of complexity classes with advice. Given a class $\mathcal{C}$ of languages and a function $\ell : \mathbb{N} \to \mathbb{N}$, we denote by $\mathcal{C}/\ell(n)$ the class of languages $L$ for which there exists a language $L' \in \mathcal{C}$ and a sequence $a_0, a_1, a_2, \ldots$ of strings with $|a_n| \leq \ell(n)$ such that for any input $x$, we have that $x \in L$ if and only if $\langle x, a_{|x|} \rangle \in L'$, where $\langle \cdot, \cdot \rangle$ denotes a standard pairing function. We call $a_n$ the advice at length $n$. $\mathcal{C}/\text{poly}$ is a shorthand for $\cup_{c>0} \mathcal{C}/n^c$. P$/\text{poly}$ consists exactly of the languages that can be decided by Boolean circuits of polynomial size. Similarly, NP$/\text{poly}$ consists exactly of the languages that can be decided by nondeterministic Boolean circuits of polynomial size. A nondeterministic circuit has two types of inputs – the actual input $x$ and auxiliary input $y$. It accepts an actual input $x$ if and only if there exists a setting of the auxiliary input $y$ such that the circuit outputs 1 on the combined input $x$ and $y$.

*Communication Protocols.* In general, a two-player communication protocol is described by strategies that tell each of the players when and what to communicate to the other player and how to further behave as a function of the input and the communication history. In the specific case of our oracle communication protocols of Definition 1, there is an asymmetry between the two players. We model the first player as a polynomial-time Turing machine $M$ and the second player as a function $f$. The machine $M$ has a special oracle query tape, oracle query symbol, and oracle answer tape. Whenever $M$ writes the special oracle query symbol on the oracle query tape, in a single computation step the contents of the answer tape is replaced by $f(q)$, where $q$ represents the contents of the oracle query tape at that time. Note that the function $f$ is independent of $M$'s input $x$, which reflects the fact that the second player does not have direct access to the input. The oracle query tape is one-way and is never erased, which allows the strategy of the second player to depend on the entire communication history.

We say that the oracle communication protocol decides a parameterized problem $(L, k)$ if $M$ with oracle $f$ accepts an input $x$ if and only if $x \in L$. The cost $c(k)$ of the protocol is the maximal number of bits written on the oracle query tape over all inputs $x$ with parameter $k(x) = k$.

By considering Turing machines other than the standard deterministic model for the first player, we obtain corresponding variants of oracle communication protocols. For example, we can let the first player be a polynomial-time conondeterministic Turing machine. The second player is always modeled as a function. Whenever there are multiple possible valid executions (as in the case of conondeterministic protocols),

we define the cost as the maximum cost over all of them, that is, we consider the worst case.

## 3. MAIN THEOREM

In this section, we establish Theorem 2 – that $d$-VERTEX COVER has no oracle communication protocol of cost $O(n^{d-\epsilon})$ for any positive constant $\epsilon$ unless coNP $\subseteq$ NP/poly, where $n$ represents the number of vertices of the $d$-uniform hypergraph. For ease of exposition, we actually develop the equivalent result for $d$-CLIQUE rather than for $d$-VERTEX COVER. Theorem 2 then follows by hypergraph complementation.

We follow the approach outlined in the introduction, based on the framework captured by Lemma 1. We first develop the reduction we use in the application of the lemma, and then establish the lemma itself.

### 3.1. Reduction

For some NP-complete language $L$, we need to translate OR($L$) into an equivalent instance of $d$-CLIQUE with few vertices. The choice of the NP-complete language $L$ does not matter. For convenience, we pick it to be 3-SAT. Thus, given $t$ 3-CNF formulas $\varphi_1, \ldots, \varphi_t$, we need to construct a $d$-uniform hypergraph $G$ on few vertices $n$ and an integer $\ell$ such that at least one of the $\varphi_i$'s is satisfiable if and only if $G$ has a clique of size at least $\ell$. We first apply a standard translation of the $t$ individual 3-SAT-instances $\varphi_1, \ldots, \varphi_t$, say of size $s$, into equivalent $d$-CLIQUE-instances consisting of $d$-uniform hypergraphs $G_1, \ldots, G_t$ on $3s$ vertices each, such that $G_i$ has a clique of size $s$ if and only if $\varphi_i$ is satisfiable. All that is left then is to turn these $t$ instances into a single instance of $d$-CLIQUE which is positive if and only if at least one of the $t$ instances is. If we take $G$ as the disjoint union of the $G_i$'s, then $G$ is a $d$-uniform hypergraph that has a clique of size $s$ if and only if at least one of the $G_i$'s has a clique of size $s$. However, this $G$ contains $n = s \cdot t$ vertices, which is too many for our purposes. In order to do better, we need to pack the graphs $G_i$ more tightly while maintaining the properties required of the reduction. The following almost-optimal packing of cliques is the critical ingredient in our construction and allows us to achieve the almost-optimal lower bounds given in Theorem 2.

LEMMA 2 (PACKING LEMMA). *For any integers $s \geq d \geq 2$ and $t > 0$, there exists a $d$-uniform hypergraph $P$ on $O(s \cdot \max(s, t^{1/d+o(1)}))$ vertices such that*

*(i) the hyperedges of $P$ partition into $t$ cliques $K_1, \ldots, K_t$ on $s$ vertices each, and*
*(ii) $P$ contains no cliques on $s$ vertices other than the $K_i$'s.*

*Furthermore, for any fixed $d$, the hypergraph $P$ and the $K_i$'s can be constructed in time polynomial in $s$ and $t$.*

Condition (i) in Lemma 2 formalizes the notion of a packing. The part that $P$ contains the $t$ cliques $K_i$ ensures the completeness of the reduction, that is, that $G$ has a clique of size $s$ if at least one of the $G_i$'s does. The part that the $K_i$'s are edge-disjoint and condition (ii) guarantee the soundness of the reduction, that is, that $G$ has a clique of size $s$ only if at least one of the $G_i$'s does.

We defer the proof of Lemma 2 to Section 4. Using it as sketched here, we obtain the following reduction.

LEMMA 3. *For any integer $d \geq 2$, there is a $\leq_m^p$-reduction from OR(3-SAT) to $d$-CLIQUE that maps $t$-tuples of instances of bitlength $s$ each to instances on $O\left(s \cdot \max(s, t^{1/d+o(1)})\right)$ vertices.*

PROOF. Let $\varphi_1, \ldots, \varphi_t$ be the $t$ instances of 3-SAT. Without loss of generality, assume that each formula has exactly $s$ clauses, each consisting of a sequence of 3 literals. Let $P$ and $K_1, \ldots, K_t$ be the hypergraphs provided by Lemma 2. Along the lines of the standard reduction from 3-SAT to 2-CLIQUE by Karp [1972], we first translate the 3-CNF formulas $\varphi_i$ into $d$-uniform hypergraphs $G_i$ on the vertex sets $V(K_i) \times [3]$. For each $i$, we identify the elements of $V(K_i) \times [3]$ with (positions of) literals of $\varphi_i$: The first component selects a clause from $\varphi_i$ and the second component selects a literal from the clause. We let $G_i$ be the $d$-uniform hypergraph with as edges all subsets $e \subseteq V(K_i) \times [3]$ of size $d$ such that no two elements of $e$ correspond to the same clause $\varphi_i$ or represent complementary literals. Note that each such $e$ induces a satisfying assignment of the conjunction of the $d$ clauses touched by $e$, and that $G_i$ has a clique of size $s$ if and only if $\varphi_i$ is satisfiable.

Let $G$ be the union of the $G_i$'s, that is, the graph with $V(G) = \bigcup_{i \in [t]} V(G_i) \subseteq V(P) \times [3]$ and $E(G) = \bigcup_{i \in [t]} E(G_i)$. If $\varphi_i$ has a satisfying assignment, then $G_i$ has a clique of size $s$ and so has $G$. For the other direction, let $K$ be a clique of size $s$ in $G$. The projection $K'$ of $K$ onto the first component is a clique of size $s$ in $P$. By property (ii) of Lemma 2, $K' = K_i$ for some $i \in [t]$. Moreover, by property (i) of Lemma 2, the projections of $E(G_i)$ and $E(G_j)$ for $j \neq i$ are disjoint. It follows that $K$ is a clique of size $s$ in $G_i$, and therefore $\varphi_i$ is satisfiable.

Thus, $(G, s) \in d$-CLIQUE if and only if $(\varphi_1, \ldots, \varphi_t) \in \text{OR}(3\text{-SAT})$. Since $G$ and $s$ are computable in time polynomial in the bitlength of $(\varphi_1, \ldots, \varphi_t)$ and $|V(G)| \leq 3|V(P)| \leq O(s \cdot \max(s, t^{1/d+o(1)}))$, we have established the $\leq_m^p$-reductions claimed in Lemma 3. □

Subsequent to our work, Dell and Marx [2012] found a more elementary proof of Lemma 3 that does not rely on the packing lemma and the existence of dense sets without arithmetic progressions of length 3. On the other hand, they also *use* the packing lemma to establish a kernel lower bound for a different problem, where their more elementary techniques do not seem to apply.

### 3.2. Framework

We now argue in general the framework given by Lemma 1. The key is the following lemma.

LEMMA 4 (COMPLEMENTARY WITNESS LEMMA). *Let $L$ be a language and $t : \mathbb{N} \to \mathbb{N} \setminus \{0\}$ be polynomially bounded such that the problem of deciding whether at least one out of $t(s)$ inputs of length at most $s$ belongs to $L$ has an oracle communication protocol of cost $O(t(s) \log t(s))$, where the first player can be conondeterministic. Then $L \in \text{coNP/poly}$.*

We defer the proof of Lemma 4 to Section 6 but use it now to prove the framework lemma.

PROOF OF LEMMA 1. We show that if there exists a mapping reduction $f$ from $\text{OR}(L)$ to $\Pi$ as given by the hypothesis, and if $\Pi$ has an oracle communication protocol $P$ of cost $O(k^{c-\epsilon})$ for some constant $\epsilon > 0$, then $\text{coNP} \subseteq \text{NP/poly}$.

Consider the following oracle communication protocol $Q$: On input $(x_1, \ldots, x_t)$, compute $y = f(x_1, \ldots, x_t)$ and run $P$ on input $y$. Since $f$ is a polynomial-time mapping reduction from $\text{OR}(L)$ to $\Pi$, $Q$ is a valid oracle communication protocol for $\text{OR}(L)$. Since $k \leq t^{1/c+o(1)}$, the cost of $Q$ is

$$O(k^{c-\epsilon}) = O((t^{1/c+o(1)} \cdot \text{poly}(s))^{c-\epsilon}),$$

which is $O(t)$ for sufficiently large polynomials $t(s)$. Lemma 4 then shows that $L \in$ coNP/poly. Since $L$ is NP-complete, this implies NP $\subseteq$ coNP/poly, or equivalently, coNP $\subseteq$ NP/poly. □

### 3.3. Proof of the Theorem

Having described all ingredients, the formal proof of Theorem 2 follows.

PROOF OF THEOREM 2. Let $\Pi$ denote $d$-VERTEX COVER parameterized by the number of vertices, i.e., $k = n$. Lemma 3 shows that $\Pi$ has the property in the hypothesis of Lemma 1 with $c = d$. The conclusion of Lemma 1 then coincides with the statement of the theorem. □

## 4. THE PACKING LEMMA

In this section, we establish Lemma 2, which is a critical ingredient in the proof of Theorem 2. We first develop the construction for the case $d = 2$, that is, for standard graphs, and then show how to generalize it to $d$-uniform hypergraphs for arbitrary $d \geq 2$. We also discuss the relationship of our construction to earlier ones.

### 4.1. Our Construction

We need to construct a graph $P$ on few vertices such that

(i) the edges of $P$ partition into $t$ cliques $K_1, \ldots, K_t$ on $s$ vertices each, and
(ii) $P$ contains no other cliques on $s$ vertices.

We first focus on realizing condition (i) and then see how to modify the construction to also realize (ii).

We construct $P$ as an $s$-partite graph and think of the $s$ parts as the columns of a two-dimensional array of vertices, say of size $p$ by $s$. Each of the $K_i$'s then contains exactly one vertex from each of the $s$ columns. Condition (i) expresses that $P$ is a packing of the $K_i$'s. The trivial packing consists of the disjoint union and requires $p = t$ rows, resulting in $s \cdot t$ vertices in total. The trivial packing is wasteful because it leaves many of the potential edges unused. In an ideal packing each of the $p^2$ potential edges between two columns of the array are assigned to some $K_i$. This would only require a number of rows $p = \sqrt{t}$ and therefore $s \cdot \sqrt{t}$ vertices. We can realize such a tight packing by picking the vertex of $K_i$ in column $j$ as the value of $j$ under a hash function $h_i$ from a minimum 2-universal family. If $p$ is a prime at least $s$, we can identify the rows as well as the columns with elements of $\mathbb{F}_p$ and use the family of linear functions over $\mathbb{F}_p$. More precisely, we construct $P$ on the vertex set $V(P) = [s] \times \mathbb{F}_p$ as the union of the $t$ cliques $K_i$ on the vertex sets $V(K_i) = \{(j, h_i(j)) \mid j \in [s]\}$, where $h_i$ is a linear function over $\mathbb{F}_p$ uniquely associated with $K_i$. See Figure 1(a). Note that there are $p^2$ distinct linear functions $h_i$ over $\mathbb{F}_p$, so we can accommodate that many cliques $K_i$. Moreover, since two points define a line, every edge of $P$ is contained in exactly one of the $K_i$'s. For arbitrary values of $s$ and $t$, we can pick $p$ to be the first prime $p \geq \max(s, \sqrt{t})$, resulting in a packing with $O(s \cdot \max(s, \sqrt{t}))$ vertices.

Note that this $P$ is in fact a complete $s$-partite graph and therefore fails to satisfy condition (ii) miserably – every clique of size $s$ that has one vertex from each column is present in $P$, which is many more than just the $K_i$'s. In order to remedy that problem, let us analyze the cliques of size $s$ in $P$ more closely.

Let $K$ denote a clique of size $s$ in $P$. Each of the $s$ columns of $P$ has to contain exactly one vertex of $K$, that is, there exists a function $h : [s] \rightarrow \mathbb{F}_p$ such that $V(K) = \{(j, h(j)) \mid j \in [s]\}$. We would like to ensure that $K$ coincides with one of the cliques $K_i$, or equivalently, that the function $h$ coincides with one of the linear functions $h_i$.
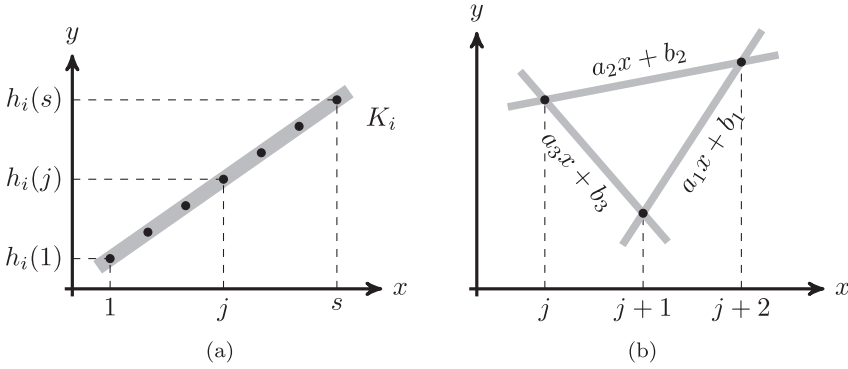
Fig. 1.   (a) The placement of one of the $K_i$'s. (b) Triangle on three consecutive abscissae.

Consider three consecutive columns, $j, j+1$, and $j+2$, and the triangle that $K$ induces between them – see Figure 1(b), where each edge is labeled by the linear function $h_i$ defining the clique $K_i$ to which the edge belongs. We claim that the highest-order coefficients of those linear functions have to form an arithmetic progression. This follows by considering the two paths in Figure 1(b) that go from the vertex in column $j$ to the one in column $j+2$. The direct path on top involves an increase in $y$-value of $2a_2$, whereas the indirect path on the bottom involves an increase in $y$ of $a_3$ followed by an increase of $a_1$. Since both paths end up at the same point, we have that

$$2a_2 = a_1 + a_3, \tag{1}$$

or equivalently, that $a_3 - a_2 = a_2 - a_1$, or yet equivalently, that the sequence $a_1, a_2, a_3$ forms an arithmetic progression. If we restrict the highest-order coefficients of the linear functions to come from a subset $A \subseteq \mathbb{F}_p$ that contains no nontrivial arithmetic progressions of length three, the arithmetic progression $a_1, a_2, a_3$ has to be trivial, that is, $a_1 = a_2 = a_3$. The latter implies that the three lines in Figure 1(b) coincide. As this implication holds for all choices of three consecutive columns, we conclude that all vertices of $K$ lie on a single line defined by one of the $h_i$'s, as we wanted.

Of course, the additional restriction on the highest-order coefficients means that we need to choose $p$ larger. However, we only need to increase $p$ slightly thanks to the existence of efficiently constructible subsets $A \subseteq \mathbb{F}_p$ of high density that contain no nontrivial arithmetic progressions of length three. For our purposes, the following classical result from additive combinatorics suffices.

LEMMA 5 (AP$_3$-FREE SETS [SALEM AND SPENCER 1942]).   *For every positive integer $p$, there exists a subset $A \subseteq \mathbb{Z}_p$ of size at least $p^{1-o(1)}$ which contains no nontrivial arithmetic progressions of length three. Furthermore, such a set $A$ can be determined in time polynomial in $p$.*

For completeness, we provide a proof of Lemma 5 in the Appendix. The resulting graph $P$ has $s \cdot p$ vertices where $p = O\left(\max\left(s, \sqrt{t}^{1+o(1)}\right)\right)$.

This finishes the construction of the packing lemma for the case of standard graphs. The generalization to $d$-uniform hypergraphs follows by using polynomials of degree $d - 1$ instead of linear functions over $\mathbb{F}_p$. Their use guarantees requirement (i) in Lemma 2. Regarding requirement (ii), the following proof shows that the case $d > 2$ reduces to the case $d = 2$. For arbitrary $d \geq 2$, we fulfill requirement (ii) by restricting

the coefficient of degree $d - 1$ to a set that contains no nontrivial arithmetic progressions of length three, namely the set $A \subseteq \mathbb{F}_p$ determined in Lemma 5.

PROOF OF LEMMA 2. Let $p$ be the smallest prime such that $p \geq s$ and $|A| \cdot p^{d-1} \geq t$, where $A$ denotes the set given by Lemma 5. We have that $p = O(\max(s, t^{1/d+o(1)}))$ and can compute $p$ and the set $A$ in time polynomial in $s$ and $t$.

Let $V(P) = [s] \times \mathbb{F}_p$. We consider polynomials of degree at most $d - 1$ over $\mathbb{F}_p$ whose coefficient of $x^{d-1}$ belongs to $A$. Note that there are $|A| \cdot p^{d-1} \geq t$ such polynomials. For $i \in [t]$, let $h_i$ denote the $i$th such polynomial in lexicographic order, and let $K_i$ be the complete $d$-uniform hypergraph on vertex set $V(K_i) = \{(j, h_i(j)) \mid j \in [s]\}$. We define the $d$-uniform hypergraph $P$ as the union of the $t$ cliques $K_i$. The hypergraphs $P$ and $K_i$ can be constructed in time polynomial in $s$ and $t$.

In order to argue property (i), it suffices to observe that every hyperedge of $P$ is contained in at most one of the $K_i$'s. This follows because the requirement that a given hyperedge of $P$ belongs to $K_i$ is equivalent to stipulating the value of $h_i$ on $d$ distinct values $j \in [s]$, which uniquely determines $h_i$ as a polynomial of degree at most $d - 1$ over $\mathbb{F}_p$, and therefore determines $i$.

In order to argue property (ii), we need to establish the following for any function $h : [s] \to \mathbb{F}_p$: If for every subset $D \subseteq [s]$ of size $d$ there exists an $i \in [t]$ such that $h$ and $h_i$ agree on $D$, then there exists an $i \in [t]$ such that $h$ and $h_i$ agree on all of $[s]$. The property follows by applying the next claim to successive values of $j \in [s - d]$, where $q_k$ denotes the polynomial $h_i$ which the hypothesis gives for the subset $D = [j, j + d] \setminus \{k\}$.

*Claim.* For each $k \in [j, j + d]$, let $q_k$ be a polynomial of degree at most $d - 1$ such that the set of coefficients of degree $d - 1$ of the $q_k$'s contains no nontrivial arithmetic progression of length three. If, for all $k, \ell \in [j, j+d]$, the polynomials $q_k$ and $q_\ell$ agree on $[j, j + d] \setminus \{k, \ell\}$, then the polynomials $q_k$ are all the same.

We prove the claim by induction on $d$. We already argued the base case $d = 2$, captured by Figure 1(b), earlier in Section 4. For the inductive step, assume the claim holds for $d - 1$ and let us prove it for $d$. Let $q_j, \ldots, q_{j+d}$ be polynomials as in the claim. For each $k \in [j, j + d - 1]$, define $q'_k$ as the difference quotient $\Delta_{j+d}(q_k)$, that is, $q'_k : [j, j+d-1] \to \mathbb{F}_p$ such that $q'_k(x) = (q_k(x) - q_k(j+d))/(x-j-d)$ for $x \in [j, j+d-1]$. Note that $q'_k$ is a polynomial of degree at most $d - 2$ whose coefficient of degree $d - 2$ equals the coefficient of $q_k$ of degree $x^{d-1}$. Moreover, for $k, \ell \in [j, j + d - 1]$, the polynomials $q'_k$ and $q'_\ell$ agree on each $x \in [j, j + d - 1] \setminus \{k, \ell\}$ because the polynomials $q_k$ and $q_\ell$ agree on both $x$ and $j+d$. Thus, by the induction hypothesis, all polynomials $q'_k$ are the same. By the definition of $q'_k = \Delta_{j+d}(q_k)$ and the fact that the polynomials $q_k$ for $k \in [j, j + d - 1]$ agree on $j + d$, this implies that the polynomials $q_k$ for $k \in [j, j + d - 1]$ are all the same, say $q$. All that remains to show is that the polynomial $q_{j+d}$ also coincides with $q$. The latter follows because $q_{j+d}$ is a polynomial of degree at most $d - 1$, which agrees with the polynomial $q$ of degree at most $d - 1$ on all $d$ points in $[j, j + d - 1]$. □

### 4.2. Related Constructions

After we developed our construction, we learned about similar applications of high-density subsets of the integers without nontrivial arithmetic progressions of length three.

Back in 1976, Ruzsa and Szemerédi [1978] constructed dense three-partite graphs whose edges partition into triangles and that contain no other triangles. Their construction corresponds to the case $(d, s) = (2, 3)$ of our Packing Lemma, and appears

between any three consecutive columns of our construction for $d = 2$ and general $s$. Our geometric derivation of the arithmetic progression condition (1), as captured in Figure 1(b), may be new; all the derivations we have found in the literature work by manipulating equations in a – to us – less intuitive way.

Different aspects of the Ruzsa-Szemerédi construction matter for the various applications we know of in the theory of computing. For their soundness analysis of graph tests for linearity, Håstad and Wigderson [2003] use the interpretation that for each of the $p$ points in the first column, the triangles involving that point span an induced matching of $p^{1-o(1)}$ edges between the other columns.

Another application area is the lower bounds for testing the graph property of being $F$-free, where $F$ is some fixed graph. An $\epsilon$-tester for this property accepts all graphs that are $F$-free, and rejects all graphs that are at least $\epsilon$ away from being $F$-free, that is, from which at least $\epsilon n^2$ edges need to be removed to make it $F$-free [Goldreich et al. 1998]. A strategy for proving lower bounds on the number of queries of such a tester is to construct high-density graphs $G$ with the following properties: (i) the edges of $G$ partition into copies of $F$, and (ii) $G$ contains few other copies of $F$ so the total number of copies of $F$ in $G$ is significantly less than expected in a random graphs of the same density as $G$ [Alon 2002]. Qualitatively, (i) implies that $G$ is far from being $F$-free, and (ii) implies that testers with few queries have a small probability of detecting a violation of $F$-freeness on input $G$. Alon and coauthors [Alon 2002; Alon and Shapira 2004, 2005, 2006; Alon et al. 2008] constructed such graphs $G$ for various $F$ based on the work of Ruzsa and Szemerédi [1978].

The requirements for our application are similar but not identical to the ones for property testing. On the one hand, we only need to consider the cases where $F$ is a clique; on the other hand, the graphs $G$ cannot contain any copy of $F$ other than those in which the edges partition. Our actual construction is very similar to the one Alon and Shapira [2005] develop. Their construction would also work for our purposes. Our proof differs from theirs and makes the arithmetic progression condition more transparent. Our construction slightly improves[2] on theirs as we only restrict the highest-order coefficient to the set $A$, whereas they restrict all coefficients to that set.

## 5. CONSEQUENCES OF THE MAIN THEOREM

Our lower bound for oracle communication protocols for $d$-VERTEX COVER, Theorem 2, has two types of consequences. The first is similar lower bounds for other parameterized NP-complete problems, and follow from parameter-frugal reductions from $d$-VERTEX COVER to these problems. The second type involves lower bounds for parameters of interest in settings that are captured by our oracle communication model. In this section, we first cover the consequences for satisfiability and then those for vertex deletion problems.

### 5.1. Satisfiability

Theorem 1, our tight oracle communication lower bound for $d$-SAT parameterized by the number of variables of the formula, immediately follows from Theorem 2 and the next lemma.

LEMMA 6. *For every $d \geq 3$, there is a $\leq_m^p$-reduction from $d$-VERTEX COVER to $d$-SAT that maps $d$-uniform hypergraphs on $n$ vertices to $d$-CNF formulas on $O(n)$ variables.*

---

[2]This allows us to relax the condition $q(\epsilon) = \max\{m : (f(m))^k \geq \epsilon\}$ in Lemma 4.1 of Alon and Shapira [2005] to $q(\epsilon) = \max\{m : f(m) \geq \epsilon\}$.

PROOF. Let $(G, k)$ be an $n$-vertex instance of $d$-VERTEX COVER. The following $d$-CNF formula on variables $x_v$ for $v \in V(G)$ has as satisfying assignments precisely the characteristic vectors of vertex covers of $G$:

$$\varphi := \bigwedge_{e \in E(G)} \bigvee_{v \in e} x_v.$$

Using at most $O(n)$ new variables, we construct a 3-CNF formula $\psi$ that is satisfied by all assignments in which at most $k$ distinct $x_v$ are set to true. Then, $\varphi \wedge \psi$ is satisfiable if and only if $G$ has a vertex cover of size at most $k$.

For the construction of $\psi$, we use a Boolean circuit of constant fan-in that has at most $O(n)$ gates and checks whether at most $k$ of the $n$ input variables are set to true. Such circuits can be constructed for any symmetric function in time polynomial in $n$ when given oracle access to the function [Wegener 1987, Theorem 4.1]. Once we have that circuit, we construct $\psi$ in a standard way by introducing a new variable for each gate, and letting $\psi$ be the conjunction of clauses that express the correct behavior of each of the gates, and the clause stipulating that the output gate is set. □

PROOF OF THEOREM 1. Suppose there exists an oracle communication protocol of cost $O(n^{d-\epsilon})$ for $n$-variable instances of $d$-SAT. By combining the $\leq_m^p$-reduction from Lemma 6 with the former, we obtain an oracle communication protocol of cost $O(n^{d-\epsilon})$ for $n$-vertex instances of $d$-VERTEX COVER. By Theorem 2, the latter implies that coNP $\subseteq$ NP/poly. □

The following corollary to Theorem 1 embodies the consequences for sparsification, kernelization, and lossy compression.

COROLLARY 1. *Let $d \geq 3$ be an integer. If coNP $\not\subseteq$ NP/poly, then there is no polynomial-time reduction from $d$-SAT to any problem that makes at most $O(n^b)$ queries and only queries strings of bitlength $O(n^c)$, where $b$ and $c$ are any nonnegative reals with $b + c < d$.*

In particular, under the hypothesis that coNP $\not\subseteq$ NP/poly, Corollary 1 implies that $\leq_m^p$-reductions cannot reduce the density of $n$-variable $d$-SAT instances to $O(n^c)$ clauses for any constant $c$ below the trivial $c = d$. This is what the title of this article refers to, and contrasts the situation at the subexponential-time level. The sparsification lemma of Impagliazzo et al. [2001] gives a reduction which, on input an $n$-variable $d$-CNF formula and a rational $\epsilon > 0$, runs in time $2^{\epsilon n} \cdot \mathrm{poly}(n)$ and makes $2^{\epsilon n}$ nonadaptive queries, each of which are $d$-CNF formulas with at most $f(d, \epsilon) \cdot n$ clauses. The best-known bound on the sparsification constant $f(d, \epsilon)$ is $(d/\epsilon)^{3d}$ [Calabro et al. 2006]. The sparsification lemma implies that sparse instances of $d$-SAT are hard under subexponential-time reductions while Corollary 1 suggests that such a result is impossible under $\leq_m^p$-reductions. Interpretations of Corollary 1 in terms of kernelization and lossy compression follow along the same lines.

Another consequence of Theorem 1 deals with the size of probabilistically checkable proofs for satisfiability. Recall that Dinur [2007] constructed such PCPs of size $O(s \cdot \mathrm{poly} \log s)$, where $s$ denotes the bitlength of the formula. Based on a connection due to Harnik and Naor [2010] between PCPs and lossy compression, Fortnow and Santhanam [2011] showed that satisfiability of Boolean formulas does not have PCPs of size bounded by a polynomial in the number of variables only, unless coNP $\subseteq$ NP/poly. Plugging in our lower bound for $d$-SAT into their argument shows that $d$-SAT does not have $q$-query PCPs of size $O(n^{d/q-\epsilon})$ unless coNP $\subseteq$ NP/poly. Since $q \geq 3$ this bound is not tight. Using a different argument and exploiting the fact that

Theorem 1 also holds for conondeterministic protocols, we can close the gap between the upper and lower bound.

COROLLARY 2. *Let $d \geq 3$ be an integer and $\epsilon$ a positive real. If* coNP $\not\subseteq$ NP/poly, *then $d$-SAT does not have probabilistically checkable proofs of bitlength $O(n^{d-\epsilon})$ where $n$ denotes the number of variables of the input formula.*

PROOF. Suppose that $d$-SAT has PCPs of size $s = O(n^c)$ that make $q$ nonadaptive queries, where $c$ and $q$ are constants. We claim that this implies a conondeterministic multivalued mapping reduction from $d$-SAT to $q$-SAT that maps formulas on $n$ variables to instances of bitlength $O(n^c \log n)$ in the following sense: There exists a nondeterministic polynomial-time Turing machine $M$ which outputs a $q$-CNF formula on each computation path (where the formula may depend on the input and the computation path) such that (i) if the input is in $d$-SAT then every output is in $q$-SAT, and (ii) otherwise at least one output is not in $q$-SAT. For $c < d$, Theorem 1 then shows that coNP $\subseteq$ NP/poly.

All that remains is to argue the claim. For a given formula $\varphi$ on $n$ variables, introduce $s$ new variables $y$, namely one for each bit position in a candidate PCP of size $s$. If the PCP system reads at most $q$ bits of the proof, each condition the PCP system checks can be expressed efficiently as a $q$-CNF. By picking a condition according to the distribution of the PCP system and a clause of the corresponding $q$-CNF formula uniformly at random, we obtain a polynomial-time randomized procedure that produces a $q$-clause on the variables $y$ with the property that if $\varphi$ is satisfiable, then all $q$-clauses produced are simultaneously satisfiable, and otherwise less than a constant fraction $\rho < 1$ is. By averaging, the latter implies that for every collection of candidate PCPs of size $s$ for an unsatisfiable input $\varphi$, there exists a produced $q$-clause that is violated by more than a fraction $1 - \rho$ of the collection. Since there are $2^s$ candidate PCPs of size $s$ in total, this means that there is a set of $s/\log(1/\rho)$ produced $q$-clauses that cannot be satisfied by any PCP of size $s$. The reduction nondeterministically guesses $s/\log(1/\rho)$ many $q$-clauses that are produced by the PCP system on input $\varphi$, and outputs their conjunction. The conjunction has bitlength $O(n^c \log n)$, is always satisfiable if $\varphi$ is, and is not satisfiable on at least one computation path otherwise.  □

## 5.2. Vertex Cover and Other Vertex Deletion Problems

Theorem 2 yields applications for $d$-VERTEX COVER similar to Corollaries 1 and 2 for $d$-SAT, using the number of vertices $n$ as the parameter. A more natural parameter for $d$-VERTEX COVER is the size $k$ of the vertex cover. We now investigate the consequences of Theorem 2 for this parameterization, first for the case $d = 2$, that is, for standard graphs, and then for $d$-uniform hypergraphs for general $d$.

*Result for Standard Graphs.* We consider the following generalization of the vertex cover problem. Recall that a graph property is a predicate on graphs that is invariant under graph isomorphism.

*Definition* 2 (*Vertex Deletion*). Fix a graph property $\Pi$. The $\Pi$-VERTEX DELETION problem is to decide, for a given graph $G$ and integer $k$, whether there exists a subset $S$ of at most $k$ vertices such that $G \setminus S$ satisfies $\Pi$.

We say that a graph property $\Pi$ is inherited by subgraphs if whenever a graph $G$ satisfies $\Pi$, every subgraph of $G$ also satisfies $\Pi$. The following natural graph problems are special cases of $\Pi$-VERTEX DELETION for a graph property $\Pi$ that is inherited by subgraphs.

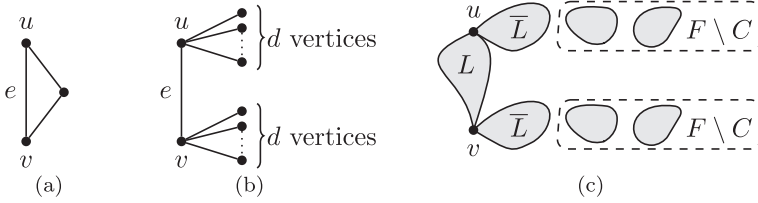— VERTEX COVER. Can we delete $k$ vertices to destroy all edges?

Fig. 2. Replacement of an edge $e = \{u, v\}$ in the transformation from $G$ to $G'$ in the proof of Lemma 7. (a) FEEDBACK VERTEX SET. (b) BOUNDED-DEGREE DELETION. (c) The general case.

— FEEDBACK VERTEX SET. Can we delete $k$ vertices to destroy all cycles?
— BOUNDED-DEGREE DELETION. Can we delete $k$ vertices to get a maximum degree of $d$?
— NON-PLANAR DELETION. Can we delete $k$ vertices to make the graph planar?
— Can we delete $k$ vertices to make the graph embeddable into some surface?
— Can we delete $k$ vertices to make the graph exclude any fixed set of minors?

As mentioned in the introduction, if only finitely many graphs satisfy Π or if all graphs satisfy Π, Π-VERTEX DELETION is trivially decidable in polynomial time. For all other graph properties Π that are inherited by subgraphs, Theorem 3 implies that Π-VERTEX DELETION does not have kernels with $O(k^{2-\epsilon})$ edges unless coNP $\subseteq$ NP/poly.

We now prove Theorem 3 by constructing a $\leq_m^p$-reduction from VERTEX COVER to Π-VERTEX DELETION that blows up the size of the deletion set by no more than a constant factor. In order to develop some intuition, we first consider the standard reduction from VERTEX COVER to FEEDBACK VERTEX SET [Karp 1972]. The reduction replaces every edge $e$ of a VERTEX COVER-instance $G$ by a cycle of length three using an additional new vertex, as depicted in Figure 2(a). Let us denote the resulting graph by $G'$. Since every cycle in $G'$ contains two vertices that are adjacent in $G$, every vertex cover of $G$ hits every cycle of $G'$ and therefore is a feedback vertex set of $G'$. Conversely, every feedback vertex set of $G'$ contains a vertex of every triangle we created, and can therefore be turned into a vertex cover of $G$ of at most the same size. Thus, $G$ has a vertex cover of size $k$ if and only if $G'$ has a feedback vertex set of size $k$.

As another example, consider the case of BOUNDED-DEGREE DELETION. In the known reduction from VERTEX COVER to this problem [Krishnamoorthy and Deo 1979], $d$ new edges are attached to every vertex of $G$ (see Figure 2(b)). Removing any vertex cover of $G$ from $G'$ reduces the maximum degree to $d$. Vice-versa, any set that reduces the maximum degree in $G'$ to $d$ can be transformed into a vertex cover of $G$ of at most the same size.

Next consider the more general case in which the minimal graphs that violate Π are connected. Generalizing these two examples we obtain $G'$ by replacing every edge of the VERTEX COVER-instance $G$ by a copy of a fixed connected graph $F$ violating Π. We refer to $F$ as a "forbidden" graph since no graph satisfying Π can contain $F$ as a subgraph. Thus, any deletion set in $G'$ has to pick at least one vertex from every copy of $F$. Projecting the deletion set back onto the graph $G$ yields a vertex cover of size no more than the deletion set. This way we can guarantee the soundness of the reduction – if $G'$ has a deletion set of size at most $k$ then $G$ has a vertex cover of size at most $k$.

For the completeness of the reduction, we would like to ensure that removing a vertex cover $S$ of $G$ from $G'$ leaves a graph $G' \setminus S$ satisfying Π. This is not automatically the case because $G' \setminus S$ may contain components of the form depicted in Figure 3(a), where the bullets are vertices of $G$ and the hashed vertices are part of the vertex cover $S$ (and are therefore not part of $G' \setminus S$) but the center vertex is not. Such a
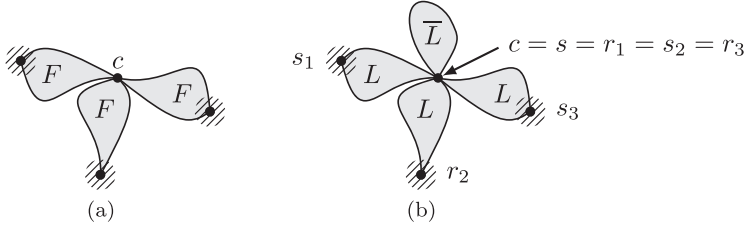
Fig. 3. Connected component $C'$ that might remain after removing a vertex cover $S$ of $G$ from $G'$, centered around a vertex $c$ that has degree 3 in $G$ and does not belong to $S$. (a) Naïve construction. (b) Final construction.

component could contain a copy of $F$, in which case $G' \setminus S$ would not satisfy $\Pi$. However, by attaching the copies of $F$ in an appropriate way we can make sure that the connected components of $G' \setminus S$ are all "simpler" than $F$. Picking $F$ to be a "simplest" connected graph that violates $\Pi$ then does the job as long as all minimal graphs violating $\Pi$ are connected.

More generally, consider a graph $F$ violating $\Pi$ whose most complex connected component $C$ is as simple as possible among all graphs violating $\Pi$. If $F$ has no other connected component of the same complexity as $C$, then the above construction still works, using a copy of $C$ to replace every edge in $G$ and including a copy of $F \setminus C$ for every vertex of $G$.

In the most general case, where minimal graphs violating $\Pi$ can have multiple components of the same complexity, we use a slightly different construction that involves multiple copies of $G$. The graph $F$ now becomes a "simplest" graph for which the number of disjoint copies of $F$ that satisfies $\Pi$ is bounded. The reduction is no longer parameter preserving in general, but the parameter $k'$ for $G'$ is still linearly bounded by the parameter $k$ for $G$. The latter ensures that the lower bound for $\Pi$-VERTEX DELETION is as strong as for VERTEX COVER modulo a constant factor.

The simplicity measure we use is the same as the one of Lewis and Yannakakis [1980] but the construction is a bit different: their construction blows up the parameter $k'$ to $\Theta(nk)$, but a straightforward modification reduces $k'$ to $\Theta(k^2)$. We further reduce $k'$ to $\Theta(k)$ using a matching argument.

LEMMA 7. *Let $\Pi$ be a graph property that is inherited by subgraphs, and is satisfied by infinitely many but not all graphs. There is a $\leq_m^p$-reduction from* VERTEX COVER *to* $\Pi$-VERTEX DELETION *that maps instances with parameter $k$ to instances with parameter $O(k)$.*

PROOF. We start by spelling out the simplicity measure for graphs. We first consider a connected graph $C$. For any vertex $s$ of $C$, we define the character of $C$ relative to $s$ as the sequence $\chi = (\chi_i)_{i \in \mathbb{N}}$ where $\chi_i$ denotes the number of connected components of $C \setminus \{s\}$ that have exactly $i$ vertices. We compare two characters $\chi$ and $\eta$ using the colexicographical order, that is, $\chi < \eta$ if there exists a positive integer $i$ such that $\chi_j = \eta_j$ for all integers $j > i$ and $\chi_i < \eta_i$. The corresponding relation $\leq$ defines a well-order on the set of characters, that is, a total order in which every nonempty subset has a smallest element. We define the *character of $C$* as a smallest character of $C$ relative to $s$ over all vertices $s$ of $C$.

For an arbitrary graph $G$, we define its *signature* as a mapping $\sigma$ from the set of all characters to $\mathbb{N}$, where $\sigma(\chi)$ equals the number of connected components of $G$ with character $\chi$. We compare two signatures $\sigma$ and $\tau$ using the colexicographical order induced by the order on characters, that is, $\sigma < \tau$ if there exists a character $\chi$ such

that $\sigma(\eta) = \tau(\eta)$ for all characters $\eta > \chi$ and $\sigma(\chi) < \tau(\chi)$. The corresponding relation $\leq$ defines a well-order on the set of signatures.

Our simplicity measure on graphs is induced by the $\leq$-relation on their signatures. We choose a graph $F$ with the smallest signature among all graphs for which the number of disjoint copies that satisfy $\Pi$ is bounded. Note that $F$ exists because not all graphs satisfy $\Pi$. Let $t$ be the positive integer such that the disjoint union of $t-1$ copies of $F$ satisfies $\Pi$ but $t$ disjoint copies do not. Let $C$ denote a connected component of $F$ with largest character and let $s \in V(C)$ be a witness for that character. Let $L$ be the subgraph of $C$ spanned by $s$ and the vertices of a largest connected component of $C \setminus \{s\}$, and let $\overline{L}$ be the subgraph of $C$ spanned by $s$ and the vertices of $C \setminus L$. Note that $L$ contains at least one other vertex than $s$. Otherwise, $F$ would consist of isolated vertices only and only finitely many graphs would satisfy $\Pi$. Let $r$ be an arbitrary vertex of $L \setminus \{s\}$.

We are now in position to describe the reduction transforming an instance $(G, k)$ of VERTEX COVER into an instance $(G', k')$ of $\Pi$-VERTEX DELETION such that $G$ has a vertex cover of size $k$ if and only if $k'$ vertices can be deleted from $G'$ to make the residual graph satisfy $\Pi$. For the construction of $G'$, we start with $2t - 1$ disjoint copies $G_1, \ldots, G_{2t-1}$ of $G$. We replace every edge $e$ of $G_i$ by a copy $L_e$ of the component $L$ such that the endpoints of $e$ are identified with $s$ and $r$ in an arbitrary way; the vertices of $L_e$ outside of $e$ are new. Furthermore, we attach to every vertex $v \in V(G)$ a graph $R_v$ that consists of a copy of $\overline{L}$ and disjoint copies of $F \setminus C$; here we identify $v$ with the vertex $s$ of $\overline{L}$ and create all other vertices of $R_v$ new. See Figure 2(c). In the remainder, we show that the reduction works when we set $k' = (2t - 1)k$.

For the soundness of the reduction, let $S'$ be a set of $k'$ vertices in $G'$ such that $G' \setminus S'$ satisfies $\Pi$. Let $S$ denote the projection of $S'$ onto $V(G_1) \cup \cdots \cup V(G_{2t-1})$, where the projection of a vertex $u \in V(G')$ is one of the vertices of $e$ (chosen arbitrarily) in case $u \in V(L_e) \setminus e$ and the vertex $v$ in case $u \in V(R_v)$. We claim that $S$ is at most $2t - 2$ vertices away from being a vertex cover of $G_1 \cup \cdots \cup G_{2t-1}$. Let $M$ be a maximal matching in $(G_1 \cup \cdots \cup G_{2t-1}) \setminus S$. If $M$ contains at least $t$ edges, then $S'$ avoids at least $t$ disjoint subgraphs $L_e \cup R_u \cup R_v$ for $e = (u, v)$. In particular, $G' \setminus S'$ contains $t$ copies of $F$ as subgraphs, which contradicts the fact that $G' \setminus S'$ satisfies $\Pi$. Thus, $M$ contains at most $t - 1$ edges. Adding $V(M)$ to $S$, we thus get a vertex cover of $G_1 \cup \cdots \cup G_{2t-1}$ of size at most $(2t - 1)k + 2t - 2$. By averaging, there is an $i$ with $|S \cap V(G_i)| \leq \lfloor k + 1 - \frac{1}{2t-1} \rfloor = k$. Hence, $G$ has a vertex cover of size at most $k$.

For the completeness of the reduction, let $S$ be a vertex cover of $G$ of size at most $k$. Let $S'$ consist of the $2t - 1$ copies of $S$ in the graphs $G_1, \ldots, G_{2t-1}$. Clearly, $|S'| \leq (2t-1)k$. Let $H$ be obtained from $G' \setminus S'$ by removing duplicate isomorphic copies of connected components. Note that $G' \setminus S'$ is a subgraph of finitely many disjoint copies of $H$. Thus, if we can show that $H$ has a strictly smaller signature than $F$, then any number of disjoint copies of $H$ satisfies $\Pi$ and by inheritance the subgraph $G' \setminus S'$ also satisfies $\Pi$. Therefore, $S'$ is a set of at most $k' = (2t - 1)k$ vertices such that $G' \setminus S'$ satisfies $\Pi$.

It remains to argue that $H$ has a strictly smaller signature than $F$. In order to do so we consider the connected components of $H$, and we distinguish four types: (1) components isomorphic to components of $F \setminus C$, (2) components isomorphic to components of $L \setminus \{s, r\}$, (3) components isomorphic to components of $\overline{L} \setminus \{s\}$, and (4) components as in Figure 3(b) consisting of a single copy of $\overline{L}$ and one or more copies of $L \setminus \{s\}$ and $L \setminus \{r\}$ in which all remaining copies of $s$ and $r$ have been identified with the vertex $c$. We show that for each of the connected components of types (2), (3), and (4), the character is strictly less than for $C$. Since $C$ is the connected component of $F$ with the largest character and $H$ has no duplicate isomorphic connected components, this implies that no connected component of $H$ has a character larger than $C$, and that the number of

connected components of $H$ with the same character as $C$ is strictly less than in $F$. Therefore, the signature of $H$ is strictly less than the one of $F$.

Let us first consider a connected component $C'$ of $H$ of type (4). Consider removing the vertex $c$ in Figure 3(b). Since $L \setminus \{s\}$ is a largest connected component of $C \setminus \{s\}$, no connected component of $C' \setminus \{c\}$ can have more vertices than $L \setminus \{s\}$. Moreover, the only components in $C' \setminus \{c\}$ that can have $|V(L \setminus \{s\})|$ vertices must come from the part $\overline{L} \setminus \{s\}$. Since $C = L \cup \overline{L}$, this means that $C \setminus \{s\}$ has one more connected component with $|V(L \setminus \{s\})|$ vertices than $C' \setminus \{c\}$. Thus, the character of $C'$ relative to $c$, and a fortiori the character of $C'$, is strictly less than the character of $C$.

The claim that connected components of types (2) and (3) have characters strictly less than $C$ follows from the corresponding claim for type (4) since (2) and (3) are subgraphs of a graph of type (4) and taking subgraphs cannot result in larger characters.  □

We point out that the proof of Lewis and Yannakakis [1980] only needs inheritance by *induced* subgraphs. The only step in the proof of Lemma 7 that requires the stronger property of inheritance by subgraphs is the matching argument. That step is vacuous when $t = 1$, for example, when all minimal graphs violating $\Pi$ are connected. The stronger property is also not necessary when the vertex $s$ is not connected to all vertices of $L$ (and we choose $r$ as one of those vertices). In such cases, our proof can do with inheritance by induced subgraphs.

PROOF OF THEOREM 3. Suppose that $\Pi$-VERTEX DELETION parameterized by the size of the deletion set has a cost $O(k^{2-\epsilon})$ protocol. By combining the $\leq^p_m$-reduction from Lemma 7 with that protocol, we obtain a cost $O(k^{2-\epsilon})$ protocol for VERTEX COVER parameterized by the size of the vertex cover. Since $k \leq n$, the case $d = 2$ of Theorem 2 then implies that coNP $\subseteq$ NP/poly.  □

Theorem 3 applies, among others, to FEEDBACK VERTEX SET, another problem whose kernelization has received considerable attention in parameterized complexity. Theorem 3 implies that FEEDBACK VERTEX SET does not have kernels consisting of $O(k^{2-\epsilon})$ edges unless coNP $\subseteq$ NP/poly. This result is tight – a kernel with $O(k^2)$ edges follows from recent work by Thomassé [2010]. He constructs a kernel with at most $4k^2$ vertices and maximum degree at most $4k$. For such an instance to be positive, the number of edges can be no larger than $8k^2$. Indeed, suppose that $S$ is a feedback vertex set of $G$ of size at most $k$. Then the graph induced by $V(G) \setminus S$ is a forest and has at most $4k^2$ edges. All other edges of $G$ are incident to a vertex of $S$. As the maximum degree is no larger than $4k$, at most $4k^2$ edges are incident to $S$. Summing up, $G$ has at most $8k^2$ edges. Thus, if $G$ has more than $8k^2$ edges, we can reduce to a trivial negative instance; otherwise, we reduce to $G$. This results in a kernel with $O(k^2)$ edges.

*Extension to Hypergraphs.* We now turn to vertex cover and related problems on $d$-uniform hypergraphs. Since $k \leq n$, Theorem 2 implies that $d$-VERTEX COVER does not have kernels with $O(k^{d-\epsilon})$ edges unless coNP $\subseteq$ NP/poly. We point out that kernels with $O(k^d)$ edges exist for $d$-VERTEX COVER. This follows from a generalization of Buss' high-degree rule (see the introduction) and a folklore application of the sunflower lemma (see Flum and Grohe [2006, chapter 9.1], for example). Recall that for a hypergraph $G$, a sunflower with heart $h \subseteq V(G)$ and $p$ petals is a set of distinct edges whose pairwise intersection is exactly $h$. The kernelization proceeds by repeatedly picking a sunflower with at least $k+1$ petals, removing the involved edges, and adding the heart as a new edge to the graph. Note that in this process, edges of size less than $d$ may be added to $G$. To get back a $d$-uniform graph, one can complete those edges with fresh

vertices, which doesn't affect the number of edges nor the minimum size of a vertex cover. The process continues until no sunflower with $k+1$ petals exists, which is bound to happen as the number of edges decreases in every step. The sunflower lemma of Erdős and Rado [1960] states that any $d$-uniform hypergraph with more than $d! \cdot k^d$ edges has a sunflower with $k+1$ petals. Thus, the hypergraph that remains at the end has at most $d \cdot d! \cdot k^d = O(k^d)$ edges, and has a vertex cover of size at most $k$ if and only if the original hypergraph does.

Regarding extensions of Theorem 3 to $d$-uniform hypergraphs for $d > 2$, we cannot expect to rule out protocols of cost $O(k^{d-\epsilon})$ for all hypergraph properties $\Pi$ that are inherited by subgraphs and for which the deletion problem is nontrivial. This is because the property $\Pi$ could only depend on the primal graph underlying the hypergraph, for which protocols of cost $O(k^2)$ are known in some cases.

## 6. THE COMPLEMENTARY WITNESS LEMMA

In this section, we prove Lemma 4 and mention some applications other than the main theorem of this article.

### 6.1. Proof of the Lemma

We first describe the special case of Lemma 4 where the language $L$ is P-selective. The simpler argument for that case provides a good starting point for the proof of the general case.

A P-selector for a language $L$ is a polynomial-time algorithm that takes two instances $x$ and $y$ as input and outputs one of them, with the guarantee that, if at least one of the inputs belongs to $L$, then so does the one that is output. Note that a P-selector for $L$ immediately yields a low-cost oracle communication protocol for deciding OR($L$) on inputs consisting of $t$ instances of size $s$ each – the first player uses the selector $t-1$ times to determine which of the instances is "most likely" to be in $L$, sends that instance to the oracle, who responds with the membership of that instance to $L$. Since the cost of this protocol is $s$, any P-selective language satisfies the promise of Lemma 4 with $t=s$.

Ko [1983] showed that the existence of a P-selector for $L$ implies that $\overline{L}$ (and thus $L$) can be decided by circuits of polynomial size. The key insight is the following way to prove that an instance $x$ belongs to $\overline{L}$: Exhibit an instance $y$ that is known to be in $\overline{L}$ and which the selector $S$ outputs when given $x$ and $y$ as input. We call such a $y$ a complementary witness. By viewing $S$ on all pairs of a given subset $F \subseteq \overline{L}$ as a tournament, there always exists a $y \in F$ that beats at least half of the $x \in F$ and therefore can be used as a proof of membership of $x$ to $\overline{L}$. Starting from the set of all instances of size $s$ in $\overline{L}$, we repeatedly apply this procedure to the remaining set $F$ of instances that have not yet been beaten by some of the $y$'s we picked, until the set becomes empty. This way, we obtain a collection $A_s$ of at most $s$ elements $y$ such that $x \in \overline{L}$ if and only if there exists a $y \in A_s$ such that $S(x,y) = y$. Using the set $A_s$ as advice, this shows that $\overline{L} \in$ P/poly. If we allow the selector $S$ to be nondeterministic (even multivalued), we similarly obtain that $\overline{L} \in$ NP/poly [Hemaspaandra et al. 1995].

Fortnow and Santhanam [2011] established the case of Lemma 4 where the protocol implements a $\leq_m^p$-reduction from OR($L$) to some language $L'$ such that $t$-tuples consisting of instances of bitlength $s$ are mapped to an instance of bitlength bounded by some fixed polynomial in $s$, independent of $t$. Their proof can be viewed as an extension of the above argument. The witnesses $y$ are now elements from $\overline{L'}$, and the requirement on the bitlength of the reduced instances guarantees that sufficiently popular $y$'s exist, so we don't need too many of them. The statement of Lemma 4 results from a more

careful analysis of that argument for bounds that can grow slowly with $t$, and from the
extension to the general setting of our oracle communication model.

PROOF OF LEMMA 4. Let us first consider the case of a deterministic oracle com-
munication protocol $P$ modeled by a deterministic polynomial-time Turing machine $M$
and a function $f$ (see Section 2 for the notation). In this proof, we make use of the no-
tion of a communication transcript on a given input $x$. Such a transcript consists of the
sequence of all queries $P$ makes on input $x$ (i.e., the contents of $M$'s oracle query tape
at the end of the protocol) as well as the answers $f(q)$ to each of the oracle queries $q$.

The key ingredient of the proof is the following equivalence: An instance $x$ of
bitlength $s$ is in $\overline{L}$ if and only if there exists a sequence $x_2, \ldots, x_{t(s)}$ of instances of
bitlength $s$ such that $P(x, x_2, \ldots, x_{t(s)})$ rejects. The forward direction of the equivalence
follows from the soundness of the protocol by setting $x_2 = \cdots = x_{t(s)} = x$, and the back-
wards direction follows from the protocol's completeness. By including a large enough
set $A_s$ of communication transcripts and the value of $t(s)$ as advice, this leads to the
following proof system with advice for $\overline{L}$. On input an instance $x$ of bitlength $s$:

(1) guess a sequence $x_2, \ldots, x_{t(s)}$ where each $x_i$ has bitlength $s$;
(2) check whether there is a communication transcript $\tau$ in $A_s$ that is consistent with
    $P$ on input $(x, x_2, \ldots, x_{t(s)})$ and that $P(x, x_2, \ldots, x_{t(s)})$ rejects. If so, accept; otherwise,
    reject.

The check for a given transcript $\tau$ involves running the first player on the input
$(x, x_2, \ldots, x_{t(s)})$. Whenever the first player sends a bit to the second player (by writing
on the oracle query tape), verify that it agrees with the corresponding bit in $\tau$. When-
ever the first player expects a bit from the second player (by reading from the oracle
answer tape), use the corresponding bit in $\tau$. This process continues until a discrepancy
is detected or the first player halts.

This proof system is sound as long as all communication transcripts in $A_s$ are con-
sistent with the protocol $P$. All that remains to show is the existence of a small subset
$A_s$ of such transcripts that guarantees completeness.

We construct $A_s$ for a fixed $s$ in the following greedy way. Consider instances,
$x_1, \ldots, x_{t(s)}$ of $L$ of bitlength $s$, and let $T(x_1, \ldots, x_{t(s)})$ denote the communication tran-
script of $P$ on input $(x_1, \ldots, x_{t(s)})$. Since the second player is not given the input
$(x_1, \ldots, x_{t(s)})$, the transcript $T(x_1, \ldots, x_{t(s)})$ is determined solely by the bits sent from
the first player to the second player. Therefore, the number of distinct such transcripts
is less than $2^{c(s)+1}$, where $c(s)$ denotes the cost of the protocol on inputs consisting of
$t(s)$ instances of bitlength $s$ each. We say that a rejecting transcript $\tau$ *covers* an instance
$x \in \overline{L}$ of bitlength $s$ if there exists a sequence $x_2, \ldots, x_{t(s)}$ of instances of bitlength $s$ each
such that $T(x, x_2, \ldots, x_{t(s)}) = \tau$. We start with $A_s$ empty and successively pick a reject-
ing communication transcript $\tau$ that covers the largest number of instances $x \in \overline{L}$ of
length $s$ that are not covered thus far, and add $\tau$ to $A_s$. We keep doing so until there
are no more instances $x \in \overline{L}$ of bitlength $s$ left to cover.

Consider one step in the construction of $A_s$ and let $F$ denote the set of uncovered
instances $x \in \overline{L}$ of bitlength $s$ at the beginning of the step. Since every tuple in $F^{t(s)}$ is
mapped by $T$ to one of the rejecting transcripts and there are less than $2^{c(s)+1}$ distinct
such transcripts, there exists a rejecting transcript $\tau^*$ such that at least a fraction
$1/2^{c(s)+1}$ of the tuples in $F^{t(s)}$ are mapped by $T$ to this particular $\tau^*$, that is, $|T^{-1}(\tau^*) \cap
F^{t(s)}| \geq |F|^{t(s)}/2^{c(s)+1}$. Now, each component of a tuple in $T^{-1}(\tau^*) \cap F^{t(s)}$ is covered
by $\tau^*$ since we can regard the input of $T$ as an unordered sequence. Thus, if we let $G$

denote the subset of $F$ that is covered by $\tau^*$, we have that $T^{-1}(\tau^*) \cap F^{t(s)} \subseteq G^{t(s)}$. We conclude that

$$|G|^{t(s)} \geq |T^{-1}(\tau^*) \cap F^{t(s)}| \geq |F|^{t(s)}/2^{c(s)+1},$$

whence $|G| \geq \varphi(s) \cdot |F|$ where $\varphi(s) = 1/2^{(c(s)+1)/t(s)}$.

Thus, every step covers a fraction at least $\varphi(s)$ of the remaining instances to be covered. Since there are at most $2^s$ instances of bitlength $s$ to begin with, after $\ell$ steps there are no more than $(1 - \varphi(s))^\ell \cdot 2^s \leq \exp(-\varphi(s)\ell) \cdot 2^s$ instances left to cover, so the process ends after $O(s/\varphi(s))$ steps. Now, $1/\varphi(s) = 2^{(c(s)+1)/t(s)}$ is polynomially bounded in $t(s)$ as long as $c(s) = O(t(s) \log t(s))$. Since each transcript as well as the running time of the proof system are polynomially bounded in $s$ and $t(s)$, for polynomially bounded $t(s)$ the resulting algorithm for $\overline{L}$ runs in NP/poly.

This finishes the proof for the case of deterministic protocols $P$. For cononondeterministic protocols we can define $T(x_1, \ldots, x_{t(s)})$ to be an arbitrary transcript of an execution on which $P$ produces the correct output. The check in step (2) now involves nondeterminism. The fact that $P$ has no valid rejecting executions for inputs $(x_1, \ldots, x_{t(s)})$ in OR($L$) guarantees the soundness of the proof system, and the existence of at least one valid rejecting execution of $P$ on an input $(x_1, \ldots, x_{t(s)})$ outside of OR($L$) guarantees completeness. The counting argument carries over verbatim. □

## 6.2. Other Applications

To illustrate the use of our oracle communication model, we describe two applications in the original framework of Bodlaender et al. [2009a].

For several NP-hard parameterized problems $L$ there exists a $\leq_m^p$-reduction from OR($L$) to $L$ that maps $t$ instances of size $s$ each to a single instance of $L$ of size poly($s$) $\cdot t^{1+o(1)}$ and parameter $k = $ poly($s$). For example, for problems like SAT and CLIQUE, such reductions follow from the disjoint union construction mentioned in the introduction. For certain other problems such reductions are more involved but still exist (see Binkele-Raible et al. [2012], Bodlaender et al. [2009b], Chen et al. [2011], Dom et al. [2009], Kratsch and Wahlström [2010, 2013], for examples). Whenever such reductions exist, Lemma 4 (via Lemma 1) implies that $L$ does not have an oracle communication protocol of cost poly($k$) unless coNP $\subseteq$ NP/poly. In particular, such problems do not have kernels of polynomial size unless coNP $\subseteq$ NP/poly.

*Turing Kernelizations.* Binkele-Raible et al. [2012] exhibit a parameterized problem that has no standard kernel of polynomial size unless coNP $\subseteq$ NP/poly, but does have a "Turing kernelization" of size $O(k^3)$ in the following sense: The problem has a self-reduction which, on inputs of size $s$ and parameter $k$, makes at most $s$ queries, all of which are of size $O(k^3)$. Using oracle communication protocols that implement general reductions rather than mapping reductions, Lemma 4 allows us to rule out the following for that problem, assuming coNP $\not\subseteq$ NP/poly: Reductions that, on inputs of size $s$ and parameter $k$, make at most $s^{1-\epsilon}$ queries for some positive real $\epsilon$ and only query instances of bitlength bounded by a polynomial in $k$. In particular, this shows that the number of queries in the Turing kernel of Binkele-Raible et al. [2012] is likely to be tight – reducing it from $s$ to $s^{1-\epsilon}$ for some positive real $\epsilon$ would collapse the polynomial-time hierarchy.

*Density of* NP-*Hard Languages.* Buhrman and Hitchcock [2008] showed that a language $S$ that contains no more than $2^{n^{o(1)}}$ strings of any length $n$ cannot be hard for NP under reductions that make $n^{1-\epsilon}$ queries for some positive real $\epsilon$ unless coNP $\subseteq$ NP/poly. The proof in Buhrman and Hitchcock [2008] is a modification of

the proof in Fortnow and Santhanam [2011]. As an illustration of the power of our oracle communication model, we show that this result immediately follows from Lemma 4 using an oracle that actively tries to extract enough information from the first player to decide the membership to $S$ of any query that the first player wants to make.

Suppose such an NP-hard language $S$ does exist and consider the reduction from SAT to $S$ that makes $n^{1-\epsilon}$ queries. Since the reduction runs in polynomial time, the size of the queries is bounded by $m = \text{poly}(n)$. Consider the lexicographic ordering of all strings of length up to $m$. The set $S$ breaks up this ordering into at most $2 \cdot |S \cap \{0, 1\}^{\leq m}| + 1$ intervals on which the membership to $S$ is constant. In order for the oracle to decide the membership to $S$ of a query, it suffices for the oracle to figure out which interval the query falls in. It can do so by running a binary search with the help of the first player, who knows the exact query. The binary search only takes $\log(2 \cdot |S \cap \{0, 1\}^{\leq m}| + 1)$ bits of communication from the first player to the oracle. Overall, this leads to a communication protocol for SAT of cost $O(n^{1-\epsilon} \cdot \log(2 \cdot |S \cap \{0, 1\}^{\leq m}| + 1)) = O(n^{1-\epsilon+o(1)})$. Combining this protocol with the $\leq_m^p$-reduction from OR(SAT) to SAT mentioned previously, we obtain an oracle communication protocol for OR(SAT) of cost $O(\text{poly}(s) \cdot t^{1-\epsilon+o(1)})$ on inputs consisting of $t$ instances of size $s$ each. As the latter quantity is $O(t \log t)$ for $t$ a sufficiently large polynomial in $s$, Lemma 4 implies that coNP $\subseteq$ NP/poly.

## 7. CONCLUSION

In this article, we introduced a model of communication that captures various settings of interest in the theory of computing. For NP-complete problems like $d$-SAT, $d$-VERTEX COVER, and $d$-CLIQUE we showed that trivial protocols are essentially optimal as function of the witness size, unless the polynomial-time hierarchy collapses. Under the hypothesis that the latter does not happen, the result implies tight lower bounds for parameters captured by the communication model, including the size of PCPs, and polynomial-time sparsification, kernelization, and lossy compression. Under stronger hypotheses, similar results hold for larger time bounds.

As directions for further research we suggest the development of more applications that exploit the full power of our oracle communication model, and to relax the hypothesis coNP $\not\subseteq$ NP/poly to P $\neq$ NP.

## APPENDIX
### Appendix: Behrend's Construction

We now prove Lemma 5, following an elegant construction due to Behrend [1946], which improves on the original construction due to Salem and Spencer [1942].

PROOF OF LEMMA 5. Let $p$ be a positive integer. We want to construct a set $A \subseteq \mathbb{Z}_p$ of size $p^{1-o(1)}$ that contains no nontrivial arithmetic progressions of length three over $\mathbb{Z}_p$.

For positive integers $d, m$ and a real $r$ to be chosen later, let $S_r \subseteq \mathbb{R}^d$ denote the $d$-dimensional sphere of radius $r$ restricted to vectors whose components are from $\mathbb{Z}_m$:

$$S_r = \left\{ (a_1, \ldots, a_d) \in \mathbb{Z}_m^d \;\middle|\; a_1^2 + \cdots + a_d^2 = r^2 \right\}.$$

The midpoint between any two distinct points $\vec{a}$ and $\vec{b}$ on a sphere is not itself on the sphere. This means that

$$\vec{a} + \vec{b} \neq 2\vec{c} \qquad\qquad \text{for all distinct } \vec{a}, \vec{b}, \vec{c} \in S_r. \qquad\qquad (2)$$

This is the type of property we need except that we want it for a subset of integers rather than vectors with integer coordinates. We can transform $S_r$ into a set of integers and maintain (2) by applying a linear mapping $\langle . \rangle : \mathbb{N}^d \to \mathbb{N}$ that is 1-to-1 on $\mathbb{Z}^d_{2m-1}$. Then the set $\langle S_r \rangle = \{\langle \vec{a} \rangle \mid \vec{a} \in S_r\}$ satisfies

$$\langle \vec{a} \rangle + \langle \vec{b} \rangle = \langle \vec{a} + \vec{b} \rangle \neq \langle 2\vec{c} \rangle = 2\langle \vec{c} \rangle \qquad \text{for all distinct } \langle \vec{a} \rangle, \langle \vec{b} \rangle, \langle \vec{c} \rangle \in \langle S_r \rangle. \qquad (3)$$

Moreover, if

$$\max_{\vec{a} \in \mathbb{Z}^d_{2m-1}} \langle \vec{a} \rangle < p, \qquad (4)$$

then $\langle S_r \rangle \subseteq \mathbb{Z}_p$ and (3) implies that

$$\langle \vec{a} \rangle + \langle \vec{b} \rangle \not\equiv 2\langle \vec{c} \rangle \mod p \qquad \text{for all distinct } \langle \vec{a} \rangle, \langle \vec{b} \rangle, \langle \vec{c} \rangle \in \langle S_r \rangle.$$

That is, $\langle S_r \rangle \subseteq \mathbb{Z}_p$ contains no nontrivial arithmetic progressions of length three over $\mathbb{Z}_p$.

We define the function $\langle . \rangle$ by interpreting a vector $\vec{a} = (a_1, \ldots, a_d) \in \mathbb{Z}^d_{2m-1}$ as a $d$-digit number in base $2m - 1$, that is, $\langle \vec{a} \rangle = \sum_{i=1}^{d} a_i (2m-1)^{i-1}$. This yields a linear function from $\mathbb{N}^d$ to $\mathbb{N}$ which is 1-to-1 on $\mathbb{Z}^d_{2m-1}$ and achieves a maximum value of $(2m-1)^d - 1$ on $\mathbb{Z}^d_{2m-1}$. Thus, (4) is satisfied if $(2m-1)^d \leq p$.

It remains to choose $d, r, m$ such that $(2m-1)^d \leq p$ and $|\langle S_r \rangle| = |S_r| \geq p^{1-o(1)}$. For this, note that the sets $S_r$ partition the set $\mathbb{Z}^d_m$. The number of $r$ for which $S_r$ has a non-empty intersection with $\mathbb{Z}^d_m$ is less than $dm^2$. By averaging, for each $m$ there exists an $r$ for which $|S_r| \geq |\mathbb{Z}^d_m|/(dm^2) = m^{d-2}/d$. Setting $d = \sqrt{\log p}$ and $m = 2^{\sqrt{\log p}-1}$ ensures that $(2m-1)^d \leq p$ and that $m^{d-2}/d = (2^{\sqrt{\log p}-1})^{(\sqrt{\log p}-2)}/\sqrt{\log p} \geq p^{1-O(1/\sqrt{\log p})}$. We set $r^*$ as the first $r$ for which $|S_r| \geq m^{d-2}/d$. We can compute $r^*$ and $\langle S_{r^*} \rangle$ in time polynomial in $p$. Thus, setting $A = \langle S_{r^*} \rangle$ satisfies all the requirements. □

We point out that the construction in the proof of Lemma 5 guarantees that the cardinality of the set $A$ is at least $p^{1-O(1/\sqrt{\log p})}$ rather than just $p^{1-o(1)}$. By considering a thin annulus rather than a sphere for the set $S$, Elkin [2011] and Green and Wolf [2010] recently further improved the cardinality by a factor of the form $\log^c p$ for some positive constant $c$. However, the analysis becomes more complicated and Behrend's already gives us more than we need.

## REFERENCES

Achlioptas, D. and Moore, C. 2007. Random $k$-SAT: Two moments suffice to cross a sharp threshold. *SIAM J. Comput. 36*, 3, 740–762.

Achlioptas, D. and Peres, Y. 2004. The threshold for random $k$-SAT is $2^k \log 2 - O(k)$. *J. Amer. Math. Soc. 17*, 4, 947–973.

Alon, N. 2002. Testing subgraphs in large graphs. *Rand. Struct. Algor. 21*, 3–4, 359–370.

Alon, N., Fischer, E., Krivelevich, M., and Szegedy, M. 2000. Efficient testing of large graphs. *Combinatorica 20*, 4, 451–476.

Alon, N., Kaufman, T., Krivelevich, M., and Ron, D. 2008. Testing triangle-freeness in general graphs. *SIAM J. Disc. Math. 22*, 2, 786–819.

Alon, N. and Shapira, A. 2004. Testing subgraphs in directed graphs. *J. Comput. Syst. Sci. 69*, 3, 354–382.

Alon, N. and Shapira, A. 2005. Linear equations, arithmetic progressions and hypergraph property testing. *Theory Comput. 1*, 1, 177–216.

Alon, N. and Shapira, A. 2006. A characterization of easily testable induced subgraphs. *Combinat. Probab. Comput. 15*, 6, 791–805.

Arora, S. and Barak, B. 2009. *Computational Complexity: A Modern Approach*. Cambridge University Press, New York.

Behrend, F. A. 1946. On sets of integers which contain no three terms in arithmetic progression. *Proc. Nat. Acad. Sci. 32*, 12, 331–332.

Binkele-Raible, D., Fernau, H., Fomin, F. V., Lokshtanov, D., Saurabh, S., and Villanger, Y. 2012. Kernel(s) for problems with no kernel: On out-trees with many leaves. *ACM Trans. Algor. 8*, 4, 38.

Bodlaender, H. L., Downey, R. G., Fellows, M. R., and Hermelin, D. 2009a. On problems without polynomial kernels. *J. Comput. Syst. Sci. 75*, 8, 423–434.

Bodlaender, H. L., Thomassé, S., and Yeo, A. 2009b. Kernel bounds for disjoint cycles and disjoint paths. In *Proceedings of the 17th Annual European Symposium on Algorithms (ESA'09)*. Lecture Notes in Computer Science, vol. 5757, Springer, 635–646.

Buhrman, H. and Hitchcock, J. M. 2008. NP-hard sets are exponentially dense unless NP is contained in coNP/poly. In *Proceedings of the 23rd IEEE Conference on Computational Complexity (CCC'08)*. IEEE Computer Society, 1–7.

Buss, J. F. and Goldsmith, J. 1993. Nondeterminism within P. *SIAM J. Comput. 22*, 3, 560–572.

Calabro, C., Impagliazzo, R., and Paturi, R. 2006. A duality between clause width and clause density for SAT. In *Proceedings of the 21st IEEE Conference on Computational Complexity (CCC'06)*. IEEE Computer Society, 252–260.

Chandra, A. K., Furst, M. L., and Lipton, R. J. 1983. Multi-party protocols. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC'83)*. ACM, 94–99.

Chen, Y., Flum, J., and Müller, M. 2007. Lower bounds for kernelizations. Tech report TR07-137, Electronic Colloquium on Computational Complexity (ECCC).

Chen, Y., Flum, J., and Müller, M. 2011. Lower bounds for kernelizations and other preprocessing procedures. *Theory Comput. Syst. 48*, 4, 803–839.

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

Coppersmith, D. and Winograd, S. 1990. Matrix multiplication via arithmetic progressions. *J. Symb. Computat. 9*, 3, 251–280.

Davie, A. M. and Stothers, A. J. 2013. Improved bound for complexity of matrix multiplication. *Proc. Roy. Soc. Edinburgh: Sect. A, Mathematics 143*, 2, 351–369.

Dell, H. and Marx, D. 2012. Kernelization of packing problems. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. SIAM, 68–81.

Dinur, I. 2007. The PCP theorem by gap amplification. *J. ACM 54*, 3, 12.

Dom, M., Lokshtanov, D., and Saurabh, S. 2009. Incompressibility through colors and IDs. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP'09)*. Lecture Notes in Computer Science, vol. 5555, Springer, 378–389.

Downey, R. G. and Fellows, M. R. 1999. *Parameterized Complexity*. Springer New York.

Drucker, A. 2012. New limits to classical and quantum instance compression. In *Proceedings of the 53rd Annual Symposium on Foundations of Computer Science (FOCS'12)*. 609–618.

Elkin, M. 2011. An improved construction of progression-free sets. *Israel J. Math. 184*, 1, 93–128.

Erdős, P. and Rado, R. 1960. Intersection theorems for systems of sets. *J. Lond. Math. Soc. 35*, 85–90.

Fellows, M. R., Guo, J., Moser, H., and Niedermeier, R. 2011. A generalization of Nemhauser and Trotter's local optimization theorem. *J. Comput. Syst. Sci. 77*, 6, 1141–1158.

Flum, J. and Grohe, M. 2006. *Parameterized Complexity Theory*. Springer.

Fortnow, L. and Santhanam, R. 2011. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci. 77*, 1, 91–106.

Friedgut, E. and Bourgain, J. 1999. Sharp thresholds of graph properties, and the $k$-SAT problem. *J. Amer. Math. Soc. 12*, 4, 1017–1054.

Goldreich, O. 2008. *Computational Complexity: A Conceptual Perspective*. ACM New York.

Goldreich, O., Goldwasser, S., and Ron, D. 1998. Property testing and its connection to learning and approximation. *J. ACM 45*, 4, 653–750.

Green, B. and Wolf, J. 2010. A note on Elkins improvement of Behrends construction. In *Additive Number Theory*, Springer, 141–144.

Guo, J. and Niedermeier, R. 2007. Invitation to data reduction and problem kernelization. *SIGACT News 38*, 1, 31–45.

Harnik, D. and Naor, M. 2010. On the compressibility of NP instances and cryptographic applications. *SIAM J. Comput. 39*, 5, 1667–1713.

Håstad, J. and Wigderson, A. 2003. Simple analysis of graph tests for linearity and PCP. *Rand. Struct. Algor. 22*, 2, 139–160.

Hemaspaandra, L. A., Hoene, A., Naik, A. V., Ogihara, M., Selman, A. L., Thierauf, T., and Wang, J. 1995. Nondeterministically selective sets. *Int. J. Found. Comput. Sci. 6*, 4, 403–416.

Impagliazzo, R., Paturi, R., and Zane, F. 2001. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci. 63*, 4, 512–530.

Karp, R. M. 1972. Reducibility among combinatorial problems. *Complex. Comput. Computat. 43*, 85–103.

Ko, K.-I. 1983. On self-reducibility and weak P-selectivity. *J. Comput. Syst. Sci. 26*, 2, 209–221.

Kratsch, S. 2012. Co-nondeterminism in compositions: A kernelization lower bound for a Ramsey-type problem. In *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. SIAM, 114–122.

Kratsch, S., Philip, G., and Ray, S. 2014. Point line cover: The easy kernel is essentially tight. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'14)*. SIAM, 1596–1606.

Kratsch, S. and Wahlström, M. 2010. Preprocessing of min ones problems: A dichotomy. In *Proceedings of the 37th Colloquium on Automata, Languages and Programming (ICALP'10)*. Lecture Notes in Computer Science, vol. 6198, Springer, 653–665.

Kratsch, S. and Wahlström, M. 2013. Two edge modification problems without polynomial kernels. *Disc. Optim.*

Krishnamoorthy, M. S. and Deo, N. 1979. Node-deletion NP-complete problems. *SIAM J. Comput. 8*, 4, 619–625.

Levin, L. A. 1973. Universal search problems (Russian: Universal'nye perebornye zadachi). *Prob. Inf. Trans.* (Russian: Problemy Peredachi Informatsii) *9*, 3, 265–266.

Lewis, J. M. and Yannakakis, M. 1980. The node-deletion problem for hereditary properties is NP-complete. *J. Comput. Syst. Sci. 20*, 2, 219–230.

Niedermeier, R. 2006. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, Oxford, UK.

Ruzsa, I. Z. and Szemerédi, E. 1978. Triple systems with no six points carrying three triangles. In *Combinatorics (Proceedings of the 5th Hungarian Colloquium, Keszthely, 1976), Vol. II*, Colloquia Mathematica Societatis János Bolyai, vol. 18, North-Holland, Amsterdam, 939–945.

Salem, R. and Spencer, D. C. 1942. On sets of integers which contain no three terms in arithmetical progression. *Proc. Nat. Acad. Sci. 28*, 12, 561–563.

Stothers, A. J. 2010. On the complexity of matrix multiplication. Ph.D. thesis, University of Edinburgh.

Thomassé, S. 2010. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algor. 6*, 2, 32:1–32:8.

Vassilevska Williams, V. 2012. Multiplying matrices faster than Coppersmith–Winograd. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC'12)*. ACM, 887–898.

Wegener, I. 1987. *The Complexity of Boolean Functions*. B. G. Teubner, and John Wiley & Sons.

Yap, C.-K. 1983. Some consequences of non-uniform conditions on uniform classes. *Theoret. Comput. Sci. 26*, 3, 287–300.