On Multidimensional and Monotone k-SUM

Chloe Ching-Yun Hsu[1] **and Chris Umans** *[2]

1 **Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, USA**
chhsu@caltech.edu

2 **Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, USA**
umans@cms.caltech.edu

Abstract

The well-known k-SUM conjecture is that integer k-SUM requires time $\Omega(n^{\lceil k/2\rceil - o(1)})$. Recent work has studied multidimensional k-SUM in $\mathbb{F}_p^d$, where the best known algorithm takes time $\tilde{O}(n^{\lceil k/2\rceil})$. Bhattacharyya et al. [ICS 2011] proved a $\min(2^{\Omega(d)}, n^{\Omega(k)})$ lower bound for k-SUM in $\mathbb{F}_p^d$ under the Exponential Time Hypothesis. We give a more refined lower bound under the standard k-SUM conjecture: for sufficiently large p, k-SUM in $\mathbb{F}_p^d$ requires time $\Omega(n^{k/2-o(1)})$ if k is even, and $\Omega(n^{\lceil k/2\rceil - 2k\frac{\log k}{\log p} - o(1)})$ if k is odd.

For a special case of the multidimensional problem, *bounded monotone d-dimensional 3SUM*, Chan and Lewenstein [STOC 2015] gave a surprising $\tilde{O}(n^{2-2/(d+13)})$ algorithm using additive combinatorics. We show this algorithm is essentially optimal. To be more precise, bounded monotone d-dimensional 3SUM requires time $\Omega(n^{2-\frac{4}{d}-o(1)})$ under the standard 3SUM conjecture, and time $\Omega(n^{2-\frac{2}{d}-o(1)})$ under the so-called strong 3SUM conjecture. Thus, even though one might hope to further exploit the structural advantage of monotonicity, no substantial improvements beyond those obtained by Chan and Lewenstein are possible for bounded monotone d-dimensional 3SUM.

1998 ACM Subject Classification F.2.1 Numerical Algorithms and Problems

Keywords and phrases 3SUM; kSUM; monotone 3SUM; strong 3SUM conjecture

Digital Object Identifier 10.4230/LIPIcs.MFCS.2017.50

1 Introduction

The k-SUM problem and the k-SUM conjecture are related to a large number of problems in computational geometry [11], dynamic data structures, and graph theory. For example, Pătraşcu [17] showed lower bounds for dynamic problems under the 3SUM conjecture, and Kopelowitz, Pettie, and Porat [16] improved Pătraşcu's framework to give better reductions from 3SUM to SetIntersection, SetDisjointness, and triangle enumeration. Goldstein, Kopelowitz, Lewenstein, and Porat [13] showed several reporting problems are 3SUM-hard. Vassilevska and Williams [19], and Jafargholi and Viola [15] used 3SUM to study triangle problems. Abboud and Lewi [1] proved tight lower and upper bounds for the exact-weight subgraph finding problem under the k-SUM conjecture.

▶ **Definition 1** (k-SUM)**.** Given subsets $A_1, \dots, A_k$ of size n of an abelian group G, the k-SUM problem asks whether there are $a_1 \in A_1, \dots, a_k \in A_k$ such that $\sum_{i=1}^{k} a_i = 0$.

* Supported by NSF grant CCF-1423544 and a Simons Foundation Investigator grant.

A simple meet-in-the-middle algorithm can solve k-SUM in time $\tilde{O}(n^{\lceil k/2 \rceil})$,[1] and it is widely believed that this is the optimal time up to polylogarithmic factors. This is known as the k-SUM conjecture:

▶ **Conjecture 2** (k-SUM Conjecture). *For $k \geq 2$, k-SUM in $\mathbb{Z}$ requires randomized time $\Omega(n^{\lceil k/2 \rceil - o(1)})$.*

To support the k-SUM conjecture, Erickson [9] and Ailon and Chazelle [3] proved that k-linear decision trees cannot solve k-SUM with fewer than $n^{\lceil k/2 \rceil}$ queries. On the other hand, Freund [10], and Gold and Sharir [12] recently gave $O(n^2 \log\log n / \log n)$ algorithms for 3SUM. Hence, the standard 3SUM conjecture (Conjecture 7) is stated as an $\Omega(n^{2-o(1)})$ lower bound instead of $\Omega(n^2)$.

Intriguingly, in non-uniform models, substantially lower complexities are known: Grønlund and Pettie [14] showed that the decision tree complexity of 3SUM is $O(n^{3/2} \log n)$. Gold and Sharir [12] showed that the randomized $(2k-2)$-linear decision tree complexity of k-SUM is $O(n^{k/2})$ for any odd $k \geq 3$.

Multidimensional k-SUM in $\mathbb{F}_p^d$

One can also consider the k-SUM problem over domains other than $\mathbb{Z}$. The focus of this paper will be on the multidimensional case, where the domain is $\mathbb{F}_p^d$. The k-SUM problem in $\mathbb{F}_p^d$ is a problem of independent interest. For example, Jafargholi and Viola [15, 20] reduced listing triangles to 3SUM in $\mathbb{F}_2^d$. In coding theory, k-SUM in $\mathbb{F}_p^d$ is studied and known as WeightDistribution [8].

Bhattacharyya et. al. [6] recently gave a $\min(2^{\Omega(d)}, n^{\Omega(k)})$ lower bound for k-SUM in $\mathbb{F}_p^d$ under the Exponential Time Hypothesis. Pătraşcu and Williams [18] proved that the Exponential Time Hypothesis implies a weak version of the k-SUM conjecture - there is no $n^{o(k)}$ algorithm for k-SUM for all k. However, prior to this paper, no connection was known between integer k-SUM and k-SUM in $\mathbb{F}_p^d$. In this paper, we use the k-SUM conjecture to prove a more refined lower bound for k-SUM in $\mathbb{F}_p^d$:

▶ **Theorem 3.** *Under the k-SUM conjecture, for any $k \geq 2$, k-SUM in $\mathbb{F}_p^d$ requires time $\Omega(n^{k/2-o(1)})$ for even k, and time $\Omega(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - o(1)})$ for odd k, when p is sufficiently large.*

Like the one-dimensional case, the fastest known algorithm for k-SUM in $\mathbb{F}_p^d$ is the meet-in-the-middle algorithm in time $\tilde{O}(n^{\lceil k/2 \rceil})$, which matches with the above conditional lower bound for even k.

Our conditional lower bound is meaningful for each $k \geq 2$, which is a stronger statement than the asymptotic result by Bhattacharyya et. al.

Monotone d-dimensional 3SUM

Chan and Lewenstein [7] first studied bounded monotone d-dimensional 3SUM, motivated by bounded monotone $(\min, +)$-convolution and histogram indexing. Chan and Lewenstein gave a remarkable $\tilde{O}(n^{2-\frac{2}{d+13}})$ algorithm with techniques from additive combinatorics. One of our main result is to show this algorithm is essentially optimal under the 3SUM conjecture.

[1] $\tilde{O}(f(n))$ is a notation for $O(f(n)\text{polylog}(n))$.

▶ **Definition 4** (Bounded Monotone d-dimensional 3SUM)**.** A set $A \subset \mathbb{Z}^d$ is monotone increasing if it can be sorted as $A = \{a_1, \ldots, a_n\}$ such that the j-th coordinates of $a_1, \ldots, a_n$ form a monotone increasing sequence for each $j = 1, \ldots, d$. Given monotone sets $A, B, S \subset [n]^d$, bounded monotone d-dimensional 3SUM asks if there exist $a \in A, b \in B, s \in S$ such that $a + b = s$.[2]

Chan and Lewenstein's subquadratic $\tilde{O}(n^{2-\frac{2}{d+13}})$ algorithm shows that bounded monotone d-dimensional 3SUM is easier than integer 3SUM, but how much easier? Since monotonicity is a strong restriction on the set structure, one may wonder whether further improvements are possible. We show, under the 3SUM conjecture, the answer is no:

▶ **Theorem 5.** *Under the standard 3SUM conjecture, bounded d-dimensional monotone 3SUM requires time $\Omega(n^{2-\frac{4}{d}-o(1)})$.*

One can also define a strong version of the 3SUM conjecture (see Conjecture 8) and this yields a slightly stronger result:

▶ **Theorem 6.** *Under the strong 3SUM conjecture, bounded d-dimensional monotone 3SUM requires time $\Omega(n^{2-\frac{2}{d}-o(1)})$.*

In Chan and Lewenstein's $\tilde{O}(n^{2-\frac{2}{d+13}})$ upper bound, the exponent $2 - 2/(d+13)$ comes from solving a quadratic equation capturing a fairly involved recurrence; it is surprising to see essentially the same exponent arise for completely different reasons in our lower bound.

Standard vs Strong 3SUM Conjecture

In this section we discuss the so-called "strong" 3SUM conjecture. For clarity, we refer to the well-known 3SUM conjecture (a special case of Conjecture 2) as the "standard" 3SUM conjecture:

▶ **Conjecture 7** (Standard 3SUM Conjecture)**.** *Integer 3SUM requires time $\Omega(n^{2-o(1)})$.*

It is known that 3SUM on a set of n integers can be reduced to the bounded domain of $\{-n^3, \ldots, n^3\}$ via a randomized reduction [5, 17]. The strong 3SUM conjecture further restricts the domain to $\{-n^2, \ldots, n^2\}$. It was first proposed by Amir, Chan, Lewenstein, and Lewenstein to obtain better conditional lower bounds for jumbled indexing [4]. [3]

▶ **Conjecture 8** (Strong 3SUM Conjecture)**.** *3SUM on a set of n integers in the domain of $\{-n^2, \ldots, n^2\}$ requires time $\Omega(n^{2-o(1)})$.*

As a context for the strong 3SUM conjecture, 3SUM in the domain of $\{-n^{2-\delta}, \ldots, n^{2-\delta}\}$ can be solved in time $\tilde{O}(n^{2-\delta})$ by Fast Fourier Transform (FFT). However, it is a long-standing open problem whether there is a truly subquadratic algorithm for 3SUM in the domain of $\{-n^2, \ldots, n^2\}$.

It is an open problem whether the strong 3SUM conjecture is equivalent to the standard 3SUM conjecture. In this paper, we prove a partial result along these lines in Theorem 9. This result may be a folklore in some communities, but it seems that it has not been written down, so we include a formal analysis for completeness.

[2] $[m]$ is a notation for $\{0, 1, \ldots, m-1\}$.
[3] Recently, Goldstein, Kopelowitz, Lewenstein, and Porat [13] showed that the strong 3SUM conjecture is not necessary for the hardness of jumbled indexing, and improved the result by basing on the standard 3SUM conjecture.

▶ **Theorem 9.** *Under the standard 3SUM conjecture, 3SUM$^+$ in the domain of* $\{-n^{2+\delta}, ..., n^{2+\delta}\}$ *requires time* $\Omega(n^{2-o(1)})$ *for any* $\delta > 0$.

Here, 3SUM$^+$ is the extension of 3SUM that reports all $a_3 \in A_3$ such that $a_1+a_2+a_3 = 0$ for some $a_1 \in A_1, a_2 \in A_2$, i.e. it outputs $A_3 \cap -(A_1+A_2)$. As noted by Chan and Lewenstein [7], all the known 3SUM algorithms actually solve 3SUM$^+$, including Fast Fourier Transfrom and Baren et al.'s slightly subquadratic $O((n^2/\log^2 n)(\log\log n)^2)$ algorithm [5].

If Theorem 9 still holds with $\delta = 0$ and 3SUM in place of 3SUM$^+$, then the strong 3SUM conjecture would be equivalent to the standard 3SUM conjecture.

Organization

The next three sections contain the technical proofs of the main theorems. In Section 2, we prove Theorem 3, the lower bound for k-SUM in $\mathbb{F}_p^d$ under the k-SUM conjecture. Section 3 contains the proofs for Theorem 5 and Theorem 6, which are lower bounds for bounded monotone 3SUM under the standard and strong 3SUM conjectures. In Section 4, we prove Theorem 9.

2 Reductions Used for the Lower Bound for k-SUM in $\mathbb{F}_p^d$

Underlying the lower bound for k-SUM in $\mathbb{F}_p^d$ is a pair of reductions: a reduction from integer k-SUM in $\mathbb{F}_p^d$ to integer $(k+1)$-SUM, and a reduction from integer k-SUM to $(k+1)$-SUM in $\mathbb{F}_p^d$. From the reductions, we deduce conditional lower bounds for k-SUM in $\mathbb{F}_p^d$, for bounded monotone d-dimensional 3SUM, and for 3SUM$^+$ in bounded domain $\{-n^{2+\delta}, ..., n^{2+\delta}\}$.

A natural idea for reduction is to use the bijection between $\mathbb{F}_p^d$ and $[p^d] \subset \mathbb{Z}$, seeing $\mathbb{F}_p^d$ as a base-p representation of integers. However, the bijection is not an abelian group homomorphism, due to the "carries" in integer addition and the mod p effect in $\mathbb{F}_p^d$-addition. The main challenge is to simulate the carries while preserving the k-SUM structure.

Lemma 10 is the reduction from k-SUM in $\mathbb{F}_p^d$ to integer $(k+1)$-SUM. This is the easier direction among the two reductions. We map $a = (a_0, ..., a_{d-1}) \in \mathbb{F}_p^d \mapsto \sum_{i=0}^{d-1} a_i(kp)^i \in \mathbb{Z}$, treating $\mathbb{F}_p^d$ coordinates as digits in a base-kp number. Since the digits are blown up by powers of kp, there are no "carries" in integer addition.

▶ **Lemma 10.** *Given a k-SUM instance in* $\mathbb{F}_p^d$ *on k sets* $A^{(1)}, \cdots, A^{(k)}$ *of size n, it can be reduced to an integer* $(k+1)$*-SUM instance on k sets of size n and an additional set of size* k^d.

Proof. Let $g : \mathbb{F}_p^d \to \mathbb{Z}$ be the injective map $a = (a_0, ..., a_{d-1}) \mapsto \sum_{i=0}^{d-1} a_i(kp)^i$.

For any $a^{(1)}, \ldots, a^{(k)} \in \mathbb{F}_p^d$, the sum $a^{(1)} + \cdots + a^{(k)}$ is zero in $\mathbb{F}_p^d$ if and only if the i-th coordinate sum $\sum_{j=1}^k a_i^{(j)}$ is a multiple of p for all $0 \le i < d$. Since $a_i^{(j)} < p$, we know $\sum_{j=1}^k a_i^{(j)} < kp$. The above condition can be rewritten as $\sum_{j=1}^k a_i^{(j)} = \lambda_i p$, where $\lambda_i \in \{0, ..., k-1\}$. This is further equivalent to

$$g(a^{(1)}) + \cdots + g(a^{(k)}) = \sum_{i=0}^{d-1} \lambda_i p(kp)^i \text{ for some } (\lambda_1, \ldots, \lambda_d) \in [k]^d.$$

Therefore, the original k-SUM instance in $\mathbb{F}_p^d$ can be reduced to the integer $(k+1)$-SUM instance on $g(A^{(1)}), \ldots, g(A^{(k)})$, and an additional set $\{-\sum_{i=0}^{d-1} \lambda_i p(kp)^i : (\lambda_i) \in [k]^d\}$. ◀

▶ **Corollary 11.** *Given a k-SUM instance in $\mathbb{F}_p^d$ on k sets of size n, it can be reduced to k^d instances of integer k-SUM.*

Proof. Using the reduction in Lemma 10, we can solve the $(k+1)$-SUM instance by enumerating the additional set of size k^d. For each element a in the additional set of size k^d, subtract a from the first set, and solve the k-SUM instance on the first k sets. ◀

Since the proof only uses the additive structure of $\mathbb{F}_p^d$, Lemma 10 also holds more generally for k-SUM instance in $\mathbb{Z}_q^d$, where q is not necessarily a prime number. In fact, all proofs in this section can be adapted to $\mathbb{Z}_q^d$ with some modification.

In the reverse direction, it is more challenging to design a reduction from integer to $\mathbb{F}_p^d$, since the reduction needs to simulate integer addition carries with $\mathbb{F}_p^d$.

In the proof of Lemma 12, we start with the bijection between $\mathbb{F}_p^d$ and $[p^d] \subset \mathbb{Z}$, viewing $\mathbb{F}_p^d$ as a base-p representation of integers. This provides a way to map integers to $\mathbb{F}_p^d$, but unfortunately the map does not preserve the additive structure in k-SUM. To fix this problem, the key observation is that the map does preserve the additive structure with respect to k-SUM when all coordinates are between 0 and $\lfloor \frac{p}{k} \rfloor$. Points in $\{0,\dots,\lfloor \frac{p}{k} \rfloor\}^d \subset \mathbb{F}_p^d$ behaves nicely for our purpose. Therefore, we divide $\mathbb{F}_p$ into k chunks: $\{0,\dots,\lfloor \frac{p}{k} \rfloor\}, \{\lceil \frac{p}{k} \rceil,\dots,2\lfloor \frac{p}{k} \rfloor\}$, $\dots$, $\{\lambda \lceil \frac{p}{k} \rceil,\dots,(\lambda+1)\lfloor \frac{p}{k} \rfloor\}$, and so on. Accordingly, $\mathbb{F}_p^d$ is divided into k^d cubes $\{S_\lambda : \lambda = (\lambda_1,\dots,\lambda_d) \in [k]^d\}$. For each cube, we shift it to align with the nice cube $S_0 = \{0,\dots,\lfloor \frac{p}{k} \rfloor\}^d$ where the reduction preserves the additive structure, perform the reduction, and then shift it back.

A similar technique was first used by Abboud, Lewi, and Williams [2] to reduce integer k-SUM to k-VECTOR-SUM.

▶ **Lemma 12.** *Assume $p > k^2$. Given an integer k-SUM instance on k sets $X^{(1)},\dots,X^{(k)}$ of size n in the bounded universe $[m]$, it can be reduced to a $(k+1)$-SUM instance in $\mathbb{F}_p^{2d}$ on k sets of size n and a set of size k^{2d}, where $d = \log_p m$.*

Proof. Let $f : \mathbb{F}_p^d \to \{0,\dots,m\}$ be the bijection $a = (a_0,\dots,a_{d-1}) \mapsto \sum_{i=1}^{d-1} a_i p^i$. Note this is a bijection but does not preserve the additive structure.

Define $\mu : \mathbb{F}_p^d \to [k]^d$ as the following index function. For any $a = (a_0,\dots,a_{d-1}) \in \mathbb{F}_p^d$, $\mu(a) = (\mu_0,\dots,\mu_{d-1})$, where μ_i is defined to be the integer such that $\mu_i \lceil \frac{p}{k} \rceil \le a_i < (\mu_i+1)\lfloor \frac{p}{k} \rfloor$.

For any $\lambda \in [k]^d$, define $S_\lambda := \{a \in \mathbb{F}_p^d : \mu(a) = \lambda\}$. Define $\overline{a} = a - \lfloor \frac{p}{k} \rfloor \cdot \mu(a)$, then we can write

$$a = \overline{a} + \left\lfloor \frac{p}{k} \right\rfloor \cdot \mu(a),$$

where $\overline{a} \in S_0$.

$$f(a) = f(\overline{a}) + \sum_{i=1}^{d-1} \mu_i(a) \left\lfloor \frac{p}{k} \right\rfloor p^i.$$

Observe that for any $\overline{a^{(1)}},\dots,\overline{a^{(k)}} \in S_0$, it is true that $\sum_{j=1}^k f(\overline{a^{(j)}}) = f(\sum_{j=1}^k \overline{a^{(j)}})$. This equality does not hold in general for elements outside S_0.

Thus, for any $a^{(1)},\dots,a^{(k)} \in \mathbb{F}_p^d$,

$$\sum_{j=1}^k f(a^{(j)}) = \sum_{j=1}^k f(\overline{a^{(j)}}) + \sum_{i=1}^d \left(\sum_{j=1}^k \mu_i(a^{(j)}) \right) \left\lfloor \frac{p}{k} \right\rfloor p^i.$$

Fix $\lambda^{(1)}, ..., \lambda^{(k)} \in [k]^d$. For any $(x^1, ..., x^{(k)})$ where $x^{(j)} \in X^{(j)} \cap S_{\lambda^{(j)}}$, if we denote $a^{(j)} = f^{-1}(x^{(j)})$, then

$$\sum_{j=1}^{k} x^{(j)} = \sum_{j=1}^{k} f(a^{(j)}) = \sum_{j=1}^{k} f(\overline{a^{(j)}}) + \sum_{i=1}^{d}\sum_{j=1}^{k} \lambda_i^{(j)} \left\lfloor \frac{p}{k} \right\rfloor p^i = f(\sum_{j=1}^{k} \overline{a^{(j)}}) + \sum_{i=1}^{d}\sum_{j=1}^{k} \lambda_i^{(j)} \left\lfloor \frac{p}{k} \right\rfloor p^i.$$

Thus,

$$\sum_{j=1}^{k} x^{(j)} = 0 \iff f(\sum_{j=1}^{k} \overline{a^{(j)}}) + \sum_{i=1}^{d}\sum_{j=1}^{k} \lambda_i^{(j)} \left\lfloor \frac{p}{k} \right\rfloor p^i = 0,$$

and

$$\sum_{j=1}^{k} x^{(j)} = 0 \iff \sum_{j=1}^{k} \overline{a^{(j)}} = f^{-1}\left(-\sum_{i=1}^{d}\sum_{j=1}^{k} \lambda_i^{(j)} \left\lfloor \frac{p}{k} \right\rfloor p^i\right). \tag{*}$$

Note that the right hand side only depends on $\lambda^{(1)}, ..., \lambda^{(k)}$, or more specifically $\sum_{j=1}^{k} \lambda_i^{(j)}$.

This gives a reduction from the integer k-SUM instance to a $(k+1)$-SUM instance on k sets of size n in $\mathbb{F}_p^d \times \mathbb{Z}^d$ of the form

$$A^{(j)} = \{(\overline{a}, \mu(a)) : f(a) \in X^{(j)}\},$$

and a set of size k^{2d} consisting of elements in the form of

$$-\left(f^{-1}(-\sum_{i=1}^{d} \sigma_i \left\lfloor \frac{p}{k} \right\rfloor p^i),\ \ \sigma\right),$$

for all $\sigma \in [k^2]^d \subset \mathbb{Z}^d$. The range $[k^2]^d$ is determined by the fact that since $\lambda^{(j)} \in [k]^d$ for each j, the sum σ is bounded above by k^2 in each coordinate. The $\mathbb{Z}^d$ component in $\mathbb{F}_p^d \times \mathbb{Z}^d$ is always bounded in $[k^2]^d$, so $\mathbb{F}_p^d \times \mathbb{Z}^d$ can be viewed as $\mathbb{F}_p^{2d}$ when $p > k^2$. ◀

The assumption $p > k^2$ in Lemma 12 is adopted only to simplify some details at the end when turning $\mathbb{F}_p^d \times [k^2]^d$ into $\mathbb{F}_p^{2d}$. When $p \leq k^2$, the same techniques apply with more care in dealing with the $[k^2]^d$ component. All previous parts before (*) in the proof hold for $p > k$, so the following corollary only requires $p > k$.

▶ **Corollary 13.** *Assume $p > k$. Given an integer k-SUM instance on k sets of size n in the bounded universe $[m]$, it can be reduced to k^{2d} instances of k-SUM in $\mathbb{F}_p^d$ on k sets of size n, where $d = \log_p m$.*

Proof. This follows from the reduction up to (*) in the proof of the previous theorem. ◀

Lower Bound for k-SUM in $\mathbb{F}_p^d$

The following lower bound under the k-SUM conjecture follows from Lemma 12. For even k, our result matches with the $\tilde{O}(n^{k/2})$ upper bound. For odd k, our lower bound is $\Omega(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - o(1)})$, while the best known upper bound is $\tilde{O}(n^{\lceil k/2 \rceil})$. The conditional lower bound converges to the upper bound as $p \to \infty$, which is expected since the group $\mathbb{F}_p$ behaves "more and more like $\mathbb{Z}$" as p increases.

The assumption that p is large enough such that $p \geq k^{2k}$ is still a meaningful assumption, because k^{2k} is a constant for fixed k.

▶ **Theorem 3. (Restated)** *For any $k \geq 2$, assume p is sufficiently large such that $p \geq k^{2k}$. Under the k-SUM conjecture, k-SUM in $\mathbb{F}_p^d$ requires time $\Omega(n^{k/2-o(1)})$ for even k, and $\Omega(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - o(1)})$ for odd k, for $d \geq 2k \log_p n$.*

Proof. Using a randomized reduction [5, 17], we can assume without loss of generality that a given integer k-SUM instance is in the bounded domain $[n^k] = \{0, \ldots, n^k - 1\}$.

Suppose k is even. By Lemma 12, any integer $(k-1)$-SUM instance can be reduced to a k-SUM instance in $\mathbb{F}_p^d$ on $k-1$ sets of size n and a set of size $(k-1)^d$, where $d = 2\log_p n^{(k-1)}$. (The d used here is $2d$ in Lemma 12.) Assuming $p > k^{2k}$, then the size of the last set can be bounded above by $(k-1)^d \leq k^d = n^{2k\frac{\log k}{\log p}} \leq n$. Hence, integer $(k-1)$-SUM can be reduced to k-SUM in $\mathbb{F}_p^d$ on k sets of size n. If k-SUM in $\mathbb{F}_p^d$ can be solved in time $O(n^{k/2-\epsilon})$, then integer $(k-1)$-SUM could be solved in time $O(n^{k/2-\epsilon}) = O(n^{\lceil \frac{k-1}{2} \rceil - \epsilon})$, violating the k-SUM conjecture.

Suppose k is odd. By Corollary 13, integer k-SUM can be reduced to k^d instances of k-SUM in $\mathbb{F}_p^d$. If k-SUM in $\mathbb{F}_p^d$ can be solved in time $O(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - \epsilon})$, then integer $(k-1)$-SUM could be solved in time $k^d \times O(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - \epsilon})$. Since $d = 2\log_p n^{(k-1)} \leq 2k\frac{\log n}{\log p}$, we can bound k^d by $k^d = n^{d\frac{\log k}{\log n}} \leq n^{2k\frac{\log k}{\log p}}$. Therefore, integer $(k-1)$-SUM could be solved in time $k^d \times O(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - \epsilon}) \leq O(n^{\lceil k/2 \rceil - \epsilon})$, violating the k-SUM conjecture. ◀

In particular, under the 3SUM conjecture, 3SUM in $\mathbb{F}_p^d$ requires time $\Omega(n^{2-\frac{6}{\log_3 p} - o(1)})$. This implies that there does not exist an $\epsilon > 0$ such that 3SUM in $\mathbb{F}_p^d$ can be solved in $O(n^{2-\epsilon})$ for all prime p, under the 3SUM conjecture. Combined with the reduction in Lemma 10, we obtain the following weak equivalence between integer k-SUM and k-SUM in $\mathbb{F}_p^d$.

▶ **Theorem 14.** *For any $k \geq 2$, integer k-SUM can be solved in $O(n^{\lceil k/2 \rceil - \delta})$ time for some $\delta > 0$ if and only if there exists an $\epsilon > 0$ such that k-SUM in $\mathbb{F}_p^d$ can be solved in $O(n^{\lceil k/2 \rceil - \epsilon})$ for all sufficiently large prime p.*

Proof. Suppose integer k-SUM can be solved in $O(n^{\lceil k/2 \rceil - \delta})$ time for some $0 < \delta < 1$. Take $\epsilon = \delta/2$, and take p large enough such that $k\frac{\log k}{\log p} < \delta/2$. Using a randomized reduction, we may assume without loss of generality that $d \leq k \log_p n$. By Corollary 11, k-SUM in $\mathbb{F}_p^d$ can be reduced to k^d instances of integer k-SUM, so k-SUM in $\mathbb{F}_p^d$ can be solved in time $k^d \cdot O(n^{\lceil k/2 \rceil - \delta}) \leq n^{k\frac{\log k}{\log p}} \cdot O(n^{\lceil k/2 \rceil - \delta}) = O(n^{\lceil k/2 \rceil - \delta/2})$.

Conversely, suppose there exists an $\epsilon > 0$ such that k-SUM in $\mathbb{F}_p^d$ can be solved in $O(n^{\lceil k/2 \rceil - \epsilon})$ for all sufficiently large prime p. Then, there exists some p large enough such that $\epsilon > 2k\frac{\log k}{\log p}$. By the proof of Theorem 3, integer k-SUM can be solved in time $O(n^{\lceil k/2 \rceil - \epsilon + 2k\frac{\log k}{\log p}})$. ◀

3 Proof of Lower Bound for Monotone 3SUM in $[n]^d$

In Definition 4, we defined bounded monotone d-dimensional 3SUM to be 3SUM on a monotone set in $[n]^d$, where n is the range of the coordinates. Alternatively, one could also define bounded monotone 3SUM to be on a monotone set of size n. Note that a monotone set in $[n]^d$ can have size at most dn, so the two definitions are equivalent up to a factor of d.

To prove the lower bound for bounded monotone 3SUM, we reduce from Convolution-3SUM to bounded monotone 3SUM. The Convolution-3SUM problem is a more restricted version of 3SUM:

▶ **Definition 15** (Convolution-3SUM)**.** Given an array $A[1 \ldots n]$, determine whether there exist $i \neq j$ with $A[i] + A[j] = A[i + j]$.

Pǎtraşcu [17] first proved that if 3SUM requires $\Omega(n^{2-o(1)})$ time, then so does Convolution-3SUM. Kopelowitz, Pettie, and Porat [16] showed that the randomized complexities of 3SUM and Convolution-3SUM differ by at most a logarithmic factor.

Note that the brute force algorithm for Convolution-3SUM runs in $O(n^2)$ time, as there are only $O(n^2)$ possible pairs to try. Amir, Chan, Lewenstein, and Lewenstein [4] pointed out a (randomized) reduction from Convolution-3SUM in any large domain to Convolution-3SUM in $\{-n^2, \ldots, n^2\}$.

The above results imply that the 3SUM conjecture is equivalent to the hardness of Convolution-3SUM in $\{-n^2, \ldots, n^2\}$, making Convolution-3SUM a useful tool for our purpose. Amir, Chan, Lewenstein, and Lewenstein [4] also restated the strong 3SUM conjecture as: Any algorithm for Convolution-3SUM in the domain of $\{-n, \ldots, n\}$ requires $\Omega(n^{2-o(1)})$ time.

Previously we defined k-SUM as determining whether there exists $a_1 + \cdots + a_k = 0$ for $a_i \in A_i$. In this section, for convenience we use an equivalent formulation of 3SUM: given sets A, B, S, determine whether there exist $a \in A, b \in B, s \in S$ such that $a + b = s$. The benefit is to avoid negative integer values. For 3SUM or Convolution-3SUM on A, B, S in $\{-u, \ldots, u\}$, add u to all values in A, B, and add $2u$ to all values in S. Therefore, it is sufficient to work with Convolution-3SUM in $[n^2]$ instead of $\{-n^2, \ldots, n^2\}$, and $[n]$ instead of $\{-n, \ldots, n\}$.

Convolution-3SUM in $[m]$ is a special case of 3SUM in $[n] \times [m]$, by mapping the array indices to the first coordinate, i.e. integer a_i in a Convolution-3SUM instance is mapped to the pair $(i, a_i) \in [n] \times [m]$. We will use this notation for simplicity. Since array indices are unique, no two values have the same first coordinate.

First, we reduce integer Convolution-3SUM to d-dimensional Convolution-3SUM, using the same techniques as in Lemma 12 (the reduction from integer 3SUM to d-dimensional 3SUM). Here we replace $\mathbb{F}_p^d$ with $[p]^d$, but the proof is nearly identical to Lemma 12, so we do not repeat it here.

▶ **Lemma 16.** *For any given dimension d, Convolution-3SUM in $[n^c]$ can be reduced to 4^d instances of Convolution-3SUM in $[n^{c/d}]^d$.*

Next we reduce from d-dimensional Convolution-3SUM to monotone d-dimensional Convolution-3SUM, by blowing up the bounded domain.

▶ **Lemma 17.** *Convolution-3SUM in $[m]^d$ can be reduced to Convolution-3SUM on monotone sets in $[nm]^d$.*

Proof. Let $f : [n] \times [m]^d \to [n] \times [nm]^d$ be the map $(a_0, a_1, ..., a_d) \mapsto (a_0, ma_0 + a_1, ..., ma_0 + a_d)$. Take $A' = f(A), B' = f(B), S' = f(S)$. Since f is linear in each coordinate, $a + b + s = 0$ implies $f(a) + f(b) + f(s) = 0$. Conversely, if $f(a) + f(b) + f(s) = 0$, the first coordinate guarantees $a_0 + b_0 + s_0 = 0$, and then it follows that $a_i + b_i + s_i = 0$ in each coordinate.

Since no two points in A (resp. B, S) share the same first coordinate, the new sets A' (resp. B', S') are monotone if we order them in increasing a_0 (resp. b_0, s_0), because the a_0 dominates in $ma_0 + a_i$.

◀

The following lemma is the main technical lemma from which Theorem 5 and Theorem 6 easily follow.

▶ **Lemma 18.** *Let c be a real constant between $1 \le c \le 2$, let $d > c$ be an integer, and let $\delta > 0$ be an arbitrarily small constant. Assume n is large enough such that $n > 2^{4d/\delta}$. If 3SUM for monotone sets in $[n]^d$ can be solved in time $O(n^{2-2c/d-\delta})$, then Convolution-3SUM in $[n^c]$ can be solved in time $O(n^{2-\delta/2})$.*

Proof. Let $\gamma = \frac{c}{d} < 1$. By Lemma 16 and Lemma 17, Convolution-3SUM in $[n^c]$ can be reduced to 4^d instances of Convolution-3SUM on monotone sets in $[n^{(1+\gamma)}]^d$. By writing the array index as an additional dimension, i.e. mapping a_i in a Convolution-3SUM instance to (i, a_i), each of the 4^d Convolution-3SUM instances can be seen as $(d+1)$-dimensional monotone 3SUM in $[n^{(1+\gamma)}]^{(d+1)}$.

By assumption, 4^d instances of $(d+1)$-dimensional monotone 3SUM in $[n^{(1+\gamma)}]^{(d+1)}$ can be solved in time

$$4^d \cdot O(n^{(1+\gamma)(2-2c/d-\delta)}) = O(n^{\frac{d}{\log_4 n}+(1+\gamma)(2(1-c/d)-\delta)}).$$

The exponent can be simplified as follows. When $n > 2^{4d/\delta}$,

$$\frac{d}{\log_4 n} + (1+\gamma)(2(1-c/d)-\delta) = 2 - \delta - \gamma\delta - 2\gamma^2 + \frac{d}{\log_4 n} < 2 - \delta/2.$$

Thus, when n is sufficiently large, the 4^d instances of $(d+1)$-dimensional monotone 3SUM can be solved in time $O(n^{2-\delta/2})$.

◀

▶ **Theorem 5. (Restated)** *Under the standard 3SUM conjecture, bounded d-dimensional monotone 3SUM requires time $\Omega(n^{2-\frac{4}{d}-o(1)})$.*

Proof. This is a immediate corollary to Lemma 18 with $c = 2$. ◀

The lower bound can be improved to $\Omega(n^{2-\frac{4}{d+1}-o(1)})$ specifically for $c = 2$ with a little extra computation.

One can also define a strong version of the 3SUM conjecture (see Definition 8) and this yields a slightly stronger result:

▶ **Theorem 6. (Restated)** *Under the strong 3SUM conjecture, bounded d-dimensional monotone 3SUM requires time $\Omega(n^{2-\frac{2}{d}-o(1)})$.*

Proof. This is a immediate corollary to Lemma 18 with $c = 1$. ◀

4 Strong 3SUM Conjecture vs 3SUM Conjecture

The 3SUM conjecture states that integer 3SUM requires time $\Omega(n^{2-o(1)})$, while the strong 3SUM conjecture states that integer 3SUM in the bounded domain of $\{-n^2, \ldots, n^2\}$ requires time $\Omega(n^{2-o(1)})$. We would like to understand whether the 3SUM conjecture is equivalent to the strong 3SUM conjecture. To add evidence to this, we prove a partial result that 3SUM$^+$ in the domain of $\{-n^{2+\delta}, ..., n^{2+\delta}\}$ is as hard as unbounded integer 3SUM, using ideas about multidimensional 3SUM from previous sections.

▶ **Lemma 19** (Lemma 1, [5]). *Let A be a sorted list of n integers. For any fixed $c \in A$, we can decide whether c is a hit (i.e., whether there exist $a, b \in A$ such that $a + b = c$) in $O(n)$ time.*

The proof idea for Lemma 20 is similar to our reduction from multidimensional 3SUM to integer 3SUM in Lemma 10. Instead of $\mathbb{F}_p^3$, we consider 3SUM in the group $\mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$ for three different primes.

▶ **Lemma 20.** *Let $0 < \epsilon < 1$. If 3SUM$^+$ in $[M]$ can be solved in $O(n^{2-\epsilon})$ time, then for any primes p_1, p_2, p_3 such that $p_1 p_2 p_3 \le \frac{M}{32}$, 3SUM$^+$ in $\mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$ can be solved in $O(n^{2-\epsilon})$ time.*

Proof. Let $S \subset \mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$ be a subset of size n. Let $\varphi : \mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3} \times \{0,1\}^3 \to [32 p_1 p_2 p_3] \subset \mathbb{Z}$ be the injective map

$$\varphi : (a_1, a_2, a_3, t_1, t_2, t_3) \mapsto_\varphi (t_1 p_1 + a_1) \cdot (16 p_2 p_3) + (t_2 p_2 + a_2) \cdot (4 p_3) + (t_3 p_3 + a_3).$$

Let A be the image $A := \varphi(S \times \{0,1\}^3)$. The size of A is $8n$.

It is easy to check $\varphi(a_1, a_2, a_3, t_1, t_2, t_3) + \varphi(b_1, b_2, b_3, r_1, r_2, r_3) = \varphi(c_1, c_2, c_3, w_1, w_2, w_3)$ implies $(a_1, a_2, a_3) + (b_1, b_2, b_3) = (c_1, c_2, c_3)$ in $\mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$.

Conversely, suppose $(a_1, a_2, a_3) + (b_1, b_2, b_3) = (c_1, c_2, c_3)$ in $\mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$. For $i = 1, 2, 3$, since $a_i + b_i = c_i \pmod{p_i}$, either $a_i + b_i = c_i$ or $a_i + b_i = c_i + p_i$. Thus, there exists $w_1, w_2, w_3 \in \{0,1\}$ such that $\varphi(a_1, a_2, a_3, 0, 0, 0) + \varphi(b_1, b_2, b_3, 0, 0, 0) = \varphi(c_1, c_2, c_3, w_1, w_2, w_3)$.

To solve 3SUM$^+$ on S, first solve 3SUM$^+$ on A, which reports all $(A + A) \cap A$. The 3SUM$^+$ instance on S then outputs all $(c_1, c_2, c_3) \in S$ such that $\varphi(c_1, c_2, c_3, w_1, w_2, w_3) \in (A + A) \cap A$ for some $w_1, w_2, w_3 \in \{0,1\}$. The reduction takes $O(n)$ time. ◀

▶ **Lemma 21.** *Let $a_1, \ldots, a_k$ be k non-zero integers in the domain $\{-u, \ldots, u\}$. For any sufficiently large M,*

$$\Pr_{p_1, p_2, p_3 \in \mathcal{P}_M}\left[\#\{i : a_i \equiv 0 \mod p_1 p_2 p_3\} < 24 \frac{\log^3(M) \log^3(u)}{M^3} k\right] \ge \frac{2}{3}.$$

Proof. Let $X_i = 1$ if $a_i \equiv 0 \mod p_1 p_2 p_3$, and $X_i = 0$ otherwise. For each $i = 1, \ldots, k$, the number of distinct prime factors of a_i is at most $\log u$. By the Prime Number Theorem, $\pi(M) \sim \frac{M}{\log M}$, where $\pi(M)$ is the number of primes in $\{1, \ldots, M\}$. Hence, for M sufficiently large, $\pi(M) \ge \frac{1}{2} \frac{M}{\log M}$, and

$$\Pr_{p \in \mathcal{P}_M}[a_i \equiv 0 \mod p] \le \frac{\log u}{\frac{1}{2} \frac{M}{\log M}} = 2 \frac{\log M \log u}{M}.$$

Since p_1, p_2, p_3 are independently chosen primes,

$$E[X_i] = \Pr_{p_1, p_2, p_3 \in \mathcal{P}_M}[X_i = 1] = \prod_{j=1}^{3} \Pr_{p_j \in \mathcal{P}_M}[a_i \equiv 0 \mod p_j] \le \left(2 \frac{\log M \log u}{M}\right)^3.$$

Let $X = \sum_i X_i = \#\{i : a_i \equiv 0 \mod p_1 p_2 p_3\}$. By linearity of expectation,

$$E[X] = \sum_i E[X_i] \le k \cdot \left(2 \frac{\log M \log u}{M}\right)^3.$$

By Markov's Inequality, $\Pr[X \ge 3E[X]] \le \frac{1}{3}$, as desired. ◀

▶ **Lemma 22.** *For any $0 < \delta < 1$, suppose 3SUM$^+$ in the bounded domain $[n^{2+\delta}]$ can be solved in $O(n^{2-\epsilon})$ time, then 3SUM can be solved in $O(n^{1+\delta} + n^{2-\epsilon} + n^{2-\delta} \log^6(n))$ time with probability $2/3$. (With probability $2/3$, the algorithm answers yes or no correctly, otherwise it outputs "Don't Know".)*

Proof. Using a randomized reduction [5, 17], assume without loss of generality the given 3SUM instance is in the domain of $[n^3]$. Also assume without loss of generality that the given 3SUM instance is on the same set A, asking whether there exist $a, b, c \in A$ such that $a + b = c$. Consider the following algorithm.

Step 1. Sort A. Choose $2n^\delta$ elements in A with uniform probability, and determine whether each of them is a hit. If a hit is found among the $2n^\delta$ elements, output yes and terminate. Otherwise, proceed to step 2. By Lemma 19, this step takes $O(n^{1+\delta})$ time.

Step 2. Choose three primes p_1, p_2, p_3 independently with uniform probability among all primes less than $n^{(2+\delta)/3}$. Map A to a set in A' in $\mathbb{F}_{p_1} \times \mathbb{F}_{p_2} \times \mathbb{F}_{p_3}$ by $a \mapsto (a \bmod p_1, a \bmod p_2, a \bmod p_3)$. By Lemma 20, under the given assumption, 3SUM$^+$ on A' can be solved in $O(n^{2-\epsilon})$ time.

Step 3. If the 3SUM$^+$ instance on A' reports more than $648 \log^6(n) n^{1-\delta}$ hits, the algorithm fails and outputs "Don't Know".

Step 4. If the 3SUM$^+$ instance on A' reports no more than $648 n^{1-\delta} \log^6(n)$ hits, for each pre-image of the hits, determine whether it is a hit in integer 3SUM. If a hit is found, output yes. Otherwise, output no. By Lemma 19, this step runs in time $O(n^{2-\delta} \log^6(n))$.

The total runtime of the algorithm is $O(n^{1+\delta} + n^{2-\epsilon} + n^{2-\delta} \log^6(n))$. To analyze the probability bound:

Suppose there are at least $n^{1-\delta}$ hits in A, then each randomly chosen element in Step 1 is a hit with probability at least $n^{-\delta}$. Let X be the total number of hits among the $2n^\delta$ randomly chosen elements in Step 1, and let $\mu = E[X]$. Since $\mu \geq 2$, by Multiplicative Chernoff Bound,

$$\Pr[X = 0] \leq \Pr[X < (1 - \frac{1}{2})\mu] < \left(\frac{e^{-\frac{1}{2}}}{(1-\frac{1}{2})^{1-\frac{1}{2}}}\right)^\mu = (2e)^{-\frac{\mu}{2}} \leq \frac{1}{2e}.$$

Thus, if the integer 3SUM instance on A has more than $n^{1-\delta}$ hits, the algorithm correctly outputs yes in Step 1 with probability at least $1 - \frac{1}{2e} \geq \frac{2}{3}$.

Suppose the 3SUM instance on A has fewer than $n^{1-\delta}$ hits. By applying Lemma 21 to all non-zero $a_i + b_j - c_k$ (at most n^3 triples), we know with probability at least 2/3, the number of false positive triples is no more than

$$24n^3 \frac{\log^3(n^{(2+\delta)/3}) \log^3(n^3)}{(n^{(2+\delta)/3})^3} \leq 24(2+\delta)^3 \log^6(n) n^{1-\delta} \leq 648 n^{1-\delta} \log^6(n).$$

Thus, the algorithm outputs yes/no correctly in Step 4 with probability at least 2/3, and outputs "Don't Know" in Step 3 with probability at most 1/3. ◀

▶ **Theorem 9.** **(Restated)** *Under the standard 3SUM conjecture, 3SUM$^+$ in the domain of $\{-n^{2+\delta}, ..., n^{2+\delta}\}$ requires time $\Omega(n^{2-o(1)})$ for any $\delta > 0$.*

Proof. This is an immediate corollary to Lemma 22. ◀

5 Conclusion and Open Problems

Under the standard k-SUM conjecture, we proved lower bounds for k-SUM in $\mathbb{F}_p^d$, for bounded monotone d-dimensional 3SUM, and for 3SUM$^+$ in the bounded domain of $\{-n^{2+\delta}, \ldots, n^{2+\delta}\}$ for arbitrarily small δ.

One open problem is whether there is a lower bound for 3XOR (3SUM in $\mathbb{F}_2^d$) under the 3SUM conjecture. Our result is only meaningful for p that are sufficiently large as a function of k. In particular, it does not assert anything for $p = 2$. 3XOR is related to other well known problems such as listing triangles [15]. Not much is known about the complexity of 3XOR. Even an $O(n^{1.99})$ algorithm or an $\Omega(n^{1.01})$ lower bound for 3XOR would be significant.

Another open problem is to obtain a faster k-SUM algorithm in $\mathbb{F}_p^d$ for odd k. Our conditional lower bound for k-SUM in $\mathbb{F}_p^d$ is tight for even k. However, for odd k, there is a gap between the best known $\tilde{O}(n^{\lceil k/2 \rceil})$ algorithm and our $\Omega(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p} - o(1)})$ lower bound. From an optimistic perspective, for odd k, our results suggest that there may be an $O(n^{\lceil k/2 \rceil - 2k\frac{\log k}{\log p}})$ algorithm for k-SUM in $\mathbb{F}_p^d$ without violating the k-SUM conjecture. In particular, there may be an $O(n^{2-\frac{c}{\log p}})$ algorithm for 3SUM in $\mathbb{F}_p^d$, even under the 3SUM conjecture.

References

1 Amir Abboud and Kevin Lewi. Exact weight subgraphs and the k-SUM conjecture. In *International Colloquium on Automata, Languages, and Programming*, pages 1–12. Springer, 2013.

2 Amir Abboud, Kevin Lewi, and Ryan Williams. Losing weight by gaining edges. In *European Symposium on Algorithms*, pages 1–12. Springer, 2014.

3 Nir Ailon and Bernard Chazelle. Lower bounds for linear degeneracy testing. *Journal of the ACM (JACM)*, 52(2):157–171, 2005.

4 Amihood Amir, Timothy M Chan, Moshe Lewenstein, and Noa Lewenstein. On hardness of jumbled indexing. In *Automata, Languages, and Programming*, pages 114–125. Springer, 2014.

5 Ilya Baran, Erik D Demaine, and Mihai Pătraşcu. Subquadratic algorithms for 3SUM. *Algorithmica*, 50(4):584–596, 2008.

6 Arnab Bhattacharyya, Piotr Indyk, David P Woodruff, and Ning Xie. The complexity of linear dependence problems in vector spaces. In *ICS*, pages 496–508, 2011.

7 Timothy M Chan and Moshe Lewenstein. Clustered integer 3SUM via additive combinatorics. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 31–40. ACM, 2015.

8 Rod G Downey, Michael R Fellows, Alexander Vardy, and Geoff Whittle. The parametrized complexity of some fundamental problems in coding theory. *SIAM Journal on Computing*, 29(2):545–570, 1999.

9 Jeff Erickson. Lower bounds for linear satisfiability problems. In *Chicago Journal of Theoretical Computer Science*, volume 8, 1999.

10 Ari Freund. Improved subquadratic 3sum. *Algorithmica*, pages 1–19, 2015.

11 Anka Gajentaan and Mark H Overmars. On a class of $O(n^2)$ problems in computational geometry. *Computational Geometry*, 5(3):165–185, 1995.

12 Omer Gold and Micha Sharir. Improved bounds for 3sum, k-sum, and linear degeneracy. *arXiv preprint arXiv:1512.05279*, 2015.

13 Isaac Goldstein, Tsvi Kopelowitz, Moshe Lewenstein, and Ely Porat. How hard is it to find (honest) witnesses? In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 57. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

14 Allan Grønlund and Seth Pettie. Threesomes, degenerates, and love triangles. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 621–630. IEEE, 2014.

15 Zahra Jafargholi and Emanuele Viola. 3SUM, 3XOR, triangles. *Algorithmica*, 74(1):326–343, 2016.

16 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1272–1287. Society for Industrial and Applied Mathematics, 2016.

17 Mihai Pătraşcu. Towards polynomial lower bounds for dynamic problems. In *Proceedings of the Forty-Second ACM Symposium on Theory of computing*, pages 603–610. ACM, 2010.

18 Mihai Pătraşcu and Ryan Williams. On the possibility of faster SAT algorithms. In *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1065–1075. SIAM, 2010.

19 Virginia Vassilevska and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, pages 455–464. ACM, 2009.

20 Emanuele Viola. Reducing 3XOR to listing triangles, an exposition. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 18, page 113, 2011.