Satisfiability Allows No Nontrivial Sparsification Unless The Polynomial-Time Hierarchy Collapses

Holger Dell*
Humboldt University of Berlin, Germany
dell@informatik.hu-berlin.de

Dieter van Melkebeek†
University of Wisconsin-Madison, USA
dieter@cs.wisc.edu

ABSTRACT

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 cooperatively decide 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 ϵ 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, the above 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

*Supported by the Deutsche Forschungsgemeinschaft within the research training group "Methods for Discrete Structures" (GRK 1408).

†Research mostly done while visiting the Humboldt University of Berlin. Partially supported by the Humboldt Foundation and by NSF award CCF-0728809.

STOC'10, June 5–8, 2010, Cambridge, Massachusetts, USA.

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 $d \geq 3$ [15, 35, 31].

In this paper 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 [25, 1, 2]. 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. [30] 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 ϵ?

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 [28] 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 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 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 [18] 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 $O(n^{d-\epsilon})$?

Our Results for Satisfiability.

We give evidence that the answer to all four of the above 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 [40].

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, i.e., 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 φ, but such that (i) if φ is satisfiable then every valid execution comes to that conclusion, and (ii) if φ is not satisfiable then at least one valid execution comes to that conclusion.

Theorem 1. *Let $d \geq 3$ be an integer and ϵ 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. The above 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.

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 ϵ 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 $d = 2$, i.e., 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 [11] came up with a kernelization avant la lettre. 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 [27] 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 Π, the corresponding vertex deletion problem asks whether removing at most k vertices from a given graph G can yield a graph that satisfies Π. A host of well-studied specific problems can be cast as the vertex deletion problem corresponding to some graph property Π 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 Π or if all graphs satisfy Π, the vertex deletion problem is trivially decidable in polynomial time. For all other graph properties Π that are inherited by subgraphs, Lewis and Yannakakis [36] 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 Π be a graph property that is inherited by subgraphs, and is satisfied by infinitely many but not all graphs. Let ϵ 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 Π 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 Thomassé's recent kernel [39] does the job; for bounded-degree deletion a kernel with $O(k^2)$ edges was known to exist [21].

Techniques and Related Work.

At a high level our approach refines the framework developed by Bodlaender et al. [9] to show that certain parameterized NP-hard problems are unlikely to have kernels of polynomial size. Harnik and Naor [28] realized the connection between their notion of lossy compression and kernelization and PCPs for satisfiability of general Boolean formulas, and Fortnow and Santhanam [24] proved the connection with the hypothesis coNP $\not\subseteq$ NP/poly in the superpolynomial setting. Several authors subsequently applied the framework in that setting [14, 10, 19, 22, 32, 33].

We develop the first application of the framework in the polynomial setting, i.e., 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. In order to obtain these tight results, a crucial new ingredient is the use of high-density subsets of the integers without nontrivial arithmetic progressions of length three.

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 the number of vertices. The proof consists of two steps.

Step 1. Assuming the clique problem on n-vertex d-uniform hypergraphs has a protocol of cost $O(n^c)$ for some constant $c < d$, some NP-complete language L has the following property: The problem $\mathrm{OR}(L)$ of deciding whether at least one of t given instances $x_1, x_2, \ldots, x_t$ is in L has a protocol of cost $O(t \log t)$ for instances where t is a sufficiently large polynomial in $s = \max_{1 \leq i \leq t} |x_i|$.

Step 2. Any language L with that property is in coNP/poly.

Combining the two steps we conclude that the hypothesis of Step 1 fails unless coNP $\subseteq$ NP/poly.

Since the clique problem on d-uniform hypergraphs is NP-complete for any integer $d \geq 2$, without loss of generality we

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

can take L in Step 1 to be this language. In order to obtain a low-cost protocol for OR(L) it suffices 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, and then run the presumed protocol of cost $O(n^c)$ on the latter hypergraph.

As observed by Harnik and Naor [28], the disjoint union of the given hypergraphs provides such a reduction. However, the number of vertices is $n = s \cdot t$, so even for $c = 1$ the cost of the resulting protocol for OR(L) is $\omega(t \log t)$, which is too much for Step 2. As a critical piece in our proof, we present a reduction that works for an NP-hard subset of clique instances and only needs $n = s \cdot t^{1/d+o(1)}$ vertices. The cost of the resulting protocol for OR(L) then goes down to $O(n^c) = O\big((s \cdot t^{1/d+o(1)})^c\big)$, which is $O(t \log t)$ for sufficiently large polynomials $t(s)$ as $c < d$.

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 [13], the asymptotically best known algorithm for matrix multiplication [16], the soundness analysis of graph tests for linearity [29], and lower bounds for property testing [4, 3, 8, 6, 5, 7]. The latter two applications as well as ours implicitly or explicitly rely on a connection due to Ruzsa and Szemerédi [37] 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 [7] in the context of property testing. We refer to Section 4 for a more detailed discussion of the relationships.

Step 2 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 Step 2 as the Complementary Witness Lemma. It involves a refined analysis and generalization of a result by Fortnow and Santhanam [24] 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.

Organization.

We review some preliminaries in Section 2. Section 3 contains the proof of the main result (Theorem 2) following the two-step approach outlined above, and fleshes out the first step modulo the packing construction. We develop the latter in Section 4. Section 5 expands on the implications for satisfiability (Theorem 1 and its corollaries) and for vertex deletion problems (Theorem 3). We provide the intuition underlying our results and the most important formalities here, but refer to [17] for more discussion and omitted proofs.

2. PRELIMINARIES

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 $x_i \in L$. A *mapping reduction*, or $\leq_m^p$*-reduction*, from L to L' is a polynomial-time 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) consisting of a language L and a parameterization $k : \{0,1\}^* \to \mathbb{N}$ 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.

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, i.e., 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, i.e., a truth assignment to its variables that makes the formula evaluate to true.

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.

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 ϵ 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 two-step approach outlined in the introduction. In the first step we use a presumed low-cost protocol for d-Clique to devise a low-cost protocol for the OR of some NP-complete language L. We do so by first translating the given instance of OR(L) into an equivalent instance of d-Clique with few vertices, and then running the presumed low-cost protocol for d-Clique on that instance.

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 k such that at least one of the φ_i's is satisfiable if and only if G has a clique of size at least k. 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 φ_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 1 (Packing Lemma). *For any integers $s \geq d \geq 2$ and $t > 0$ there exists a d-uniform hypergraph P on $O\big(s \cdot \max(s, t^{1/d+o(1)})\big)$ 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 1 formalizes the notion of a packing. The part that P contains the t cliques K_i ensures the completeness of the reduction, i.e., 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, i.e., 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 1 to Section 4. Using it as sketched above we obtain the following reduction.

Lemma 2. *For any integer $d \geq 2$, there is a $\leq^p_m$-reduction from* OR(3-SAT) *to* d-CLIQUE *that maps t-tuples of instances of bitlength s each to instances on $O\big(s \cdot \max(s, t^{1/d+o(1)})\big)$ 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 1. Along the lines of the standard reduction from 3-SAT to 2-CLIQUE [31], we first translate the 3-CNF formulas φ_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 φ_i: The first component selects a clause from φ_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 φ_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 φ_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 φ_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 1, $K' = K_i$ for some $i \in [t]$. Moreover, by property (i) of Lemma 1, 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 φ_i is satisfiable.

Thus, $(G, s) \in d$-CLIQUE if and only if $(\varphi_1, \ldots, \varphi_t) \in$ OR(3-SAT). Since G and s are computable in time polynomial in the bitlength of $(\varphi_1, \ldots, \varphi_t)$ and $|V(G)| \leq 3|V(P)| \leq O\big(s \cdot \max(s, t^{1/d+o(1)})\big)$, we have established the $\leq^p_m$-reductions claimed in Lemma 2. ∎

Lemma 2 represents the essence of the first step of the proof of Theorem 2 – obtaining a low-cost protocol for OR(3-SAT) out of a low-cost protocol for d-CLIQUE. The second step shows in general how to use a low-cost protocol for OR(L) to build a proof system with advice for $\overline{L}$. That step is captured in the following lemma.

Lemma 3 (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 $L \in$* coNP/poly.

The proof and further discussion of Lemma 3 can be found in [17]. Theorem 2 follows by combining Lemma 2 with Lemma 3.

4. THE PACKING LEMMA

In this section we establish Lemma 1, which is a critical ingredient in the proof of Theorem 2. We first develop the construction for the case $d = 2$, i.e., 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.

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 partitions 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 1a. 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,

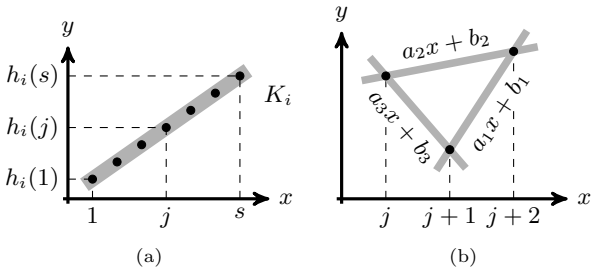

Figure 1: (a) The placement of one of the K_i's. (b) Triangle on three consecutive abscissae.

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, i.e., there exists a function $h : [s] \to \mathbb{F}_p$ 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.

Consider three consecutive columns, j, $j+1$, and $j+2$, and the triangle that K induces between them – see Figure 1b, 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 1b 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 $2a_2 = a_1 + a_3$, 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, i.e., $a_1 = a_2 = a_3$. The latter implies that the three lines in Figure 1b 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 4 ($\mathbf{AP_3}$-Free Sets [38]). *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.*

The resulting graph P has $s \cdot p$ vertices where $p = O\big(\max(s, \sqrt{t}^{1+o(1)})\big)$.

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 1. 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 4.

Proof (of Lemma 1). 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 4. We have that $p = O\big(\max(s, t^{1/d+o(1)})\big)$ 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 ith 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_ℓ 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 1b, 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 $\Delta_{j+d}(q_k)$, i.e., $q'_k : [j, j+d-1] \to \mathbb{F}_p$ 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'_ℓ agree on each $x \in [j, j+d-1] \setminus \{k, \ell\}$ because the polynomials q_k and q_ℓ 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 $[j, j+d-1]$. ∎

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 [37] 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 $2a_2 = a_1 + a_3$, as captured in Figure 1b, 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 [29] 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 ϵ-tester for this property accepts all graphs that are F-free, and rejects all graphs that are at least ϵ away from being F-free, i.e., from which at least ϵn^2 edges need to be removed to make it F-free [26]. 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 [3]. 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 [3, 8, 6, 5, 7] constructed such graphs G for various F based on [37].

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 develop in [7]. 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.

[2]This allows us to relax the condition $q(\epsilon) = \max\{m : (f(m))^k \geq \epsilon\}$ in Lemma 4.1 of [7] to $q(\epsilon) = \max\{m : f(m) \geq \epsilon\}$.

5. CONSEQUENCES OF MAIN THEOREM

Our lower bound for oracle communication protocols for d-Vertex Cover, Theorem 2, has two types of consequences. The first are 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 5. *For every $d \geq 3$, there is a $\leq^p_m$-reduction from d-Vertex Cover to d-Sat that maps d-uniform hypergraphs on n vertices to d-CNF formulas on $O(n)$ variables.*

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 $\leq^p_m$-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 the paper refers to, and contrasts the situation at the subexponential-time level. The sparsification lemma of [30] gives a reduction which, on input an n-variable d-CNF formula and a rational $\epsilon > 0$, runs in time $2^{\epsilon n} \cdot \text{poly}(n)$ 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 $(d/\epsilon)^{3d}$ [12]. 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 $\leq^p_m$-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 [18] constructed such PCPs of size $O(s \cdot \text{poly} \log s)$, where s denotes the bitlength of the formula. Based on a connection due to Harnik and Naor between PCPs and lossy compression [28], Fortnow and Santhanam [24] 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 ϵ 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 multi-valued 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 φ 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 φ 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 φ, 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 φ, and outputs their conjunction. The conjunction has bitlength $O(n^c \log n)$, is always satisfiable if φ is, and is not satisfiable on at least one computation path otherwise. ∎

5.2 Vertex Cover and 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 $d = 2$, i.e., 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* Π*. The problem* Π-Vertex Deletion *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* Π*.*

We say that a graph property Π is inherited by subgraphs if whenever a graph G satisfies Π, every subgraph of G also satisfies Π. The following natural graph problems are special cases of Π-Vertex Deletion for a graph property Π that is inherited by subgraphs.

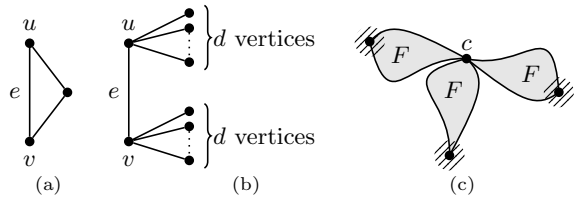

Figure 2: (a) and (b) Replacement of an edge $e = \{u, v\}$ in the transformation from G to G' in the proof of Lemma 6 for Feedback Vertex Set and Bounded-Degree Deletion, respectively. (c) Connected component C' that might remain after removing a vertex cover S of G from the naïve construction of G', centered around a vertex c that has degree 3 in G and does not belong to S.

- Vertex Cover: Can we delete k vertices to destroy all edges?
- 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 $\leq^p_m$-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 [31]. 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 2a. 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 [34], d new edges are attached to every vertex of G (see Figure 2b). 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 the above 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 2c, 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 component could contain a copy of F, in which case $G' \setminus S$ would not satisfy Π. 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 Π then does the job as long as all minimal graphs violating Π are connected.

More generally, consider a graph F violating Π whose most complex connected component C is as simple as possible among all graphs violating Π. 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 Π 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 Π 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 Π-Vertex Deletion is as strong as for Vertex Cover modulo a constant factor.

The simplicity measure we use is the same as in [36] but the construction is a bit different. The construction in [36] blows up the parameter k' to $\Theta(nk)$. A straightforward modification reduces k' to $\Theta(k^2)$. We further reduce k' to $\Theta(k)$ using a matching argument.

Lemma 6. *Let Π be a graph property that is inherited by subgraphs, and is satisfied by infinitely many but not all graphs. There is a $\leq^p_m$-reduction from* Vertex Cover *to* Π-Vertex Deletion *that maps instances with parameter k to instances with parameter $O(k)$.*

The proof of Lemma 6 can be found in [17]. Theorem 3 then follows by combining Lemma 6 with Theorem 2.

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 [39]. 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 [23, 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 [20] 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 Π that are inherited by subgraphs and for which the deletion problem is nontrivial. This is because the property Π could only depend on the primal graph underlying the hypergraph, for which protocols of cost $O(k^2)$ are known in some cases.

6. CONCLUSION

In this paper 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.

Acknowledgments

We would like to thank the following people for discussions, comments, pointers to the literature, and guidance: Matt Anderson, Albert Atserias, Kord Eickmeyer, Martin Grohe, Johan Håstad, Danny Hermelin, Daniel Lokshtanov, Moritz Müller, Saket Saurabh, Mathias Schacht, Asaf Shapira, Luca Trevisan, Chris Umans, Thomas Watson, Dalibor Zelený.

7. REFERENCES

[1] D. Achlioptas and C. Moore. Random k-SAT: two moments suffice to cross a sharp threshold. *SICOMP*, 36(3):740–762, 2007.

[2] D. Achlioptas and Y. Peres. The threshold for random k-SAT is $2^k \log 2 - O(k)$. *Journal of the AMS*, 17(4):947–973, 2004.

[3] N. Alon. Testing subgraphs in large graphs. *RSA*, 21(3–4):359–370, 2002.

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

[5] N. Alon, T. Kaufman, M. Krivelevich, and D. Ron. Testing triangle-freeness in general graphs. *SIAM Journal on Discrete Mathematics*, 22(2):786–819, 2008.

[6] N. Alon and A. Shapira. Testing subgraphs in directed graphs. *JCSS*, 69(3):354–382, 2004.

[7] N. Alon and A. Shapira. Linear equations, arithmetic progressions and hypergraph property testing. *Theory of Computing*, 1(1):177–216, 2005.

[8] N. Alon and A. Shapira. A characterization of easily testable induced subgraphs. *Combinatorics, Probability and Computing*, 15(6):791–805, 2006.

[9] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *JCSS*, 75(8):423–434, 2009.

[10] H. L. Bodlaender, S. Thomassé, and A. Yeo. Kernel bounds for disjoint cycles and disjoint paths. In *ESA*, pages 635–646, 2009.

[11] J. F. Buss and J. Goldsmith. Nondeterminism within P. *SICOMP*, 22(3):560–572, 1993.

[12] C. Calabro, R. Impagliazzo, and R. Paturi. A duality between clause width and clause density for SAT. In *CCC*, pages 252–260, 2006.

[13] A. K. Chandra, M. L. Furst, and R. J. Lipton. Multi-party protocols. In *STOC*, pages 94–99, 1983.

[14] Y. Chen, J. Flum, and M. Müller. Lower bounds for kernelizations. *ECCC*, 14(137), 2007.

[15] S. A. Cook. The complexity of theorem-proving procedures. In *STOC*, pages 151–158, 1971.

[16] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.

[17] H. Dell and D. van Melkebeek. Satisfiability allows no nontrivial sparsification unless the polynomial-time hierarchy collapses. *ECCC*, 17(38), 2010.

[18] I. Dinur. The PCP theorem by gap amplification. *JACM*, 54(3):12, 2007.

[19] M. Dom, D. Lokshtanov, and S. Saurabh. Incompressibility through colors and IDs. In *ICALP*, pages 378–389, 2009.

[20] P. Erdős and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35:85–90, 1960.

[21] M. R. Fellows, J. Guo, H. Moser, and R. Niedermeier. A generalization of Nemhauser and Trotter's local optimization theorem. In *STACS*, pages 409–420, 2009.

[22] H. Fernau, F. V. Fomin, D. Lokshtanov, D. Raible, S. Saurabh, and Y. Villanger. Kernel(s) for problems with no kernel: On out-trees with many leaves. In *STACS*, pages 421–432, 2009.

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

[24] L. Fortnow and R. Santhanam. Infeasibility of instance compression and succinct PCPs for NP. In *STOC*, pages 133–142, 2008.

[25] E. Friedgut and J. Bourgain. Sharp thresholds of graph properties, and the k-SAT problem. *Journal of the AMS*, 12(4):1017–1054, 1999.

[26] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *JACM*, 45(4):653–750, 1998.

[27] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.

[28] D. Harnik and M. Naor. On the compressibility of NP instances and cryptographic applications. In *FOCS*, pages 719–728, 2006.

[29] J. Håstad and A. Wigderson. Simple analysis of graph tests for linearity and PCP. *RSA*, 22(2):139–160, 2003.

[30] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *JCSS*, 63(4):512–530, 2001.

[31] R. M. Karp. Reducibility among combinatorial problems. *Complexity of computer computations*, 43:85–103, 1972.

[32] S. Kratsch and M. Wahlström. Preprocessing of min ones problems: A dichotomy. *Arxiv Preprint*, 2009.

[33] S. Kratsch and M. Wahlström. Two edge modification problems without polynomial kernels. In *IWPEC*, pages 264–275, 2009.

[34] M. S. Krishnamoorthy and N. Deo. Node-deletion NP-complete problems. *SICOMP*, 8(4):619–625, 1979.

[35] L. A. Levin. Universal search problems. *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.

[36] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *JCSS*, 20(2):219–230, 1980.

[37] I. Z. Ruzsa and E. Szemerédi. Triple systems with no six points carrying three triangles. In *Combinatorics, Vol. II*, volume 18 of *Colloquia Mathematica Societatis János Bolyai*, pages 939–945. North-Holland, 1978.

[38] R. Salem and D. C. Spencer. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences, USA*, 28(12):561–563, 1942.

[39] S. Thomassé. A quadratic kernel for feedback vertex set. In *SODA*, pages 115–119, 2009.

[40] C.-K. Yap. Some consequences of non-uniform conditions on uniform classes. *TCS*, 26(3):287–300, 1983.