On the Parameterized Complexity of Approximating Dominating Set

KARTHIK C. S., Weizmann Institute of Science, Israel
BUNDIT LAEKHANUKIT, Shanghai University of Finance and Economics, China
PASIN MANURANGSI, University of California, Berkeley, USA

We study the parameterized complexity of approximating the k-Dominating Set (DomSet) problem where an integer k and a graph G on n vertices are given as input, and the goal is to find a dominating set of size at most $F(k) \cdot k$ whenever the graph G has a dominating set of size k. When such an algorithm runs in time $T(k) \cdot \text{poly}(n)$ (i.e., FPT-time) for some computable function T, it is said to be an *$F(k)$-FPT-approximation algorithm* for k-DomSet. Whether such an algorithm exists is listed in the seminal book of Downey and Fellows (2013) as one of the "most infamous" open problems in parameterized complexity. This work gives an almost complete answer to this question by showing the non-existence of such an algorithm under W[1] $\neq$ FPT and further providing tighter running time lower bounds under stronger hypotheses. Specifically, we prove the following for every computable functions T, F and every constant $\varepsilon > 0$:

- Assuming W[1] $\neq$ FPT, there is no *$F(k)$-FPT-approximation algorithm* for k-DomSet.
- Assuming the Exponential Time Hypothesis (ETH), there is no $F(k)$-approximation algorithm for k-DomSet that runs in $T(k) \cdot n^{o(k)}$ time.
- Assuming the Strong Exponential Time Hypothesis (SETH), for every integer $k \geq 2$, there is no $F(k)$-approximation algorithm for k-DomSet that runs in $T(k) \cdot n^{k-\varepsilon}$ time.
- Assuming the k-SUM Hypothesis, for every integer $k \geq 3$, there is no $F(k)$-approximation algorithm for k-DomSet that runs in $T(k) \cdot n^{\lceil k/2 \rceil - \varepsilon}$ time.

Previously, only constant ratio FPT-approximation algorithms were ruled out under W[1] $\neq$ FPT and $(\log^{1/4-\varepsilon} k)$-FPT-approximation algorithms were ruled out under ETH [Chen and Lin, FOCS 2016]. Recently, the non-existence of an $F(k)$-FPT-approximation algorithm for any function F was shown under Gap-ETH [Chalermsook et al., FOCS 2017]. Note that, to the best of our knowledge, no running time lower bound of the form $n^{\delta k}$ for any absolute constant $\delta > 0$ was known before even for any constant factor inapproximation ratio.

Our results are obtained by establishing a connection between communication complexity and hardness of approximation, generalizing the ideas from a recent breakthrough work of Abboud et al. [FOCS 2017].

Karthik C. S. is supported by ERC-CoG grant no. 772839, BSF grant no. 2014371, and ISF-UGC grant no. 1399/14. Bundit Laekhanukit is partially supported by ISF grant no. 621/12, I-CORE grant no. 4/11 and by the DIMACS/Simons Collaboration on Bridging Continuous and Discrete Optimization through NSF grant no. CCF-1740425. Pasin Manurangsi is supported by NSF grants no. CCF 1540685 and CCF 1655215, and by ISF-UGC grant no. 1399/14. Parts of the work were done while all the authors were at the Weizmann Institute of Science and while the second author was visiting the Simons Institute for the Theory of Computing.

Authors' addresses: Karthik C. S., Department of Applied Mathematics and Computer Science, Weizmann Institute of Science, 234 Herzl street, Rehovot, Israel - 7610001; email: karthik.srikanta@weizmann.ac.il; B. Laekhanukit, Institute for Theoretical Computer Science, School of Information Management and Engineering, Shanghai University of Finance and Economics, 100 Wudong Street, Yangpu, Shanghai, 200433 China; email: bundit@sufe.edu.cn; P. Manurangsi, Department of Electrical Engineering and Computer Sciences, UC Berkeley, CA 94720, USA; email: pasin@berkeley.edu.

0004-5411/2019/08-ART33 $15.00
https://doi.org/10.1145/3325116

Specifically, we show that to prove hardness of approximation of a certain parameterized variant of the label cover problem, it suffices to devise a specific protocol for a communication problem that depends on which hypothesis we rely on. Each of these communication problems turns out to be either a well-studied problem or a variant of one; this allows us to easily apply known techniques to solve them.

CCS Concepts: • **Theory of computation → Parameterized complexity and exact algorithms**; Problems, reductions and completeness; Communication complexity;

Additional Key Words and Phrases: Parameterized inapproximability, dominating set, set cover

ACM Reference format:
Karthik C. S., Bundit Laekhanukit, and Pasin Manurangsi. 2019. On the Parameterized Complexity of Approximating Dominating Set. *J. ACM* 66, 5, Article 33 (August 2019), 38 pages.
https://doi.org/10.1145/3325116

1 INTRODUCTION

In the *dominating set* problem (DomSet), we are given an undirected graph G on n vertices and an integer k, and the goal is to decide whether there is a subset of vertices $S \subseteq V(G)$ of size k such that every vertex outside S has a neighbor in S (i.e., S *dominates* every vertex in G and is thus called a *dominating set*). Often regarded as one of the classical problems in computational complexity, DomSet was first shown to be NP-complete in the seminal work of Karp [49].[1] Thus, its optimization variant, namely the *minimum dominating set*, where the goal is to find a dominating set of smallest possible size, is also NP-hard. To circumvent this apparent intractability of the problem, the study of an approximate version was initiated. The quality of an approximation algorithm is measured by the *approximation ratio*, which is the ratio between the size of the solution output by an algorithm and the size of the minimum dominating set. A simple greedy heuristic for the problem, which has by now become one of the first approximation algorithms taught in undergraduate and graduate algorithm courses, was intensively studied and was shown to yield a $(\ln n - \ln \ln n + \Theta(1))$-approximation for the problem [24, 45, 56, 73, 74]. On the opposite side, a long line of works in hardness of approximation [6, 36, 57, 60, 67] culminated in the work of Dinur and Steurer [29], who showed that obtaining a $(1-\varepsilon)\ln n$-approximation for the problem is NP-hard for every $\varepsilon > 0$. This essentially settles the approximability of the problem.

Besides approximation, another widely used technique to cope with NP-hardness is *parameterization*. The parameterized version of DomSet, which we will refer to simply as k-DomSet, is exactly the same as the original decision version of the problem except that now we are not looking for a polynomial time algorithm but rather a *fixed parameter tractable* (FPT) algorithm—one that runs in time $T(k) \cdot \mathrm{poly}(n)$ for some computable function T (e.g., $T(k) = 2^k$ or 2^{2^k}). Such running time will henceforth be referred to as *FPT time*. Alas, even with this relaxed requirement, k-DomSet still remains intractable: In the same work that introduced the W-hierarchy, Downey and Fellows [30] showed that k-DomSet is complete for the class W[2], which is generally believed to not be contained in FPT, the class of fixed parameter tractable problems. In the ensuing years, stronger running time lower bounds have been shown for k-DomSet under strengthened assumptions. Specifically, Chen et al. [20] ruled out $T(k) \cdot n^{o(k)}$-time algorithm for k-DomSet assuming the *Exponential Time Hypothesis* (ETH)[2]. Furthermore, Pătraşcu and Williams [64] proved, for every $k \geq 2$, that, under the *Strong Exponential Time Hypothesis* (SETH),[3] not even an $O(n^{k-\varepsilon})$ algorithm

[1] To be precise, Karp showed NP-completeness of Set Cover, which is well known to be equivalent to DomSet.
[2] ETH [43] states that no subexponential time algorithm can solve 3-CNF-SAT; see Hypothesis 3.3.
[3] SETH [43], a strengthening of ETH, states that, for every $\varepsilon > 0$, there exists $k = k(\varepsilon) \in \mathbb{N}$ such that no $O(2^{(1-\varepsilon)n})$-time algorithm can solve k-CNF-SAT; see Hypothesis 3.4.

exists for k-DomSet for any $\varepsilon > 0$. Note that the trivial algorithm that enumerates through every k-size subset and checks whether it forms a dominating set runs in $O(n^{k+1})$ time. It is possible to speed up this running time using fast matrix multiplication [35, 64]. In particular, Pătraşcu and Williams [64] themselves also gave an $n^{k+o(1)}$-time algorithm for every $k \geq 7$, painting an almost complete picture of the complexity of the problem.

Given the strong negative results for k-DomSet discussed in the previous paragraph, it is natural to ask whether we can somehow incorporate the ideas from the area of approximation algorithms to come up with a *fix parameter approximation (FPT-approximation) algorithm* for k-DomSet. To motivate the notion of FPT-approximation algorithms, first notice that the seemingly reasonable $O(\log n)$-approximation given by the greedy algorithm can become unjustifiable when the optimum k is small, since it is even possible that the overhead paid is unbounded in terms of k. As a result, FPT-approximation algorithms require the approximation ratios to be bounded only in terms of k; specifically, for any computable function F, we say that an algorithm is an $F(k)$-FPT-approximation algorithm for k-DomSet if it runs in FPT time and, on any input (G, k) such that the minimum dominating set of G is of size at most k, it outputs a dominating set of size at most $F(k) \cdot k$.

This brings us to the main question addressed in our work: *Is there an $F(k)$-FPT-approximation algorithm for k-DomSet for some computable function F?* This question, which dates back to the late 1990s (see, e.g., Reference [33]), has attracted significant attention in the literature [14–16, 21–23, 32–34, 42]. In fact, it is even listed in the seminal textbook of Downey and Fellows [32] as one of the six "most infamous" open questions[4] in the area of parameterized complexity. While earlier attempts fell short of ruling out either $F(k)$ that is super constant or all FPT algorithms (see Section 1.2 for more details), the last couple of years have seen significant progresses on the problem. In a remarkable result by Chen and Lin [22], it was shown that no FPT-approximation for k-DomSet exists for any constant ratio unless W[1] = FPT. They also proved that, assuming ETH, the inapproximability ratio can be improved to $\log^{1/4-\varepsilon} k$ for any constant $\varepsilon > 0$. Very recently, Chalermsook et al. [16] proved, under the *Gap Exponential Time Hypothesis* (Gap-ETH)[5], that no $F(k)$-approximation algorithm for k-DomSet exists for any computable function F. Such non-existence of FPT-approximation algorithms is referred to in literature as the *total FPT-inapproximability* of k-DomSet.

Although Chalermsook et al.'s result on the surface seems to settle the parameterized complexity of approximating dominating set, several aspects of the result are somewhat unsatisfactory. First, while Gap-ETH may be plausible, it is quite strong and, in a sense, does much of the work in the proof. Specifically, Gap-ETH itself already gives the gap in hardness of approximation; once there is such a gap, it suffices to design gap-preserving reductions[6] to prove other inapproximability results. As an example, in the analogous situation in NP-hardness of approximation, once one inapproximability result can be shown, others follow via relatively simple gap-preserving reductions (see, e.g., Reference [62]). However, creating a gap in the first place requires the PCP Theorem [8, 9], which involves several new technical ideas such as local checkability of codes and proof composition.[7] Hence, it is desirable to bypass Gap-ETH and prove total FPT-inapproximability under

[4] Since its publication, one of the questions, the parameterized complexity of biclique, has been resolved [54].

[5] Gap-ETH [28, 58], another strengthening of ETH, states that no subexponential time algorithm can distinguish satisfiable 3-CNF formulae from ones that are not even $(1 - \varepsilon)$-satisfiable for some $\varepsilon > 0$.

[6] One issue grossed over in this discussion is that of *gap amplification*. While Gap-ETH gives some constant gap, Chalermsook et al. still needed to amplify the gap to arrive at total FPT-inapproximability. Fortunately, unlike the NP-hardness regime that requires Raz's parallel repetition theorem [66], the gap amplification step in Reference [16], while non-trivial, only involved relatively simple combinatorial arguments. (See Reference [16, Theorem 4.3].)

[7] Even in the "combinatorial proof" of the PCP Theorem [27], many of these tools still remain in use, specifically in the alphabet reduction step of the proof.

assumptions that do not involve hardness of approximation in the first place. Drawing a parallel to the theory of NP-hardness of approximation once again, it is imaginable that a success in bypassing Gap-ETH may also reveal a "PCP-like Theorem" for parameterized complexity.

An additional reason one may wish to bypass Gap-ETH for the total FPT-inapproximability of k-DomSet is that the latter is a statement purely about parameterized complexity, so one expects it to hold under a standard parameterized complexity assumption. Given that Chen and Lin [22] proved W[1]-hardness of approximating k-DomSet to within any constant factor, a concrete question here is whether we can show W[1]-hardness of approximation for every function $F(k)$:

Question 1.1. *Can we base the total FPT-inapproximability of* k-DomSet *on* W[1] ≠ FPT*?*

Another issue not completely resolved in Reference [16] is the running time lower bound. While the work gives a quite strong running time lower bound that rules out any $T(k) \cdot n^{o(k)}$-time $F(k)$-approximation algorithm for any computable functions T and F, it is still possible that, say, an $O(n^{0.5k})$-time algorithm can provide a very good (even constant ratio) approximation for k-DomSet. Given the aforementioned $O(n^{k-\varepsilon})$ running time lower bound for exact algorithms of k-DomSet by Pătraşcu and Williams [64], it seems reasonable to ask whether such a lower bound can also be established for approximation algorithms:

Question 1.2. *Is it hard to approximate* k-DomSet *in* $O(n^{k-\varepsilon})$*-time?*

This question has perplexed researchers, as even with the running time of, say, $O(n^{k-0.1})$, no $F(k)$-approximation algorithm is known for k-DomSet for any computable function F.

1.1 Our Contributions

Our contributions are twofold. First, we prove parameterized inapproximabilty results for k-DomSet, answering the two aforementioned open questions (and more). Second, we demonstrate a connection between communication complexity and parameterized inapproximability, allowing us to translate running time lower bounds for parameterized problems into parameterized hardness of approximation. This latter part of the contribution extends ideas from a recent breakthrough of Abboud et al. [4], who discovered similar connections and used them to establish inapproximability for problems in ű. In this subsection, we only focus on the first part of our contributions. The second part will be discussed in detail in Section 2.

Our first batch of results are the inapproximability results for k-DomSet under various standard assumptions in parameterized complexity and fine-grained complexity: W[1] ≠ FPT, ETH, SETH, and the k-SUM Hypothesis. First, we show total inapproximability of k-DomSet under W[1] ≠ FPT. In fact, we show an even stronger[8] inapproximation ratio of $(\log n)^{1/\text{poly}(k)}$:

Theorem 1.3. *Assuming* W[1] ≠ FPT*, no FPT time algorithm can approximate* k-DomSet *to within a factor of* $(\log n)^{1/\text{poly}(k)}$.

Our result above improves upon the constant factor inapproximability result of Chen and Lin [22] and resolves the question of whether we can base total FPT inapproximability of k-DomSet on a purely parameterized complexity assumption. Furthermore, if we are willing to assume the stronger ETH, then we can even rule out all $T(k) \cdot n^{o(k)}$-time algorithms:

[8]Note that the factor of the form $(\log n)^{1/\text{poly}(k)}$ is stronger than that of the form $F(k)$. To see this, assume that we have an $F(k)$-FPT-approximation algorithm for some computable function F. We can turn this into a $(\log n)^{1/\text{poly}(k)}$-approximation algorithm by first checking which of the two ratios is smaller. If $F(k)$ is smaller, then just run the $F(k)$-FPT-approximation algorithm. Otherwise, use brute-force search to solve the problem. Since the latter case can only occur when $n \leq \exp(F(k)^{\text{poly}(k)})$, we have that the running time remains FPT.

Table 1. Summary of Our and Previous Results on k-DomSet

Summary of Previous Works and The Results in This Paper			
Complexity Assumption	Inapproximability Ratio	Running Time Lower Bound	Reference
W[1] ≠ FPT	Any constant	$T(k) \cdot \text{poly}(n)$	[22]
	$(\log n)^{1/\text{poly}(k)}$	$T(k) \cdot \text{poly}(n)$	This article
ETH	$(\log k)^{1/4+\varepsilon}$	$T(k) \cdot n^{o(\sqrt{k})}$	[22]
	$(\log n)^{1/\text{poly}(k)}$	$T(k) \cdot n^{o(k)}$	This article
Gap-ETH	$(\log n)^{1/\text{poly}(k)}$	$T(k) \cdot n^{o(k)}$	[16]
SETH	Exact	$O(n^{k-\varepsilon})$	[64]
	$(\log n)^{1/\text{poly}(k, e(\varepsilon))}$	$O(n^{k-\varepsilon})$	This article
k-SUM Hypothesis	$(\log n)^{1/\text{poly}(k)}$	$O(n^{\lceil k/2 \rceil - \varepsilon})$	This article

We only show those whose inapproximability ratios are at least some constant greater than 1 (i.e., we exclude additive inapproximability results). Here $e : \mathbb{R}^+ \to \mathbb{N}$ is some function, $T : \mathbb{N} \to \mathbb{N}$ can be any computable function, and ε can be any positive constant. The Gap-ETH has been bypassed in this article, and prior to this article, the k-SUM Hypothesis had never been used in proving inapproximability of k-DomSet.

Theorem 1.4. *Assuming* ETH, *no* $T(k) \cdot n^{o(k)}$*-time algorithm can approximate* k-DomSet *to within a factor of* $(\log n)^{1/\text{poly}(k)}$.

Note that the running time lower bound and approximation ratio ruled out by the above theorem are exactly the same as those of Charlermsook et al.'s result based on Gap-ETH [16]. In other words, we successfully bypass Gap-ETH from their result completely. Prior to this, the best-known ETH-based inapproximability result for k-DomSet due to Chen and Lin [22] ruled out only $(\log^{1/4+\varepsilon} k)$-approximation for $T(k) \cdot n^{o(\sqrt{k})}$-time algorithms.

Assuming the even stronger hypothesis, SETH, we can rule out $O(n^{k-\varepsilon})$-time approximation algorithms for k-DomSet, matching the running time lower bound from Reference [64] while excluding not only exact but also approximation algorithms. We note, however, that the approximation ratio we get in this case is not $(\log n)^{1/\text{poly}(k)}$ anymore but rather $(\log n)^{1/\text{poly}(k, e(\varepsilon))}$ for some function e, which arises from SETH and the Sparsification Lemma [44].

Theorem 1.5. *There is a function* $e : \mathbb{R}^+ \to \mathbb{N}$ *such that, assuming* SETH, *for every integer* $k \geq 2$ *and for every* $\varepsilon > 0$, *no* $O(n^{k-\varepsilon})$*-time algorithm can approximate* k-DomSet *to within a factor of* $(\log n)^{1/\text{poly}(k, e(\varepsilon))}$.

Finally, to demonstrate the flexibility of our proof techniques (which will be discussed at length in the next section), we apply the framework to the k-SUM Hypothesis,[9] which yields an $n^{\lceil k/2 \rceil - \varepsilon}$ running time lower bound for approximating k-DomSet as stated below.

Theorem 1.6. *Assuming the* k-SUM *Hypothesis, for every integer* $k \geq 3$ *and for every* $\varepsilon > 0$, *no* $O(n^{\lceil k/2 \rceil - \varepsilon})$*-time algorithm can approximate* k-DomSet *to within a factor of* $(\log n)^{1/\text{poly}(k)}$.

We remark here that the k-SUM problem is known to be W[1]-hard [3, 30] and our proof of Theorem 1.6 indeed yields an alternative proof of W[1]-hardness of approximating k-DomSet (Theorem 1.3). Nevertheless, we provide a different self-contained W[1]-hardness reduction directly from Clique, since the ideas there are also useful for our ETH-hardness result (Theorem 1.4).

The summary of our results and those from previous works are shown in Table 1.

[9]The k-SUM Hypothesis states that, for every $\varepsilon > 0$, $k \in \mathbb{N}$ such that $k \geq 3$, no $O(n^{\lceil k/2 \rceil - \varepsilon})$-time algorithm solves the k-SUM problem; see Hypothesis 3.5.

1.2 Comparison to Previous Works

In addition to the lower bounds previously mentioned, the parameterized inapproximability of k-DomSet has also been investigated in several other works [14, 23, 34, 42]. Specifically, Downey et al. [34] showed that obtaining an *additive* constant approximation for k-DomSet is W[2]-hard. However, in References [23, 42], the authors ruled out $(\log k)^{1+\varepsilon}$-approximation in time $\exp(\exp((\log k)^{1+\varepsilon})) \cdot \text{poly}(n)$ for some fixed constant $\varepsilon > 0$ by assuming ETH and the *projection game conjecture* proposed in Reference [60]. Further, Bonnet et al. [14] ruled out $(1+\varepsilon)$-FPT-approximation for some fixed constant $\varepsilon > 0$, assuming Gap-ETH.[10] We note that, with the exception of W[2]-hardness results [30, 34], our results subsume all other aforementioned lower bounds regarding k-DomSet, for both approximation [14, 16, 22, 23, 42] and exact algorithms [20, 64].

While our techniques will be discussed at a much greater length in the next section (in particular, we compare our technique with Reference [4] in Section 2.2), we note that our general approach is to first show inapproximability of a parameterized variant of the *Label Cover* problem called MaxCover and then reduce MaxCover to k-DomSet. The first step employs the connection between communication complexity and inapproximability of MaxCover, whereas the second step follows directly from the reduction in Reference [16] (which is in turn based on Reference [36]). While MaxCover was not explicitly defined until Reference [16], its connection to k-DomSet had been implicitly used both in the work of Pătraşcu and Williams [64] and that of Chen and Lin [22].

From this perspective, the main difference between our work and References [16, 22, 64] is the source of hardness for MaxCover. Recall that Pătraşcu and Williams [64] ruled out only exact algorithms; in this case, a relatively simple reduction gave hardness for the exact version of MaxCover. However, both Chalermsook et al. [16] and Chen and Lin [22] ruled out approximation algorithms, meaning that they needed gaps in their hardness results for MaxCover. Chalermsook et al. obtained their initial gap from their assumption (Gap-ETH), after which they amplified it to arrive at an arbitrarily large gap for MaxCover. However, Reference [22] derived their gap from the hardness of approximating Maximum k-Intersection shown in Lin's earlier breakthrough work [54]. Lin's proof [54] made use of certain combinatorial objects called *threshold graphs* to prove inapproximability of Maximum k-Intersection. Unfortunately, this construction was not very flexible, in the sense that it produced MaxCover instances with parameters that were not sufficient for proving total-FPT-inapproximability for k-DomSet. Moreover, his technique (i.e., threshold graphs) was limited to reductions from k-Clique and was unable to provide a tight running time lower bound under ETH. By resorting to the connection between MaxCover and communication complexity, we can generate MaxCover instances with wider ranges of parameters from much more general assumptions, allowing us to overcome the aforementioned barriers.

1.3 Organization of the Paper

In the next section, we give an overview of our lower level contributions; for readers interested in the general ideas without too much notational overhead, this section covers most of the main ideas from our article through a proof sketch of our W[1]-hardness of approximation result (Theorem 1.3). After that, in Section 3, we define additional notations and preliminaries needed to formalize our proofs. Section 4 provides a definition for Product Space Problems (PSP) and rewrites the hypotheses in these terms. Next, in Section 5, we establish a general theorem converting communication protocols to a reduction from PSP to MaxCover. Sections 6, 7, and 8 provide

[10] The authors assume the same statement as Gap-ETH (albeit, with imperfect completeness) but have an additional assertion that it is implied by ETH (see Hypothesis 1 in Reference [14]). It is not hard to see that their assumption can be replaced by Gap-ETH.

communication protocols for our problems of interest: Set Disjointness, Multi-Equality, and Sum-Zero. Finally, in Section 9, we conclude with a few open questions and research directions.

2 CONNECTING COMMUNICATION COMPLEXITY AND PARAMETERIZED INAPPROXIMABILITY: AN OVERVIEW

This section is devoted to presenting our connection between communication complexity and parameterized inapproximability (which is one of our main contributions as discussed in the introduction) and serves as an overview for all the proofs in this article. As mentioned previously, our discovery of this connection is inspired by the work of Abboud et al. [4], who showed the connection between the communication protocols and hardness of approximation for problems in P. More specifically, they showed how a Merlin-Arthur protocol for *Set Disjointness* with certain parameters implies the SETH-hardness of approximation for a problem called *PCP-Vectors* and used it as the starting point to prove inapproximability of other problems in P. We extend this idea by identifying a communication problem associated with each of the complexity assumptions (W[1] ≠ FPT, ETH, SETH and k-SUM Hypothesis) and then prove a generic theorem that translates communication protocols for these problems to conditional hardness of approximation for a parameterized variant of the *Label Cover* problem called MaxCover [16]. Since the hardness of MaxCover is known to imply the hardness of k-DomSet [16] (see Section 3.3), we have arrived at our inapproximability results for k-DomSet. As the latter part is not the contribution of this article, we will focus on explaining the connection between communication complexity and the hardness of approximating MaxCover. We start by defining MaxCover:

Definition 2.1. The input for MaxCover is a *label cover instance* Γ, which consists of a bipartite graph $G = (U, W; E)$ such that U is partitioned into $U_1 \cup \cdots \cup U_q$ and W is partitioned into $W_1 \cup \cdots \cup W_h$. We sometimes refer to U_i's and W_j's as left and right *super-nodes* of Γ, respectively.

A solution to MaxCover is called a *labeling*, which is a subset of vertices $S \subseteq W$ formed by picking a vertex w_j from each W_j (i.e., $|S \cap W_j| = 1$ for all $j \in [h]$). We say that a labeling S *covers* a left super-node U_i if there exists a vertex $u_i \in U_i$ such that u_i is a neighbor of every vertex in S. The goal in MaxCover is to find a labeling that covers the maximum fraction of left super-nodes.

For concreteness, we focus on the W[1]-hardness proof (Theorem 1.3); at the end of this subsection, we will discuss how this fits into a larger framework that encapsulates other hypotheses as well.

For the purpose of our current discussion, it suffices to think of MaxCover as being parameterized by h, the number of right super-nodes; from this viewpoint, we would like to show that it is W[1]-hard to approximate MaxCover to within $(\log n)^{1/\text{poly}(h)}$ factor. For simplicity, we shall be somewhat imprecise in our overview below, all proofs will be formalized later in the article.

We reduce from the k-Clique problem, which is well-known to be W[1]-hard [30]. The input to k-Clique is an integer k and a graph that we will call $G' = (V', E')$ to avoid confusion with the label cover graph. The goal is to determine whether G' contains a clique of size k. Recall that, to prove the desired W[1]-hardness, it suffices to provide an *FPT-reduction* from any k-Clique instance (G', k) to approximate MaxCover instance $G = (U, W; E)$; this is an FPT-time reduction such that the new parameter h is bounded by a function of the original parameter k. Furthermore, since we want a hardness of approximation result for the latter, we will also show that, when (G', k) is a YES instance of k-Clique, there is a labeling of G that covers all the left super-nodes. However, when (G', k) is a NO instance of k-Clique, we wish to show that every labeling of G will cover at most $1/(\log n)^{1/\text{poly}(h)}$ fraction of the left super-nodes. Once we have such a reduction, we would have arrived at the total FPT-inapproximability of MaxCover under W[1] ≠ FPT. But how would

we come up with such a reduction? We will do this by devising a specific kind of protocol for a communication problem.

2.1 A Communication Problem for k-Clique

The communication problem related to k-Clique we consider is a multi-party problem where there are $h = \binom{k}{2}$ players, each associated with a two-element subset $\{i, j\}$ of $[k]$. The players cannot communicate with each other. Rather, there is a referee that they can send messages to. Each player $\{i, j\}$ is given two vertices $u_i^{\{i,j\}}$ and $u_j^{\{i,j\}}$ such that $\{u_i^{\{i,j\}}, u_j^{\{i,j\}}\}$ forms an edge in G'. The vertices $u_i^{\{i,j\}}$ and $u_j^{\{i,j\}}$ are allegedly the ith and jth vertices of a clique, respectively. The goal is to determine whether there is indeed a k-clique in G' such that, for every $\{i, j\} \subseteq [k]$, $u_i^{\{i,j\}}$ and $u_j^{\{i,j\}}$ are the ith and jth vertices of the clique.

The communication protocol that we are looking for is a one-round protocol with public randomness and by the end of which the referee is the one who outputs the answer. Specifically, the protocol proceeds as follows. First, the players and the referee together toss r random coins. Then, each player sends an ℓ-bit message to the referee. Finally, the referee decides, based on the messages received and the randomness, either to accept or reject. The protocol is said to have perfect completeness and soundness s if (1) when there is a desired clique, the referee always accepts and (2) when there is no such clique, the referee accepts with probability at most s. The model described here is referred to in the literature as the multi-party *Simultaneous Message Passing* (SMP) model [10, 37, 79]. We refer to a protocol in the SMP model as an SMP protocol.

From Communication Protocol to MaxCover. Before providing a protocol for the previously described communication problem, let us describe how to turn the protocol into a label cover instance $G = (U = U_1 \cup \cdots \cup U_q, W = W_1 \cup \cdots \cup W_h; E)$.

- Let $h = \binom{k}{2}$. Again, we associate elements in $[h]$ with two-element subsets of $[k]$. Each right super-node $W_{\{i,j\}}$ represents Player $\{i, j\}$. Each vertex in $W_{\{i,j\}}$ represents a possible input to the player, i.e., we have one vertex $a_{\{u,v\}} \in W_{\{i,j\}}$ for each edge $\{u, v\} \in E'$ in the graph G'. Assume w.l.o.g. that $i < j$ and $u < v$. This vertex $a_{\{u,v\}}$ represents player $\{i, j\}$ receiving u and v as the alleged ith and jth vertices of the clique, respectively.
- Let $q = 2^r$. We associate each element in $[q]$ with an r-bit string. For each $\gamma \in \{0, 1\}^r$, the left super-node U_γ contains one node corresponding to each *accepting configuration* on randomness γ; that is, for each h-tuple of ℓ-bit strings $(m_{\{1,2\}}, \ldots, m_{\{k-1,k\}}) \in (\{0, 1\}^\ell)^h$, there is a node $(m_{\{1,2\}}, \ldots, m_{\{k-1,k\}})$ in U_γ iff the referee on randomness γ and message $m_{\{1,2\}}, \ldots, m_{\{k-1,k\}}$ from all the players accepts.
- The edges in E are defined as follows. Recall that each node a in a right super-node $W_{\{i,j\}}$ corresponds to an input that each player receives in the protocol. For each $\gamma \in \{0, 1\}^r$, suppose that the message produced on this randomness by the $\{i, j\}$th player on the input corresponding to a is $m^{a,\gamma}$. We connect a to every accepting configuration on randomness γ that agrees with the message $m^{a,\gamma}$. More specifically, for every $\gamma \in \{0, 1\}^r$, a is connected to every vertex $(m_{\{1,2\}}, \ldots, m_{\{k-1,k\}}) \in U_\gamma$ iff $m_{\{i,j\}} = m^{a,\gamma}$.

Consider any labeling $S \subseteq W$. It is not hard to see that, if we run the protocol where the $\{i, j\}$th player is given the edge corresponding to the unique element in $S \cap W_{\{i,j\}}$ as an input, then the referee accepts a random string $\gamma \in \{0, 1\}^r$ if and only if the left super-node U_γ is covered by the labeling S. In other words, the fraction of the left super-nodes covered by S is exactly equal to the acceptance probability of the protocol. This means that if (G', k) is a YES-instance of k-Clique, then we can select S corresponding to the edges of a k-clique, and every left super-node will be

covered. However, if (G', k) is a NO-instance of k-Clique, then there is no subset S that corresponds to a valid k-clique, meaning that every labeling S covers at most s fraction of the edges. Hence, we have indeed arrived at hardness of approximation for MaxCover. Before we move on to describe the protocol, let us note that the running time of the reduction is $\text{poly}(2^{r+\ell h}, |E'|)$, which also gives an upper bound on the number of vertices in the label cover graph G.

SMP Protocol. Observe, first, that the trivial protocol, one where every player sends the whole input to the referee, does not suffice for us; this is because the message length ℓ is $\Omega(\log n)$, meaning that the running time of the reduction is $n^{\Omega(h)} = n^{\Omega(k^2)}$, which is not FPT time.

Nevertheless, there still is a simple protocol that does the job. Note that the input vertices $u_i^{\{i,j\}}$ and $u_j^{\{i,j\}}$ given to Player $\{i, j\}$ are already promised to form an edge. Hence, the only thing the referee needs to check is whether each alleged vertex of the clique sent to different players are the same; namely, he or she only needs to verify that, for every $i \in [k]$, we have $u_i^{\{i,1\}} = u_i^{\{i,2\}} = \cdots = u_i^{\{i,i-1\}} = u_i^{\{i,i+1\}} = \cdots = u_i^{\{i,k\}}$. In other words, he or she only needs to check equalities for each of the k unknowns. The equality problem and its variants are extensively studied in communication complexity (see, e.g., References [51, 79]). In our case, the protocol can be easily derived using any error-correcting code. Specifically, for an outcome $\gamma \in \{0, 1\}^r$ of the random coin tosses, every Player $\{i, j\}$ encodes each of his or her input ($u_i^{\{i,j\}}$ and $u_j^{\{i,j\}}$) using a binary error-correcting code and sends only the γth bit of each encoded word to the referee. The referee then checks whether, for every $i \in [k]$, the received γth bits of the encodings of $u_i^{\{i,1\}}, u_i^{\{i,2\}}, \ldots, u_i^{\{i,k\}}$ are equal.

In the protocol described above, the message length ℓ is now only two bits (one bit per vertex), the randomness r used is logarithmic in the block length of the code, the soundness s is one minus the relative distance of the code. If we use a binary code with constant rate and constant relative distance (aka *good code*), then r will be simply $O(\log \log n)$; this means that the running time of the reduction is $\text{poly}(n, \exp(O(k^2)))$ as desired. While the soundness in this case will just be some constant less than 1, we can amplify the soundness by repeating the protocol multiple times independently; this increases the randomness and message length, but it is still not hard to see that, with the right number of repetitions, all parameters lie within the desired ranges. With this, we have completed our sketch for the proof of W[1]-hardness of approximating MaxCover.

2.2 A Framework for Parameterized Hardness of Approximation

The W[1]-hardness proof sketched above is an example of a much more general connection between communication protocol and the hardness of approximating MaxCover. To gain insight on this, consider any function $f : X_1 \times \cdots \times X_k \to \{0, 1\}$. This function naturally induces both a communication problem and a computational problem. The communication problem for f is one where there are k players, each player i receives an input $a_i \in X_i$, and they together wish to compute $f(a_1, \ldots, a_k)$. The computational problem for f, which we call the *Product Space Problem*[11] of f (abbreviated as PSP(f)), is one where the input consists of subsets $A_1 \subseteq X_1, \ldots, A_k \subseteq X_k$ and the goal is to determine whether there exists $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ such that $f(a_1, \ldots, a_k) = 1$. The sketch reduction to MaxCover above in fact not only applies to the specific communication problem of k-Clique: The analogous construction is a generic way to translate any SMP protocol for the communication problem of any function f to a reduction from PSP(f) to MaxCover. To phrase it somewhat differently, if we have an SMP protocol for f with certain parameters and PSP(f) is hard to solve, then MaxCover is hard to approximate.

This brings us to the framework we use in this article. It consists of only two steps. First, we rewrite the problem in the hypotheses as Product Space Problems of some family of functions $\mathcal{F}$.

[11] The naming comes from the product structure of the domain of f.

This gives us the conditional hardness for solving PSP($\mathcal{F}$). Second, we devise an SMP protocol for every function $f \in \mathcal{F}$. Given the connection outlined in the previous paragraph, this automatically yields the parameterized hardness of approximating MaxCover.

To gain more intuition into the framework, note that in the case of k-Clique above, the function $f \in \mathcal{F}$ we consider is just the function $f : X_{\{1,2\}} \times \cdots \times X_{\{k,k-1\}}$, where each of $X_{\{1,2\}}, \ldots, X_{\{k,k-1\}}$ is a copy of the edge set. The function f "checks" that the edges selected form a clique, i.e., that, for every $i \in [k]$, the alleged ith vertex of the clique specified in the $\{i, j\}$-coordinate is equal for every $j \neq i$. Since this is a generalization of the equality function, we call such a class of functions "multi-equality." It turns out that 3-CNF-SAT can also be written as PSP of multi-equality; each X_i contains assignments to $1/k$ fraction of the clauses and the function f checks that each variable is assigned the same value across all X_i's they appear in. A protocol essentially the same as the one given above also works in this setting and immediately gives our ETH-hardness result (Theorem 1.4). Unfortunately, this does not suffice for our SETH-hardness. In that case, the function used is the k-way set disjointness; this interpretation of SETH is well known (see, e.g., Reference [78]) and is also used in Reference [4]. Last, the k-SUM problem is already written in PSP form where f is just the Sum-Zero function that checks whether the sum of k specified numbers equals zero.

Let us note that in the actual proof, we have to be more careful about the parameters than in the above sketch. Specifically, the reduction from MaxCover to k-DomSet from [16] incurs a blow-up in size that is exponential in terms of the number of vertices in each left super-node (i.e., exponential in $|U_\gamma|$). This means that we need $|U_1|, \ldots, |U_r| = o(\log n)$. In the context of communication protocol, this translates to keeping the message length $O(\log \log n)$ where $O(\cdot)$ hides a sufficiently small constant. Nevertheless, for the protocol for k-Clique reduction (and more generally for multi-equality), this does not pose a problem for us, since the message length before repetitions is $O(1)$ bits; we can make sure that we apply only $O(\log \log n)$ repetitions to the protocol.

For Sum-Zero, known protocols either violate the above requirement on message length [61] or use too much randomness [76]. Nonetheless, a simple observation allows us to compose Nisan's protocol [61] and Viola's protocol [76] and arrive at a protocol with the best of both parameters. This new protocol may also be of independent interest beyond the scope of our work.

However, well-known communication complexity lower bounds on set disjointness [11, 47, 68] rule out the existence of protocols with parameters we wish to have. Reference [4] also ran into this issue; in our language, they got around this problem by allowing the referee to receive an advice. This will also be the route we take. Even with advice, however, devising a protocol with the desired parameters is a technically challenging issue. In particular, until very recently, no protocol for set disjointness with $O(\log \log n)$ message length (and $o(n)$ advice length) was known. This was overcome in the work of Rubinstein [70], who used algebraic geometric codes to give such a protocol for the two-player case. We extend his protocol in a straightforward manner to the k-player case; this extension was also suggested to us by Rubinstein [69].

A diagram illustrating the overview of our approach can be found in Figure 1.

Comparison to Abboud et al. In Reference [4], the authors show the first subquardatic hardness result (under SETH) for many important *gap* problems. Their main result is the (SETH) hardness of a problem they refer to as the PCP-Vectors problem. They then design gap-preserving reductions from PCP-Vectors problem to other natural gap problems in P. We remark that PCP-Vectors is equivalent to MaxCover when $h = 2$ (i.e., the number of right super nodes is 2). However, formulating the label cover problem as MaxCover instead of PCP-Vectors, is beneficial for us, as our goal is to reduce to graph problems.

Also, in their work, they merge the roles of the referee and the first player, as it is necessary to achieve the goal of proving hardness of approximation for important problems in P (which are usually defined on one or two sets of vectors). However, by doing this the details of the proof

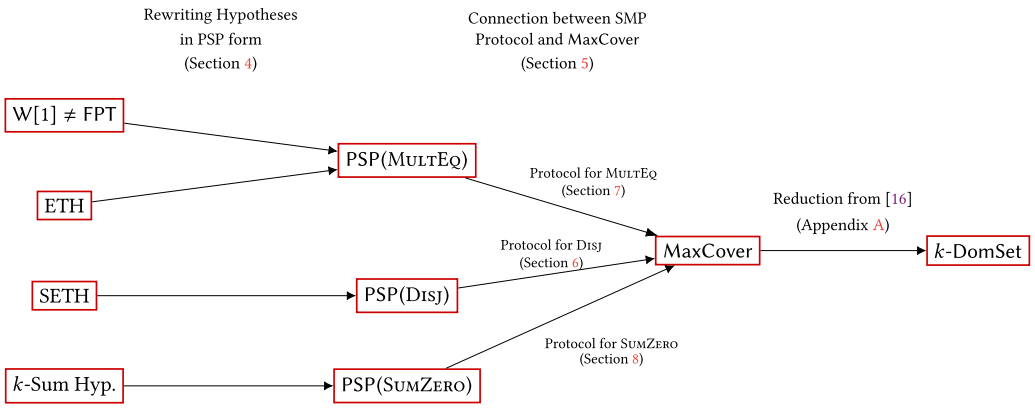

Fig. 1. Overview of our framework. The first step is to reformulate each hypothesis in terms of hardness of a PSP problem, which is done in Section 4. Using the connection between SMP protocols and MaxCover outlined earlier (and formalized in Section 5), our task is now to devise SMP protocols with certain parameters for the corresponding communication problems; these are taken care of in Sections 6, 7, and 8. For completeness, the final reduction from MaxCover to k-DomSet, which was shown in Reference [16], is included in Appendix A.

become a little convoluted. On the contrary, our framework with the SMP model is arguably a cleaner framework to work with and it works well for our goal of proving hardness of approximation for parameterized problems.

Finally, we note that our observation that the hardness of approximating MaxCover can be obtained from any arbitrary hypothesis as long as there is an underlying product structure (as formalized via PSPs) is a new contribution of this article.

Comparison to Subsequent Work of Lin. In Reference [55], the author provides a one-step reduction from an instance of k-set cover[12] on a universe of size $O(\log n)$ (where n is the number of subsets given in the collection) to an instance of k-set cover on a universe of size poly(n) with a gap of $(\frac{\log n}{\log\log n})^{1/k}$. The author then uses this gap producing self-reduction to provide running time lower bounds (under different time hypotheses) for approximating k-set cover to a factor of $(1 - o(1)) \cdot (\frac{\log n}{\log\log n})^{1/k}$, improving on the results in Table 1 with a better dependence on k in the exponent.

At a high level, the NP-hardness of gap set cover proceeds by combining the gap label cover instance generated from the PCP theorem with the hypercube partition gadget (described in Appendix A.2). In this article, we proceed in a similar way by combining the gap MaxCover instance generated from the (generalized) Distributed PCP framework (see Figure 1), with the hypercube partition gadget. In Reference [55], the author seems to first combine the hypercube partition gadget with a derandomizing combinatorial object called *universal set*, to obtain a gap gadget, and then combines the gap gadget with the input k-set cover instance (on small universe but with no gap) to obtain a gap k-set cover instance.

Finally, we remark that unlike our proof framework, Lin's technique seems to be specifically tailored for the parameterized set cover problem; case in point, his technique does not give the inapproximability of the MaxCover problem, which we believe to be a canonical parameterized gap problem.

[12]Recall that there is a pair of polynomial-time L-reductions between the minimum dominating set problem and the set cover problem [48].

3 PRELIMINARIES AND BACKGROUNDS

We use standard graph terminology. Let G be any graph. The vertex and edge sets of G are denoted by $V(G)$ and $E(G)$, respectively. We say that a subset of vertices $S \subseteq V(G)$ *dominates* G if every vertex $v \in S \setminus V(G)$ has a neighbor in S, and we call S a *dominating set* of G. A *k-dominating set* of G is a dominating set of G with cardinality k. The *domination number*, denoted by (DomSet (G)) of G is the size of the smallest dominating set of G. A *clique* H in G is a complete subgraph of G, and we say that H is a *k-clique* if H contains k vertices. The *clique number* of G is the size of the largest clique in G. Sometime we abuse notation and call a subset of vertices $S \subseteq V(G)$, a clique, if S induces a complete subgraph in G.

3.1 Problem Definitions

Below are the list of problems considered in this article.

- **Dominating Set.** In the *k-Dominating Set* problem (k-DomSet), we are given a graph G, and the goal is to decide whether G has a dominating set of size k. In the minimization version, called *Minimum Dominating Set* (DomSet, for short), the goal is to find a dominating set in G of minimum size.
- **Clique.** In the *k-Clique* problem (k-Clique), we are given a graph G, and the goal is to decide whether G has a clique of size k. In the maximization version, called *Maximum Clique* (Clique, for short), the goal is to find a clique in G of maximum size.
- ***k*-SAT.** In the k-SAT problem (k-CNF-SAT), we are given a CNF formula Φ with **at most** k literals in a clause and the goal is to decide whether Φ is satisfiable.
- ***k*-Sum.** In k-SUM, we are given k subsets $S_1, \ldots, S_k \subseteq ([-M, M] \cap \mathbb{Z})$ of integers between $-M$ and M (inclusive), and the goal is to determine whether there exist $x_1 \in S_1, \ldots, x_k \in S_k$ such that $x_1 + \cdots + x_k = 0$. That is, we wish to pick one integer from each subset so that they sum to zero.

In addition to the above problems, we devote the next section to define and discuss a variant of the *label cover* problem, namely MaxCover.

3.2 MaxCover —A Variant of Label Cover

We now define a variant of the label cover problem called MaxCover, which was introduced by Chalermsook et al. [16], to capture the parameterized inapproximability of k-Clique and k-DomSet. The approximation hardness of MaxCover will be the basis of our hardness results.

The input of MaxCover is a *label cover instance*; a label cover instance Γ consists of a bipartite graph $G = (U, W; E)$ such that U is partitioned into $U = U_1 \cup \cdots \cup U_\ell$ and W is partitioned into $W = W_1 \cup \cdots \cup W_h$. We sometimes refer to U_i's and W_j's as *left super-nodes* and *right super-nodes* of Γ, respectively. Another parameter we will be interested in is the maximum size of left super nodes, i.e., $\max_{i \in [\ell]} |U_i|$; we refer to this quantity as the *left alphabet size* of the instance.

A solution to MaxCover is called a *labeling*, which is a subset of vertices $S \subseteq W$ formed by picking a vertex w_j from each W_j (i.e., $|S \cap W_j| = 1$ for all $j \in [h]$). We say that a labeling S *covers* a left super-node U_i if there exists a vertex $u_i \in U_i$ such that u_i is a neighbor of every vertex in S. The goal in MaxCover is to find a labeling that covers the maximum fraction of left super-nodes. We abuse the notation MaxCover and also use it for the optimum as well, i.e.,

$$\text{MaxCover}(\Gamma) = \frac{1}{\ell} \left(\max_{\text{labeling } S} |\{i \in [\ell] \mid U_i \text{ is covered by } S\}| \right).$$

The above terminologies for MaxCover are from Reference [16]. Note, however, that our definitions are phrased somewhat differently than theirs; in our definitions, the input graphs are the

so-called *label-extended graphs*, whereas in their definitions, the input graphs are the *constraint graphs*. Nevertheless, it is not hard to see that the two versions are in fact equivalent. Another difference is that we use MaxCover to denote the *fraction* of left super-nodes covered by the optimal labeling, whereas Reference [16] uses the notion for the *number* of covered left super-nodes. The former notation is somewhat more convenient for us as the value is between zero and 1.

3.3 Inapproximability of DomSet from MaxCover

The relation between MaxCover and DomSet has been observed in the literature. The *k-prover system* introduced by Feige [36] can be cast as a special case of MaxCover with *projection property*, and it has been shown that this proof system can be transformed into an instance of DomSet. We note, however, that the optimal value of the DomSet instance produced by Feige's k-prover system has size dependent on the number of left super-nodes rather than k, the number of right super-nodes. Recently, Chalermsook et al. [16] observed that even without the projection property, the relation between MaxCover and DomSet still holds, and the value of the optimal solution can be reduced to k. This is stated formally below.

Theorem 3.1 (Reduction from MaxCover to DomSet [16]). *There is an algorithm that, given a* MaxCover *instance* $\Gamma = (U = \bigcup_{j=1}^{q} U_j, W = \bigcup_{i=1}^{k} W_i, E)$*, outputs a* k-DomSet *instance* G *such that*

- *If* $\mathsf{MaxCover}(\Gamma) = 1$*, then* $\mathsf{DomSet}(G) = k$.
- *If* $\mathsf{MaxCover}(\Gamma) \leq \varepsilon$*, then* $\mathsf{DomSet}(G) \geq (1/\varepsilon)^{1/k} \cdot k$.
- $|V(G)| = |W| + \sum_{j \in [q]} k^{|U_j|}$.
- *The reduction runs in time* $O(|W|(\sum_{j \in [q]} k^{|U_j|}))$.

For the sake of self-containedness, we provide the proof of Theorem 3.1 in Appendix A.

3.4 The Hypotheses

We now list and discuss the computational complexity hypotheses on which our results are based.

3.4.1 $\mathsf{W}[1] \neq \mathsf{FPT}$ *Hypothesis.* The first hypothesis is $\mathsf{W}[1] \neq \mathsf{FPT}$, which is one of the most popular hypotheses used in the area of parameterized complexity, since many fundamental parameterized problems turn out to be W[1]-hard. For the interest of space, we do not give the full definition of W-hierarchy; we refer the readers to standard textbooks in the field (e.g. [26, 32]) for the definition and discussions regarding the hierarchy. Rather, since it is well known that k-Clique is W[1]-complete, we will use a more convenient formulation of $\mathsf{W}[1] \neq \mathsf{FPT}$, which simply states that k-Clique is not in FPT:

Hypothesis 3.2 ($\mathsf{W}[1] \neq \mathsf{FPT}$ Hypothesis). *For any computable function* $T : \mathbb{N} \to \mathbb{N}$*, no algorithm can solve* k-Clique *in* $T(k) \cdot \mathsf{poly}(n)$ *time where* n *denotes the number of vertices in the input graph.*

3.4.2 Exponential Time Hypothesis and Strong Exponential Time Hypothesis. Our second hypothesis is the Exponential Time Hypothesis (ETH), which can be stated as follows.

Hypothesis 3.3 (Exponential Time Hypothesis (ETH) [43, 44, 75]). *There exists* $\delta > 0$ *such that no algorithm can solve 3-CNF-SAT in* $O(2^{\delta n})$ *time where* n *is the number of variables. Moreover, this holds even when restricted to formulae in which each variable appears in at most three clauses.*

Note that the original version of the hypothesis from Reference [43] does not enforce the requirement that each variable appears in at most three clauses. To arrive at the above formulation, we first apply the Sparsification Lemma of Reference [44], which implies that we can assume without loss of generality that the number of clauses m is $O(n)$. We then apply Tovey's reduction [75],

which produces a 3-CNF instance with at most $3m + n = O(n)$ variables, and every variable occurs in at most three clauses. This means that the bounded occurrence restriction is also without loss of generality.

We will also use a stronger hypothesis called the Strong Exponential Time Hypothesis (SETH):

Hypothesis 3.4 (Strong Exponential Time Hypothesis (SETH) [43, 44]). *For every $\varepsilon > 0$, there exists $k = k(\varepsilon) \in \mathbb{N}$ such that no algorithm can solve k-CNF-SAT in $O(2^{(1-\varepsilon)n})$ time, where n is the number of variables. Moreover, this holds even when the number of clauses m is at most $c(\varepsilon) \cdot n$, where $c(\varepsilon)$ denotes a constant that depends only on ε.*

Again, we note that, in the original form [43], the bound on the number of clauses is not enforced. However, the Sparsification Lemma [44] allows us to do so without loss of generality.

3.4.3 k-SUM Hypothesis. Our final hypothesis is the k-SUM Hypothesis, which can be stated as follows.

Hypothesis 3.5 (k-SUM Hypothesis [2]). *For every integer $k \geq 3$ and every $\varepsilon > 0$, no $O(n^{\lceil k/2 \rceil - \varepsilon})$ time algorithm can solve k-SUM, where n denotes the total number of input integers, i.e., $n = |S_1| + \cdots + |S_k|$. Moreover, this holds even when $M = n^{2k}$.*

The above hypothesis is a natural extension of the more well-known 3-SUM Hypothesis [38, 63], which states that 3-SUM cannot be solved in $O(n^{2-\varepsilon})$ time for any $\varepsilon > 0$. Moreover, the k-SUM Hypothesis is closely related to the question of whether SUBSET-SUM can be solved in $O(2^{(1/2-\varepsilon)n})$ time; if the answer to this question is negative, then k-SUM cannot be solved in $O(n^{k/2-\varepsilon})$ time for every $\varepsilon > 0, k \in \mathbb{N}$. We remark that if one is only willing to assume this latter weaker lower bound of $O(n^{k/2-\varepsilon})$ instead of $O(n^{\lceil k/2 \rceil - \varepsilon})$, our reduction would give an $O(n^{k/2-\varepsilon})$ running time lower bound for approximating k-DomSet. Finally, we note that the assumption that $M = n^{2k}$ can be made without loss of generality, since there is a randomized reduction from the general version of the problem (where M is, say, 2^n) to this version of the problem, and this reduction can be derandomized under a certain circuit complexity assumption [3].

3.5 Error-Correcting Codes

An error-correcting code C over alphabet Σ is a function $C : \Sigma^m \to \Sigma^d$, where m and d are positive integers that are referred to as the *message length* and *block length* of C, respectively. Intuitively, the function C encodes an original message of length m to an encoded message of length d. Since we will also deal with communication protocols, for which "message length" has another meaning, we will sometimes refer to the message length of codes as *code message length* whenever there is an ambiguity. The *rate* of a code $\rho(C)$ is defined as the ratio between its message length and its block length, i.e., $\rho(C) = m/d$. The *relative distance* of a code, denoted by $\delta(C)$, is defined as $\min_{x \neq y \in \Sigma^m} \delta(C(x), C(y))$, where $\delta(C(x), C(y))$ is the *relative Hamming distance* between $C(x)$ and $C(y)$, i.e., the fraction of coordinates on which $C(x)$ and $C(y)$ disagree.

3.5.1 Good Codes. In the construction of our communication protocol in Section 7, we require our codes to have constant rate and constant relative distance (referred to as *good codes*). It is not hard to see that random codes, ones where each codeword $C(x)$ is randomly selected from Σ^d independently from each other, satisfy these properties. For binary codes (i.e., $|\Sigma| = 2$), one can explicitly construct such codes using expander graphs (so-called *Expander Codes* [72]); alternatively *Justesen Code* [46] also have the same property (see Appendix E.1.2.5 from Reference [40] for an excellent exposition).

Fact 3.6. *For some absolute constant $\delta, \rho > 0$, there exists a family of codes $C := \{C_m : \{0,1\}^m \to \{0,1\}^{d(m)}\}_{m\in\mathbb{N}}$ such that for every $m \in \mathbb{N}$ the rate of C_m is at least ρ and the relative distance of C_m is at least δ. Moreover, any codeword of C_m can be computed (i.e., encoded) in time* $\mathrm{poly}(m)$.

3.5.2 Algebraic Geometric Codes. In the construction of our communication protocol in Section 6, we require our codes to have some special algebraic properties that have been shown to be present in algebraic geometric codes [39]. First, we will introduce a couple of additional definitions, namely, systematicity and degree-t closure.

Definition 3.7 (Systematicity). Given $s \in \mathbb{N}$, a code $C : \Sigma^m \to \Sigma^d$ is *s-systematic* if there exists a size-s subset of $[d]$, which for convenience we identify with $[s]$, such that for every $x \in \Sigma^s$ there exists $w \in \Sigma^m$ in which $x = C(w)\,|_{[s]}$.

Definition 3.8 (Degree-t Closure). Let Σ be a finite field. Given two codes $C : \Sigma^m \to \Sigma^d, C' : \Sigma^{m'} \to \Sigma^d$ and positive integer t, we say that C' is a degree-t closure of C if, for every $w_1, \ldots, w_r \in \Sigma^m$ and $P \in \mathbb{F}[X_1, \ldots, X_r]$ of total degree at most t, it holds that $\omega := P(C(w_1), \ldots, C(w_r))$ is in the range of C', where $\omega \in \Sigma^d$ is defined coordinate-wise by the equation $\omega_i := P(C(w_1)_i, \ldots, C(w_r)_i)$.

The notion of degree-t closure can be seen as a natural extension of linearity and informally says that for any multivariate polynomial P (of total degree at most t) where the variables are codewords of C (and the addition and multiplication operations are coordinate-wise over Σ), the evaluation of P is a codeword of C'. Below we provide a self-contained statement of the result we rely on in Section 6; it follows from Theorem 7 in Reference [71], which gives an efficient construction of the algebraic geometric codes based on Reference [39]'s explicit towers of function fields.

Theorem 3.9 ([39, 71]). *There are two polynomial functions $\hat{r}, \hat{q} : \mathbb{N} \to \mathbb{N}$ such that for every $k \in \mathbb{N}$ and any prime square $q > \hat{q}(k)$, there are two code families $\mathcal{A} = \{A_n\}_{n\in\mathbb{N}}$, $\mathcal{B} = \{B_n\}_{n\in\mathbb{N}}$ such that the following holds for all $n \in \mathbb{N}$,*

- *A_n and B_n are n-systematic code with alphabet $\mathbb{F}_q$,*
- *A_n and B_n have block length less than $n \cdot \hat{r}(k)$.*
- *B_n has relative distance $\geq 1/2$,*
- *B_n is a degree-k closure of A_n, and,*
- *Any codeword in A_n or B_n can be computed (i.e., encoded) in time polynomial in n.*

We remark here that variants of the above theorem have previously found applications in the construction of special kinds of PCPs [12, 13]. In these works, the theorems are also stated in a language similar to Theorem 3.9 above.

We conclude this section by a proof sketch of the above theorem that was suggested to us by Kopparty [50]. Fix $k \in \mathbb{N}$ and define $\varepsilon := \frac{1}{20k}$ (the constant 20 is an arbitrarily chosen large number). The authors of Reference [39] provide us with a family of curves $\mathcal{C} = \{C_i\}_{i\in\mathbb{N}}$ over a field $\mathbb{F}_q$ for any prime square $q > \frac{1}{\varepsilon^2}$, such that for every $n \in \mathbb{N}$, we have that C_n has at least $\ell := 4nk$ rational points and genus at most $g := 4\varepsilon k \cdot n$. Fix $n \in \mathbb{N}$. Let P be a rational point on C_n. Consider the Riemann-Roch space $\mathcal{L}(m \cdot P)$ for some $m \in \mathbb{N}$. This has dimension at least $m + 1 - g$, and any two elements have at most m common zeroes among the rational points of C_n.

Now we can define our codes A_n and B_n as specified in Theorem 3.9. Pick any set S of ℓ $\mathbb{F}_q$-rational points of C_n that does not contain P. Then A_n (respectively, B_n) is the code of evaluations of elements of $\mathcal{L}((\frac{\ell}{2k}) \cdot P)$ (respectively, $\mathcal{L}((\frac{\ell}{2}) \cdot P)$) at the points of S. Note that B_n is a degree-k closure of A_n. The dimensions of both A_n and B_n are at least $\frac{\ell}{2k} + 1 - 4\varepsilon k \cdot n \geq n$. Therefore, the rate of both A_n and B_n is at least $\frac{n}{\ell} > \frac{1}{4k}$. Also the relative distance of B_n is at least $\frac{1}{2}$ as any two codewords in B_n agree on at most $\ell/2$ coordinates. Further, every linear code is systematic,

and to have the same set of coordinates to be systematic for B_n and A_n, it suffices to note that a systematic set of coordinates for A_n is also one for B_n. Finally, the polynomial time encoding of the above codes was shown in Reference [71].

4 PRODUCT SPACE PROBLEMS AND POPULAR HYPOTHESES

In this section, we define a class of computational problems called Product Space Problems (PSP). As the name suggests, a problem in this class is defined on a class of functions whose domain is a k-ary Cartesian Product, i.e., $f : X_1 \times \cdots \times X_k \to \{0,1\}$. The input of the problem are subsets[13] $A_1 \subseteq X_1, \ldots, A_k \subseteq X_k$, and the goal is to determine whether there exists $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ such that $f(a_1, \ldots, a_k) = 1$. The size of the problem is determined by $\max_{i \in [k]} |A_i|$. A formal definition of PSP can be found below.

Definition 4.1 (Product Space Problem). Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be any function and $\mathcal{F} := \{f_{N,k} : \{0,1\}^{m(N,k) \times k} \to \{0,1\}\}_{N,k \in \mathbb{N}}$ be a family of Boolean functions indexed[14] by N and k. For each $k \in \mathbb{N}$, the *product space problem* PSP $(k, \mathcal{F})$ of order N is defined as follows: Given k subsets $A_1, \ldots, A_k$ of $\{0,1\}^{m(N,k)}$, each of cardinality at most N as input, determine whether there exists $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ such that $f_{N,k}(a_1, \ldots, a_k) = 1$. We use the following shorthand PSP$(k, \mathcal{F}, N)$ to describe PSP$(k, \mathcal{F})$ of order N.

In all the PSPs considered in this article, the input length $m(N,k)$ is always at most poly$(k) \cdot \log N$ and $f_{N,k}$ is always computable in time poly$(m(N,k))$. In such a case, there is a trivial $N^{k+o_k(1)}$-time algorithm to solve PSP$(k, \mathcal{F}, N)$: enumerating all $(a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k$ and check whether $f_{N,k}(a_1, \ldots, a_k) = 1$. The rest of this section is devoted to rephrasing the hypotheses (SETH, ETH, W[1] $\neq$ FPT and the k-SUM Hypothesis) in terms of lower bounds for PSPs. The function families $\mathcal{F}$'s, and running time lower bounds will depend on the hypotheses. For example, SETH will corresponds to set disjointness, whereas W[1] $\neq$ FPT will correspond to a generalization of equality called "multi-equality"; the former will give an $N^{k(1-o(1))}$ running time lower bound, whereas the latter only rules out FPT time algorithms.

We note that the class of problems called "locally characterizable sets" introduced by Goldreich and Rothblum [41] is closely related to PSPs. Elaborating, we may interpret locally characterizable sets as the negation of PSPs, i.e., for any PSP$(k, \mathcal{F}, N)$, we may define the corresponding locally characterizable set $\mathcal{S}$ as follows:

$$\mathcal{S} = \{(A_1, \ldots, A_k) \mid \text{for all } (a_1, \ldots, a_k) \in A_1 \times \cdots \times A_k \text{ we have } f_{N,k}(a_1, \ldots, a_k) = 0\}.$$

4.1 k-SUM Hypothesis

To familiarize the readers with our notations, we will start with the k-SUM Hypothesis, which is readily in the PSP form. Namely, the functions in the family are the Sum-Zero functions that checks whether the sum of k integers is zero:

Definition 4.2 (Sum-Zero). Let $k, m \in \mathbb{N}$. $\textsc{SumZero}_{m,k} : (\{0,1\}^m)^k \to \{0,1\}$ is defined by

$$\textsc{SumZero}_{m,k}(x_1, \ldots, x_k) = \begin{cases} 1 \text{ if } \sum\limits_{i \in [k]} x_i = 0, \\ 0 \text{ otherwise,} \end{cases}$$

where we think of each x_i as a number in $[-2^{m-1}, 2^{m-1} - 1]$, and the addition is over $\mathbb{Z}$.

[13] Each A_i will be explicitly given as part of the input through the elements that it contains.

[14] Here we choose to use N instead of n, to index $\mathcal{F}$, as it is a well-established convention to use n for the number of variables of a SAT formula (and SAT formulas do appear later in this section), and our index N is not the same as the number of variables n of the SAT formulas.

The function family $\mathcal{F}^{\textsc{SumZero}}$ can now be defined as follows.

Definition 4.3 (Sum-Zero Function Family). Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function defined by $m(N,k) = 2k\lceil \log N \rceil$. $\mathcal{F}^{\textsc{SumZero}}$ is defined as $\{\textsc{SumZero}_{m(N,k),k}\}_{N\in\mathbb{N},k\in N}$.

The following proposition is immediate from the definition of the k-SUM Hypothesis.

Proposition 4.4. *Assuming the k-SUM Hypothesis, for every integer $k \geq 3$ and every $\varepsilon > 0$, no $O(N^{\lceil k/2\rceil-\varepsilon})$-time algorithm can solve $\mathsf{PSP}(k, \mathcal{F}^{\textsc{SumZero}}, N)$ for all $N \in \mathbb{N}$.*

4.2 Set Disjointness and SETH

We recall the k-way disjointness function, where given k bit strings as input the function evaluates to 1 if and only if the coordinate-wise AND of the k strings is the all zeroes string. The k-way disjointness function has been studied extensively in the literature (see, e.g., Reference [52] and references therein), and we formally define it below.

Definition 4.5 (Set Disjointness). Let $k, m \in \mathbb{N}$. $\textsc{Disj}_{m,k} : (\{0,1\}^m)^k \to \{0,1\}$ is defined by

$$\textsc{Disj}_{m,k}(x_1,\ldots,x_k) = \neg\left(\bigvee_{i\in[m]}\left(\bigwedge_{j\in[k]}(x_j)_i\right)\right).$$

The function family $\mathcal{F}_c^{\textsc{Disj}}$ can now be defined as follows.

Definition 4.6 (Set Disjointness Function Family). For every $c \in \mathbb{N}$, let $m_c : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be a function defined by $m_c(N,k) = c\lceil k \log N\rceil$. $\mathcal{F}_c^{\textsc{Disj}}$ is defined as $\{\textsc{Disj}_{m_c(N,k),k}\}_{N\in\mathbb{N},k\in N}$.

We have the following proposition, which follows easily from the definition of SETH and its well-known connection to the Orthogonal Vectors Hypothesis [78].

Proposition 4.7. *Let $k \in \mathbb{N}$ such that $k > 1$. Assuming* SETH, *for every $\varepsilon > 0$ there exists $c := c_\varepsilon \in \mathbb{N}$ such that no $O(N^{k(1-\varepsilon)})$-time algorithm can solve $\mathsf{PSP}(k, \mathcal{F}_c^{\textsc{Disj}}, N)$ for all $N \in \mathbb{N}$.*

Proof. Fix $\varepsilon > 0$ and $k > 1$. By SETH, there exists $w := w(\varepsilon) \in \mathbb{N}$ and $c := c(\varepsilon) \in \mathbb{N}$ such that no algorithm can solve w-CNF-SAT in $O(2^{(1-\varepsilon)n})$ time, where n is the number of variables and $m \leq cn$ is the number of clauses. For every w-CNF-SAT formula ϕ, we will build $A_1^\phi, \ldots, A_k^\phi \subseteq \{0,1\}^m$ each of cardinality $N := 2^{n/k}$ such that there exists $(a_1,\ldots,a_k) \in A_1^\phi \times \cdots \times A_k^\phi$ such that $\textsc{Disj}_{m,k}(a_1,\ldots,a_k) = 1$ if and only if ϕ is satisfiable. Thus, if there was an $O(N^{k(1-\varepsilon)})$-time algorithm that can solve $\mathsf{PSP}(k, \mathcal{F}_c^{\textsc{Disj}}, N)$ for all $N \in \mathbb{N}$, then it would violate SETH.

All that remains is to show the construction of $A_1^\phi, \ldots, A_k^\phi$ from ϕ. Fix $i \in [k]$. For every partial assignment σ to the variables $x_{(i-1)\cdot(n/k)+1}, \ldots, x_{i\cdot(n/k)}$ we build an m-bit vector $a_\sigma \in A_i^\phi$ as follows: $\forall j \in [m]$, we have $a_\sigma(j) = 0$ if σ satisfies the jth clause, and $a_\sigma(j) = 1$ otherwise (i.e., the clause is not satisfied, or its satisfiability is indeterminate). It is easy to verify that there exists $(a_1,\ldots,a_k) \in A_1^\phi \times \cdots \times A_k^\phi$ such that $\textsc{Disj}_{m,k}(a_1,\ldots,a_k) = 1$ if and only if ϕ is satisfiable. □

We remark that we can prove a similar statement as that of Proposition 4.7 for ETH: Assuming ETH, there exists k_0 such that for every $k > k_0$ there exists $c := c_{k_0} \in \mathbb{N}$ such that no $O(N^{o(k)})$-time algorithm can solve $\mathsf{PSP}(k, \mathcal{F}_c^{\textsc{Disj}}, N)$ for all $N \in \mathbb{N}$. However, instead of associating ETH with Disj, we will associate with the Boolean function MultEq (which will be defined in the next subsection) and its corresponding PSP. This is because associating ETH with MultEq provides a more elementary proof of Theorem 1.4 (in particular, we will not need to use algebraic geometric codes—which are essentially inevitable if we associate ETH with Disj).

4.3 W[1] ≠ FPT Hypothesis and ETH

Again, we recall the k-way Equality function that has been studied extensively in literature (see, e.g., References [5, 7, 19, 25, 53, 65] and references therein).

Definition 4.8 (Equality). Let $k, m \in \mathbb{N}$. $\text{EQ}_{m,k} : (\{0,1\}^m)^k \to \{0,1\}$ is defined by

$$\text{EQ}_{m,k}(x_1, \ldots, x_k) = \bigwedge_{i,j \in [k]} (x_i = x_j),$$

where $x_i = x_j$ is a shorthand for $\bigwedge_{p \in [m]} (x_i)_p = (x_j)_p$.

Unfortunately, the PSP associated with EQ is in fact not hard: Given sets $A_1, \ldots, A_k$, it is easy to find whether they share an element by just sorting the combined list of $A_1 \cup \cdots \cup A_k$. Hence, we will need a generalization of the equality function to state our hard problem. Before we do so, let us first state an intermediate helper function, which is a variant of the usual equality function where some of the k inputs may be designed as "null" and the function only checks the equality over the non-null inputs. We call this function the Selective-Equality (SELEQ) function. For notational convenience, in the definition below, each of the k inputs is now viewed as $(x_i, y_i) \in \{0,1\}^{m-1} \times \{\bot, \top\}$; if $y_i = \bot$, then (x_i, y_i) represents the "null" input.

Definition 4.9 (Selective-Equality). Let $k, m \in \mathbb{N}$. $\text{SELEQ}_{m,k} : (\{0,1\}^{m-1} \times \{\bot, \top\})^k \to \{0,1\}$ is defined by

$$\text{SELEQ}_{m,k}((x_1, y_1), \ldots, (x_k, y_k)) = \bigwedge_{i,j \in [k]} ((y_i = \bot) \vee (y_j = \bot) \vee (x_i = x_j)).$$

Next, we introduce the variant of EQ whose associated PSP is hard under W[1] ≠ FPT and ETH. In the settings of both Equality and Selective-Equality defined above, there is only one unknown that is given in each of the k inputs $a_1 \in A_1, \ldots, a_k \in A_k$, and the functions check whether they are equal. The following function, which we name Multi-Equality, is the t-unknown version of Selective-Equality. Specifically, the ith part of the input is now a tuple $((x_{i,1}, y_{i,1}), \ldots, (x_{i,t}, y_{i,t}))$, where $x_{i,1}, \ldots, x_{i,t}$ are bit strings representing the supposed values of the t unknowns while, similar to Selective-Equality, each $y_{i,q} \in \{\bot, \top\}$ is a symbol indicating whether $(x_{i,q}, y_{i,q})$ is the "null" input. Below is the formal definition of MULTEQ; note that for convenience, we use $(x_{i,q}, y_{i,q})_{q \in [t]}$ as a shorthand for $((x_{i,1}, y_{i,1}), \ldots, (x_{i,t}, y_{i,t}))$, i.e., the ith part of the input.

Definition 4.10 (Multi-Equality). Let $k, t \in \mathbb{N}$ and let $m \in \mathbb{N}$ be any positive integer such that m is divisible by t. Let $m' = m/t$. $\text{MULTEQ}_{m,k,t} : (\{0,1\}^{m'-1} \times \{\bot, \top\})^t)^k \to \{0,1\}$ is defined by

$$\text{MULTEQ}_{m,k,t}((x_{1,q}, y_{1,q})_{q \in [t]}, \ldots, (x_{k,q}, y_{k,q})_{q \in [t]}) = \bigwedge_{q \in [t]} \text{SELEQ}_{m',k}((x_{1,q}, y_{1,q}), \ldots, (x_{k,q}, y_{k,q})).$$

Next, we define the family $\mathcal{F}^{\text{MULTEQ}}$; note that in the definition below, we simply choose $t(k)$, the number of unknowns, to be $k + \binom{k}{2} + \binom{k}{3}$. As we will see later, this is needed for ETH-hardness. For W[1]-hardness, it suffices to use a smaller number of variables. However, we choose to define $t(k)$ in such a way so that we can conveniently use one family for both ETH and W[1]-hardness.

Definition 4.11. Let $t : \mathbb{N} \to \mathbb{N}$ be defined by $t(k) = k + \binom{k}{2} + \binom{k}{3}$. Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be defined by $m(N,k) = t(k)(1 + k\lceil \log N \rceil)$. We define $\mathcal{F}^{\text{MULTEQ}}$ as $\{\text{MULTEQ}_{m(N,k),k,t(k)}\}_{N \in \mathbb{N}, k \in \mathbb{N}}$.

We next show a reduction from k-Clique to PSP$(k', \mathcal{F}^{\text{MULTEQ}})$, where $k' = \binom{k}{2}$. The overall idea of the reduction is simple. First, we associate the integers in $[k']$ naturally with the elements of $\binom{[k]}{2}$. We then create the sets $(A_{\{i,j\}})_{\{i,j\} \subseteq [k], i \neq j}$ in such a way that each element of the set $A_{\{i,j\}}$

corresponds to picking an edge between the ith and the jth vertices in the supposed k-clique. Then, MultEq is used to check that these edges are consistent, i.e., that, for every $i \in [k]$, $a_{\{i,j\}}$ and $a_{\{i,j'\}}$ pick the same vertex to be the ith vertex in the clique for all $j, j' \in [k] \setminus \{i\}$. This idea is formalized in the following proposition and its proof.

Proposition 4.12. *Let $k \in \mathbb{N}$ and $k' = \binom{k}{2}$. There exists a $\mathsf{poly}(N,k)$-time reduction from any instance (G,k) of Clique to an instance $(A_1, \ldots, A_{k'})$ of the $\mathsf{PSP}(k', \mathcal{F}^{\textsc{MultEq}}, N')$, where N denotes the number of vertices of G and $N' = \binom{N}{2}$.*

Proof. Given a Clique instance[15] (G,k), the reduction proceeds as follows. For convenience, we assume that the vertex set $V(G)$ is $[N]$. Furthermore, we associate the elements of $[k']$ naturally with the elements of $\binom{[k]}{2}$. For the sake of conciseness, we sometimes abuse notation and think of $\{i,j\}$ as an ordered pair (i,j), where $i < j$. For every $\{i,j\} \in \binom{[k]}{2}$ such that $i < j$, the set $A_{\{i,j\}}$ contains one element $a^{\{u,v\}}_{\{i,j\}} = (a^{\{u,v\}}_{\{i,j\},1}, \ldots, a^{\{u,v\}}_{\{i,j\},t(k')})$ for each edge $\{u,v\} \in E(G)$ such that $u < v$, where

$$a^{\{u,v\}}_{\{i,j\},q} = \begin{cases} (u, \top) & \text{if } q = i, \\ (v, \top) & \text{if } q = j, \\ (0, \bot) & \text{otherwise.} \end{cases}$$

Note that in the definition above, we view u, v, and 0 as $(m(N',k')/t(k') - 1)$-bit strings, where $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ is as in Definition 4.11. Also note that each set $A_{\{i,j\}}$ has size at most $\binom{N}{2} = N'$, meaning that $(A_{\{i,j\}})_{\{i,j\} \subseteq [k]}$ is indeed a valid instance of $\mathsf{PSP}(k', \mathcal{F}^{\textsc{MultEq}}, N')$. For brevity, below we will use f as a shorthand for $\textsc{MultEq}_{m(N',k'),k',t(k')}$.

($\Rightarrow$) Suppose that (G,k) is a YES instance for Clique, i.e., there exists a k-clique $\{u_1, \ldots, u_k\}$ in G. Assume without loss of generality that $u_1 < \cdots < u_k$. We claim that

$$f\left(\left(a^{\{u_i,u_j\}}_{\{i,j\}}\right)_{i,j \in [k], i<j}\right) = 1.$$

To see that this is the case, observe that for every $q \in [t(k')]$ and for every $\{i,j\} \subseteq [k]$ such that $i < j$, we have either $a^{\{u_i,u_j\}}_{\{i,j\},q} = (0, \bot)$ or $a^{\{u_i,u_j\}}_{\{i,j\},q} = (u_q, \top)$. This means that, $\textsc{SelEq}((a^{\{u_i,u_j\}}_{\{i,j\},q})_{i,j \in [k], i<j}) = 1$ for every $q \in [t(k')]$.

($\Leftarrow$) Suppose that $(A_{\{i,j\}})_{\{i,j\} \subseteq [k]}$ is a YES instance for $\mathsf{PSP}(k', \mathcal{F}^{\textsc{MultEq}}, N')$, i.e., there exists $a^*_{\{i,j\}} \in A_{\{i,j\}}$ for every $\{i,j\} \subseteq [k]$ such that $f((a^*_{\{i,j\}})_{\{i,j\} \subseteq [k]}) = 1$. Suppose that $a^*_{\{i,j\}} = (x^*_{\{i,j\},1}, y^*_{\{i,j\},1}, \ldots, x^*_{\{i,j\},t(k')}, y^*_{\{i,j\},t(k')})$. From this solution $\{a^*_{\{i,j\}}\}_{\{i,j\} \subseteq [k]}$, we can recover the k-clique as follows. For each $i \in [k]$, pick an arbitrary $j(i) \in [k]$ that is not equal to i. Let u_i be $x^*_{\{i,j\},i}$. We claim that $u_1, \ldots, u_k$ forms a k-clique in G. To show this, it suffices to argue that, for every distinct $i, i' \in [k]$, there is an edge between u_i and $u_{i'}$ in G. To see that this holds, consider $a^*_{\{i,i'\}}$. Since $y^*_{\{i,i'\},i} = y^*_{\{i,j(i)\},i} = \top$, we have $x^*_{\{i,i'\},i} = x^*_{\{i,j(i)\},i} = u_i$. Similarly, we have $x^*_{\{i,i'\},i'} = u_{i'}$. Since $a^*_{\{i,i'\}} \in A_{\{i,i'\}}$ and from how the set $A_{\{i,i'\}}$ is defined, we have $\{u_i, u_{i'}\} \in E(G)$, which concludes our proof. □

Lemma 4.13. *Assuming $\mathsf{W}[1] \neq \mathsf{FPT}$, for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot \mathsf{poly}(N)$ time algorithm that can solve $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$ for every $N, k \in \mathbb{N}$.*

Proof. Suppose for the sake of contradiction that, for some computable function $T : \mathbb{N} \to \mathbb{N}$, there is a $T(k) \cdot \mathsf{poly}(N)$ time algorithm $\mathcal{A}$ that can solve $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$ for every $N, k \in \mathbb{N}$.

[15] We assume without loss of generality that G does not contain any self-loop.

We will show that this algorithm can also be used to solve k-Clique parameterized by k in FPT time.

Given an instance (G, k) of k-Clique, we first run the reduction from Proposition 4.12 to produce an instance $(A_1, \ldots, A_{k'})$ of $\mathsf{PSP}(k', \mathcal{F}^{\textsc{MultEq}}, N')$ in $\mathrm{poly}(N, k)$ time, where $N = |V(G)|$, $N' = \binom{N}{2}$ and $k' = \binom{k}{2}$. We then run $\mathcal{A}$ on $(A_1, \ldots, A_{k'})$, which takes time $T(k') \cdot \mathrm{poly}(N')$. This means that we can also solve our k-Clique instance (G, k) in time $\mathrm{poly}(N, k) + T(k') \cdot \mathrm{poly}(N') = \mathrm{poly}(N, k) + T(\binom{k}{2}) \cdot \mathrm{poly}(N)$, which is FPT time. Since k-Clique is W[1]-complete, this contradicts with W[1] $\neq$ FPT. □

Next, we will prove ETH-hardness of $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}})$. Specifically, we will reduce a 3-CNF-SAT instance ϕ where each variable appears in at most three clauses to an instance of $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$, where $N = 2^{O(n/k)}$ and n denotes the number of variables in ϕ. The overall idea is to partition the set of clauses into k parts of equal size and use each element in A_j to represent a partial assignment that satisfies all the clauses in the jth partition. This indeed means that each group has size $2^{O(n/k)}$ as intended. However, choosing the unknowns are not as straightforward as in the reduction from k-Clique above; in particular, if we view each variable by itself as an unknown, then we would have n unknowns, which is much more than the designated $t(k) = k + \binom{k}{2} + \binom{k}{3}$ unknowns. This is where we use the fact that each variable appears in at most three clauses: We group the variables of ϕ together based on which partitions they appear in and view each group as a single variable. Since each variable appears in at most three clauses, the number of ways they can appear in the k partitions is $k + \binom{k}{2} + \binom{k}{3}$, which is indeed equal to $t(k)$. The ideas are formalized below.

Proposition 4.14. *Let $k \in \mathbb{N}$. There exists a $\mathrm{poly}(N, k)$-time reduction from any instance ϕ of 3-CNF-SAT such that each variable appears in at most three clauses in an instance $(A_1, \ldots, A_k)$ of the $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$, where $N = 2^{3\lceil m/k \rceil}$ and m denotes the number of clauses in ϕ.*

Proof. Given a 3-CNF-SAT formula ϕ such that each variable appears in at most three clauses. Let the variable set of ϕ be $\mathcal{Z} = \{z_1, \ldots, z_n\}$ and the clauses of ϕ be $C = \{C_1, \ldots, C_m\}$. Then for every $k \in \mathbb{N}$, we produce an instance $(A_1, \ldots, A_k)$ of $\mathsf{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$, where $N = 2^{3\lceil m/k \rceil}$ as follows.

First, we partition the clause set C into k parts $C_1, \ldots, C_k$ each of size at most $\lceil m/k \rceil$. For each variable z_i, let S_i denote $\{j \in [k] \mid \exists C_h \in C_j$ such that $z_i \in C_h$ or $\overline{z}_i \in C_h\}$. Since every z_i appears in at most three clauses, we have $S_i \in \binom{[k]}{\leq 3}$. For each $S \in \binom{[k]}{\leq 3}$, let $\nu(S)$ denote the set of all variables z_i's such that $S_i = S$ (i.e., S is exactly equal to the set of all partitions that z_i appears in). The general idea of the reduction is that we will view a partial assignment to the variables in $\nu(S)$ as an unknown for MultEq; let us call this unknown X_S (hence there are $k + \binom{k}{2} + \binom{k}{3} = t(k)$ unknowns). For each $j \in [k]$, A_j contains one element for each partial assignment to the variables that appear in the clauses in C_j and that satisfies all the clauses in C_j. Such a partial assignment specifies $(1 + k + \binom{k}{2})$ unknowns: All the X_S such that $j \in S$. The MultEq function is then used to check the consistency between the partial assignments to the variables from different A_j's.

To formalize this intuition, we first define more notations. Let $\widetilde{m} = 3k\lceil m/k \rceil$. For every subsets $T \subseteq T' \subseteq \mathcal{Z}$ and every partial assignment $\alpha : T' \to \{0, 1\}$, the restriction of α to T, denoted by $\alpha|_T$ is the function from T to $\{0, 1\}$ where $\alpha|_T(z) = \alpha(z)$ for every $z \in T$. Furthermore, we define the operator $\mathrm{ext}(\alpha)$, which "extends" α to $\widetilde{m}$ bits, i.e., the ith bit of $\mathrm{ext}(\alpha)$ is $\alpha(z_i)$ if $z_i \in T$ and is zero otherwise. Finally, we use $\mathrm{var}(C_j)$ to denote the set of all variables that appear in at least one of the clauses from C_j, i.e., $\mathrm{var}(C_j) = \bigcup_{C \in C_j} \mathrm{var}(C)$, where $\mathrm{var}(C)$ denotes $\{z_i \in \mathcal{Z} \mid z_i \in C$ or $\overline{z}_i \in C\}$.

Now, since our $t(k)$ is exactly $\binom{k}{\leq 3}$, we can associate each element of $[t]$ with a subset $S \in \binom{[k]}{\leq 3}$. Specifically, for each partial assignment $\alpha : \mathrm{var}(C_j) \to \{0,1\}$ such that α satisfies all the clauses in C_j, the set A_j contains an element $a_j^\alpha = (a_{j,S}^\alpha)_{S \in \binom{[k]}{\leq 3}}$ where, for every $S \in \binom{[k]}{\leq 3}$,

$$a_{j,S}^\alpha = \begin{cases} (\mathrm{ext}(\alpha|_{\nu(S)}), \top) & \text{if } j \in S, \\ (0^{\widetilde{m}}, \bot) & \text{otherwise.} \end{cases}$$

Fix $S \in \binom{[k]}{\leq 3}$. For every $j \in S$, observe that $\nu(S) \subseteq \mathrm{var}(C_j)$. Moreover, since each C_j contains at most $\lceil m/k \rceil$ clauses, there are at most $3\lceil m/k \rceil$ variables in $\mathrm{var}(C_j)$. This means that A_j has size at most $2^{3\lceil m/k \rceil}$. Hence, $(A_1, \ldots, A_k)$ is indeed a valid instance of PSP $(k, \mathcal{F}^{\textsc{MultEq}}, N)$, where $N = 2^{3\lceil m/k \rceil}$. For brevity, below we will use f as a shorthand for $\textsc{MultEq}_{m(N,k),k,t(k)}$.

($\Rightarrow$) Suppose that ϕ is satisfiable. Let $\alpha : C \to \{0,1\}$ be an assignment that satisfies all the clauses. Let $a_j^* = a_j^{\alpha|_{\mathrm{var}(C_j)}} \in A_j$ for every $j \in [k]$. Observe that, for every $S \in \binom{[k]}{\leq 3}$ and every $j \in [k]$, we have either $a_{j,S}^* = (0^{\widetilde{m}}, \bot)$ or $a_{j,S}^* = (\mathrm{ext}(\alpha|_{\nu(S)}), \top)$. This indeed implies that $f(a_1^*, \ldots, a_k^*) = 1$.

($\Leftarrow$) Suppose that there exists $(a_1^{\alpha_1}, \ldots, a_k^{\alpha_k}) \in A_1 \times \cdots \times A_k$ such that $f(a_1^{\alpha_1}, \ldots, a_k^{\alpha_k}) = 1$. We construct an assignment $\alpha : \mathcal{Z} \to \{0,1\}$ as follows. For each $i \in [n]$, pick an arbitrary $j(i) \in [k]$ such that $z_i \in \mathrm{var}(C_{j(i)})$ and let $\alpha(z_i) = \alpha_{j(i)}(z_i)$. We claim that α satisfies every clause. To see this, consider any clause $C \in \mathcal{C}$. Suppose that C is in the partition C_j. It is easy to check that $f(a_1^{\alpha_1}, \ldots, a_k^{\alpha_k}) = 1$ implies that $\alpha|_{\mathrm{var}(C)} = \alpha_j|_{\mathrm{var}(C)}$. Since α_j is a partial assignment that satisfies C, α must also satisfy C. In other words, α satisfies all clauses of ϕ. □

Lemma 4.15. *Assuming* ETH, *for any computable function* $T : \mathbb{N} \to \mathbb{N}$, *there is no* $T(k) \cdot N^{o(k)}$ *time algorithm that can solve* $\mathrm{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$ *for every* $N, k \in \mathbb{N}$.

Proof. Let $\delta > 0$ be the constant in the running time lower bound in ETH. Suppose for the sake of contradiction that ETH holds but, for some function T, there is a $T(k) \cdot N^{o(k)}$ time algorithm $\mathcal{A}$ that can solve $\mathrm{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$ for every $N, k \in \mathbb{N}$. Thus, there exists a sufficiently large k such that the running time of $\mathcal{A}$ for solving $\mathrm{PSP}(k, \mathcal{F}^{\textsc{MultEq}}, N)$ is at most $O(N^{\delta k/10})$ for every $N \in \mathbb{N}$.

Given a 3-CNF formula ϕ such that each variable appears in at most three clauses. Let n, m denote the number of variables and the number of clauses of ϕ, respectively. We first run the reduction from Proposition 4.14 on ϕ with this value of k. This produces an instance $(A_1, \ldots, A_k)$ of PSP $(k, \mathcal{F}^{\textsc{MultEq}}, N)$, where $N = 2^{3\lceil m/k \rceil}$. Since each variable appears in at most three clauses, we have $m \leq 3n$, meaning that $N = O(2^{9n/k})$. By running $\mathcal{A}$ on this instance, we can decide whether ϕ is satisfiable in time $O(N^{\delta k/10}) = O(2^{0.9\delta n})$, contradicting ETH. □

5 COMMUNICATION PROTOCOLS AND REDUCTION TO GAP LABEL COVER

In this section, we first introduce a communication model for multiparty communication known in literature as the Simultaneous Message Passing model. Then, we introduce a notion of "efficient" communication protocols and connect the existence of such protocols to a reduction from PSP to a gap version of MaxCover.

5.1 Efficient Protocols in Simultaneous Message Passing Model

The two-player Simultaneous Message Passing (SMP) model was introduced by Yao [79] and has been extensively studied in the literature [51]. In the multiparty setting, the SMP model is considered popularly with the number-on-forehead model, where each player can see the input of all

the other players but not his own [10, 17]. In this article, we consider the multiparty SMP model where the inputs are given as in the number-in-hand model (like in References [37, 77]).

Simultaneous Message Passing Model. Let $f : \{0,1\}^{m \times k} \to \{0,1\}$. In the k-player simultaneous message passing communication model, we have k players each with an input $x_i \in \{0,1\}^m$ and a referee who is given an advice $\mu \in \{0,1\}^*$ (at the same time when the players are given the input). The communication task is for the referee to determine whether $f(x_1, \ldots, x_k) = 1$. The players are allowed to only send messages to the referee. In the randomized setting, we allow the players *and* the referee to jointly toss some random coins before sending messages, i.e., we allow public randomness.

Next, we introduce the notion of *efficient* protocols, which are in a nutshell one-round randomized protocols where the players and the referee are in a computationally bounded setting.

Efficient Protocols. Let π be a communication protocol for a problem in the SMP model. We say that π is a (w, r, ℓ, s)-efficient protocol if the following holds:

- The referee receives w bits of advice.
- The protocol is one-round with public randomness, i.e., the following actions happen sequentially:
  (1) The players receive their inputs and the referee receives his advice.
  (2) The players and the referee jointly toss r random coins.
  (3) Each player on seeing the randomness (i.e., results of r coin tosses) deterministically sends an ℓ-bit message to the referee.
  (4) Based on the advice, the randomness, and the total $\ell \cdot k$ bits sent from the players, the referee outputs accept or reject.
- The protocol has completeness 1 and soundness s, i.e.,
  —If $f(x_1, \ldots, x_k) = 1$, then there exists an advice on which the referee always accepts.
  —If $f(x_1, \ldots, x_k) = 0$, then, on any advice, the referee accepts with probability at most s.
- The players and the referee are computationally bounded, i.e., all of them perform all their computations in poly(m)-time.

The following proposition follows immediately from the definition of an efficient protocol and will be very useful in later sections for gap amplification.

Proposition 5.1. *Let $z \in \mathbb{N}$ and π be a communication protocol for a problem in the SMP model. Suppose π is a (w, r, ℓ, s)-efficient protocol. Then there exists a $(w, z \cdot r, z \cdot \ell, s^z)$-efficient protocol for the same problem.*

Proof. The proof follows by a simple repetition argument. More precisely, we repeat steps (2)–(4) in the protocol z times, each time using fresh randomness, but note that the z steps of drawing random coins can be clubbed into one step, and the decision by the referee can be reserved for the end of the entire protocol, wherein he or she accepts if and only if he or she would accept in each of the individual repetitions. □

5.2 Lower Bounds on Gap-MaxCover

The following theorem is the main conceptual contribution of the article: We show below that the existence of efficient protocols can translate (exact) hardness of PSPs to hardness of approximating MaxCover.

Theorem 5.2. *Let $m : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ be any function. Let $\mathcal{F} := \{f_{N,k} : \{0,1\}^{m(N,k) \times k} \to \{0,1\}\}_{N,k \in \mathbb{N}}$ be a family of Boolean functions indexed by N, k. Suppose there exists a (w, r, ℓ, s)-*

efficient protocol[16] *for* $f_{N,k}$ *in the* k*-player SMP model for every* $N, k \in \mathbb{N}$*. Then, there is a reduction from any instance* $(A_1, \ldots, A_k)$ *of* $\mathsf{PSP}(k, \mathcal{F}, N)$ *to* 2^w *label cover instances* $\{\Gamma^\mu\}_{\mu \in \{0,1\}^w}$ *such that*

- *The running time of the reduction is* $2^{w+r+\ell k}\mathsf{poly}(m(N,k))$.
- *Each* $\Gamma^\mu = (U^\mu = U_1^\mu \cup \cdots \cup U_q^\mu, W^\mu = W_1^\mu \cup \cdots \cup W_h^\mu; E^\mu)$ *has the following parameters:*
  —Γ^μ *has at most* Nk *right nodes, i.e.,* $|W^\mu| \leq Nk$,
  —Γ^μ *has* k *right super nodes, i.e.,* $h = k$,
  —Γ^μ *has* 2^r *left super nodes, i.e.,* $q = 2^r$,
  —Γ^μ*'s left alphabet size is at most* $2^{\ell k}$*, i.e.,* $|U_1^\mu|, \ldots, |U_q^\mu| \leq 2^{\ell k}$.
- *If* $(A_1, \ldots, A_k)$ *is a YES instance of* $\mathsf{PSP}(k, \mathcal{F}, N)$*, then* $\mathsf{MaxCover}(\Gamma^\mu) = 1$ *for some* $\mu \in \{0,1\}^w$.
- *If* $(A_1, \ldots, A_k)$ *is a NO instance of* $\mathsf{PSP}(k, \mathcal{F}, N)$*, then* $\mathsf{MaxCover}(\Gamma^\mu) \leq s$ *for every* $\mu \in \{0,1\}^w$.

Proof. Given a (w, r, ℓ, s)-efficient protocol π of $f_{N,k}$ and an instance $(A_1, \ldots, A_k)$ of $\mathsf{PSP}(k, \mathcal{F}, N)$, we will generate 2^w instances of MaxCover. Specifically, for each $\mu \in \{0,1\}^w$, we construct an instance $\Gamma^\mu = (U^\mu = U_1^\mu \cup \cdots \cup U_q^\mu, W^\mu = W_1^\mu \cup \cdots \cup W_h^\mu; E^\mu)$ of MaxCover as follows.

- Let $h = k$. For each $j \in [h]$, the right super-node W_j^μ contains one node for each $x_j \in A_j$.
- Let $q = 2^r$. For each random string $\gamma \in \{0,1\}^r$, the left-super node U_γ^μ contains one node for each of the possible accepting messages from the k players. Elaborating, each vertex in U_γ^μ corresponds to $(m_1, \ldots, m_k) \in (\{0,1\}^\ell)^k$ such that the following holds: In the protocol π, the referee on receiving an advice μ, jointly drawing the random string γ, and receiving the message m_i from the ith player (for all $i \in [k]$), decides to accept.
- We add an edge between $x_j \in W_j^\mu$ and $(m_1, \ldots, m_k) \in U_\gamma^\mu$ if m_j is equal to the message that j sends on an input x_j and a random string γ in the protocol π.

Observe that there is a bijection between labelings of Γ^μ and elements of $A_1 \times \cdots \times A_k$.

Now consider a labeling $S \subseteq W^\mu$ of Γ^μ and the corresponding $(x_1, \ldots, x_k) \in A_1 \times \cdots \times A_k$. For each random string $\gamma \in \{0,1\}^r$, observe that the referee accepts on an input $(x_1, \ldots, x_k)$, an advice μ, and a random string γ if and only if there is a vertex $u \in U_\gamma$ (corresponding to the messages sent by the players) that has an edge to every vertex in S. Therefore, the acceptance probability of the protocol on advice μ is the same as the fraction of left super-nodes covered by S. The completeness and soundness then easily follows:

Completeness. If there exists $(x_1, \ldots, x_k) \in A_1 \times \cdots \times A_k$ such that $f_{N,k}(x_1, \ldots, x_k) = 1$, then there is an advice $\mu \in \{0,1\}^w$ on which the referee always accepts for this input $(x_1, \ldots, x_k)$, meaning that the corresponding labeling covers every left super-node of Γ^μ, i.e., $\mathsf{MaxCover}(\Gamma^\mu) = 1$.

Soundness. If $f_{N,k}(x_1, \ldots, x_k) = 0$ for every $(x_1, \ldots, x_k) \in A_1 \times \cdots \times A_k$, then, for any advice $\mu \in \{0,1\}^w$, the referee accepts with probability at most s on every input $(x_1, \ldots, x_k) \in A_1 \times \cdots \times A_k$. This means that, for any $\mu \in \{0,1\}^w$, no labeling covers more than s fraction of left the super-nodes. In other words, $\mathsf{MaxCover}(\Gamma^\mu) \leq s$ for all $\mu \in \{0,1\}^w$. □

For the rest of this subsection, we will use the following shorthand. Let $\Gamma = (U = U_1 \cup \cdots \cup U_q, W = W_1 \cup \cdots \cup W_h; E)$ be a label cover instance, and we use the shorthand $\Gamma(N, k, r, \ell)$ to say that the label cover instance has the following parameters:

[16] w, r, ℓ, and s can depend on N and k.

- Γ has at most Nk right nodes, i.e., $|W| \leq Nk$,
- Γ has k right super nodes, i.e., $h = k$,
- Γ has 2^r left super nodes, i.e., $q = 2^r$,
- Γ has left alphabet size of at most $2^{\ell k}$, i.e., $|U_1|, \ldots, |U_q| \leq 2^{\ell k}$.

The rest of this section is devoted to combining Theorem 5.2 with the results in Section 4 to obtain conditional hardness for the gap-MaxCover problem, assuming that we have efficient protocols with certain parameters. These protocols will be devised in the three subsequent sections.

Understanding the Parameters. Before we state the exact dependency of parameters, let us first discuss some intuition behind it. First, if we start with an instance of $\mathsf{PSP}(k, \mathcal{F}, N)$, Theorem 5.2 will produce 2^w instances of $\Gamma(N, k, r, \ell)$. Roughly speaking, since we want the lower bounds from PSP to translate to MaxCover, we would like the number of instances to be $N^{o(1)}$, meaning that we want $w = o(\log N)$. Recall that in all function families we consider $m = \Theta_k(\log N)$. Hence, this requirement is the same as $w = o_k(m)$. Moreover, we would like the instance size of $\Gamma(N, k, r, \ell)$ to also be $O_k(N)$, meaning that the number of left vertices, $2^{r+\ell k}$ has to be $O_k(N)$. Thus, it suffices to have a protocol where $r + \ell k = o_k(m)$.

If we additionally want the hardness to translate also to k-DomSet, then the parameter dependencies become more subtle. Specifically, applying Theorem 3.1 to the MaxCover instances results in a blow-up of $\sum_{j\in[q]} k^{|U_j|} = 2^r \cdot k^{2^{\ell k}} = 2^{r+(\log k)\cdot 2^{\ell k}}$. We also want this to be at most $N^{o(1)}$, meaning that we need $r + (\log k) \cdot 2^{\ell k} = o(\log N) = o_k(m)$. In other words, it suffices for us to require that $\ell k < {}^{(\log m)}/_{\beta}$ for some constant $\beta > 1$. The exact parameter dependencies are formalized below.

5.2.1 SETH. In this subsubsection, we prove Theorem 1.5.

Corollary 5.3. *For any $c \in \mathbb{N}$, let $\mathcal{F}_c^{\textsc{Disj}}$ be the family of Boolean functions as defined in Definition 4.6. For every $\delta > 0$, suppose there exists a (w, r, ℓ, s)-efficient protocol for $\textsc{Disj}_{m,k}$ in the k-player SMP model for every $k \in \mathbb{N}$ and every $m \in \mathbb{N}$, such that $w \leq \delta m$ and $r + \ell k = o_k(m)$. Then, assuming* SETH, *for every $\varepsilon > 0$ and integer $k > 1$, no $O(N^{k(1-\varepsilon)})$-time algorithm can distinguish between* $\mathsf{MaxCover}(\Gamma) = 1$ *and* $\mathsf{MaxCover}(\Gamma) \leq s$ *for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N \in \mathbb{N}$. Moreover, if $\ell < {}^{(\log m)}/_{\beta \cdot k}$ for some constant $\beta > 1$, then assuming* SETH, *for every $\varepsilon > 0$ and integer $k > 1$, no $O(N^{k(1-\varepsilon)})$-time algorithm can distinguish between* $\mathsf{DomSet}(G) = k$ *and* $\mathsf{DomSet}(G) \geq (\frac{1}{s})^{\frac{1}{k}} \cdot k$ *for any graph G with at most $O_k(N)$ vertices, for all $N \in \mathbb{N}$.*

Proof. The proof of the first part of the theorem statement is by contradiction. Suppose there is an $O(N^{k(1-\varepsilon)})$-time algorithm $\mathcal{A}$ for some fixed constant $\varepsilon > 0$ and integer $k > 1$, which can distinguish between $\mathsf{MaxCover}(\Gamma) = 1$ and $\mathsf{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N \in \mathbb{N}$. From Proposition 4.7, we have that there exists $c_\varepsilon \in \mathbb{N}$ such that no $O(N^{k(1-\varepsilon/2)})$-time algorithm can solve $\mathsf{PSP}(k, \mathcal{F}_{c_\varepsilon}^{\textsc{Disj}}, N)$ for all $N \in \mathbb{N}$. Fix $\delta = {}^{\varepsilon}/_{3c_\varepsilon}$. Next, by considering Theorem 5.2 for the case of (w, r, ℓ, s)-efficient protocols, we have that there are 2^w label cover instances $\{\Gamma^\mu\}_{\mu\in\{0,1\}^w}$, which can be constructed in $2^{\delta m(1+o_k(1))}$ time. Note that $2^{\delta m(1+o_k(1))} = N^{k\varepsilon/3(1+o_k(1))}$ by our choice of δ. Thus, we can run $\mathcal{A}$ on each Γ^μ and solve $\mathsf{PSP}(k, \mathcal{F}_c^{\textsc{Disj}}, N)$ for all $N, k \in \mathbb{N}$ in time less than $N^{k(1-\varepsilon/2)}$. This contradicts Proposition 4.7.

To prove the second part of the theorem statement, we apply the reduction described in Theorem 3.1 and note that $2^r = N^{o(1)}$ and $k^{2^{\ell k}} = 2^{(\log_2 k)\cdot(m(N,k))^{1/\beta}} = N^{o(1)}$. □

The proof of Theorem 1.5 follows by plugging in the parameters of the protocol described in Corollary 6.2 to the above corollary.

5.2.2 ETH. In this subsubsection, we prove Theorem 1.4.

Corollary 5.4. *Let $\mathcal{F}^{\text{MultEq}}$ be the family of Boolean functions as defined in Definition 4.11. Suppose there exists a (w, r, ℓ, s)-efficient protocol for $\mathsf{MultEq}_{m,k,t}$ in the k-player SMP model for every $k, t, m \in \mathbb{N}$ such that $w + r + \ell k = o_k(m)$. Then, assuming ETH, for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot N^{o(k)}$ time algorithm that can distinguish between $\mathsf{MaxCover}(\Gamma) = 1$ and $\mathsf{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N, k \in \mathbb{N}$. Moreover, if $\ell < (\log m)/\beta \cdot k$ for some constant $\beta > 1$, then assuming ETH, for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot N^{o(k)}$ time algorithm that can distinguish between $\mathsf{DomSet}(G) = k$ and $\mathsf{DomSet}(G) \geq (\frac{1}{s})^{\frac{1}{k}} \cdot k$ for any graph G with at most $O_k(N)$ vertices, for all $N, k \in \mathbb{N}$.*

Proof. The proof of the first part of the theorem statement is by contradiction. Suppose there is an algorithm $\mathcal{A}$ running in time $\widetilde{T}(k) \cdot N^{o(k)}$ for some computable function $\widetilde{T} : \mathbb{N} \to \mathbb{N}$ that can distinguish between $\mathsf{MaxCover}(\Gamma) = 1$ and $\mathsf{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N, k \in \mathbb{N}$. From Lemma 4.15, we have that for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot N^{o(k)}$ time algorithm that can solve $\mathsf{PSP}(k, \mathcal{F}^{\text{MultEq}}, N)$ for every $N, k \in \mathbb{N}$. Next, by considering Theorem 5.2 for the case of (w, r, ℓ, s)-efficient protocols, we have that there are 2^w label cover instances $\{\Gamma^\mu\}_{\mu \in \{0,1\}^w}$, which can be constructed in $2^{o_k(m)}$ time. Note that $2^{o_k(m)} = O_k(N^{o(1)})$ by the choice of $m(N, k)$ in Definition 4.11. Thus, we can run $\mathcal{A}$ on each Γ^μ and solve $\mathsf{PSP}(k, \mathcal{F}^{\text{MultEq}}, N)$ for all $N, k \in \mathbb{N}$ in time $\widetilde{T}(k) \cdot N^{o(k)}$. This contradicts Lemma 4.15.

To prove the second part of the theorem statement, we apply the reduction described in Theorem 3.1 and note that $2^r = N^{o(1)}$ and $k^{2^{\ell k}} = 2^{(m(N,k))^{1/\beta} \cdot \log_2 k} = N^{o(1)}$. □

The proof of Theorem 1.4 follows by plugging in the parameters of the protocol described in Corollary 7.2 to the above corollary.

5.2.3 $\mathsf{W}[1] \neq \mathsf{FPT}$. In this subsubsection, we prove Theorem 1.3.

Corollary 5.5. *Let $\mathcal{F}^{\text{MultEq}}$ be the family of Boolean functions as defined in Definition 4.11. Suppose there exists a (w, r, ℓ, s)-efficient protocol for $\mathsf{MultEq}_{m,k,t}$ in the k-player SMP model for every $k, t, m \in \mathbb{N}$ such that $w + r + \ell k < m/tk$. Then, assuming $\mathsf{W}[1] \neq \mathsf{FPT}$, for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot \mathsf{poly}(N)$-time algorithm that can distinguish between $\mathsf{MaxCover}(\Gamma) = 1$ and $\mathsf{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N, k \in \mathbb{N}$. Moreover, if $r < m/2tk$ and $\ell < (\log m)/\beta \cdot k$ for some constant $\beta > 1$, then assuming $\mathsf{W}[1] \neq \mathsf{FPT}$, for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot \mathsf{poly}(N)$-time algorithm that can distinguish between $\mathsf{DomSet}(G) = k$ and $\mathsf{DomSet}(G) \geq (\frac{1}{s})^{\frac{1}{k}} \cdot k$ for any graph G with at most $O_k(N)$ vertices, for all $N, k \in \mathbb{N}$.*

Proof. The proof of the first part of the theorem statement is by contradiction. Suppose there is an algorithm $\mathcal{A}$ running in time $\widetilde{T}(k) \cdot \mathsf{poly}(N)$ for some computable function $\widetilde{T} : \mathbb{N} \to \mathbb{N}$ that can distinguish between $\mathsf{MaxCover}(\Gamma) = 1$ and $\mathsf{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N, k \in \mathbb{N}$. From Lemma 4.13, we have that for any computable function $T : \mathbb{N} \to \mathbb{N}$, there is no $T(k) \cdot \mathsf{poly}(N)$ time algorithm that can solve $\mathsf{PSP}(k, \mathcal{F}^{\text{MultEq}}, N)$ for every $N, k \in \mathbb{N}$. Next, by considering Theorem 5.2 for the case of (w, r, ℓ, s)-efficient protocols, we have that there are 2^w label cover instances $\{\Gamma^\mu\}_{\mu \in \{0,1\}^w}$, which can be constructed in $2^{m(N,k)/k \cdot t(k)} \cdot \mathsf{poly}(m(N, k))$ time. Note that $2^{m(N,k)/k \cdot t(k)} = O(N)$ and $\mathsf{poly}(m(N, k)) = N^{o(1)}$ by the choice of $m(N, k)$ in Definition 4.11. Thus, we can run $\mathcal{A}$ on each Γ^μ and solve $\mathsf{PSP}(k, \mathcal{F}^{\text{MultEq}}, N)$ for all $N, k \in \mathbb{N}$ in time less than $\widetilde{T}(k) \cdot \mathsf{poly}(N)$. This contradicts Lemma 4.13.

To prove the second part of the theorem statement, we apply the reduction described in Theorem 3.1 and note that $2^r = O(\sqrt{N})$ and $k^{2^{\ell k}} = 2^{(m(N,k))^{1/\beta} \cdot \log_2 k} = N^{o(1)}$. □

The proof of Theorem 1.3 follows by plugging in the parameters of the protocol described in Corollary 7.2 to the above corollary.

5.2.4 k-SUM Hypothesis. In this subsubsection, we prove Theorem 1.6.

Corollary 5.6. *Let $\mathcal{F}^{\text{SumZero}}$ be the family of Boolean functions as defined in Definition 4.3. Suppose there exists a (w, r, ℓ, s)-efficient protocol for $\text{SumZero}_{m,k}$ in the k-player SMP model for every $m, k \in \mathbb{N}$, such that $w + r + \ell k = o_k(m)$. Then assuming the k-SUM Hypothesis, for every integer $k \geq 3$ and every $\varepsilon > 0$, no $O(N^{\lceil k/2 \rceil - \varepsilon})$-time algorithm can distinguish between $\text{MaxCover}(\Gamma) = 1$ and $\text{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N \in \mathbb{N}$. Moreover, if $\ell < (\log m)/\beta \cdot k$ for some constant $\beta > 1$, then assuming the k-SUM Hypothesis, for every $\varepsilon > 0$ no $O(N^{\lceil k/2 \rceil - \varepsilon})$-time algorithm can distinguish between $\text{DomSet}(G) = k$ and $\text{DomSet}(G) \geq (\frac{1}{s})^{\frac{1}{k}} \cdot k$ for any graph G with at most $O_k(N)$ vertices for all $N \in \mathbb{N}$.*

Proof. The proof of the first part of the theorem statement is by contradiction. Suppose there is an algorithm $\mathcal{A}$ running in time $O(N^{\lceil k/2 \rceil - \varepsilon})$ for some fixed constant $\varepsilon > 0$ and some integer $k \geq 3$ that can distinguish between $\text{MaxCover}(\Gamma) = 1$ and $\text{MaxCover}(\Gamma) \leq s$ for any label cover instance $\Gamma(N, k, r, \ell)$ for all $N \in \mathbb{N}$. From Proposition 4.4, we have that no $O(N^{\lceil k/2 \rceil - \varepsilon/2})$-time algorithm can solve $\text{PSP}(k, \mathcal{F}^{\text{SumZero}}, N)$ for all $N \in \mathbb{N}$. Next, by considering Theorem 5.2 for the case of (w, r, ℓ, s)-efficient protocols, we have that there are 2^w label cover instances $\{\Gamma^\mu\}_{\mu \in \{0,1\}^w}$, which can be constructed in $2^{o_k(m)}$ time. Note that $2^{o_k(m)} = O_k(N^{o(1)})$ by the choice of $m(N, k)$ in Definition 4.3. Thus, we can run $\mathcal{A}$ on each Γ^μ and solve $\text{PSP}(k, \mathcal{F}^{\text{SumZero}}, N)$ for all $N \in \mathbb{N}$ in time $O(N^{\lceil k/2 \rceil - \varepsilon})$. This contradicts Proposition 4.4.

To prove the second part of the theorem statement, we apply the reduction described in Theorem 3.1 and note that $2^r = N^{o(1)}$ and $k^{2^{\ell k}} = 2^{(m(N,k))^{1/\beta} \cdot \log_2 k} = N^{o(1)}$. □

The proof of Theorem 1.6 follows by plugging in the parameters of the protocol described in Corollary 8.6 to the above corollary.

6 AN EFFICIENT PROTOCOL FOR SET DISJOINTNESS

Set Disjointness has been extensively studied primarily in the two-player setting (i.e., $k = 2$). In that setting, we know that the randomized communication complexity is $\Omega(m)$ [11, 47, 68], where m is the input size of each player. Surprisingly, Reference [1] showed that the MA-complexity of two-player set disjointness is $\tilde{O}(\sqrt{m})$. Their protocol was indeed an $(\tilde{O}(\sqrt{m}), O(\log m), \tilde{O}(\sqrt{m}), 1/2)$-efficient protocol for the case when $k = 2$. Recently, Reference [4] improved (in terms of the message size) the protocol to be an $(m/\log m, O(\log m), O((\log m)^3), 1/2)$-efficient protocol for the case when $k = 2$. Both these results can be extended naturally for all $k > 1$, to give an $(mk/\log m, O_k(\log m), O_k((\log m)^3), 1/2)$-efficient protocol. However, this does not suffice for proving Theorem 1.5, since we need a (w, r, ℓ, s)-efficient protocol with $w = o(m)$ and $\ell = o(\log m)$. Fortunately, Rubinstein [70] recently showed that the exact framework of the MA-protocols as in References [1, 4] but with the use of algebraic geometric codes instead of Reed Muller or Reed Solomon codes gives the desired parameters in the two-player case. Below we naturally extend Rubinstein's protocol to the k-player setting. This extension was suggested to us by Rubinstein [69].

Theorem 6.1. *There is a polynomial function $\hat{\ell} : \mathbb{N} \times \mathbb{N} \to [1, \infty)$ such that for every $k \in \mathbb{N}$ and every $\alpha \in \mathbb{N}$, there is a protocol for k-player $\text{Disj}_{m,k}$ in the SMP model, which is an $(m/\alpha, \log_2 m, \hat{\ell}(k, \alpha), 1/2)$-efficient protocol, where each player is given m bits as input, and the referee is given at most m/α bits of advice.*

Proof. Fix $k, m, \alpha \in \mathbb{N}$. Let q be the smallest prime greater than $\hat{q}(k)$ such that $q > (2\alpha\hat{r}(k))^2$, where the functions $\hat{r}$ and $\hat{q}$ are as defined in Theorem 3.9. Let $\mathcal{G} = \mathbb{F}_{q^2}$. Let $T = 2\alpha\hat{r}(k) \log_2 q$.

We associate $[m]$ with $[T] \times [m/T]$ and write the input $\mathbf{x}_j \in \{0,1\}^m$ as vectors $\mathbf{x}_j^1, \ldots, \mathbf{x}_j^T$, where $\mathbf{x}_j^t \in \{0,1\}^{m/T}$. For every $j \in [k]$, Player j computes $A_{m/T}(\mathbf{x}_j^t)$ for every $t \in [T]$. We denote the block length of $A_{m/T}$ by d. From the systematicity guaranteed by Theorem 3.9 we have that $A_{m/T}(\mathbf{x}_j^t)\,|_{[m/T]} = \mathbf{x}_j^t$. Also, notice that for all $t \in [T]$, we have $\wedge_{j\in[k]}\mathbf{x}_j^t = 0^{m/T}$ if and only if $\prod_{j\in[k]} A_{m/T}(\mathbf{x}_j^t) = 0^{m/T}$.

With the above observation in mind, we define the marginal sum $\Gamma \in \mathcal{G}^d$ as follows:

$$\forall i \in [d],\ \Gamma_i = \sum_{t\in[T]} \prod_{j\in[k]} A_{m/T}(\mathbf{x}_j^t)_i.$$

Again, notice that for all $t \in [T]$, we have $\wedge_{j\in[k]}\mathbf{x}_j^t = 0^{m/T}$ if and only if $\Gamma_i = 0$ for all $i \in [m/T]$. This follows from the following:

- For all $i \in [m/T]$, we have $A_{m/T}(\mathbf{x}_j^t)_i \in \{0,1\}$ and thus $\prod_{j\in[k]} A_{m/T}(\mathbf{x}_j^t)_i \in \{0,1\}$.
- The characteristic of $\mathcal{G}$ being greater than $(2\alpha\hat{r}(k)) \cdot \sqrt{q} \geq (2\alpha\hat{r}(k)) \cdot \log_2 q = T$.

More importantly, we remark that Γ is a codeword[17] in $B_{m/T}$. This follows because $B_{m/T}$ is a degree k closure code of $A_{m/T}$. To see this, in Definition 3.8, set $t = k, r = k \cdot T$, and $P[x_{1,1}, \ldots, x_{k,T}] = \sum_{i\in[T]} \prod_{j\in[k]} x_{i,j}$.

The Protocol

(1) Merlin sends the referee Φ, which is allegedly equal to the marginal sums codeword Γ defined above.
(2) All players jointly draw $r \in [d]$ uniformly at random.
(3) For every $j \in \{1, \ldots, k\}$, Player j sends to the referee $A_{m/T}(\mathbf{x}_j^t)_r$, $\forall t \in [T]$.
(4) The referee accepts if and only if both of the following hold:

$$\forall i \in [m/T], \quad \Phi_i = 0, \tag{1}$$

$$\Phi_r = \sum_{t\in[T]} \prod_{j\in[k]} A_{m/T}(\mathbf{x}_j^t)_r. \tag{2}$$

Analysis

Advice Length. To send the advice, Merlin only needs to send a codeword in $B_{m/T}$ to the verifier. This means that the advice length (in bits) is no more than $\log_2 q^2$ times the block length, which is $d \leq (m/T)\hat{r}(k) = m/2\alpha \log_2 q$, where the equality comes from our choice of T and the inequality from Theorem 3.9.

Message Length. Each player sends T elements of $\mathcal{G}$. Hence, the message length is $T \log_2 q^2 \leq \alpha\hat{r}(k)(2\log_2 q)^2$. Recall that q can be upper bounded by a polynomial in k and α. Hence, the message length is upper bounded entirely as a polynomial in k and α as desired.

Randomness. The number of coin tosses is $\log_2(d) \leq \log_2(m\hat{r}(k)/T) = \log_2(m/2\alpha \log_2 q) < \log_2 m$.

Completeness. If the k sets are disjoint, then Merlin can send the true Γ, and the verifier always accepts.

Soundness. If the k sets are not disjoint and Φ is actually Γ, then (1) is false and the verifier always rejects. However, if $\Phi \neq \Gamma$, then, since both are codewords of $B_{m/T}$, from Theorem 3.9 their relative

[17] We note that we use the multiplication property of Algebraic Geometric codes to find a non-trivial advice. On a related note, Meir [59] had previously shown that error-correcting codes with the multiplication property suffice to show the IP theorem (i.e., the IP=PSPACE result).

distance must be at least 1/2. As a result, with probability at least 1/2, $\Phi_r \neq \Gamma_r$. Since the right-hand side of Equation (2) is simply Γ_r, the verifier will reject for such r. Hence, the rejection probability is at least 1/2. □

The following corollary follows immediately by applying Proposition 5.1 with $z = {}^{(\log_2 m)}/_{2k\cdot\ell(k,\alpha)}$ to the above theorem.

Corollary 6.2. *There is a polynomial function $\hat{s} : \mathbb{N} \times \mathbb{N} \to [1, \infty)$ such that for every $k \in \mathbb{N}$ and every $\alpha \in \mathbb{N}$ there is a protocol for k-player $\mathsf{Disj}_{m,k}$ in the SMP model, which is an $({}^{m}/_{\alpha}, O((\log_2 m)^2), {}^{(\log_2 m)}/_{2k}, ({}^{1}/_{m})^{1/\hat{s}(k,\alpha)})$-efficient protocol, where each player is given m bits as input, and the referee is given at most ${}^{m}/_{\alpha}$ bits of advice.*

7 AN EFFICIENT PROTOCOL FOR MULTI-EQUALITY

Equality has been extensively studied, primarily in the two-player setting (i.e., $k = 2$). In that setting, when public randomness is allowed, we know that the randomized communication complexity is $O(1)$ [51, 79], and the protocols can be naturally extended to the k-player SMP model that yields a randomized communication complexity of $O(k)$. There are many protocols that achieve this complexity bound but for the purposes of proving Theorems 1.3 and 1.4, we will use the protocol where the players encode their input using a fixed good binary code and then send a jointly agreed random location of the encoded input to the referee who checks if all the messages he received are equal. Below we extend that protocol for Multi-Equality.

Theorem 7.1. *For some absolute constant $\delta > 0$, for every $t, k \in \mathbb{N}$ and every $m \in \mathbb{N}$ such that m is divisible by t, there is a $(0, \log m + O(1), 2t, 1 - \delta)$-efficient protocol for $\mathsf{MultEq}_{m,k,t}$ in the k-player SMP model.*

Proof. Let $C = \{C_m : \{0,1\}^m \to \{0,1\}^{d(m)}\}_{m\in\mathbb{N}}$ be the family of good codes with rate at least ρ and relative distance at least δ as guaranteed by Fact 3.6. Fix $m, k, t \in \mathbb{N}$ as in the theorem statement. Let $m' := {}^{m}/_{t}$.

The Protocol

(1) All players jointly draw $i^\star \in [d(m')]$ uniformly at random.
(2) If player j's input is $x_j = (x_{j,1}, y_{j,1}, \ldots, x_{j,t}, y_{j,t})$, then he sends $(C(x_{j,1})_{i^*}, y_{j,1}, \ldots, C(x_{j,t})_{i^*}, y_{j,t})$ to the referee.
(3) The referee accepts if and only if the following holds:

$$\textsc{MultEq}_{2t,k,t}((C(x_{1,1})_{i^*}, y_{1,1}, \ldots, C(x_{1,t})_{i^*}, y_{1,t}), \ldots, (C(x_{k,1})_{i^*}, y_{k,1}, \ldots, C(x_{k,t})_{i^*}, y_{k,t})) = 1.$$

Analysis

Parameters of the Protocol. In the first step of the protocol all the players jointly draw $i^\star \in [d(m')]$ uniformly, which requires $\lceil \log d(m') \rceil \leq \log m' + \log(1/\rho) \leq \log m + O(1)$ random bits. Then, for every $j \in [k]$, player j sends the referee $2t$ bits. Finally, since the code $C_{m'}$ is efficient, it is easy to see that the players and referee run in poly(m)-time.

Completeness. If $\textsc{MultEq}_{m,k,t}(x_1, \ldots, x_k) = 1$, then, for every $q \in [t]$ and $i, j \in [k]$, we have $(y_{i,q} = \perp) \vee (y_{j,q} = \perp) \vee (x_{i,q} = x_{j,q}) = 1$. This implies that, for every $q \in [t], i, j \in [k]$ and $i^* \in [d(m)]$, we have $(y_{i,q} = \perp) \vee (y_{j,q} = \perp) \vee (C(x_{i,q})_{i^*} = C(x_{j,q})_{i^*})$. In other words, $\textsc{MultEq}_{2t,k}((C(x_{1,1})_{i^*}, y_{1,1}, \ldots, C(x_{1,t})_{i^*}, y_{1,t}), \ldots, (C(x_{k,1})_{i^*}, y_{k,1}, \ldots, C(x_{k,t})_{i^*}, y_{k,t})) = 1$ for every $i^* \in [d(m)]$, meaning that the referee always accepts.

Soundness. Suppose that $\textsc{MultEq}_{m,k,t}(x_1, \ldots, x_k) = 0$. Then, there exists some $q \in [t]$ and $i, j \in [k]$ such that $y_{i,q} = \top, y_{j,q} = \top$, and $x_{i,q} \neq x_{j,q}$. Since the code C has relative distance

at least δ, $C(x_{i,q})$ and $C(x_{j,q})$ must differ on at least δ fraction of the coordinates. When the randomly selected i^* is such a coordinate, we have that $\textsc{MultEQ}_{2t,k}((C(x_{1,1})_{i^*}, y_{1,1}, \ldots, C(x_{1,t})_{i^*}, y_{1,t}), \ldots, (C(x_{k,1})_{i^*}, y_{k,1}, \ldots, C(x_{k,t})_{i^*}, y_{k,t})) = 0$, i.e., the referee rejects. Hence, the referee rejects with probability at least δ. □

The following corollary follows immediately by applying Proposition 5.1 to the above theorem with $z = \log_2 m/4kt$.

Corollary 7.2. *For every $t, k \in \mathbb{N}$ and every $m \in \mathbb{N}$ such that m is divisible by t, there is a $(0, O((\log m)^2), (\log_2 m)/2k, (1/m)^{1/O(kt)})$-efficient protocol for $\textsc{MultEQ}_{m,k,t}$ in the k-player SMP model.*

8 AN EFFICIENT PROTOCOL FOR SUM-ZERO

The Sum-Zero problem has been studied in the SMP model and efficient protocols with the following parameters have been obtained.

Theorem 8.1 ([61]). *For every $k \in \mathbb{N}$ and $m \in \mathbb{N}$ there is a $(0, O(\log(m + \log k)), O(\log(m + \log k)), 1/2)$-efficient protocol for $\textsc{SumZero}_{m,k}$ in the k-player SMP model.*

The above protocol is based on a simple (yet powerful) idea of picking a small random prime p and checking if the numbers sum to zero modulo p. Viola put forth a protocol with better parameters than the above protocol (i.e., smaller message length) using specialized hash functions and obtained the following:

Theorem 8.2 ([76]). *For every $k \in \mathbb{N}$ and $m \in \mathbb{N}$ there is a $(0, O(m), O(\log k), 1/2)$-efficient protocol for $\textsf{SumZero}_{m,k}$ in the k-player SMP model.*

To prove Theorem 1.6, we need a protocol with $o(m)$ randomness *and* $o(\log m)$ message length, and both the protocols described above do not meet these conditions. We show below that the above two results can be composed to get a protocol with $O(\log k)$ communication complexity and $O_k(\log m)$ randomness. In fact, we will use a slightly different protocol from Reference [76], namely, the protocol for the Sum-Zero($\mathbb{Z}_p$) problem as stated below.

*Definition 8.3 (*Sum-Zero($\mathbb{Z}_p$) *Problem).* Let $k, m, p \in \mathbb{N}$. $\mathbb{Z}_p$-$\textsc{SumZero}_{m,k} : (\{0,1\}^m)^k \to \{0,1\}$ is defined by

$$\mathbb{Z}_p\text{-}\textsc{SumZero}_{m,k}(x_1, \ldots, x_k) = \begin{cases} 1 \text{ if } \sum_{i \in [k]} x_i = 0 \mod p, \\ 0 \text{ otherwise,} \end{cases}$$

where we think of each x_i as a number in $[-2^{m-1}, 2^{m-1} - 1]$.

Theorem 8.4 ([76]). *For every $p, k \in \mathbb{N}$ and $m \in \mathbb{N}$, there is a $(0, O(\log p), O(\log k), 1/2)$-efficient protocol*[18] *for $\mathbb{Z}_p-\textsf{SumZero}_{m,k}$ in the k-player SMP model.*

Below is our theorem that essentially combines Theorem 8.1 and Theorem 8.4.

Theorem 8.5. *For every $k \in \mathbb{N}$ and $m \in \mathbb{N}$ there is a $(0, O(\log(m + \log k)), O(\log k), 3/4)$-efficient protocol for $\textsf{SumZero}_{m,k}$ in the k-player SMP model.*

Proof. Let π^* be the protocol of Viola from Theorem 8.4.

The Protocol

Let $t = 2(m - 1 + \log k)$. Let $p_1, \ldots, p_t$ be the first t primes.

[18] As written in Reference [76], the protocol has the following steps. First, the players send some messages to the referee. Then the players and the referee jointly draw some random coins, and, finally, the players send some more messages to the referee. However, we note that the first and second steps of the protocol can be swapped in Reference [76] to obtain an efficient protocol as the drawing of randomness do not depend on the messages sent by the players to the referee.

(1) All players jointly draw $i^\star \in [t]$ uniformly at random.
(2) The players and the referee run $\pi^\star$ where each player now has input $y_i = x_i \mod p_{i^\star}$, and the referee accepts if and only if $\sum_{i\in[k]} y_i = 0 \mod p_{i^\star}$.

Notice that the above protocol is still an efficient protocol as the first step of the above protocol can be combined with the draw of random coins in the first step of $\pi^\star$ to form a single step in which the players and the referee jointly draw random coins from the public randomness.

Analysis

Randomness. In the first step of the protocol all the players jointly draw $i^\star \in [t]$ uniformly, which requires $\lceil \log_2 t \rceil$ random bits. Then, the players draw $O(\log p_{i^\star})$ additional random coins for $\pi^\star$. The bound on the randomness follows by noting that $p_{i^\star} \leq p_t = O(t \log t)$.

Message Length. For every $j \in [k]$, Player j sends the referee $O(\log k)$ bits as per $\pi^\star$.

Completeness. If the k numbers sum to zero, then for any $p \in \mathbb{N}$, they sum to zero mod p and thus the referee always accepts.

Soundness. If the k numbers do not sum to zero, then let $\mathsf{s}(\pi^\star)$ be the soundness of $\pi^\star$ and let S be the subset of the first t primes defined as follows:

$$S = \left\{ p_i \middle| i \in [t], \sum_{j\in[k]} x_j = 0 \bmod p_i \right\}.$$

It is clear that the referee rejects with probability at least $(1 - |S|/t) \cdot (1 - \mathsf{s}(\pi^\star))$. Therefore, it suffices to show that $|S| < t/2$ as $\mathsf{s}(\pi^\star) \leq 1/2$. Let $x := \sum_{j\in[k]} x_j$. We have that $x \in [-k \cdot 2^{m-1}, k \cdot 2^{m-1}]$ and that, for every $p \in S$, p divides x. Since $x \neq 0$, we know that $|x| \geq \prod_{p\in S} p \geq \prod_{i=1}^{|S|} p_i \geq 2^{|S|}$. Since $|x| \leq k \cdot 2^{m-1}$, we have that $|S| < m - 1 + \log k = t$, and the proof follows. □

The following corollary follows immediately by applying Proposition 5.1 with $z = \log_2 m/2k \cdot c \log k$ to the above theorem, where c is some constant such that the message length of the protocol in Theorem 8.5 is at most $c \log k$.

Corollary 8.6. *For every $k, m \in \mathbb{N}$ there is a $(0, O((\log(m + \log k))^2), (\log_2 m)/2k, (1/m)^{1/O(k \log k)})$-efficient protocol for $\mathsf{SumZero}_{m,k}$ in the k-player SMP model.*

9 CONCLUSION AND OPEN QUESTIONS

We showed the parameterized inapproximability results for k-DomSet under W[1] ≠ FPT, ETH, SETH and k-SUM Hypothesis, which almost resolve the complexity status of approximating parameterized k-DomSet. Although we showed the W[1]-hardness of the problem, the exact version of k-DomSet is W[2]-complete. Thus, a remaining question is whether approximating k-DomSet is W[2]-hard:

Open Question 9.1. *Can we base total inapproximability of k-DomSet on W[2] ≠ FPT?*

We note that even 1.01-approximation of k-DomSet is not known to be W[2]-hard.

Another direction is to look beyond k-DomSet and try to prove inapproximability of other parameterized problems. Since k-DomSet is W[2]-complete, there are already known reductions from it to other W[2]-hard problems. Some of these reductions such as those for k-Set Cover, k-Hitting Set, k-DomSet on tournaments [31] and k-Contiguous Set [18] are gap-preserving reductions (in the sense of References [16, Definition 3.4]) and total inapproximability under W[1] ≠ FPT

translate directly to those problems. In this sense, one may wish to go beyond W[2]-hard problems and prove hardness of approximation for problems in W[1]. One of the most well-studied W[1]-complete problems is k-Clique, which we saw earlier in our proof; in its optimization variant, one wishes to find a clique of maximum size in the graph. Note that in this case a k-approximation is trivial, since we can just output one vertex. It was shown in Reference [16] that no $o(k)$-approximation is possible in FPT time assuming Gap-ETH. Since this rules out all non-trivial approximation algorithms, such a non-existential result is also referred to as the *total FPT-inapproximability* of k-Clique. Hence, the question is whether one can bypass Gap-ETH for this result as well:

Open Question 9.2. *Can we base total inapproximability of* k-Clique *on* W[1] $\neq$ FPT *or* ETH*?*

Again, we note that the current state-of-the-art results do not even rule out 1.01-FPT-approximation algorithms for k-Clique under W[1] $\neq$ FPT or ETH.

Another direction is to look beyond parameterized complexity questions. As mentioned earlier, Abboud et al. [4] used the hardness of approximating of PCP-Vectors as a starting point of their inapproximability results of problems in P. Since MaxCover is equivalent to PCP-Vectors when the number of right super-nodes is two, it may be possible that MaxCover for larger number of right super-nodes can also be used to prove hardness of problems in P as well. At the moment, however, we do not have any natural candidate in this direction (see Appendix B for further discussions).

APPENDICES

A A REDUCTION FROM MAXCOVER TO DOMSET

In this section, we provide the reduction from MaxCover to DomSet (i.e., Theorem 3.1). The reduction and its proof presented here are quoted almost directly from Chalermsook et al. [16] except with appropriate notational adjustments. We include it here only for the sake of self-containedness.

To ease the presentation, we will present the reduction in terms of a different variant of the label cover problem called MinLabel. The input to MinLabel is the same as that of MaxCover, i.e., it is a label cover instance $\Gamma = (U = \bigcup_{i\in[q]} U_i, W = \bigcup_{j\in[k]} W_j; E)$ as defined in Subsection 3.2.

However, the solution and the objective of the problem will be different from MaxCover. Specifically, a solution of MinLabel is called a *multi-labeling*, which is simply a subset of vertices $\widehat{S} \subseteq W$. We say that a multi-labeling $\widehat{S}$ *covers* a left super-node U_i if there exists a vertex $u_i \in U_i$ that has a neighbor in $\widehat{S} \cap W_j$ for every $j \in [k]$. The goal in MinLabel is to find a minimum-size multi-labeling that covers all the left super-nodes U_i's. Again, we overload the notation and use MinLabel to also denote the optimum, i.e.,

$$\mathsf{MinLabel}(\Gamma) = \min_{\substack{\text{multi-labeling } \widehat{S} \\ \text{that covers every } U_i}} |\widehat{S}|.$$

The following lemma demonstrates relations between MaxCover and MinLabel on the same input label cover instance. These relations in turns allow us to deduce inapproximability results of MinLabel from that of MaxCover.

Proposition A.1. *Let* Γ *be any label cover instance with* k *right super-nodes. Then, we have*

- *If* $\mathsf{MaxCover}(\Gamma) = 1$, *then* $\mathsf{MinLabel}(\Gamma) = k$.
- *If* $\mathsf{MaxCover}(\Gamma) \leq \varepsilon$ *for some* $0 < \varepsilon$, *then* $\mathsf{MinLabel}(\Gamma) \geq (1/\varepsilon)^{1/k} \cdot k$.

Before proceeding to prove Proposition A.1, let us state the desired reduction from label cover to DomSet (Theorem 3.1) in terms of MinLabel instead of MaxCover:

Lemma A.2 (Reduction from MinLabel to DomSet [16, Theorem 5.4])). *There is an algorithm that, given a label cover instance* $\Gamma = (U = \bigcup_{j=1}^{q} U_j, W = \bigcup_{i=1}^{k} W_i, E)$, *outputs a* k-DomSet *instance* G *such that*

- $\mathsf{MinLabel}(\Gamma) = \mathsf{DomSet}(G)$.
- $|V(G)| = |W| + \sum_{i\in[q]} k^{|U_q|}$.
- *The reduction runs in time* $O(|W|(\sum_{i\in[q]} k^{|U_q|}))$.

It is obvious that Proposition A.1 and Lemma A.2 together imply Theorem 3.1. The rest of this section is devoted to proving Proposition A.1 and Lemma A.2. Their proofs can be found in Subsections A.1 and A.2, respectively.

A.1 From MaxCover to MinLabel: Proof of Proposition A.1

In this section, we prove Proposition A.1. The proof presented here is due to [16] with slight changes to match our notations.

Proof of Proposition A.1.

(1) Suppose $\mathsf{MaxCover}(\Gamma) = 1$. Then there exists a labeling $S \subseteq W$ that covers every left super-node U_i. Hence, S is also a multi-labeling that covers every left super-node, which implies that $\mathsf{MinLabel}(\Gamma) = k$.

(2) We prove by contrapositive. Assume that $\mathsf{MinLabel}(\Gamma) < (1/\varepsilon)^{1/k}k$. Then there exists a multi-labeling $\widehat{S} \subseteq W$ of size less than $(1/\varepsilon)^{1/k}k$ that covers every left super-node. Observe that we can construct a feasible S' solution to MaxCover by uniformly and independently choosing one node from $\widehat{S} \cap W_j$ at random, for each of the right super-node W_j.

Thus, the expected number of left super-nodes covered by S' is

$$\begin{aligned}
\mathbb{E}_{S'}\left[|\{i \in [q] : S' \text{ covers } U_i\}|\right] &\geq \sum_{i\in[q]} \prod_{j\in[k]} |\widehat{S} \cap W_j|^{-1} \\
\text{(By AM-GM inequality)} &\geq \sum_{i\in[q]} \left(\frac{1}{k}\sum_{j\in[k]} |\widehat{S} \cap W_j|\right)^{-k} \\
&> q \cdot \left(\frac{1}{k} \cdot \left(\left(\frac{1}{\varepsilon}\right)^{1/k} k\right)\right)^{-k} \\
&= q \cdot \varepsilon
\end{aligned}$$

Hence, there is a labeling that covers $> \varepsilon q$ left super-nodes, i.e., $\mathsf{MaxCover}(\Gamma) > \varepsilon$. □

A.2 From MinLabel to DomSet: Proof of Lemma A.2

We devote this subsection to show a reduction from MinLabel to DomSet as stated in Lemma A.2.

Proof of Lemma A.2. We show the reduction from MinLabel to the bipartite version of DomSet, where the input graph $G = (A, B; E)$ is bipartite, and we are asked to find a subset of (left) vertices of A that dominates all the (right) vertices in B. This variant of DomSet is known as Red-Blue DomSet, which is equivalent to the *Set Cover* problem. One can transform the bipartite version of DomSet into the general case just by forming a clique on A. It is not hard to see that the optimum remains unchanged.

We apply the reduction from Reference [16], which is in turn taken from Reference [36] (and Reference [57]) with small adjustments. Our construction requires the *hypercube partition system*.

Hypercube Partition System. The (κ, ρ)-hypercube partition system consists of the universe $\mathcal{M}$ and a collection of subset $\{P_{x,y}\}_{x\in[\rho],y\in[\kappa]}$, where

$$\mathcal{M} = [\kappa]^\rho \quad \text{and} \quad P_{x,y} = \{z \in \mathcal{M} : z_x = y\}.$$

The following are properties that can be observed from the hypercube partition system:

- **Partition Cover:** For each row $x \in [\rho]$, $\mathcal{P}_x = (P_{x,1}, \ldots, P_{x,\kappa})$ is a partition of $\mathcal{M}$. Thus, $\bigcup_{y\in[\kappa]} P_{x,y} = \mathcal{M}$ and $P_{x,y} \cap P_{x,y'} = \emptyset$ for $y \neq y'$.
  That is, one can cover the universe $\mathcal{M}$ by picking all κ subsets from some row $x \in [\kappa]$.
- **Robust Property:** For any $y_1^*, \ldots, y_\rho^* \in [\kappa]$, $\bigcup_{x\in[\rho],y\in[\kappa]\setminus\{y_x^*\}} P_{x,y} \neq \mathcal{M}$.
  That is, the only way to cover the universe $\mathcal{M}$ is to pick all the κ subsets from the same row. Otherwise, even if we include $\kappa - 1$ subsets from every row $x \in [\rho]$, it is not possible to cover the universe $\mathcal{M}$.
- **Size:** The number of elements in the universe is κ^ρ.

The Reduction. Now we present our reduction. Let $\Gamma = (U = \bigcup_{j=1}^{q} U_j, W = \bigcup_{i=1}^{k} W_i; E')$ be an instance of MinLabel. We will construct a bipartite graph $G = (A, B; E)$ of Red-Blue DomSet from Γ. The vertices of A are taken from the right vertices of Γ, i.e., $A = W$.

For each super-node U_i, we create the $(k, |U_i|)$-hypercube partition system $(\mathcal{M}^i, \{P^i_{x,y}\}_{x\in[|U_i|],y\in[k]})$ (i.e., $\rho = |U_i|$ and $\kappa = k$); then we take the elements of $\mathcal{M}^i$ as left vertices of G.

We name vertices in U_i by $u^i_1, u^i_2, \ldots, u^i_{|U_i|}$ for $i \in [q]$. For each $j \in [k]$ and each $w^j \in W_j$, we join w^j to every vertex $z \in P^i_{x,j}$ if $\{u^i_x, w^j\}$ is an edge in Γ. More precisely, the bipartite graph $G = (A, B; E)$ is defined such that

$$A := W = \bigcup_{j}^{k} W_j,$$

$$B := \bigcup_{i=1}^{q} \mathcal{M}^i,$$

$$E := \{\{w^j, z\} : i \in [q], j \in [k], x \in [|U_i|], w^j \in W_j, z \in P^i_{x,j}, \{u^i_x, w^j\} \in E'\}.$$

Size of the Construction. It is clear that the number of left and right vertices in our construction are as follows.

$$\text{The Number of Left veritces of } G = |W|$$

$$\text{The Number of Right vertices of } G = \sum_{i\in[q]} k^{|U_i|}$$

Analysis. Clearly, the size of G satisfies the lemma statement. For the claim MinLabel(Γ) = DomSet(G), observe that any solution S to the MinLabel instance corresponds to a solution to the Red-Blue DomSet instance. We claim that S is feasible to the MinLabel instance if and only if it is feasible to the DomSet instance. This will prove the claim and thus imply the lemma.

First, suppose S is feasible to MinLabel. Then, for every U_i, there is a vertex $u^i_x \in U_i$ that has an edge to some vertex $w^j_y \in W_j \cap S$, for all $j \in [k]$. This means that we have an edge from w^j_y to every vertex $z \in P^i_{x,y}$. It then follows by the Partition Cover property of the hypercube partition system that S dominates every vertex in the universe $\mathcal{M}^i$. Consequently, we conclude that S dominates $B = \bigcup_{i\in[q]} \mathcal{M}^i$.

Conversely, suppose S is feasible to DomSet. Then S has an edge to every vertex in the universe $\mathcal{M}^i \subseteq B$. We know by construction and the Robustness Property of the hypercube partition system

that, for some $x \in [|U_i|]$, S contains vertices $w^1 \in S \cap W_1, w^2 \in S \cap W_2, \ldots, w^k \in S \cap W_k$, which correspond to the subsets $P^i_{x,y}$'s in the partition system, and that $w^1, w^2, \ldots, w^k$ are incident to the vertex $u^i_x \in U_i$. This means that S covers the left super-node U_i. Consequently, we conclude that S is a feasible multi-labeling, thus completing the proof. □

B CONNECTION TO FINE-GRAINED COMPLEXITY

In this section, we will demonstrate the conditional hardness of problems in P by basing them on the conditional hardness of PSP. First, we define the problem in P of interest to this section.

Definition B.1 (k-linear form inner product). Let $k, m \in \mathbb{N}$. Given $x_1, \ldots, x_k \in \mathbb{R}^m$, we define the inner product of these k vectors as follows:

$$\langle x_1, \ldots, x_k \rangle = \sum_{i \in [m]} \prod_{j \in [k]} x_i(j),$$

where $x_i(j)$ denotes the jth coordinate of the vector x_i.

Definition B.2 (k-chromatic Maximum Inner Product). Let $k \in \mathbb{N}$. Given k collections $A_1, A_2, \ldots, A_k$ each of N vectors in $\{0,1\}^D$, where $D = N^{o(1)}$, and an integer s, the k-chromatic Maximum Inner Product (MIP) problem is to determine if there exists $a_i \in A_i$ for all $i \in [k]$ such that $\langle a_1, \ldots, a_k \rangle \geq s$.

We continue to use the shorthand $\Gamma(N, k, r, \ell)$ introduced in Section 5.2. Moreover, while using the shorthand $\Gamma(N, k, r, \ell)$, we will assume that for all $i \in [h]$, $|W_i| \leq N$. We define Unique MaxCover to be the MaxCover problem with the following additional structure: For every labeling $S \subseteq W$ (see Section 3.2 to recall the definition of a labeling) and any left super-node U_i, there is at most one node in U_i, which is a neighbor to all the nodes in S. We remark that the reduction in Theorem 5.2 already produces instances of Unique MaxCover. This follows from the proof of Theorem 5.2 by noting that on every random string, each player sends a message to the referee in a deterministic way. Finally, we have the following connection between unique MaxCover and MIP.

Theorem B.3. *Let $N, k, r, \ell \in \mathbb{N}$. There is a reduction from any Unique MaxCover instance $\Gamma(N, k, r, \ell)$ to k-chromatic MIP instance $(A_1, \ldots, A_k, s)$ such that*

- *For all $i \in [k]$, $|A_i| \leq N$, $s = 2^r$, and $D \leq 2^{r+\ell k}$.*
- *The running time of the reduction is $O(Nk \cdot 2^{r+\ell k})$.*
- *For any integer s^*, there exists $a_i \in A_i$ for all $i \in [k]$ such that $\langle a_1, \ldots, a_k \rangle \geq s^*$ if and only if* $\mathsf{MaxCover}(\Gamma) \geq s^*/2^r$.

Proof. For every $i \in [k]$, we associate each A_i with the right super-node W_i. Each node in W_i corresponds to a vector in A_i. Given a node w in W_i, the corresponding vector $a \in A_i$ is constructed as follows. Each coordinate of a corresponds to a left node. Let the jth coordinate of a correspond to a node $u \in U$. The jth coordinate of a is 1 if there is an edge between u and w in the graph G of the instance Γ; otherwise, the jth coordinate of a is 0. It is easy to see that $|A_i| \leq N$ and $D \leq 2^{r+\ell k}$, and the running time of the reduction is $O(Nk \cdot 2^{r+\ell k})$. It is also easy to see that if $\mathsf{MaxCover}(\Gamma) \geq s^*/2^r$ then the vectors a_i corresponding to the nodes in the labeling resulting in the maximum cover, have inner product at least s^*.

We will now show that for any integer s^*, if there exists $a_i \in A_i$ for all $i \in [k]$ such that $\langle a_1, \ldots, a_k \rangle \geq s^*$ then $\mathsf{MaxCover}(\Gamma) \geq s^*/2^r$. Fix an integer s^*. Suppose that there exists $a_i \in A_i$ for all $i \in [k]$ such that $\langle a_1, \ldots, a_k \rangle \geq s^*$. The k-tuple $(a_1, \ldots, a_k)$ corresponds to a labeling S of unique MaxCover. For each coordinate such that all the $a_i = 1$ on that coordinate, we note that

there is a left node (corresponding to the coordinate), which is a common neighbor of all the nodes in the labeling. Moreover, since in the unique MaxCover we can have at most one node in each left super-node, which is a common neighbor of the nodes in the labeling, we have that:

$$\text{MaxCover}(\Gamma) \geq \frac{1}{2^r} \cdot |\{i \in [2^r] \mid U_i \text{ is covered by } S\}| = \langle a_1, \ldots, a_k \rangle \geq s^*. \qquad \square$$

The results for hardness of approximation for problems in P is obtained by simply fixing the value of k in the above theorem to some universal constant (such as $k = 2$). For example, by fixing $k = 2$ and applying the above reduction to Corollary 5.3, we recover the main result of Reference [4] on bichromatic MIP. This is not surprising as under SETH, our framework is just a generalization of the distributed PCP framework of Reference [4].

Furthermore, we demonstrate the flexibility of our framework by fixing $k = 3$ and applying the above reduction to Corollary 5.6, to establish a hardness of approximation result of trichromatic MIP under the 3-SUM Hypothesis as stated below. We note here that the running time lower bound is only $N^{2-o(1)}$, which is likely not tight since the lower bound from SETH is $N^{3-o(1)}$.

Theorem B.4. *Assuming the 3-SUM Hypothesis, for every $\varepsilon > 0$, no $O(N^{2-\varepsilon})$ time algorithm can, given three collections A, B, and C each of N vectors in $\{0,1\}^D$, where $D = N^{o(1)}$, and an integer s, distinguish between the following two cases:*

Completeness. *There exists $a \in A, b \in B, c \in C$ such that $\langle a, b, c \rangle \geq s$.*
Soundness. *For every $a \in A, b \in B, c \in C$, $\langle a, b, c \rangle \leq s/2^{(\log N)^{1-o(1)}}$.*

Proof. We apply Proposition 5.1 with $z = m/(\log_2 m)^2$ and $k = 3$ to Theorem 8.5, to obtain a $(0, O(m/\log m), O(m/(\log m)^2), (1/2)^{m^{1-o(1)}})$-efficient protocol for $\text{SumZero}_{m,3}$ in the three-player SMP model. By plugging in the parameters of the above protocol to Corollary 5.6, we obtain that assuming the 3-SUM hypothesis, for every $\varepsilon > 0$, no $O(N^{2-\varepsilon})$-time algorithm can distinguish between $\text{MaxCover}(\Gamma) = 1$ and $\text{MaxCover}(\Gamma) \leq (1/2)^{(\log N)^{1-o(1)}}$ for any label cover instance $\Gamma(N, 3, r, \ell)$ for all $N \in \mathbb{N}$. The proof of the theorem concludes by applying Theorem B.3 to the above hardness of MaxCover (Note that Theorem 5.2 provides a reduction from $\text{PSP}(k, \mathcal{F}, N)$ to Unique MaxCover). $\square$

ACKNOWLEDGMENTS

We express our sincere gratitude and appreciation to Aviad Rubinstein for suggesting the use of algebraic geometric codes, which resolved our issues in constructing the protocol for Set Disjointness with our desired parameters. We thank Swastik Kopparty, Henning Stichtenoth, Gil Cohen, Alessandro Chiesa, and Nicholas Spooner for very helpful discussions on Algebraic Geometric Codes, Emanuele Viola for insightful comments on his protocol from Reference [76], and Guy Rothblum for discussions on locally characterizable sets. We also thank Irit Dinur, Prahladh Harsha, and Yijia Chen for general discussions and suggestions.

REFERENCES

[1] Scott Aaronson and Avi Wigderson. 2009. Algebrization: A new barrier in complexity theory. *Trans. Comput. Theory* 1, 1 (2009), 2:1–2:54. DOI: https://doi.org/10.1145/1490270.1490272

[2] Amir Abboud and Kevin Lewi. 2013. Exact weight subgraphs and the k-sum conjecture. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP'13)*. 1–12. DOI: https://doi.org/10.1007/978-3-642-39206-1_1

[3] Amir Abboud, Kevin Lewi, and Ryan Williams. 2014. Losing weight by gaining edges. In *Proceedings of the Annual Energy Storage Conference and Expo (ESA'14)*. 1–12. DOI: https://doi.org/10.1007/978-3-662-44777-2_1

[4] Amir Abboud, Aviad Rubinstein, and Ryan Williams. 2017. Distributed PCP theorems for hardness of approximation in P. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS'17)*. 25–36.

[5] Noga Alon, Ankur Moitra, and Benny Sudakov. 2012. Nearly complete graphs decomposable into large induced matchings and their applications. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'12)*. 1079–1090. DOI: https://doi.org/10.1145/2213977.2214074

[6] Noga Alon, Dana Moshkovitz, and Shmuel Safra. 2006. Algorithmic construction of sets for k-restrictions. *ACM Trans. Algor.* 2, 2 (2006), 153–177. DOI: https://doi.org/10.1145/1150334.1150336

[7] Chrisil Arackaparambil, Joshua Brody, and Amit Chakrabarti. 2009. Functional monitoring without monotonicity. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP'09)*. 95–106. DOI: https://doi.org/10.1007/978-3-642-02927-1_10

[8] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. 1998. Proof verification and the hardness of approximation problems. *J. ACM* 45, 3 (1998), 501–555. DOI: https://doi.org/10.1145/278298.278306

[9] Sanjeev Arora and Shmuel Safra. 1998. Probabilistic checking of proofs: A new characterization of NP. *J. ACM* 45, 1 (1998), 70–122. DOI: https://doi.org/10.1145/273865.273901

[10] László Babai, Anna Gál, Peter G. Kimmel, and Satyanarayana V. Lokam. 2003. Communication complexity of simultaneous messages. *SIAM J. Comput.* 33, 1 (2003), 137–166. DOI: https://doi.org/10.1137/S0097539700375944

[11] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. 2004. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.* 68, 4 (2004), 702–732. DOI: https://doi.org/10.1016/j.jcss.2003.11.006

[12] Eli Ben-Sasson, Alessandro Chiesa, Ariel Gabizon, Michael Riabzev, and Nicholas Spooner. 2016. Short interactive oracle proofs with constant query complexity, via composition and sumcheck. *Electr. Coll. Comput. Complex.* 23 (2016), 46:1–46:38.

[13] Eli Ben-Sasson, Yohay Kaplan, Swastik Kopparty, Or Meir, and Henning Stichtenoth. 2016. Constant rate PCPs for circuit-SAT with sublinear query complexity. *J. ACM* 63, 4 (2016), 32:1–32:57. DOI: https://doi.org/10.1145/2901294

[14] Edouard Bonnet, Bruno Escoffier, Eun Jung Kim, and Vangelis Th. Paschos. 2015. On subexponential and FPT-time inapproximability. *Algorithmica* 71, 3 (2015), 541–565. DOI: https://doi.org/10.1007/s00453-014-9889-1

[15] Liming Cai and Xiuzhen Huang. 2010. Fixed-parameter approximation: Conceptual framework and approximability results. *Algorithmica* 57, 2 (2010), 398–412. DOI: https://doi.org/10.1007/s00453-008-9223-x

[16] Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. 2017. From Gap-ETH to FPT-inapproximability: Clique, dominating set, and more. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS'17)*. 743–754.

[17] Ashok K. Chandra, Merrick L. Furst, and Richard J. Lipton. 1983. Multi-party protocols. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'83)*. 94–99. DOI: https://doi.org/10.1145/800061.808737

[18] Moses Charikar, Yonatan Naamad, and Anthony Wirth. 2016. On approximating target set selection. In *International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'16)*. 4:1–4:16. DOI: https://doi.org/10.4230/LIPIcs.APPROX-RANDOM.2016.4

[19] Arkadev Chattopadhyay, Jaikumar Radhakrishnan, and Atri Rudra. 2014. Topology matters in communication. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS'14)*. 631–640. DOI: https://doi.org/10.1109/FOCS.2014.73

[20] Jianer Chen, Xiuzhen Huang, Iyad A. Kanj, and Ge Xia. 2006. Strong computational lower bounds via parameterized complexity. *J. Comput. Syst. Sci.* 72, 8 (2006), 1346–1367. DOI: https://doi.org/10.1016/j.jcss.2006.04.007

[21] Yijia Chen, Martin Grohe, and Magdalena Grüber. 2006. On parameterized approximability. In *International Workshop on Parameterized and Exact Computation (IWPEC'06)*. 109–120. DOI: https://doi.org/10.1007/11847250_10

[22] Yijia Chen and Bingkai Lin. 2016. The constant inapproximability of the parameterized dominating set problem. In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS'16)*. 505–514. DOI: https://doi.org/10.1109/FOCS.2016.61

[23] Rajesh Hemant Chitnis, MohammadTaghi Hajiaghayi, and Guy Kortsarz. 2013. Fixed-parameter and approximation algorithms: A new look. In *International Symposium on Parameterized and Exact Computation (IPEC'13)*. 110–122. DOI: https://doi.org/10.1007/978-3-319-03898-8_11

[24] Vasek Chvátal. 1979. A greedy heuristic for the set-covering problem. *Math. Oper. Res.* 4, 3 (1979), 233–235. DOI: https://doi.org/10.1287/moor.4.3.233

[25] Graham Cormode, S. Muthukrishnan, and Ke Yi. 2008. Algorithms for distributed functional monitoring. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA'08)*. 1076–1085.

[26] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms*. Springer. DOI: https://doi.org/10.1007/978-3-319-21275-3

[27] Irit Dinur. 2007. The PCP theorem by gap amplification. *J. ACM* 54, 3 (2007), 12. DOI: https://doi.org/10.1145/1236457.1236459

[28] Irit Dinur. 2016. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. *Electr. Coll. Comput. Complex.* 23 (2016), 128:1–128:15.

[29] Irit Dinur and David Steurer. 2014. Analytical approach to parallel repetition. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'14)*. 624–633. DOI: https://doi.org/10.1145/2591796.2591884

[30] Rodney G. Downey and Michael R. Fellows. 1995. Fixed-parameter tractability and completeness II: On completeness for W[1]. *Theor. Comput. Sci.* 141, 1&2 (1995), 109–131. DOI: https://doi.org/10.1016/0304-3975(94)00097-3

[31] Rodney G. Downey and Michael R. Fellows. 1995. *Parameterized Computational Feasibility*. Birkhäuser Boston, Boston, MA, 219–244. DOI: https://doi.org/10.1007/978-1-4612-2566-9_7

[32] Rodney G. Downey and Michael R. Fellows. 2013. *Fundamentals of Parameterized Complexity*. Springer. DOI: https://doi.org/10.1007/978-1-4471-5559-1

[33] Rodney G. Downey, Michael R. Fellows, and Catherine McCartin. 2006. Parameterized approximation problems. In *International Workshop on Parameterized and Exact Computation (IWPEC'06)*. 121–129. DOI: https://doi.org/10.1007/11847250_11

[34] Rodney G. Downey, Michael R. Fellows, Catherine McCartin, and Frances A. Rosamond. 2008. Parameterized approximation of dominating set problems. *Inf. Process. Lett.* 109, 1 (2008), 68–70. DOI: https://doi.org/10.1016/j.ipl.2008.09.017

[35] Friedrich Eisenbrand and Fabrizio Grandoni. 2004. On the complexity of fixed parameter clique and dominating set. *Theor. Comput. Sci.* 326, 1–3 (2004), 57–67. DOI: https://doi.org/10.1016/j.tcs.2004.05.009

[36] Uriel Feige. 1998. A threshold of ln *n* for approximating set cover. *J. ACM* 45, 4 (1998), 634–652. DOI: https://doi.org/10.1145/285055.285059

[37] Orr Fischer, Rotem Oshman, and Uri Zwick. 2016. Public vs. private randomness in simultaneous multi-party communication complexity. In *Proceedings of the International Colloquium on Structural Information and Communication Complexity (SIROCCO'16)*. 60–74. DOI: https://doi.org/10.1007/978-3-319-48314-6_5

[38] Anka Gajentaan and Mark H. Overmars. 1995. On a class of O(n2) problems in computational geometry. *Comput. Geom.* 5, 3 (1995), 165–185. DOI: https://doi.org/10.1016/0925-7721(95)00022-2

[39] Arnaldo Garcia and Henning Stichtenoth. 1996. On the asymptotic behaviour of some towers of function fields over finite fields. *J. Number Theory* 61, 2 (1996), 248–273. DOI: https://doi.org/10.1006/jnth.1996.0147

[40] Oded Goldreich. 2008. *Computational Complexity: A Conceptual Perspective* (1st ed.). Cambridge University Press, New York, NY.

[41] Oded Goldreich and Guy N. Rothblum. 2018. Simple doubly-efficient interactive proof systems for locally-characterizable sets. In *Proceedings of the 9th Innovations in Theoretical Computer Science Conference (ITCS'18)*. 18:1–18:19. DOI: https://doi.org/10.4230/LIPIcs.ITCS.2018.18

[42] Mohammad Taghi Hajiaghayi, Rohit Khandekar, and Guy Kortsarz. 2013. The foundations of fixed parameter inapproximability. *CoRR* abs/1310.2711 (2013). http://arxiv.org/abs/1310.2711

[43] Russell Impagliazzo and Ramamohan Paturi. 2001. On the complexity of k-SAT. *J. Comput. Syst. Sci.* 62, 2 (2001), 367–375. DOI: https://doi.org/10.1006/jcss.2000.1727

[44] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. 2001. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.* 63, 4 (2001), 512–530. DOI: https://doi.org/10.1006/jcss.2001.1774

[45] David S. Johnson. 1974. Approximation algorithms for combinatorial problems. *J. Comput. Syst. Sci.* 9, 3 (1974), 256–278. DOI: https://doi.org/10.1016/S0022-0000(74)80044-9

[46] Jørn Justesen. 1972. Class of constructive asymptotically good algebraic codes. *IEEE Trans. Inf. Theory* 18, 5 (1972), 652–656. DOI: https://doi.org/10.1109/TIT.1972.1054893

[47] Bala Kalyanasundaram and Georg Schnitger. 1992. The probabilistic communication complexity of set intersection. *SIAM J. Discr. Math.* 5, 4 (1992), 545–557. DOI: https://doi.org/10.1137/0405044

[48] Viggo Kann. 1992. *On the Approximability of NP-complete Optimization Problems*. Ph.D. Dissertation. Royal Institute of Technology.

[49] Richard M. Karp. 1972. Reducibility among combinatorial problems. In *Proceedings of the Symposium on the Complexity of Computer Computations*. 85–103. http://www.cs.berkeley.edu/~luca/cs172/karp.pdf.

[50] Swastik Kopparty. 2017. *Personal communication*.

[51] Eyal Kushilevitz and Noam Nisan. 1997. *Communication Complexity*. Cambridge University Press, New York, NY.

[52] Troy Lee and Adi Shraibman. 2009. Lower bounds in communication complexity. *Found. Trends Theoret. Comput. Sci.* 3, 4 (2009), 263–398. DOI: https://doi.org/10.1561/0400000040

[53] Guanfeng Liang and Nitin H. Vaidya. 2011. Multiparty equality function computation in networks with point-to-point links. In *Proceedings of the International Colloquium on Structural Information and Communication Complexity (SIROCCO'11)*. 258–269. DOI: https://doi.org/10.1007/978-3-642-22212-2_23

[54] Bingkai Lin. 2015. The parameterized complexity of *k*-biclique. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA'15)*. 605–615. DOI: https://doi.org/10.1137/1.9781611973730.41

[55] Bingkai Lin. 2019. A simple gap-producing reduction for the parameterized set cover problem. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP'19)*, Vol. 132. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 81:1–81:15. http://drops.dagstuhl.de/opus/frontdoor.php?source_opus=10657.

[56] L. Lovász. 1975. On the ratio of optimal integral and fractional covers. *Discr. Math.* 13, 4 (1975), 383–390. DOI: https://doi.org/10.1016/0012-365X(75)90058-8

[57] Carsten Lund and Mihalis Yannakakis. 1994. On the hardness of approximating minimization problems. *J. ACM* 41, 5 (1994), 960–981. DOI: https://doi.org/10.1145/185675.306789

[58] Pasin Manurangsi and Prasad Raghavendra. 2016. A birthday repetition theorem and complexity of approximating dense CSPs. *CoRR* abs/1607.02986 (2016). http://arxiv.org/abs/1607.02986

[59] Or Meir. 2013. IP = PSPACE using error-correcting codes. *SIAM J. Comput.* 42, 1 (2013), 380–403. DOI: https://doi.org/10.1137/110829660

[60] Dana Moshkovitz. 2015. The projection games conjecture and the NP-hardness of ln n-approximating set-cover. *Theory Comput.* 11, 7 (2015), 221–235. DOI: https://doi.org/10.4086/toc.2015.v011a007

[61] Noam Nisan. 1994. The communication complexity of threshold gates. In *Proceedings of Combinatorics, Paul Erdos Is Eighty*. 301–315.

[62] Christos H. Papadimitriou and Mihalis Yannakakis. 1991. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.* 43, 3 (1991), 425–440. DOI: https://doi.org/10.1016/0022-0000(91)90023-X

[63] Mihai Patrascu. 2010. Towards polynomial lower bounds for dynamic problems. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC'10)*. 603–610. DOI: https://doi.org/10.1145/1806689.1806772

[64] Mihai Patrascu and Ryan Williams. 2010. On the possibility of faster SAT algorithms. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA'10)*. 1065–1075. DOI: https://doi.org/10.1137/1.9781611973075.86

[65] Jeff M. Phillips, Elad Verbin, and Qin Zhang. 2012. Lower bounds for number-in-hand multiparty communication complexity, made easy. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. 486–501. http://portal.acm.org/citation.cfm?id=2095158&CFID=63838676&CFTOKEN=79617016.

[66] Ran Raz. 1998. A parallel repetition theorem. *SIAM J. Comput.* 27, 3 (1998), 763–803. DOI: https://doi.org/10.1137/S0097539795280895

[67] Ran Raz and Shmuel Safra. 1997. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'97)*. 475–484. DOI: https://doi.org/10.1145/258533.258641

[68] Alexander A. Razborov. 1992. On the distributional complexity of disjointness. *Theor. Comput. Sci.* 106, 2 (1992), 385–390. DOI: https://doi.org/10.1016/0304-3975(92)90260-M

[69] Aviad Rubinstein. 2017. *Personal communication.*

[70] Aviad Rubinstein. 2018. Hardness of approximate nearest neighbor search. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC'18)*. 1260–1268. DOI: https://doi.org/10.1145/3188745.3188916

[71] Kenneth W. Shum, Ilia Aleshnikov, P. Vijay Kumar, Henning Stichtenoth, and Vinay Deolalikar. 2001. A low-complexity algorithm for the construction of algebraic-geometric codes better than the Gilbert-Varshamov bound. *IEEE Trans. Inf. Theory* 47, 6 (2001), 2225–2241. DOI: https://doi.org/10.1109/18.945244

[72] Michael Sipser and Daniel A. Spielman. 1996. Expander codes. *IEEE Trans. Inf. Theory* 42, 6 (1996), 1710–1722. DOI: https://doi.org/10.1109/18.556667

[73] Petr Slavík. 1996. A tight analysis of the greedy algorithm for set cover. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'96)*. 435–441. DOI: https://doi.org/10.1145/237814.237991

[74] Aravind Srinivasan. 1995. Improved approximations of packing and covering problems. In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'95)*. 268–276. DOI: https://doi.org/10.1145/225058.225138

[75] Craig A. Tovey. 1984. A simplified NP-complete satisfiability problem. *Discr. Appl. Math.* 8, 1 (1984), 85–89. DOI: https://doi.org/10.1016/0166-218X(84)90081-7

[76] Emanuele Viola. 2015. The communication complexity of addition. *Combinatorica* 35, 6 (2015), 703–747. DOI: https://doi.org/10.1007/s00493-014-3078-3

[77] Omri Weinstein and David P. Woodruff. 2015. The simultaneous communication of disjointness with applications to data streams. In *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP'15)*. 1082–1093. DOI: https://doi.org/10.1007/978-3-662-47672-7_88

[78] Ryan Williams. 2005. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.* 348, 2–3 (2005), 357–365. DOI: https://doi.org/10.1016/j.tcs.2005.09.023

[79] Andrew Chi-Chih Yao. 1979. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the Annual ACM SIGACT Symposium on Theory of Computing (STOC'79)*. 209–213. DOI: https://doi.org/10.1145/800135.804414

Received April 2018; accepted April 2019