On The Hardness of Approximate and Exact (Bichromatic) Maximum Inner Product

Lijie Chen[1]
Massachusetts Institute of Technology, USA
lijieche@mit.edu

Abstract

In this paper we study the (Bichromatic) Maximum Inner Product Problem (Max-IP), in which we are given sets A and B of vectors, and the goal is to find $a \in A$ and $b \in B$ maximizing inner product $a \cdot b$. Max-IP is very basic and serves as the base problem in the recent breakthrough of [Abboud et al., FOCS 2017] on hardness of approximation for polynomial-time problems. It is also used (implicitly) in the argument for hardness of exact ℓ_2-Furthest Pair (and other important problems in computational geometry) in poly-log-log dimensions in [Williams, SODA 2018]. We have three main results regarding this problem.

- **Characterization of Multiplicative Approximation**. First, we study the best multiplicative approximation ratio for Boolean Max-IP in sub-quadratic time. We show that, for Max-IP with two sets of n vectors from $\{0,1\}^d$, there is an $n^{2-\Omega(1)}$ time $(d/\log n)^{\Omega(1)}$-multiplicative-approximating algorithm, and we show this is conditionally optimal, as such a $(d/\log n)^{o(1)}$-approximating algorithm would refute SETH. Similar characterization is also achieved for additive approximation for Max-IP.
- **$2^{O(\log^* n)}$-dimensional Hardness for Exact Max-IP Over The Integers.** Second, we revisit the hardness of solving Max-IP exactly for vectors with integer entries. We show that, under SETH, for Max-IP with sets of n vectors from $\mathbb{Z}^d$ for some $d = 2^{O(\log^* n)}$, every exact algorithm requires $n^{2-o(1)}$ time. With the reduction from [Williams, SODA 2018], it follows that ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair in $2^{O(\log^* n)}$ dimensions require $n^{2-o(1)}$ time.
- **Connection with NP·UPP Communication Protocols.** Last, We establish a connection between conditional lower bounds for exact Max-IP with integer entries and NP·UPP communication protocols for Set-Disjointness, parallel to the connection between conditional lower bounds for approximating Max-IP and MA communication protocols for Set-Disjointness.

The lower bound in our first result is a direct corollary of the new MA protocol for Set-Disjointness introduced in [Rubinstein, STOC 2018], and our algorithms utilize the polynomial method and simple random sampling. Our second result follows from a new dimensionality self reduction from the Orthogonal Vectors problem for n vectors from $\{0,1\}^d$ to n vectors from $\mathbb{Z}^\ell$ where $\ell = 2^{O(\log^* d)}$, dramatically improving the previous reduction in [Williams, SODA 2018]. The key technical ingredient is a recursive application of *Chinese Remainder Theorem.*

As a side product, we obtain an MA communication protocol for Set-Disjointness with complexity $O\left(\sqrt{n \log n \log\log n}\right)$, slightly improving the $O\left(\sqrt{n}\log n\right)$ bound [Aaronson and Wigderson, TOCT 2009], and approaching the $\Omega(\sqrt{n})$ lower bound [Klauck, CCC 2003].

Moreover, we show that (under SETH) one can apply the $O(\sqrt{n})$ BQP communication protocol for Set-Disjointness to prove near-optimal hardness for approximation to Max-IP with vectors in $\{-1,1\}^d$. This answers a question from [Abboud et al., FOCS 2017] in the affirmative.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

[1] Supported by an Akamai Fellowship

33rd Computational Complexity Conference (CCC 2018).
Editor: Rocco A. Servedio; Article No. 14; pp. 14:1–14:45
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Keywords and phrases Maximum Inner Product, SETH, Hardness of Approximation in P, Fined-Grained Complexity, Hopcroft's Problem, Chinese Remainder Theorem

Digital Object Identifier 10.4230/LIPIcs.CCC.2018.14

Acknowledgements I would like to thank Ryan Williams for introducing the problem to me, countless encouragement and helpful discussions during this work, and also many comments on a draft of this paper. In particular, the idea of improving OV dimensionality self-reduction using CRT (the direct CRT based approach) is introduced to me by Ryan Williams.
I am grateful to Virginia Vassilevska Williams, Kaifeng Lyu, Peilin Zhong for helpful discussions and suggestions. I would like to thank Aviad Rubinstein for sharing a manuscript of his paper, and pointing out that the $O(\sqrt{n \log n \log\log n})$ MA protocol also works for Inner Product.

1 Introduction

We study the following fundamental problem from similarity search and statistics, which asks to find the most correlated pair in a dataset:

▶ **Definition 1.1** (Bichromatic Maximum Inner Product (Max-IP)). For $n, d \in \mathbb{N}$, the $\mathsf{Max\text{-}IP}_{n,d}$ problem is defined as: *given two sets A, B of vectors from $\{0,1\}^d$ compute*

$$\mathsf{OPT}(A, B) := \max_{a \in A, b \in B} a \cdot b.$$

We use $\mathbb{Z}$-$\mathsf{Max\text{-}IP}_{n,d}$ ($\mathbb{R}$-$\mathsf{Max\text{-}IP}_{n,d}$) to denote the same problem, but with A, B being sets of vectors from $\mathbb{Z}^d$ ($\mathbb{R}^d$).

Hardness of Approximation Max-IP

A natural brute-force algorithm solves Max-IP in $O(n^2 \cdot d)$-time. Assuming SETH[2], there is no $n^{2-\Omega(1)}$-time algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ when $d = \omega(\log n)$ [70].

Despite being one of the most central problems in similarity search and having numerous applications [47, 15, 61, 62, 65, 17, 16, 18, 57, 66, 68, 14, 49, 12, 67, 32, 31], until recently it was unclear whether there could be a near-linear time, 1.1-approximating algorithm, before the recent breakthrough of Abboud, Rubinstein and Williams [5].[3]

In [5], a framework for proving inapproximability results for problems in P is established (the distributed PCP framework), from which it follows:

▶ **Theorem 1.2** ([5]). *Assuming SETH, there is no $2^{(\log n)^{1-o(1)}}$-multiplicative-approximating $n^{2-\Omega(1)}$-time algorithm for $\mathsf{Max\text{-}IP}_{n,n^{o(1)}}$.*

Theorem 1.2 is an exciting breakthrough for hardness of approximation in P, implying other important inapproximability results for a host of problems including Bichromatic LCS Closest Pair Over Permutations, Approximate Regular Expression Matching, and Diameter in Product Metrics [5]. However, we still do not have a complete understanding of the approximation hardness of Max-IP yet. For instance, consider the following two concrete questions:

[2] SETH (Strong Exponential Time Hypothesis) states that for every $\varepsilon > 0$ there is a k such that k-SAT cannot be solved in $O((2-\varepsilon)^n)$ time [46].

[3] see [5] for a thorough discussion on the state of affairs on hardness of approximation in P before their work

▶ **Question 1.** *Is there a $(\log n)$-multiplicative-approximating $n^{2-\Omega(1)}$-time algorithm for Max-IP$_{n,\log^2 n}$? What about a 2-multiplicative-approximating algorithm for Max-IP$_{n,\log^2 n}$?*

▶ **Question 2.** *Is there a $(d/\log n)$-additive-approximating $n^{2-\Omega(1)}$-time algorithm for Max-IP$_{n,d}$?*

We note that the lower bound from [5] cannot answer Question 1. Tracing the details of their proofs, one can see that it only shows approximation hardness for dimension $d = \log^{\omega(1)} n$. Question 2 concerning additive approximation is not addressed at all by [5]. Given the importance of Max-IP, it is interesting to ask:

For what ratios r do $n^{2-\Omega(1)}$-time r-approximation algorithms exist for Max-IP?

Does the best-possible approximation ratio (in $n^{2-\Omega(1)}$ time) relate to the dimensionality, in some way?

In an important recent work, Rubinstein [64] improved the distributed PCP construction in a very crucial way, from which one can derive more refined lower bounds on approximating Max-IP. Building on its technique, in this paper we provide full *characterizations*, determining essentially optimal multiplicative approximations and additive approximations to Max-IP, under SETH.

Hardness of Exact $\mathbb{Z}$-Max-IP

Recall that from [70], there is no $n^{2-\Omega(1)}$-time algorithm for exact Boolean Max-IP$_{n,\omega(\log n)}$. Since in real life applications of similarity search, one often deals with real-valued data instead of just Boolean data, it is natural to ask about $\mathbb{Z}$-Max-IP (which is certainly a special case of $\mathbb{R}$-Max-IP): what is the maximum d such that $\mathbb{Z}$-Max-IP$_{n,d}$ can be solved exactly in $n^{2-\Omega(1)}$ time?

Besides being interesting in its own right, there are also reductions from $\mathbb{Z}$-Max-IP to ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair. Hence, lower bounds for $\mathbb{Z}$-Max-IP imply lower bounds for these two famous problems in computational geometry (see [72] for a discussion on this topic).

Prior to our work, it was implicitly shown in [72] that:

▶ **Theorem 1.3** ([72]). *There is no $n^{2-\Omega(1)}$-time algorithm for $\mathbb{Z}$-Max-IP$_{n,\omega((\log\log n)^2)}$ with vectors of $O(\log n)$-bit entries, assuming SETH.*

However, the best known algorithm for $\mathbb{Z}$-Max-IP runs in $n^{2-\Theta(1/d)}$ time [55, 11, 74][4], hence there is still a gap between the lower bound and the best known upper bounds. To confirm these algorithms are in fact optimal, we would like to prove a lower bound with $\omega(1)$ dimensions.

In this paper, we significantly strength the previous lower bound from $\omega((\log\log n)^2)$ dimensions to $2^{O(\log^* n)}$ dimensions ($2^{O(\log^* n)}$ is an *extremely slow-growing* function, see preliminaries for its formal definition).

[4] [11, 74] are for ℓ_2-Furthest Pair or Bichromatic ℓ_2-Closest Pair. They also work for $\mathbb{Z}$-Max-IP as there are reductions from $\mathbb{Z}$-Max-IP to these two problems, see [72] or Lemma 4.5 and Lemma 4.6.

Fine-Grained Complexity and Communication Complexity

One intriguing aspect of the distributed PCP framework is that it makes use of the $\widetilde{O}(\sqrt{n})$ MA communication protocol for Set-Disjointness [1]. Several follow-up works [50, 64] explored this connection further, and settled the hardness of approximation to several fundamental problems (under SETH).

Given the success of the interplay between these two seemingly unrelated fields, it is natural to seek more results from it. In particular, it is asked in [5] whether the $O(\sqrt{n})$ BQP communication protocol for Set-Disjointness can be utilized.

In this paper, we answer the question affirmatively by showing that BQP communication protocol implies hardness for approximation to $\{-1,1\}$-Max-IP[5]. Moreover, we also establish a connection between $\mathbb{Z}$-Max-IP lower bounds and $\mathsf{NP} \cdot \mathsf{UPP}$ communication protocols for Set-Disjointness, which suggests a new perspective on our results on $\mathbb{Z}$-Max-IP.

1.1 Our Results

We use $\mathsf{OV}_{n,d}$ to denote the Orthogonal Vectors problem: given two sets of vectors A, B each consisting of n vectors from $\{0,1\}^d$, determine whether there are $a \in A$ and $b \in B$ such that $a \cdot b = 0$.[6] Similarly, we use $\mathbb{Z}$-$\mathsf{OV}_{n,d}$ to denote the same problem except for that A, B consists of vectors from $\mathbb{Z}^d$ (which is also called Hopcroft's problem).

All our results are based on the following widely used conjecture about OV:

▶ **Conjecture 1.4** (Orthogonal Vectors Conjecture (OVC) [70, 7]). *For every $\varepsilon > 0$, there exists a $c \geq 1$ such that $\mathsf{OV}_{n,d}$ requires $n^{2-\varepsilon}$ time when $d = c \log n$.*

OVC is a plausible conjecture as it is implied by the popular Strong Exponential Time Hypothesis [46, 29] on the time complexity of solving k-SAT [70, 73].

Characterizations of Hardness of Approximate Max-IP

The first main result of our paper characterizes when there is a truly sub-quadratic time ($n^{2-\Omega(1)}$ time, for some universal constant hidden in the big-Ω) t-multiplicative-approximating algorithm for Max-IP, and characterizes the best-possible additive approximations as well. We begin with formal definitions of these two standard types of approximation:

- We say an algorithm $\mathbb{A}$ for Max-$\mathsf{IP}_{n,d}$ ($\mathbb{Z}$-Max-$\mathsf{IP}_{n,d}$) is t-multiplicative-approximating, if for all A, B, $\mathbb{A}$ outputs a value $\widetilde{\mathsf{OPT}}(A,B) \in [\mathsf{OPT}(A,B), \mathsf{OPT}(A,B) \cdot t]$.
- We say an algorithm $\mathbb{A}$ for Max-$\mathsf{IP}_{n,d}$ ($\mathbb{Z}$-Max-$\mathsf{IP}_{n,d}$) is t-additive-approximating, if for all A, B, $\mathbb{A}$ outputs a value $\widetilde{\mathsf{OPT}}(A,B)$ such that $|\widetilde{\mathsf{OPT}}(A,B) - \mathsf{OPT}(A,B)| \leq t$.
- To avoid ambiguity, we call an algorithm computing $\mathsf{OPT}(A,B)$ exactly an *exact* algorithm for Max-$\mathsf{IP}_{n,d}$ ($\mathbb{Z}$-Max-$\mathsf{IP}_{n,d}$).

Multiplicative Approximations for Max-IP. In the multiplicative case, our characterization (formally stated below) basically says that there is a t-multiplicative-approximating $n^{2-\Omega(1)}$-time algorithm for Max-$\mathsf{IP}_{n,d}$ if and only if $t = (d/\log n)^{\Omega(1)}$. Note that in the following theorem we require $d = \omega(\log n)$, since in the case of $d = O(\log n)$, there *are* $n^{2-\varepsilon}$-time algorithms for exact Max-$\mathsf{IP}_{n,d}$ [14, 13].

[5] That is, Max-IP with sets A and B being n vectors from $\{-1,1\}^d$.

[6] Here we use the bichromatic version of OV instead of the monochromatic one for convenience, as they are equivalent.

▶ **Theorem 1.5.** *Letting $\omega(\log n) < d < n^{o(1)}$ and $t \ge 2$,*[7] *the following holds:*

1. *There is an $n^{2-\Omega(1)}$-time t-multiplicative-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if*

$$t = (d/\log n)^{\Omega(1)},$$

and under SETH (or OVC), there is no $n^{2-\Omega(1)}$-time t-multiplicative-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if

$$t = (d/\log n)^{o(1)}.$$

2. *Moreover, let $\varepsilon = \min\left(\frac{\log t}{\log(d/\log n)}, 1\right)$. There are t-multiplicative-approximating deterministic algorithms for $\mathsf{Max\text{-}IP}_{n,d}$ running in time*

$$O\left(n^{2+o(1)-0.31\cdot\frac{1}{\varepsilon^{-1}+\frac{0.31}{2}}}\right) = O\left(n^{2+o(1)-\Omega(\varepsilon)}\right)$$

or time

$$O\left(n^{2-0.17\cdot\frac{1}{\varepsilon^{-1}+\frac{0.17}{2}}} \cdot \mathrm{polylog}(n)\right) = O\left(n^{2-\Omega(\varepsilon)} \cdot \mathrm{polylog}(n)\right).$$

▶ Remark 1.6. The first algorithm is slightly faster, but only sub-quadratic when $\varepsilon = \Omega(1)$, while the second algorithm still gets a non-trivial speed up over the brute force algorithm as long as $\varepsilon = \omega(\log\log n/\log n)$.

We remark here that the above algorithms indeed work for the case where the sets consisting of non-negative reals (i.e., $\mathbb{R}^+$-Max-IP):

▶ **Corollary 1.7.** *Assume $\omega(\log n) < d < n^{o(1)}$ and let $\varepsilon = \min(\frac{\log t}{\log(d/\log n)}, 1)$. There is a t-multiplicative-approximating deterministic algorithm for $\mathbb{R}^+$-$\mathsf{Max\text{-}IP}_{n,d}$ running in time*

$$O\left(n^{2-\Omega(\varepsilon)} \cdot \mathrm{polylog}(n)\right).$$

The lower bound is a direct corollary of the new improved MA protocols for Set-Disjointness from [64], which is based on Algebraic Geometry codes. Together with the framework of [5], that MA-protocol implies a reduction from OV to approximating Max-IP.

Our upper bounds are application of the polynomial method [71, 9]: defining appropriate sparse polynomials for approximating Max-IP on small groups of vectors, and use fast matrix multiplication to speed up the evaluation of these polynomials on many pairs of points.

Via the known reduction from Max-IP to LCS-Pair in [5], we also obtain a more refined lower bound for approximating the LCS Closest Pair problem (defined below).

▶ **Definition 1.8** (LCS Closest Pair). The $\mathsf{LCS\text{-}Closest\text{-}Pair}_{n,d}$ problem is: *given two sets A, B of n strings from Σ^d (Σ is a finite alphabet), determine*

$$\max_{a\in A, b\in B} \mathsf{LCS}(a, b),$$

where $\mathsf{LCS}(a, b)$ is the length of the longest common subsequence of strings a and b.

▶ **Corollary 1.9** (Improved Inapproximability for LCS-Closest-Pair). *Assuming SETH (or OVC), for every $t \ge 2$, t-multiplicative-approximating $\mathsf{LCS\text{-}Closest\text{-}Pair}_{n,d}$ requires $n^{2-o(1)}$ time, if $d = t^{\omega(1)} \cdot \log^5 n$.*

[7] Note that t and d are both functions of n, we assume they are computable in $n^{o(1)}$ time throughout this paper for simplicity.

Additive Approximations for Max-IP. Our characterization for additive approximations to Max-IP says that there is a t-additive-approximating $n^{2-\Omega(1)}$-time algorithm for $\textsf{Max-IP}_{n,d}$ if and only if $t = \Omega(d)$.

▶ **Theorem 1.10.** *Letting $\omega(\log n) < d < n^{o(1)}$ and $0 \le t \le d$, the following holds:*

1. *There is an $n^{2-\Omega(1)}$-time t-additive-approximating algorithm for $\textsf{Max-IP}_{n,d}$ if*

$$t = \Omega(d),$$

and under SETH (or OVC), there is no $n^{2-\Omega(1)}$-time t-additive-approximating algorithm for $\textsf{Max-IP}_{n,d}$ if

$$t = o(d).$$

2. *Moreover, letting $\varepsilon = \dfrac{t}{d}$, there is a randomized*

$$O\left(n^{2-\Omega(\varepsilon^{1/3}/\log \varepsilon^{-1})}\right)$$

time, t-additive-approximating algorithm for $\textsf{Max-IP}_{n,d}$ when $\varepsilon \gg \log^6 \log n / \log^3 n$.

The lower bound above is already established in [64], while the upper bound works by reducing the problem to the $d = O(\log n)$ case via random-sampling coordinates, and solving the reduced problem via known methods [14, 13].

▶ Remark 1.11. We want to remark here that the lower bounds for approximating Max-IP are direct corollaries of the new MA protocols for Set-Disjointness in [64]. Our main contribution is providing the complementary *upper bounds* to show that these lower bounds are indeed *tight* assuming SETH.

All-Pair-Max-IP. Finally, we remark that our algorithms (with slight adaptions) also work for the following stronger problem[8]: $\textsf{All-Pair-Max-IP}_{n,d}$, in which we are given two sets A and B of n vectors from $\{0,1\}^d$, and for each $x \in A$ we must compute $\textsf{OPT}(x, B) := \max\limits_{y \in B} x \cdot y$. An algorithm is t-multiplicative-approximating (additive-approximating) for All-Pair-Max-IP if for all $\textsf{OPT}(x, B)$'s, it computes corresponding approximating answers.

▶ **Corollary 1.12.** *Suppose $\omega(\log n) < d < n^{o(1)}$, and let*

$$\varepsilon_M := \min\left(\frac{\log t}{\log(d/\log n)}, 1\right) \quad \text{and } \varepsilon_A := \frac{\min(t, d)}{d}.$$

There is an $n^{2-\Omega(\varepsilon_M)}\,\mathrm{polylog}(n)$ time t-multiplicative-approximating algorithm and an $n^{2-\Omega(\varepsilon_A^{1/3}/\log \varepsilon_A^{-1})}$ time t-additive-approximating algorithm for $\textsf{All-Pair-Max-IP}_{n,d}$, when $\varepsilon_A \gg \log^6 \log n / \log^3 n$.

[8] Since All-Pair-Max-IP is stronger than Max-IP, lower bounds for Max-IP automatically apply for All-Pair-Max-IP.

BQP Communication Protocols and Approximate {-1,1}-Max-IP.

Making use of the $O(\sqrt{n})$-degree approximate polynomial for OR [27, 36], we also give a completely different proof for the hardness of multiplicative approximation to $\{-1,1\}$-Max-IP. Lower bound from that approach is inferior to Theorem 1.5: in particular, *it cannot achieve a characterization.*

It is asked in [5] that whether we can make use of the $O(\sqrt{n})$ BQP communication protocol for Set-Disjointness [28] to prove conditional lower bounds. Indeed, that quantum communication protocol is based on the $O(\sqrt{n})$-time quantum query algorithm for OR (Grover's algorithm [42]), which induces the needed approximate polynomial for OR. Hence, the following theorem in some sense answers their question in the affirmative:

▶ **Theorem 1.13** (Informal). *Assuming SETH (or OVC), there is no $n^{2-\Omega(1)}$ time $n^{o(1)}$-multiplicative-approximating algorithm for $\{-1,1\}$-Max-IP$_{n,n^{o(1)}}$.*

The full statement can be found in Theorem C.1 and Theorem C.2.

Hardness of Exact $\mathbb{Z}$-Max-IP in $2^{O(\log^* n)}$ Dimensions

Now we turn to discuss our results on $\mathbb{Z}$-Max-IP. We show that $\mathbb{Z}$-Max-IP is hard to solve in $n^{2-\Omega(1)}$ time, even with $2^{O(\log^* n)}$-dimensional vectors:

▶ **Theorem 1.14.** *Assuming SETH (or OVC), there is a constant c such that any exact algorithm for $\mathbb{Z}$-Max-IP$_{n,d}$ for $d = c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

As direct corollaries of the above theorem, using reductions implicit in [72], we also conclude hardness for ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair under SETH (or OVC) in $2^{O(\log^* n)}$ dimensions.

▶ **Theorem 1.15** (Hardness of ℓ_2-Furthest Pair in $c^{\log^* n}$ Dimensions). *Assuming SETH (or OVC), there is a constant c such that ℓ_2-Furthest Pair in $c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

▶ **Theorem 1.16** (Hardness of Bichromatic ℓ_2-Closest Pair in $c^{\log^* n}$ Dimensions). *Assuming SETH (or OVC), there is a constant c such that Bichromatic ℓ_2-Closest Pair in $c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

The above lower bounds on ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair are in sharp contrast with the case of *ℓ_2-Closest Pair*, which can be solved in $2^{O(d)} \cdot n \log^{O(1)} n$ time [23, 51, 37].

Improved Dimensionality Reduction for OV and Hopcroft's Problem

Our hardness of $\mathbb{Z}$-Max-IP is established by a reduction from Hopcroft's problem, whose hardness is in turn derived from the following significantly improved dimensionality reduction for OV.

▶ **Lemma 1.17** (Improved Dimensionality Reduction for OV). *Let $1 \le \ell \le d$. There is an*

$$O\left(n \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))} \cdot \mathrm{poly}(d)\right)\text{-time}$$

reduction from OV$_{n,d}$ to $\ell^{O(6^{\log^* d} \cdot (d/\ell))}$ instances of $\mathbb{Z}$-OV$_{n,\ell+1}$, with vectors of entries with bit-length $O\left(d/\ell \cdot \log \ell \cdot 6^{\log^* d}\right)$.

Comparison with [72]. Comparing to the old construction in [72], our reduction here is more efficient when ℓ is much smaller than d (which is the case we care about). That is, in [72], $\mathsf{OV}_{n,d}$ can be reduced to $d^{d/\ell}$ instances of $\mathbb{Z}$-$\mathsf{OV}_{n,\ell+1}$, while we get $\left\{\ell^{6^{\log^* d}}\right\}^{d/\ell}$ instances in our improved one. So, for example, when $\ell = 7^{\log^* d}$, the old reduction yields $d^{d/7^{\log^* d}} = n^{\omega(1)}$ instances (recall that $d = c \log n$ for an arbitrary constant c), while our improved one yields only $n^{o(1)}$ instances, each with $2^{O(\log^* n)}$ dimensions.

From Lemma 1.17, the following theorem follows in the same way as in [72].

▶ **Theorem 1.18** (Hardness of Hopcroft's Problem in $c^{\log^* n}$ Dimensions). *Assuming SETH (or OVC), there is a constant c such that* $\mathbb{Z}$-$\mathsf{OV}_{n,c^{\log^* n}}$ *with vectors of* $O(\log n)$*-bit entries requires* $n^{2-o(1)}$ *time.*

Connection between $\mathbb{Z}$-Max-IP Lower Bounds and NP · UPP Communication Protocols

We also show a new connection between $\mathbb{Z}$-Max-IP and a special type of communication protocol. Let us first recall the Set-Disjointness problem:

▶ **Definition 1.19** (Set-Disjointness). Let $n \in \mathbb{N}$, in Set-Disjointness (DISJ_n), Alice holds a vector $X \in \{0,1\}^n$, Bob holds a vector $Y \in \{0,1\}^n$, and they want to determine whether $X \cdot Y = 0$.

In [5], the hardness of approximating Max-IP is established via a connection to MA communication protocols (in particular, a fast MA communication protocol for Set-Disjointness). Our lower bound for (exact) $\mathbb{Z}$-Max-IP can also be connected to similar NP · UPP protocols (note that MA = NP · promiseBPP).

Formally, we define NP · UPP protocols as follows:

▶ **Definition 1.20.** For a problem Π with inputs x, y of length n (Alice holds x and Bob holds y), we say a communication protocol is an (m, ℓ)*-efficient* NP · UPP *communication protocol* if the following holds:

- There are three parties Alice, Bob and Merlin in the protocol.
- Merlin sends Alice and Bob an advice string z of length m, which is a function of x and y.
- Given y and z, Bob sends Alice ℓ bits, and Alice decides to accept or not.[9] They have an unlimited supply of private random coins (not public, which is important) during their conversation. The following conditions hold:
  - If $\Pi(x, y) = 1$, then there is an advice z from Merlin such that Alice accepts with probability $\geq 1/2$.
  - Otherwise, for all possible advice strings from Merlin, Alice accepts with probability $< 1/2$.

Moreover, we say the protocol is (m, ℓ)-computational-efficient, if in addition the probability distributions of both Alice and Bob's behavior can be computed in poly(n) time given their input and the advice.

Our new reduction from OV to Max-IP actually implies a super-efficient NP · UPP protocol for Set-Disjointness.

[9] In UPP, actually one-way communication is equivalent to the seemingly more powerful one in which they communicate [60].

▶ **Theorem 1.21.** *For all $1 \le \alpha \le n$, there is an*

$$\left(\alpha \cdot 6^{\log^* n} \cdot (n/2^{\alpha}), O(\alpha)\right)\text{-computational-efficient}$$

$\mathsf{NP} \cdot \mathsf{UPP}$ *communication protocol for* DISJ_n.

For example, when $\alpha = 3\log^* n$, Theorem 1.21 implies there is an $O(o(n), O(\log^* n))$-computational-efficient $\mathsf{NP} \cdot \mathsf{UPP}$ communication protocol for DISJ_n. Moreover, we show that if the protocol of Theorem 1.21 can be improved a little bit (like removing the $6^{\log^* n}$ term), we would obtain the desired hardness for $\mathbb{Z}$-$\mathsf{Max\text{-}IP}$ in $\omega(1)$-dimensions.

▶ **Theorem 1.22.** *Assuming SETH (or OVC), if there is an increasing and unbounded function f such that for all $1 \le \alpha \le n$, there is an*

$$(n/f(\alpha), \alpha)\text{-computational-efficient}$$

$\mathsf{NP} \cdot \mathsf{UPP}$ *communication protocol for* DISJ_n*, then* $\mathbb{Z}$-$\mathsf{Max\text{-}IP}_{n,\omega(1)}$ *requires* $n^{2-o(1)}$ *time with vectors of* $\operatorname{polylog}(n)$*-bit entries. The same holds for* ℓ_2*-Furthest Pair and Bichromatic* ℓ_2*-Closest Pair.*

Improved MA Protocols for Set-Disjointness

Finally, we also obtain a new MA protocol for Set-Disjointness, which improves on the previous $O(\sqrt{n}\log n)$ protocol in [1], and is closer to the $\Omega(\sqrt{n})$ lower bound by [52]. Like the protocol in [1], our new protocol also works for the following slightly harder problem Inner Product.

▶ **Definition 1.23** (Inner Product). Let $n \in \mathbb{N}$, in Inner Product (IP_n), Alice holds a vector $X \in \{0,1\}^n$, Bob holds a vector $Y \in \{0,1\}^n$, and they want to compute $X \cdot Y$.

▶ **Theorem 1.24.** *There is an* MA *protocol for* DISJ_n *and* IP_n *with communication complexity*

$$O\left(\sqrt{n \log n \log\log n}\right).$$

In [64], the author asked whether the MA communication complexity of DISJ (IP) is $\Theta(\sqrt{n})$ or $\Theta(\sqrt{n\log n})$, and suggested that $\Omega(n \log n)$ may be necessary for IP. Our result makes progress on that question by showing that the true complexity lies between $\Theta(\sqrt{n})$ and $\Theta(\sqrt{n\log n\log\log n})$.

1.2 Intuition for Dimensionality Self Reduction for OV

The $2^{O(\log^* n)}$ factor in Lemma 1.17 is not common in theoretical computer science[10], and our new reduction for OV is considerably more complicated than the polynomial-based construction from [72]. Hence, it is worth discussing the intuition behind Lemma 1.17, and the reason why we get a factor of $2^{O(\log^* n)}$.

[10] Other examples include an $O\left(2^{O(\log^* n)} n^{4/3}\right)$ time algorithm for $\mathbb{Z}$-$\mathsf{OV}_{n,3}$ [56], $O\left(2^{O(\log^* n)} n\log n\right)$ time algorithms (Fürer's algorithm with its modifications) for Fast Integer Multiplication [38, 34, 43] and an old $O(n^{d/2} 2^{O(\log^* n)})$ time algorithm for Klee's measure problem [30].

A Direct Chinese Remainder Theorem Based Approach. We first discuss a direct reduction based on the *Chinese Remainder Theorem* (CRT) (see Theorem 2.5 for a formal definition). CRT says that given a collection of primes $q_1, \dots, q_b$, and a collection of integers $r_1, \dots, r_b$, there exists a unique integer $t = \mathsf{CRR}(\{r_i\}; \{q_i\})$ such that $t \equiv r_i \pmod{q_i}$ for each $i \in [b]$ (CRR stands for *Chinese Remainder Representation*).

Now, let $b, \ell \in \mathbb{N}$, suppose we would like to have a dimensionality reduction φ from $\{0,1\}^{b \cdot \ell}$ to $\mathbb{Z}^{\ell}$. We can partition an input $x \in \{0,1\}^{b \cdot \ell}$ into ℓ blocks, each of length b, and represent each block via CRT: that is, for a block $z \in \{0,1\}^{b}$, we map it into a single integer $\varphi_{\mathsf{block}}(z) := \mathsf{CRR}(\{z_i\}; \{q_i\})$, and the concatenations of φ_{block} over all blocks of x is $\varphi(x) \in \mathbb{Z}^{\ell}$.

The key idea here is that, for $z, z' \in \{0,1\}^{b}$, $\varphi_{\mathsf{block}}(z) \cdot \varphi_{\mathsf{block}}(z') \pmod{q_i}$ is simply $z_i \cdot z'_i$. That is, the multiplication between two *integers* $\varphi_{\mathsf{block}}(z) \cdot \varphi_{\mathsf{block}}(z')$ simulates the coordinate-wise multiplication between two *vectors* z and z'!

Therefore, if we make all primes q_i larger than ℓ, we can in fact determine $x \cdot y$ from $\varphi(x) \cdot \varphi(y)$, by looking at $\varphi(x) \cdot \varphi(y) \pmod{q_i}$ for each i. That is,

$$x \cdot y = 0 \Leftrightarrow \varphi(x) \cdot \varphi(y) \equiv 0 \pmod{q_i} \quad \text{for all } i.$$

Hence, let V be the set of all integer $0 \le v \le \ell \cdot \left(\prod_{i=1}^{b} q_i\right)^2$ that $v \equiv 0 \pmod{q_i}$ for all $i \in [b]$, we have

$$x \cdot y = 0 \Leftrightarrow \varphi(x) \cdot \varphi(y) \in V.$$

The reduction is completed by enumerating all integers $v \in V$, and appending corresponding values to make $\varphi_A(x) = [\varphi(x), -1]$ and $\varphi_B(y) = [\varphi(y), v]$ (this step is from [72]).

Note that a nice property for φ is that each $\varphi(x)_i$ only depends on the i-th block of x, and the mapping is the same on each block (φ_{block}); we call this the *block mapping property.*

Analysis of the Direct Reduction. To continue building intuition, let us analyze the above reduction. The size of V is the number of $\mathbb{Z}\text{-}\mathsf{OV}_{n,\ell+1}$ instances we create, and $|V| \ge \prod_{i=1}^{b} q_i$. These primes q_i have to be all distinct, and it follows that $\prod_{i=1}^{b} q_i$ is $b^{\Theta(b)}$. Since we want to create at most $n^{o(1)}$ instances (or n^{ε} for arbitrarily small ε), we need to set $b \le \log n / \log\log n$. Moreover, to base our hardness on OVC which deals with $c \log n$-dimensional vectors, we need to set $b \cdot \ell = d = c \cdot \log n$ for an arbitrary constant c. Therefore, we must have $\ell \ge \log\log n$, and the above reduction only obtains the same hardness result as [72].

Key Observation: "Most Space Modulo q_i" is Actually Wasted. To improve the above reduction, we need to make $|V|$ smaller. Our key observation about φ is that, for the primes q_i's, they are mostly larger than $b \gg \ell$, but $\varphi(x) \cdot \varphi(y) \in \{0, 1, \dots, \ell\} \pmod{q_i}$ for all these q_i's. Hence, *"most space modulo q_i" is actually wasted.*

Make More "Efficient" Use of the "Space": Recursive Reduction. Based on the previous observation, we want to use the "space modulo q_i" more efficiently. It is natural to consider a *recursive reduction.* We will require all our primes q_i's to be larger than b. Let b_{micro} be a very small integer compared to b, and let $\psi : \{0,1\}^{b_{\mathsf{micro}} \cdot \ell} \to \mathbb{Z}^{\ell}$ with a set V_{ψ} and a block mapping ψ_{block} be a similar reduction on a much smaller input: for $x, y \in \{0,1\}^{b_{\mathsf{micro}} \cdot \ell}$, $x \cdot y = 0 \Leftrightarrow \psi(x) \cdot \psi(y) \in V_{\psi}$. We also require here that $\psi(x) \cdot \psi(y) \le b$ for all x and y.

For an input $x \in \{0,1\}^{b \cdot \ell}$ and a block $z \in \{0,1\}^b$ of x, our key idea is to partition z again into b/b_{micro} "micro" blocks each of size b_{micro}. And for a block z in x, let $z^1, \dots, z^{b/b_{\mathsf{micro}}}$ be its b/b_{micro} micro blocks, we map z into an integer $\varphi_{\mathsf{block}}(z) := \mathsf{CRR}(\{\psi_{\mathsf{block}}(z_i)\}_{i=1}^{b/b_{\mathsf{micro}}}; \{q_i\}_{i=1}^{b/b_{\mathsf{micro}}})$.

Now, given two blocks $z, z' \in \{0,1\}^b$, we can see that

$$\varphi_{\mathsf{block}}(z) \cdot \varphi_{\mathsf{block}}(z') \equiv \psi_{\mathsf{block}}(z_i) \cdot \psi_{\mathsf{block}}(z'_i) \pmod{q_i}.$$

That is, $\varphi(x) \cdot \varphi(y) \pmod{q_i}$ in fact is equal to $\psi(x^{[i]}) \cdot \psi(y^{[i]})$, where $x^{[i]}$ is the concatenation of the i-th micro blocks of x in each block, and $y^{[i]}$ is defined similarly. Hence, we can determine whether $x^{[i]} \cdot y^{[i]} = 0$ from $\varphi(x) \cdot \varphi(y) \pmod{q_i}$ for all i, and therefore also determine whether $x \cdot y = 0$ from $\varphi(x) \cdot \varphi(y)$.

We can now observe that $|V| \le b^{\Theta(b/b_{\mathsf{micro}})}$, smaller than before; thus we get an improvement, depending on how large can b_{micro} be. Clearly, the reduction ψ can also be constructed from even smaller reductions, and after recursing $\Theta(\log^* n)$ times, we can switch to the direct construction discussed before. By a straightforward (but tedious) calculation, we can derive Lemma 1.17.

High-Level Explanation on the $2^{O(\log^* n)}$ Factor. Ideally, we want to have a reduction from OV to $\mathbb{Z}$-OV with only $\ell^{O(b)}$ instances, in other words, we want $|V| = \ell^{O(b)}$. The reason we need to pay an extra $2^{O(\log^* n)}$ factor in the exponent is as follows:

In our reduction, $|V|$ is at least $\prod_{i=1}^{b/b_{\mathsf{micro}}} q_i$, which is also the bound on each coordinate of the reduction: $\psi(x)_i$ equals to a CRR encoding of a vector with $\{q_i\}_{i=1}^{b/b_{\mathsf{micro}}}$, whose value can be as large as $\prod_{i=1}^{b/b_{\mathsf{micro}}} q_i - 1$. That is, all we want is to control the upper bound on the coordinates of the reduction.

Suppose we are constructing an "outer" reduction $\varphi : \{0,1\}^{b \cdot \ell} \to \mathbb{Z}^\ell$ from the "micro" reduction $\psi : \{0,1\}^{b_{\mathsf{micro}} \cdot \ell} \to \mathbb{Z}^\ell$ with coordinate upper bound L_ψ ($\psi(x)_i \le L_\psi$), and let $L_\psi = \ell^{\kappa \cdot b_{\mathsf{micro}}}$ (that is, κ is the extra factor comparing to the ideal case). Recall that we have to ensure $q_i > \psi(x) \cdot \psi(y)$ to make our construction work, and therefore we have to set q_i larger than L_ψ^2.

Then the coordinate upper bound for φ becomes $L_\varphi = \prod_{i=1}^{b/b_{\mathsf{micro}}} q_i \ge (L_\psi)^{2 \cdot b/b_{\mathsf{micro}}} = \ell^{2\kappa \cdot b}$. Therefore, we can see that after one recursion, the "extra factor" κ at least doubles. Since our recursion proceeds in $\Theta(\log^* n)$ rounds, we have to pay an extra $2^{O(\log^* n)}$ factor on the exponent.

1.3 Related Works

SETH-based Conditional Lower Bound. SETH is one of the most fruitful conjectures in the Fine-Grained Complexity. There are numerous conditional lower bounds based on it for problems in P among different areas, including: dynamic data structures [58, 6, 10, 44, 53, 3, 45, 41], computational geometry [25, 72, 35], pattern matching [7, 21, 22, 26, 24], graph algorithms [63, 40, 8, 54]. See [69] for a very recent survey on SETH-based lower bounds (and more).

Communication Complexity and Conditional Hardness. The connection between communication protocols (in various model) for Set-Disjointness and SETH dates back at least to [59], in which it is shown that a sub-linear, computational efficient protocol for 3-party Number-On-Forehead Set-Disjointness problem would refute SETH. And it is worth mentioning that [4]'s result builds on the $\widetilde{O}(\log n)$ IP communication protocol for Set-Disjointness in [1].

Distributed PCP. Using Algebraic Geometry codes, [64] obtains a better MA protocol, which in turn improves the efficiency of the previous distributed PCP construction of [5]. He then shows the $n^{2-o(1)}$ time hardness for $1+o(1)$-approximation to Bichromatic Closest Pair and $o(d)$-additive approximation to $\textsf{Max-IP}_{n,d}$ with this new technique.

[50] use the Distributed PCP framework to derive inapproximability results for k-Dominating Set under various assumptions. In particular, building on the techniques of [64], it is shown that under SETH, k-Dominating Set has no $(\log n)^{1/\operatorname{poly}(k,e(\varepsilon))}$ approximation in $n^{k-\varepsilon}$ time[11].

Hardness of Approximation in P. Making use of Chebychev embeddings, [12] prove a $2^{\Omega\left(\frac{\sqrt{\log n}}{\log\log n}\right)}$ inapproximability lower bound on $\{-1,1\}$-Max-IP.[12] [2] take an approach different from Distributed PCP, and shows that under certain complexity assumptions, LCS does not have a *deterministic* $1+o(1)$-approximation in $n^{2-\varepsilon}$ time. They also establish a connection with circuit lower bounds and show that the existence of such a *deterministic* algorithm implies E^{NP} does not have non-uniform linear-size Valiant Series Parallel circuits. In [4], it is improved to that any constant factor approximation deterministic algorithm for LCS in $n^{2-\varepsilon}$ time implies that E^{NP} does not have non-uniform linear-size NC^1 circuits. See [5] for more related results in hardness of approximation in P.

Organization of the Paper

In Section 2, we introduce the needed preliminaries for this paper. In Section 3, we prove our characterizations for approximating Max-IP and other related results. In Section 4, we prove $2^{O(\log^* n)}$ dimensional hardness for $\mathbb{Z}$-Max-IP and other related problems. In Section 5, we establish the connection between $\mathsf{NP}\cdot\mathsf{UPP}$ communication protocols and SETH-based lower bounds for exact $\mathbb{Z}$-Max-IP. In Section 6, we present the $O\left(\sqrt{n\log n\log\log n}\right)$ MA protocol for Set-Disjointness.

2 Preliminaries

We begin by introducing some notation. For an integer d, we use $[d]$ to denote the set of integers from 1 to d. For a vector u, we use u_i to denote the i-th element of u.

We use $\log(x)$ to denote the logarithm of x with respect to base 2 with ceiling as appropriate, and $\ln(x)$ to denote the natural logarithm of x.

In our arguments, we use the iterated logarithm function $\log^*(n)$, which is defined recursively as follows:

$$\log^*(n) := \begin{cases} 0 & n \le 1; \\ \log^*(\log n) + 1 & n > 1. \end{cases}$$

[11] where $e : \mathbb{R}^+ \to \mathbb{N}$ is some function
[12] which is improved by Theorem 1.13

2.1 Fast Rectangular Matrix Multiplication

Similar to previous algorithms using the polynomial method, our algorithms make use of the algorithms for fast rectangular matrix multiplication.

▶ **Theorem 2.1** ([39]). *There is an $N^{2+o(1)}$ time algorithm for multiplying two matrices A and B with size $N \times N^{\alpha}$ and $N^{\alpha} \times N$, where $\alpha > 0.31389$.*

▶ **Theorem 2.2** ([33]). *There is an $N^2 \cdot \operatorname{polylog}(N)$ time algorithm for multiplying two matrices A and B with size $N \times N^{\alpha}$ and $N^{\alpha} \times N$, where $\alpha > 0.172$.*

2.2 Number Theory

Here we recall some facts from number theory. In our reduction from OV to $\mathbb{Z}$-OV, we will apply the famous prime number theorem, which supplies a good estimate of the number of primes smaller than a certain number. See e.g. [19] for a reference on this.

▶ **Theorem 2.3** (Prime Number Theorem). *Let $\pi(n)$ be the number of primes $\le n$, then we have*

$$\lim_{n\to\infty} \frac{\pi(n)}{n/\ln n} = 1.$$

From a simple calculation, we obtain:

▶ **Lemma 2.4.** *There are $10n$ distinct primes in $[n+1, n^2]$ for a large enough n.*

Proof. For a large enough n, from the prime number theorem, the number of primes in $[n+1, n^2]$ is equal to

$$\pi(n^2) - \pi(n) \sim n^2/2\ln n - n/\ln n \gg 10n. \qquad ◀$$

Next we recall the Chinese remainder theorem, and Chinese remainder representation.

▶ **Theorem 2.5.** *Given d pairwise co-prime integers $q_1, q_2, \dots, q_d$, and d integers $r_1, r_2, \dots, r_d$, there is exactly one integer $0 \le t < \prod_{i=1}^{d} q_i$ such that*

$$t \equiv r_i \pmod{q_i} \quad \textit{for all } i \in [d].$$

We call this t the Chinese remainder representation (or the CRR encoding) of the r_i's (with respect to these q_i's). We also denote

$$t = \mathsf{CRR}(\{r_i\}; \{q_i\})$$

for convenience. We sometimes omit the sequence $\{q_i\}$ for simplicity, when it is clear from the context.

Moreover, t can be computed in polynomial time with respect to the total bits of all the given integers.

2.3 Communication Complexity

In our paper we will make use of a certain kind of MA protocol, we call them (m, r, ℓ, s)-efficient protocols[13].

[13] Our notations here are adopted from [50]. They also defined similar k-party communication protocols, while we only discuss 2-party protocols in this paper.

▶ **Definition 2.6.** We say an MA Protocol is (m, r, ℓ, s)-efficient for a communication problem, if in the protocol:

- There are three parties Alice, Bob and Merlin in the protocol, Alice holds input x and Bob holds input y.
- Merlin sends an advice string z of length m to Alice, which is a function of x and y.
- Alice and Bob jointly toss r coins to obtain a random string w of length r.
- Given y and w, Bob sends Alice a message of length ℓ.
- After that, Alice decides whether to accept or not.
  - When the answer is yes, Merlin has exactly one advice such that Alice always accept.
  - When the answer is no, or Merlin sends the wrong advice, Alice accepts with probability at most s.

2.4 Derandomization

We make use of expander graphs to reduce the amount of random coins needed in one of our communication protocols. We abstract the following result for our use here.

▶ **Theorem 2.7** (see e.g. Theorem 21.12 and Theorem 21.19 in [20]). *Let m be an integer, and set $B \subseteq [m]$. Suppose $|B| \ge m/2$. There is a universal constant c_1 such that for all $\varepsilon < 1/2$, there is a $\mathrm{poly}(\log m, \log \varepsilon^{-1})$-time computable function $\mathcal{F} : \{0,1\}^{\log m + c_1 \cdot \log \varepsilon^{-1}} \to [m]^{c_1 \cdot \log \varepsilon^{-1}}$, such that*

$$\Pr_{w \in \{0,1\}^{\log m + c_1 \cdot \log \varepsilon^{-1}}} [a \notin B \text{ for all } a \in \mathcal{F}(w)] \le \varepsilon,$$

here $a \in \mathcal{F}(w)$ means a is one of the element in the sequence $\mathcal{F}(w)$.

3 Hardness of Approximate Max-IP

In this section we prove our characterizations of approximating Max-IP.

3.1 The Multiplicative Case

We begin with the proof of Theorem 1.5. We recap it here for convenience.

Reminder of Theorem 1.5 *Letting $\omega(\log n) < d < n^{o(1)}$ and $t \ge 2$, the following holds:*

1. *There is an $n^{2-\Omega(1)}$-time t-multiplicative-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if*

$$t = (d/\log n)^{\Omega(1)},$$

and under SETH (or OVC), there is no $n^{2-\Omega(1)}$-time t-multiplicative-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if

$$t = (d/\log n)^{o(1)}.$$

2. *Moreover, let $\varepsilon = \min\left(\dfrac{\log t}{\log(d/\log n)}, 1\right)$. There are t-multiplicative-approximating deterministic algorithms for $\mathsf{Max\text{-}IP}_{n,d}$ running in time*

$$O\left(n^{2+o(1)-0.31 \cdot \frac{1}{\varepsilon^{-1}+\frac{0.31}{2}}}\right) = O\left(n^{2+o(1)-\Omega(\varepsilon)}\right)$$

or time

$$O\left(n^{2-0.17 \cdot \frac{1}{\varepsilon^{-1}+\frac{0.17}{2}}} \cdot \mathrm{polylog}(n)\right) = O\left(n^{2-\Omega(\varepsilon)} \cdot \mathrm{polylog}(n)\right).$$

In Lemma 3.2, we construct the desired approximate algorithm and in Corollary 3.4 we prove the lower bound.

The Algorithm

First we need the following simple lemma, which says that the k-th root of the sum of the k-th powers of non-negative reals gives a good approximation to their maximum.

▶ **Lemma 3.1.** *Let S be a set of non-negative real numbers, k be an integer, and $x_{max} := \max_{x \in S} x$. We have*

$$\left(\sum_{x \in S} x^k\right)^{1/k} \in \left[x_{max}, x_{max} \cdot |S|^{1/k}\right].$$

Proof. Since

$$\left(\sum_{x \in S} x^k\right) \in \left[x_{max}^k, |S| \cdot x_{max}^k\right],$$

the lemma follows directly by taking the k-th root of both sides. ◀

▶ **Lemma 3.2.** *Assuming $\omega(\log n) < d < n^{o(1)}$ and letting $\varepsilon = \min\left(\frac{\log t}{\log(d/\log n)}, 1\right)$, there are t-multiplicative-approximating deterministic algorithms for* Max-IP$_{n,d}$ *running in time*

$$O\left(n^{2+o(1)-0.31\cdot\frac{1}{\varepsilon^{-1}+\frac{0.31}{2}}}\right) = O\left(n^{2+o(1)-\Omega(\varepsilon)}\right)$$

or time

$$O\left(n^{2-0.17\cdot\frac{1}{\varepsilon^{-1}+\frac{0.17}{2}}} \cdot \text{polylog}(n)\right) = O\left(n^{2-\Omega(\varepsilon)} \cdot \text{polylog}(n)\right).$$

Proof. Let $d = c \cdot \log n$. From the assumption, we have $c = \omega(1)$, and $\varepsilon = \min\left(\frac{\log t}{\log c}, 1\right)$. When $\log t > \log c$, we simply use a c-multiplicative-approximating algorithm instead, hence in the following we assume $\log t \leq \log c$. We begin with the first algorithm here.

Construction and Analysis of the Power of Sum Polynomial $P_r(z)$. Let r be a parameter to be specified later and z be a vector from $\{0,1\}^d$, consider the following polynomial

$$P_r(z) := \left(\sum_{i=1}^{d} z_i\right)^r.$$

Observe that since each z_i takes value in $\{0,1\}$, we have $z_i^k = z_i$ for $k \geq 2$. Therefore, by expanding out the polynomial and replacing all z_i^k with $k \geq 2$ by z_i, we can write $P_r(z)$ as

$$P_r(z) = \sum_{S \subseteq [d], |S| \leq r} c_S \cdot z_S.$$

In which $z_S := \prod_{i \in S} z_i$, and the c_S's are the corresponding coefficients. Note that $P_r(z)$ has

$$m := \sum_{k=0}^{r} \binom{d}{k} \leq \left(\frac{ed}{r}\right)^r$$

terms.

Then consider $P_r(x,y) := P_r(x_1 \cdot y_1, x_2 \cdot y_2, \ldots, x_d \cdot y_d)$, plugging in $z_i := x_i \cdot y_i$, it can be written as

$$P_r(x,y) := \sum_{S \subseteq [d], |S| \le r} c_S \cdot x_S \cdot y_S,$$

where $x_S := \prod_{i \in S} x_i$, and y_S is defined similarly.

Construction and Analysis of the Batch Evaluation Polynomial $\boldsymbol{P_r(X,Y)}$. Now, let X and Y be two sets of $b = t^{r/2}$ vectors from $\{0,1\}^d$, we define

$$P_r(X,Y) := \sum_{x \in X, y \in Y} P_r(x,y) = \sum_{x \in X, y \in Y} (x \cdot y)^r.$$

By Lemma 3.1, we have

$$P_r(X,Y)^{1/r} \in \left[\mathsf{OPT}(X,Y), \mathsf{OPT}(X,Y) \cdot t\right],$$

recall that $\mathsf{OPT}(X,Y) := \max_{x \in X, y \in Y} x \cdot y$.

Embedding into Rectangle Matrix Multiplication. Now, for $x, y \in \{0,1\}^d$, we define the mapping $\phi_x(x)$ as

$$\phi_x(x) := (c_{S_1} \cdot x_{S_1}, c_{S_2} \cdot x_{S_2}, \ldots, c_{S_m} \cdot x_{S_m})$$

and

$$\phi_y(y) := (y_{S_1}, y_{S_2}, \ldots, y_{S_m}),$$

where $S_1, S_2, \ldots, S_m$ is an enumeration of all sets $S \subseteq [d]$ and $|S| \le r$.

From the definition, it follows that

$$\phi_x(x) \cdot \phi_y(y) = P_r(x,y)$$

for every $x, y \in \{0,1\}^d$.

Then for each X and Y, we map them into m-dimensional vectors $\phi_X(X)$ and $\phi_Y(Y)$ simply by a summation:

$$\phi_X(X) := \sum_{x \in X} \phi_x(x) \quad \text{and} \quad \Phi_Y(Y) := \sum_{y \in Y} \phi_y(y).$$

We can see

$$\phi_X(X) \cdot \phi_Y(Y) = \sum_{x \in X} \phi_x(x) \cdot \sum_{y \in Y} \phi_y(y) = \sum_{x \in X} \sum_{y \in Y} P_r(x,y) = P_r(X,Y).$$

Given two sets A, B of n vectors from $\{0,1\}^d$, we split A into n/b sets $A_1, A_2, \ldots, A_{n/b}$ of size b, and split B in the same way as well. Then we construct a matrix $M_A(M_B)$ of size $n/b \times m$, such that the i-th row of $M_A(M_B)$ is the vector $\Phi_X(A_i)(\Phi_Y(B_i))$. After that, the evaluation of $P_r(A_i, B_j)$ for all $i, j \in [n/b]$ can be reduced to compute the matrix product $M_A \cdot M_B^T$. After knowing all $P_r(A_i, B_j)$'s, we simply compute the maximum of them, whose r-th root gives us a t-multiplicative-approximating answer of the original problem.

Analysis of the Running Time. Finally, we are going to specify the parameter r and analyze the time complexity. In order to utilize the fast matrix multiplication algorithm from Theorem 2.1, we need to have

$$m \le (n/b)^{0.313},$$

then our running time is simply $(n/b)^{2+o(1)} = n^{2+o(1)}/b^2$.

We are going to set $r = k \cdot \log n/\log c$, and our choice of k will satisfy $k = \Theta(1)$. We have

$$m \le \left(\frac{e \cdot d}{r}\right)^r \le \left(\frac{c\log n \cdot e}{k \cdot \log n/\log c}\right)^{k \cdot \log n/\log c},$$

and therefore

$$\log m \le k \cdot \log n \left[\log \frac{c \log c}{k} + 1\right] \Big/ \log c.$$

Since $c = \omega(1)$ and $k = \Theta(1)$, we have

$$\log m \le (1 + o(1)) \cdot k \log n = k \log n + o(\log n).$$

Plugging in, we have

$$\begin{aligned}
& m \le (n/b)^{0.313} \\
\Longleftarrow\ & \log m \le 0.313 \cdot (\log n - \log b) \\
\Longleftarrow\ & k \log n \le 0.31 \cdot (\log n - \log b) \\
\Longleftarrow\ & 0.31 \cdot (r/2) \cdot \log t + k \log n \le 0.31 \log n && (b = t^{r/2}) \\
\Longleftarrow\ & \frac{\log n}{\log c} \cdot k \cdot \log t \cdot \frac{0.31}{2} + k \log n \le 0.31 \log n && (r = k \cdot \log n/\log c) \\
\Longleftarrow\ & k \cdot \left\{1 + \frac{\log t}{\log c} \cdot \frac{0.31}{2}\right\} \le 0.31 \\
\Longleftarrow\ & k = \frac{0.31}{1 + \frac{\log t}{\log c} \cdot \frac{0.31}{2}} = \frac{0.31}{1 + \frac{0.31}{2} \cdot \varepsilon}.
\end{aligned}$$

Note since $\varepsilon \in [0, 1]$, k is indeed $\Theta(1)$.

Finally, with our choice of k specified, our running time is $n^{2+o(1)}/b^2 = n^{2+o(1)}/t^r$.

By a simple calculation,

$$\begin{aligned}
\log t^r &= r \cdot \log t \\
&= k \cdot \log n/\log c \cdot \log t \\
&= \log n \cdot \left\{\frac{\log t}{\log c} \cdot \frac{0.31}{1 + \frac{0.31}{2} \cdot \varepsilon}\right\} \\
&= \log n \cdot \frac{0.31\varepsilon}{1 + \frac{0.31}{2} \cdot \varepsilon} \\
&= \log n \cdot \frac{0.31}{\varepsilon^{-1} + \frac{0.31}{2}}.
\end{aligned}$$

Hence, our running time is

$$n^{2+o(1)}/t^r = n^{2+o(1) - \frac{0.31}{\varepsilon^{-1} + \frac{0.31}{2}}}$$

as stated.

The Second Algorithm. The second algorithm follows exactly the same except for that we apply Theorem 2.2 instead, hence the constant 0.31 is replaced by 0.17. ◀

Generalization to Non-negative Real Case

Note that Lemma 3.1 indeed works for a set of non-negative reals, we can observe that the above algorithm in fact works for $\mathbb{R}^{+}$-Max-$\mathsf{IP}_{n,d}$ (which is the same as Max-IP except for that the sets consisting of non-negative reals):[14]

Reminder of Corollary 1.7 *Assume $\omega(\log n) < d < n^{o(1)}$ and let $\varepsilon = \min(\frac{\log t}{\log(d/\log n)}, 1)$. There is a t-multiplicative-approximating deterministic algorithm for $\mathbb{R}^{+}$-Max-$\mathsf{IP}_{n,d}$ running in time*

$$O\left(n^{2-\Omega(\varepsilon)} \cdot \operatorname{polylog}(n)\right).$$

Proof Sketch. We can just use the same algorithm in Lemma 3.2, the only difference is on the analysis of the number of terms in $P_r(z)$: since z is no longer Boolean, $P_r(z)$ is no longer multi-linear, and we need to switch to a general upper bound $\binom{d+r}{r}$ on the number of terms for r-degree polynomials of d variables. This corollary then follows by a similar calculation as in Lemma 3.2. ◀

The Lower Bound

The lower bound follows directly from the new MA protocol for Set-Disjointness in [64]. We present an explicit proof here for completeness.

To prove the lower bound, we need the following reduction from OV to t-multiplicative-approximating Max-IP.

▶ **Lemma 3.3** (Implicit in Theorem 4.1 of [64]). *There is a universal constant c_1 such that, for every integer c, reals $\varepsilon \in (0,1]$ and $\tau \geq 2$, $\mathsf{OV}_{n,c\log n}$ can be reduced to n^{ε} Max-$\mathsf{IP}_{n,d}$ instances (A_i, B_i) for $i \in [n^{\varepsilon}]$, such that:*

- *$d = \tau^{\operatorname{poly}(c/\varepsilon)} \cdot \log n$.*
- *Letting $T = c\log n \cdot \tau^{c_1}$, if there is $a \in A$ and $b \in B$ such that $a \cdot b = 0$, then there exists an i such that $\mathsf{OPT}(A_i, B_i) \geq T$.*
- *Otherwise, for all i we must have $\mathsf{OPT}(A_i, B_i) \leq T/\tau$.*

The reduction above follows directly from the new MA communication protocols in [64] together with the use of expander graphs to reduce the amount of random coins. A proof for the lemma above can be found in Appendix D.

Now we are ready to show the lower bound on t-multiplicative-approximating Max-IP.

▶ **Corollary 3.4.** *Assuming SETH (or OVC), and letting $d = \omega(\log n)$ and $t \geq 2$. There is no $n^{2-\Omega(1)}$-time t-multiplicative-approximating algorithm for Max-$\mathsf{IP}_{n,d}$ if*

$$t = (d/\log n)^{o(1)}.$$

[14] In the following we assume a real RAM model of computation for simplicity.

Proof. Let $c = d/\log n$, then $t = c^{o(1)}$ (recall that t and d are two functions of n).

Suppose for contradiction that there is an $n^{2-\varepsilon'}$ time $t(n)$-multiplicative-approximating algorithm $\mathbb{A}$ for $\mathsf{Max\text{-}IP}(n,d)$ for some $\varepsilon' > 0$.

Let $\varepsilon = \varepsilon'/2$. Now, for every constant c_2, we apply the reduction in Lemma 3.3 with $\tau = t$ to reduce an $\mathsf{OV}_{n,c_2 \log n}$ instance to n^{ε}

$$\mathsf{Max\text{-}IP}_{n,t^{\mathrm{poly}(c_2/\varepsilon)}\cdot\log n} \equiv \mathsf{Max\text{-}IP}_{n,t^{O(1)}\cdot\log n}$$

instances. Since $t = c^{o(1)}$, which means for sufficiently large n, $t^{O(1)} \cdot \log n = c^{o(1)} \cdot \log n = o(d)$, and it in turn implies that for sufficiently large n, n^{ε} calls to $\mathbb{A}$ are enough to solve the $\mathsf{OV}_{n,c_2 \log n}$ instance.

Therefore, we can solve $\mathsf{OV}_{n,c_2 \log n}$ in $n^{2-\varepsilon'} \cdot n^{\varepsilon} = n^{2-\varepsilon}$ time for all constant c_2. Contradiction to OVC. ◀

Finally, the correctness of Theorem 1.5 follows directly from Lemma 3.2 and Corollary 3.4.

3.2 The Additive Case

In this subsection we prove Theorem 1.10. We first recap it here for convenience.

Reminder of Theorem 1.10 *Letting $\omega(\log n) < d < n^{o(1)}$ and $0 \le t \le d$, the following holds:*

1. *There is an $n^{2-\Omega(1)}$-time t-additive-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if*

   $$t = \Omega(d),$$

   and under SETH (or OVC), there is no $n^{2-\Omega(1)}$-time t-additive-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ if

   $$t = o(d).$$

2. *Moreover, letting $\varepsilon = \dfrac{t}{d}$, there is a randomized*

   $$O\left(n^{2-\Omega(\varepsilon^{1/3}/\log \varepsilon^{-1})}\right)$$

   time, t-additive-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ when $\varepsilon \gg \log^6 \log n/\log^3 n$.

We proceed similarly as in the multiplicative case by establishing the algorithm first.

The Algorithm

The algorithm is actually very easy, we simply apply the following algorithm from [13].

▶ **Lemma 3.5** (Implicit in Theorem 5.1 in [13]). *Assuming $\varepsilon \gg \log^6 \log(d \log n)/\log^3 n$, there is an*

$$n^{2-\Omega\left(\varepsilon^{1/3}/\log\left(\frac{d}{\varepsilon \log n}\right)\right)}$$

time $\varepsilon \cdot d$-additive-approximating randomized algorithm for $\mathsf{Max\text{-}IP}_{n,d}$.

▶ **Lemma 3.6.** *Let $\varepsilon = \dfrac{\min(t,d)}{d}$, there is a randomized*

$$O\left(n^{2-\Omega(\varepsilon^{1/3}/\log \varepsilon^{-1})}\right)$$

time, t-additive-approximating algorithm for $\mathsf{Max\text{-}IP}_{n,d}$ when $\varepsilon \gg \log^6 \log n/\log^3 n$.

Proof. When $t > d$ the problem becomes trivial, so we can assume $t \leq d$, and now $t = \varepsilon \cdot d$.

Let $\varepsilon_1 = \varepsilon/2$ and c_1 be a constant to be specified later. Given an $\textsf{Max-IP}_{n,d}$ instance with two sets A and B of vectors from $\{0,1\}^d$, we create another $\textsf{Max-IP}_{n,d_1}$ instance with sets $\widetilde{A}$, $\widetilde{B}$ and $d_1 = c_1 \cdot \varepsilon_1^{-2} \cdot \log n$ as follows:

- Pick d_1 uniform random indices $i_1, i_2, i_3, \dots, i_{d_1} \in [d]$, each i_k is an independent uniform random number in $[d]$.
- Then we construct $\widetilde{A}$ from A by reducing each $a \in A$ into $\tilde{a} = (a_{i_1}, a_{i_2}, \dots, a_{i_{d_1}}) \in \{0,1\}^{d_1}$ and $\widetilde{B}$ from B in the same way.

Note for each $a \in A$ and $b \in B$, by a Chernoff bound, we have

$$\Pr\left[\left|\frac{\tilde{a} \cdot \tilde{b}}{d_1} - \frac{a \cdot b}{d}\right| \geq \varepsilon_1\right] < 2e^{-2d_1\varepsilon_1^2} = 2n^{-2\cdot c_1}.$$

By setting $c_1 = 2$, the above probability is smaller than $1/n^3$.

Hence, by a simple union bound, with probability at least $1 - 1/n$, we have

$$\left|\frac{\widetilde{a} \cdot \widetilde{b}}{d_1} - \frac{a \cdot b}{d}\right| \leq \varepsilon_1$$

for all $a \in A$ and $b \in B$. Hence, it means that this reduction only changes the "relative inner product"($\dfrac{a \cdot b}{d}$ or $\dfrac{\widetilde{a} \cdot \widetilde{b}}{d_1}$) of each pair by at most ε_1. Hence, the maximum of the "relative inner product" also changes by at most ε_1, and we have $|\textsf{OPT}(A,B)/d - \textsf{OPT}(\widetilde{A},\widetilde{B})/d_1| \leq \varepsilon_1$.

Then we apply the algorithm in Lemma 3.5 on the instance with sets $\widetilde{A}$ and $\widetilde{B}$ with error $\varepsilon = \varepsilon_1$ to obtain an estimate $\widetilde{O}$, and our final answer is simply $\dfrac{\widetilde{O}}{d_1} \cdot d$.

From the guarantee from Lemma 3.5, we have $|\textsf{OPT}(\widetilde{A},\widetilde{B})/d_1 - \widetilde{O}/d_1| \leq \varepsilon_1$, and therefore we have $|\textsf{OPT}(A,B)/d - \widetilde{O}/d_1| \leq 2\varepsilon_1 = \varepsilon$, from which the correctness of our algorithm follows directly.

For the running time, note that the reduction part runs in linear time $O(n \cdot d)$, and the rest takes

$$n^{2-\Omega\left(\varepsilon^{1/3}/\log(\frac{d_1}{\varepsilon_1 \log n})\right)} = n^{2-\Omega(\varepsilon^{1/3}/\log \varepsilon^{-1})}$$

time. ◀

The Lower Bound

The lower bound is already established in [64], we show it follows from Lemma 3.3 here for completeness.

▶ **Lemma 3.7** (Theorem 4.1 of [64]). *Assuming SETH (or OVC), and letting $d = \omega(\log n)$ and $t > 0$, there is no $n^{2-\Omega(1)}$-time t-additive-approximating randomized algorithm for $\textsf{Max-IP}_{n,d}$ if*

$$t = o(d).$$

Proof. Recall that t and d are all functions of n. Suppose for contradiction that there is an $n^{2-\varepsilon'}$ time $t(n)$-additive-approximating algorithm $\mathbb{A}$ for $\textsf{Max-IP}(n,d)$ for some $\varepsilon' > 0$.

Let $\varepsilon = \varepsilon'/2$. Now, for every constant c_2, we apply the reduction in Lemma 3.3 with $\tau = 2$ to reduce an $\mathsf{OV}_{n,c_2 \log n}$ instance to n^{ε}

$$\mathsf{Max\text{-}IP}_{n,2^{\mathrm{poly}(c_2/\varepsilon)} \cdot \log n} \equiv \mathsf{Max\text{-}IP}_{n,d_1} \text{ where } d_1 = O(1) \cdot \log n$$

instances. In addition, from Lemma 3.3, to solve the $\mathsf{OV}_{c_2 \log n}$ instance, we only need to distinguish an additive gap of $\dfrac{T}{2} = \Omega(\log n) = \Omega(d_1)$ for these Max-IP instances obtained via the reduction.

This can be solved, via n^{ε} calls to $\mathbb{A}$ as follows: for each $\mathsf{Max\text{-}IP}_{n,d_1}$ instance $\mathcal{I}$ we get, since $d = \omega(\log n)$, which means for a sufficiently large n, $d_1 = O(\log n) \ll d$, and we can duplicate each coordinate d/d_1 times (for simplicity we assume $d_1 | d$ here), to obtain an $\mathsf{Max\text{-}IP}_{n,d}$ instance $\mathcal{I}^{\mathsf{new}}$, such that $\mathsf{OPT}(\mathcal{I}^{\mathsf{new}}) = d/d_1 \cdot \mathsf{OPT}(\mathcal{I})$. Then $\mathbb{A}$ can be used to estimate $\mathsf{OPT}(\mathcal{I}^{\mathsf{new}})$ within an additive error $t = o(d)$. Scaling its estimate by $\dfrac{d_1}{d}$, it can also be used to estimate $\mathsf{OPT}(\mathcal{I})$ within an additive error $o(d_1) = o(\log n) \le T/2$ for sufficiently large n.

Therefore, we can solve $\mathsf{OV}_{n,c_2 \log n}$ in $n^{2-\varepsilon'} \cdot n^{\varepsilon} = n^{2-\varepsilon}$ time for all constant c_2. Contradiction to OVC. ◀

Finally, the correctness of Theorem 1.10 follows directly from Lemma 3.6 and Lemma 3.7.

3.3 Adaption for All-Pair-Max-IP

Now we sketch the adaption for our algorithms to work for the All-Pair-Max-IP problem.

Reminder of Corollary 1.12 *Suppose $\omega(\log n) < d < n^{o(1)}$, and let*

$$\varepsilon_M := \min\left(\frac{\log t}{\log(d/\log n)}, 1\right) \text{ and } \varepsilon_A := \frac{\min(t,d)}{d}.$$

There is an $n^{2-\Omega(\varepsilon_M)}\,\mathrm{polylog}(n)$ time t-multiplicative-approximating algorithm and an $n^{2-\Omega(\varepsilon_A^{1/3}/\log \varepsilon_A^{-1})}$ time t-additive-approximating algorithm for All-Pair-Max-IP$_{n,d}$, when $\varepsilon_A \gg \log^6 \log n / \log^3 n$.

Proof Sketch. Note that the algorithm in Lemma 3.5 from [13] actually works for the All-Pair-Max-IP$_{n,d}$. Hence, we can simply apply that algorithm after the coordinate sampling phase, and obtain a t-additive-approximating algorithm for All-Pair-Max-IP$_{n,d}$.

For t-multiplicative-approximating algorithm, suppose we are given with two sets A and B of n vectors from $\{0,1\}^d$. Instead of partitioning both of them into n/b subsets A_i's and B_i's (the notations used here are the same as in the proof of Lemma 3.2), we only partition B into n/b subsets $B_1, B_2, \dots, B_{n/b}$ of size b, and calculate $P_r(x, B_i) := \sum_{y \in B_i} P_r(x,y)$ for all $x \in A$ and $i \in [n/b]$ using similar reduction to rectangle matrix multiplication as in Lemma 3.2. By a similar analysis, these can be done in $n^{2-\Omega(\varepsilon_M)} \cdot \mathrm{polylog}(n)$ time, and with these informations we can compute the t-multiplicative-approximating answers for the given All-Pair-Max-IP$_{n,d}$ instance. ◀

3.4 Improved Hardness for LCS-Closest Pair Problem

We finish this section with the proof of Corollary 1.9. First we abstract the reduction from Max-IP to LCS-Closest-Pair in [5] here.

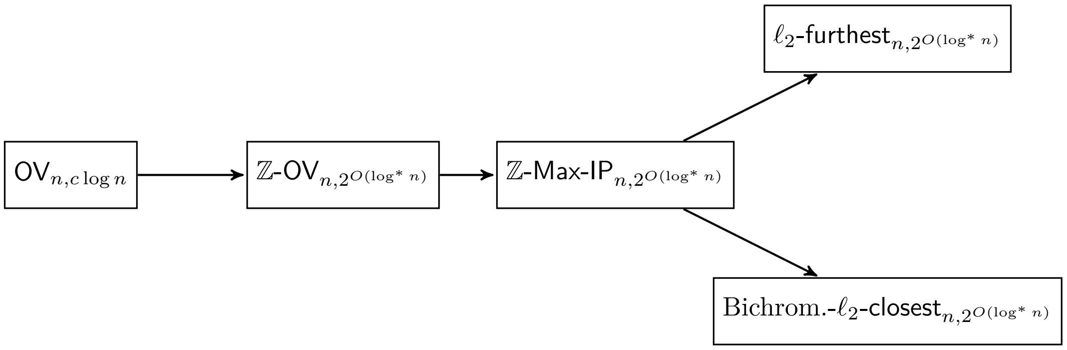
ℓ_2-furthest$_{n,2^{O(\log^* n)}}$
$\mathsf{OV}_{n,c\log n}$
$\mathbb{Z}$-$\mathsf{OV}_{n,2^{O(\log^* n)}}$
$\mathbb{Z}$-Max-IP$_{n,2^{O(\log^* n)}}$
Bichrom.-ℓ_2-closest$_{n,2^{O(\log^* n)}}$

Figure 1 A diagram for all reductions in this section.

▶ **Lemma 3.8** (Implicit in Theorem 1.6 in [5]). *For big enough t and n, t-multiplicative-approximating Max-IP$_{n,d}$ reduces to $t/2$-multiplicative-approximating LCS-Closest-Pair$_{n,d_1}$, where $d_1 = O(d^3 \log^2 n)$.*

Now we are ready to prove Corollary 1.9 (restated below for convenience).

Reminder of Corollary 1.9 *Assuming SETH (or OVC), for every $t \ge 2$, t-multiplicative-approximating LCS-Closest-Pair$_{n,d}$ requires $n^{2-o(1)}$ time, if $d = t^{\omega(1)} \cdot \log^5 n$.*

Proof. From Corollary 3.4, assuming SETH (or OVC), for every $t \ge 2$, we have that $2t$-multiplicative-approximating Max-IP$_{n,d}$ requires $n^{2-o(1)}$ time if $d = t^{\omega(1)} \cdot \log n$. Then from Lemma 3.8, t-multiplicative-approximating LCS-Closest-Pair$_{n,d_1}$ for $d_1 = O(d^3 \log^2 n) = t^{\omega(1)} \cdot \log^5 n$ requires $n^{2-o(1)}$ time. ◀

4 Hardness of Exact $\mathbb{Z}$-Max-IP, Hopcroft's Problem and More

In this section we show hardness of Hopcroft's problem, exact $\mathbb{Z}$-Max-IP, ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair. Essentially our results follow from the framework of [72], in which it is shown that hardness of Hopcroft's problem implies hardness of other three problems, and is implied by dimensionality reduction for OV.

The Organization of this Section

In Section 4.1, we prove the improved dimensionality reduction for OV. In Section 4.2, we establish the hardness of Hopcroft's problem in $2^{O(\log^* n)}$ dimensions with the improved reduction. In Section 4.3, we show Hopcroft's problem can be reduced to $\mathbb{Z}$-Max-IP and thus establish the hardness for the later one. In Section 4.4, we show $\mathbb{Z}$-Max-IP can be reduced to ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair, therefore the hardness for the later two problems follow. See Figure 1 for a diagram of all reductions covered in this section.

The reduction in last three subsections are all from [72] (either explicit or implicit), we make them explicit here for our ease of exposition and for making the paper self-contained.

4.1 Improved Dimensionality Reduction for OV

We begin with the improved dimensionality reduction for OV. The following theorem is one of the technical cores of this paper, which makes use of the CRR encoding (see Theorem 2.5) recursively.

▶ **Theorem 4.1.** *Let b, ℓ be two sufficiently large integers. There is a reduction $\psi_{b,\ell} : \{0,1\}^{b\cdot\ell} \to \mathbb{Z}^\ell$ and a set $V_{b,\ell} \subseteq \mathbb{Z}$, such that for every $x, y \in \{0,1\}^{b\cdot\ell}$,*

$$x \cdot y = 0 \Leftrightarrow \psi_{b,\ell}(x) \cdot \psi_{b,\ell}(y) \in V_{b,\ell}$$

and

$$0 \le \psi_{b,\ell}(x)_i < \ell^{6^{\log^*(b)}\cdot b}$$

for all possible x and $i \in [\ell]$. Moreover, the computation of $\psi_{b,\ell}(x)$ takes $\operatorname{poly}(b\cdot\ell)$ time, and the set $V_{b,\ell}$ can be constructed in $O\left(\ell^{O(6^{\log^*(b)}\cdot b)} \cdot \operatorname{poly}(b\cdot\ell)\right)$ time.

▶ Remark 4.2. We didn't make much effort to minimize the base 6 above to keep the calculation clean, it can be replaced by any constant > 2 with a tighter calculation.

Proof. We are going to construct our reduction in a recursive way. ℓ will be the same throughout the proof, hence in the following we use ψ_b (V_b) instead of $\psi_{b,\ell}$ $(V_{b,\ell})$ for simplicity.

Direct CRR for small b:

When $b < \ell$, we use a direct Chinese remainder representation of numbers. We pick b primes $q_1, q_2, \dots, q_b$ in $[\ell+1, \ell^2]$, and use them for our CRR encoding.

Let $x \in \{0,1\}^{b\cdot\ell}$, we partition it into ℓ equal size groups, and use x^i to denote the i-th group, which is the sub-vector of x from the $((i-1)\cdot b+1)$-th bit to the $(i\cdot b)$-th bit.

Then we define $\psi_b(x)$ as

$$\psi_b(x) := \left(\mathsf{CRR}\left(\left\{x^1_j\right\}_{j=1}^b\right), \mathsf{CRR}\left(\left\{x^2_j\right\}_{j=1}^b\right), \dots, \mathsf{CRR}\left(\left\{x^\ell_j\right\}_{j=1}^b\right)\right).$$

That is, the i-th coordinate of $\psi_b(x)$ is the CRR encoding of the i-th sub-vector x^i with respect to the primes q_j's.

Now, for $x, y \in \{0,1\}^{b\cdot\ell}$, note that for $j \in [b]$,

$$\begin{aligned}
&\psi_b(x)\cdot\psi_b(y) \pmod{q_j}\\
\equiv&\sum_{i=1}^{\ell} \mathsf{CRR}\left(\left\{x^i_j\right\}_{j=1}^b\right)\cdot\mathsf{CRR}\left(\left\{y^i_j\right\}_{j=1}^b\right) \pmod{q_j}\\
\equiv&\sum_{i=1}^{\ell} x^i_j \cdot y^i_j \pmod{q_j}.
\end{aligned}$$

Since the sum $\sum_{i=1}^{\ell} x^i_j \cdot y^i_j$ is in $[0,\ell]$, and $q_j > \ell$, we can see

$$\sum_{i=1}^{\ell} x^i_j \cdot y^i_j = 0 \Leftrightarrow \psi_b(x)\cdot\psi_b(y) \equiv 0 \pmod{q_j}.$$

Therefore, $x \cdot y = \sum_{j=1}^{b}\sum_{i=1}^{\ell} x^i_j \cdot y^i_j = 0$ is equivalent to that

$$\psi_b(x)\cdot\psi_b(y) \equiv 0 \pmod{q_j}$$

for every $j \in [b]$.

Finally, we have $0 \le \psi_b(x)_i < \prod_{j=1}^{b} p_j < \ell^{2 \cdot b} \le \ell^{6^{\log^*(b)} \cdot b}$. Therefore

$$\psi_b(x) \cdot \psi_b(y) < \ell^{6^{\log^*(b)} \cdot 2b+1},$$

and we can set V_b to be the set of all integers in $[0, \ell^{6^{\log^*(b)} \cdot 2b+1}]$ that is 0 modulo all the p_j's, and it is easy to see that

$$x \cdot y \Leftrightarrow \psi_b(x) \cdot \psi_b(y) \in V_b$$

for all $x, y \in \{0,1\}^{b \cdot \ell}$.

Recursive Construction for larger b:

When $b \ge \ell$, suppose the theorem holds for all $b' < b$. Let b_{micro} be the number such that (we ignore the rounding issue here and pretend that b_{micro} is an integer for simplicity),

$$\ell^{6^{\log^*(b_{\mathsf{micro}})} \cdot b_{\mathsf{micro}}} = b.$$

Then we pick b/b_{micro} primes $p_1, p_2, \dots, p_{b/b_{\mathsf{micro}}}$ in $[(b^2\ell), (b^2\ell)^2]$, and use them as our reference primes in the CRR encodings.

Let $x \in \{0,1\}^{b \cdot \ell}$, as before, we partition x into ℓ equal size sub-vectors $x^1, x^2, \dots, x^\ell$, where x^i consists of the $((i-1) \cdot b + 1)$-th bit of x to the $(i \cdot b)$-th bit of x. Then we partition each x^i again into b/b_{micro} micro groups, each of size b_{micro}. We use $x^{i,j}$ to denote the j-th micro group of x^i after the partition.

Now, we use $x^{[j]}$ to denote the concatenation of the vectors $x^{1,j}, x^{2,j}, \dots, x^{\ell,j}$. That is, $x^{[j]}$ is the concatenation of the j-th micro group in each of the ℓ groups. Note that $x^{[j]} \in \{0,1\}^{b_{\mathsf{micro}} \cdot \ell}$, and can be seen as a smaller instance, on which we can apply $\psi_{b_{\mathsf{micro}}}$.

Our recursive construction then goes in two steps. In the first step, we make use of $\psi_{b_{\mathsf{micro}}}$, and transform each b_{micro}-size micro group into a single number in $[0, b)$. This step transforms x from a vector in $\{0,1\}^{b \cdot \ell}$ into a vector $S(x)$ in $\mathbb{Z}^{(b/b_{\mathsf{micro}}) \cdot \ell}$. And in the second step, we use a similar CRR encoding as in the base case to encode $S(x)$, to get our final reduced vector in $\mathbb{Z}^\ell$.

$S(x)$ is simply

$$\begin{aligned} S(x) := \Big(&\psi_{b_{\mathsf{micro}}}(x^{[1]})_1, \psi_{b_{\mathsf{micro}}}(x^{[2]})_1, \dots, \psi_{b_{\mathsf{micro}}}(x^{[b/b_{\mathsf{micro}}]})_1, \\ &\psi_{b_{\mathsf{micro}}}(x^{[1]})_2, \psi_{b_{\mathsf{micro}}}(x^{[2]})_2, \dots, \psi_{b_{\mathsf{micro}}}(x^{[b/b_{\mathsf{micro}}]})_2, \\ &\dots, \dots, \dots \\ &\psi_{b_{\mathsf{micro}}}(x^{[1]})_\ell, \psi_{b_{\mathsf{micro}}}(x^{[2]})_\ell, \dots, \psi_{b_{\mathsf{micro}}}(x^{[b/b_{\mathsf{micro}}]})_\ell \Big). \end{aligned}$$

That is, we apply $\psi_{b_{\mathsf{micro}}}$ on all the $x^{[j]}$'s, and shrink all the corresponding micro-groups in x into integers. Again, we partition S into ℓ equal size groups $S^1, S^2, \dots, S^\ell$.

Then we define $\psi_b(x)$ as

$$\psi_b(x) := \left(\mathsf{CRR}\left(\{S_j^1\}_{j=1}^{b/b_{\mathsf{micro}}} \right), \mathsf{CRR}\left(\{S_j^2\}_{j=1}^{b/b_{\mathsf{micro}}} \right), \dots, \mathsf{CRR}\left(\{S_j^\ell\}_{j=1}^{b/b_{\mathsf{micro}}} \right) \right).$$

In other words, the i-th coordinate of $\psi_b(x)$ is the CRR representation of the number sequence S^i, with respect to our primes $\{q_j\}_{j=1}^{b/b_{\mathsf{micro}}}$.

Now, note that for $x,y \in \{0,1\}^{b\cdot\ell}$, $x \cdot y = 0$ is equivalent to $x^{[j]} \cdot y^{[j]} = 0$ for every $j \in [b/b_{\mathsf{micro}}]$, which is further equivalent to

$$\psi_{b_{\mathsf{micro}}}(x^{[j]}) \cdot \psi_{b_{\mathsf{micro}}}(y^{[j]}) \in V_{b_{\mathsf{micro}}}$$

for all $j \in [b/b_{\mathsf{micro}}]$, by our assumption on $\psi_{b_{\mathsf{micro}}}$.

Since $0 \le \psi_{b_{\mathsf{micro}}}(x^{[j]})_i, \psi_{b_{\mathsf{micro}}}(y^{[j]})_i < b$ for all $x,y \in \{0,1\}^{b\cdot\ell}$, $i \in [\ell]$ and $j \in [b/b_{\mathsf{micro}}]$, we also have $\psi_{b_{\mathsf{micro}}}(x^{[j]}) \cdot \psi_{b_{\mathsf{micro}}}(y^{[j]}) < b^2 \cdot \ell$, therefore we can assume that $V_{b_{\mathsf{micro}}} \subseteq [0, b^2\ell)$.

For all $x,y \in \{0,1\}^{b\cdot\ell}$ and $j \in [b/b_{\mathsf{micro}}]$, we have

$$\begin{aligned}
&\psi_b(x) \cdot \psi_b(y)\\
\equiv&\sum_{i=1}^{\ell} \mathsf{CRR}\left(\left\{S(x)_j^i\right\}_{j=1}^{b/b_{\mathsf{micro}}}\right) \cdot \mathsf{CRR}\left(\left\{S(y)_j^i\right\}_{j=1}^{b/b_{\mathsf{micro}}}\right) \pmod{p_j}\\
\equiv&\sum_{i=1}^{\ell} S(x)_j^i \cdot S(y)_j^i \pmod{p_j}\\
\equiv&\sum_{i=1}^{\ell} \psi_{b_{\mathsf{micro}}}(x^{[j]})_i \cdot \psi_{b_{\mathsf{micro}}}(y^{[j]})_i \pmod{p_j}\\
\equiv&\psi_{b_{\mathsf{micro}}}(x^{[j]}) \cdot \psi_{b_{\mathsf{micro}}}(y^{[j]}) \pmod{p_j}.
\end{aligned}$$

Since $p_j \ge b^2 \cdot \ell$, we can determine $\psi_{b_{\mathsf{micro}}}(x^{[j]}) \cdot \psi_{b_{\mathsf{micro}}}(y^{[j]})$ from $\psi_b(x) \cdot \psi_b(y)$ by taking modulo p_j. Therefore,

$$x \cdot y = 0$$

is equivalent to

$$(\psi_b(x) \cdot \psi_b(y) \mod p_j) \in V_{b_{\mathsf{micro}}},$$

for every $j \in [b/b_{\mathsf{micro}}]$.

Finally, recall that we have

$$\ell^{6^{\log^*(b_{\mathsf{micro}})} \cdot b_{\mathsf{micro}}} = b.$$

Taking logarithm of both sides, we have

$$6^{\log^*(b_{\mathsf{micro}})} \cdot b_{\mathsf{micro}} \cdot \log \ell = \log b.$$

Then we can upper bound $\psi_b(x)_i$ by

$$\begin{aligned}
\psi_b(x)_i &< \prod_{j=1}^{b/b_{\mathsf{micro}}} p_j\\
&< (b^2\ell)^{2\cdot(b/b_{\mathsf{micro}})} && (b \ge \ell.)\\
&\le 2^{6\cdot b/b_{\mathsf{micro}} \cdot \log b}\\
&\le 2^{6\cdot b/b_{\mathsf{micro}} \cdot 6^{\log^*(b_{\mathsf{micro}})} \cdot b_{\mathsf{micro}} \cdot \log \ell}\\
&\le \ell^{6 \cdot 6^{\log^*(b_{\mathsf{micro}})} \cdot b}\\
&\le \ell^{6^{\log^*(b)} \cdot b} && (b_{\mathsf{micro}} \le \log b, \log^*(b_{\mathsf{micro}}) + 1 \le \log^*(\log b) + 1 = \log^*(b).)
\end{aligned}$$

Therefore, we can set V_b as the set of integer t in $[0, \ell^{6^{\log^*(b)} \cdot 2b+1})$ such that

$$(t \mod p_j) \in V_{b_{\mathsf{micro}}}$$

for every $j \in [b/b_{\mathsf{micro}}]$. And it is easy to see this V_b satisfies our requirement.

Finally, it is easy to see that the straightforward way of constructing $\psi_b(x)$ takes $O(\mathrm{poly}(b \cdot \ell))$ time, and we can construct V_b by enumerating all possible values of $\psi_b(x) \cdot \psi_b(y)$ and check each of them in $O(\mathrm{poly}(b \cdot \ell))$ time. Since there are at most $\ell^{O(6^{\log^*(b)} \cdot b)}$ such values, V_b can be constructed in

$$O\left(\ell^{O(6^{\log^*(b)} \cdot b)} \cdot \mathrm{poly}(b \cdot \ell)\right)$$

time, which completes the proof. ◀

Now we prove Lemma 1.17, we recap its statement here for convenience.

Reminder of Lemma 1.17 *Let $1 \le \ell \le d$. There is an*

$$O\left(n \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))} \cdot \mathrm{poly}(d)\right)\text{-}\textit{time}$$

reduction from $\mathsf{OV}_{n,d}$ to $\ell^{O(6^{\log^* d} \cdot (d/\ell))}$ instances of $\mathbb{Z}$-$\mathsf{OV}_{n,\ell+1}$, with vectors of entries with bit-length $O\left(d/\ell \cdot \log \ell \cdot 6^{\log^* d}\right)$.

Proof. The proof is exactly the same as the proof for Lemma 1.1 in [72] with different parameters, we recap it here for convenience.

Given two sets A' and B' of n vectors from $\{0,1\}^d$, we apply $\psi_{d/\ell,\ell}$ to each of the vectors in A' (B') to obtain a set A (B) of vectors from $\mathbb{Z}^\ell$. From Theorem 4.1, there is a $(u,v) \in A' \times B'$ such that $u \cdot v = 0$ if and only if there is a $(u,v) \in A \times B$ such that $u \cdot v \in V_{d/\ell,\ell}$.

Now, for each element $t \in V_{d/\ell,\ell}$, we are going to construct two sets A_t and B_t of vectors from $\mathbb{Z}^{\ell+1}$ such that there is a $(u,v) \in A \times B$ with $u \cdot v = t$ if and only if there is a $(u,v) \in A_t \times B_t$ with $u \cdot v = 0$. We construct a set A_t as a collection of all vectors $u_A = [u, 1]$ for $u \in A$, and a set B_t as a collection of all vectors $v_B = [v, -t]$ for $v \in B$. It is easy to verify this reduction has the properties we want.

Note that there are at most $\ell^{O(6^{\log^* d} \cdot (d/\ell))}$ numbers in $V_{d/\ell,\ell}$, so we have such a number of $\mathbb{Z}$-$\mathsf{OV}_{n,\ell+1}$ instances. And from Theorem 4.1, the reduction takes

$$O\left(n \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))} \cdot \mathrm{poly}(d)\right)$$

time.

Finally, the bit-length of reduced vectors is bounded by

$$\log\left(\ell^{O(6^{\log^* d} \cdot (d/\ell))}\right) = O\left(d/\ell \cdot \log \ell \cdot 6^{\log^* d}\right),$$

which completes the proof. ◀

A Transformation from Nonuniform Construction to Uniform Construction

The proof for Theorem 4.1 works recursively. In one recursive step, we reduce the construction of $\psi_{b,\ell}$ to the construction of $\psi_{b_{\mathsf{micro}},\ell}$, where $b_{\mathsf{micro}} \le \log b$. Applying this reduction $\log^* n$ times, we get a sufficiently small instance that we can switch to a direct CRR construction.

An interesting observation here is that after applying the reduction only thrice, the block length parameter becomes $b' \le \log\log\log b$, which is so small that we can actually use brute force to find the "optimal" construction $\psi_{b',\ell}$ in $b^{o(1)}$ time instead of recursing deeper. Hence, to find a construction better than Theorem 4.1, we only need to prove the existence of such a construction. See Appendix B for details.

4.2 Improved Hardness for Hopcroft's Problem

In this subsection we are going to prove Theorem 1.18 using our new dimensionality reduction Lemma 1.17, we recap its statement here for completeness.

Reminder of Theorem 1.18 *[Hardness of Hopcroft's Problem in $c^{\log^* n}$ Dimension] Assuming SETH (or OVC), there is a constant c such that $\mathbb{Z}$-$\mathsf{OV}_{n,c^{\log^* n}}$ with vectors of $O(\log n)$-bit entries requires $n^{2-o(1)}$ time.*

Proof. The proof here follows roughly the same as the proof for Theorem 1.1 in [72].

Let c be an arbitrary constant and $d := c \cdot \log n$. We show that an oracle solving $\mathbb{Z}$-$\mathsf{OV}_{n,\ell+1}$ where $\ell = 7^{\log^* n}$ in $O(n^{2-\delta})$ time for some $\delta > 0$ can be used to construct an $O(n^{2-\delta+o(1)})$ time algorithm for $\mathsf{OV}_{n,d}$, and therefore contradicts the OVC.

We simply invoke Lemma 1.17, note that we have

$$\begin{aligned}\log\left\{\ell^{O\left(6^{\log^* d}\cdot(d/\ell)\right)}\right\} &= \log \ell \cdot O\left(6^{\log^* d}\cdot(d/\ell)\right)\\ &= O\left(\log^* n \cdot 6^{\log^* n}\cdot c\cdot \log n/7^{\log^* n}\right)\\ &= O\left(\log^* n\cdot (6/7)^{\log^* n}\cdot c\cdot\log n\right)\\ &= o(\log n).\end{aligned}$$

Therefore, the reduction takes $O(n\cdot \ell^{O\left(6^{\log^* d}\cdot(d/\ell)\right)}\cdot \mathrm{poly}(d)) = n^{1+o(1)}$ time, and an $\mathsf{OV}_{n,d}$ instance is reduced to $n^{o(1)}$ instances of $\mathbb{Z}$-$\mathsf{OV}_{n,\ell+1}$, and the reduced vectors have bit length $o(\log n)$ as calculated above. We simply solve all these $n^{o(1)}$ instances using our oracle, and this gives us an $O(n^{2-\delta+o(1)})$ time algorithm for $\mathsf{OV}_{n,d}$, which completes the proof. ◀

4.3 Hardness for $\mathbb{Z}$-Max-IP

Now we move to hardness of exact $\mathbb{Z}$-Max-IP.

▶ **Theorem 4.3** (Implicit in Theorem 1.2 [72]). *There is an $O(\mathrm{poly}(d)\cdot n)$-time algorithm which reduces a $\mathbb{Z}$-$\mathsf{OV}_{n,d}$ instance into a $\mathbb{Z}$-Max-IP_{n,d^2} instance.*

Proof. We remark here that this reduction is implicitly used in the proof of Theorem 1.2 in [72], we abstract it here only for our exposition.

Given a $\mathbb{Z}$-$\mathsf{OV}_{n,d}$ instance with sets A, B. Consider the following polynomial $P(x, y)$, where $x, y \in \mathbb{Z}^d$.

$$P(x,y) = -(x \cdot y)^2 = \sum_{i,j \in [d]} -(x_i \cdot y_i) \cdot (x_j \cdot y_j) = \sum_{i,j \in [d]} -(x_i \cdot x_j) \cdot (y_i \cdot y_j).$$

It is easy to see that whether there is a $(x, y) \in A \times B$ such that $x \cdot y = 0$ is equivalent to whether the maximum value of $P(x, y)$ is 0.

Now, for each $x \in A$ and $y \in B$, we identify $[d^2]$ with $[d] \times [d]$ and construct $\widetilde{x}, \widetilde{y} \in \mathbb{Z}^{d^2}$ such that

$$\widetilde{x}_{(i,j)} = x_i \cdot x_j \quad \text{and} \quad \widetilde{y}_{(i,j)} = -y_i \cdot y_j.$$

Then we have $\widetilde{x} \cdot \widetilde{y} = P(x, y)$. Hence, let $\widetilde{A}$ be the set of all these $\widetilde{x}$'s, and $\widetilde{B}$ be the set of all these $\widetilde{y}$'s, whether there is a $(x, y) \in A \times B$ such that $x \cdot y = 0$ is equivalent to whether $\mathsf{OPT}(\widetilde{A}, \widetilde{B}) = 0$, and our reduction is completed.

◀

Now, Theorem 1.14 (restated below) is just a simple corollary of Theorem 4.3 and Theorem 1.18.

Reminder of Theorem 1.14 *Assuming SETH (or OVC), there is a constant c such that every exact algorithm for $\mathbb{Z}$-Max-IP$_{n,d}$ for $d = c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

A Dimensionality Reduction for Max-IP

The reduction $\psi_{b,\ell}$ from Theorem 4.1 actually does more: for $x, y \in \{0,1\}^{b \cdot \ell}$, from $\psi_{b,\ell}(x) \cdot \psi_{b,\ell}(y)$ we can in fact determine the inner product $x \cdot y$ itself, not only whether $x \cdot y = 0$.

Starting from this observation, together with Theorem 4.3, we can in fact derive a similar dimensionality self reduction from Max-IP to $\mathbb{Z}$-Max-IP, we deter its proof to Appendix A.

▶ **Corollary 4.4.** *Let $1 \le \ell \le d$. There is an*

$$O\left(n \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))} \cdot \mathrm{poly}(d)\right) \text{-time}$$

reduction from Max-IP$_{n,d}$ to $d \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))}$ instances of $\mathbb{Z}$-Max-IP$_{n,(\ell+1)^2}$, with vectors of entries with bit-length $O\left(d/\ell \cdot \log \ell \cdot 6^{\log^* d}\right)$.

4.4 Hardness for ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair

We finish the whole section with the proof of hardness of ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair. The two reductions below are slight adaptations of the ones in the proofs of Theorem 1.2 and Corollary 2.1 in [72].

▶ **Lemma 4.5.** *Assuming $d = n^{o(1)}$, there is an $O(\mathrm{poly}(d) \cdot n)$-time algorithm which reduces a $\mathbb{Z}$-Max-IP$_{n,d}$ instance into an instance of ℓ_2-Furthest Pair on $2n$ points in $\mathbb{R}^{d+2}$. Moreover, if the $\mathbb{Z}$-Max-IP instance consists of vectors of $O(\log n)$-bit entries, so does the ℓ_2-Furthest Pair instance.*

Proof. Let A, B be the sets in the $\mathbb{Z}$-Max-IP$_{n,d}$ instance, and k be the smallest integer such that all vectors from A and B consist of $(k \cdot \log n)$-bit entries.

Let W be $n^{C \cdot k}$ where C is a large enough constant. Given $x \in A$ and $y \in B$, we construct point

$$\widetilde{x} = \left(x, \sqrt{W - \|x\|^2}, 0\right) \quad \text{and} \quad \widetilde{y} = \left(-y, 0, \sqrt{W - \|y\|^2}\right),$$

that is, appending two corresponding values into the end of vectors x and $-y$.

Now, we can see that for $x_1, x_2 \in A$, the squared distance between their reduced points is

$$\|\widetilde{x_1} - \widetilde{x_2}\|^2 = \|x_1 - x_2\|^2 \leq 4 \cdot d \cdot n^{2k}.$$

Similarly we have

$$\|\widetilde{y_1} - \widetilde{y_2}\|^2 \leq 4 \cdot d \cdot n^{2k}$$

for $y_1, y_2 \in B$.

Next, for $x \in A$ and $y \in B$, we have

$$\|\widetilde{x} - \widetilde{y}\|^2 = \|\widetilde{x}\|^2 + \|\widetilde{y}\|^2 - 2 \cdot \widetilde{x} \cdot \widetilde{y} = 2 \cdot W + 2 \cdot (x \cdot y) \geq 2 \cdot W - d \cdot n^{2k} \gg 4 \cdot d \cdot n^{2k},$$

the last inequality holds when we set C to be 5.

Putting everything together, we can see the ℓ_2-furthest pair among all points $\widetilde{x}$'s and $\widetilde{y}$'s must be a pair of $\widetilde{x}$ and $\widetilde{y}$ with $x \in A$ and $y \in B$. And maximizing $\|\widetilde{x} - \widetilde{y}\|$ is equivalent to maximize $x \cdot y$, which proves the correctness of our reduction. Furthermore, when k is a constant, the reduced instance clearly only needs vectors with $O(k) \cdot \log n = O(\log n)$-bit entries. ◀

▶ **Lemma 4.6.** *Assuming $d = n^{o(1)}$, there is an $O(\mathrm{poly}(d) \cdot n)$-time algorithm which reduces a $\mathbb{Z}$-Max-IP$_{n,d}$ instance into an instance of Bichromatic ℓ_2-Closest Pair on $2n$ points in $\mathbb{R}^{d+2}$. Moreover, if the $\mathbb{Z}$-Max-IP instance consists of vectors of $O(\log n)$-bit entries, so does the Bichromatic ℓ_2-Closest Pair instance.*

Proof. Let A, B be the sets in the $\mathbb{Z}$-Max-IP$_{n,d}$ instance, and k be the smallest integer such that all vectors from A and B consist of $(k \cdot \log n)$-bit entries.

Let W be $n^{C \cdot k}$ where C is a large enough constant. Given $x \in A$ and $y \in B$, we construct point

$$\widetilde{x} = \left(x, \sqrt{W - \|x\|^2}, 0\right) \quad \text{and} \quad \widetilde{y} = \left(y, 0, \sqrt{W - \|y\|^2}\right),$$

that is, appending two corresponding values into the end of vectors x and $-y$. And our reduced instance is to find the closest point between the set $\widetilde{A}$ (consisting of all these $\widetilde{x}$ where $x \in A$) and the set $\widetilde{B}$ (consisting of all these $\widetilde{y}$ where $y \in B$).

Next, for $x \in A$ and $y \in B$, we have

$$\|\widetilde{x} - \widetilde{y}\|^2 = \|\widetilde{x}\|^2 + \|\widetilde{y}\|^2 - 2 \cdot \widetilde{x} \cdot \widetilde{y} = 2 \cdot W - 2 \cdot (x \cdot y) \geq 2 \cdot W - d \cdot n^{2k} \gg 4 \cdot d \cdot n^{2k},$$

the last inequality holds when we set C to be 5.

Hence minimizing $\|\widetilde{x} - \widetilde{y}\|$ where $x \in A$ and $y \in B$ is equivalent to maximize $x \cdot y$, which proves the correctness of our reduction. Furthermore, when k is a constant, the reduced instance clearly only needs vectors with $O(k) \cdot \log n = O(\log n)$-bit entries. ◀

Now Theorem 1.15 and Theorem 1.16 (restated below) are simple corollaries of Lemma 4.5, Lemma 4.6 and Theorem 1.14.

Reminder of Theorem 1.15 *[Hardness of ℓ_2-Furthest Pair in $c^{\log^* n}$ Dimension] Assuming SETH (or OVC), there is a constant c such that ℓ_2-Furthest Pair in $c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

Reminder of Theorem 1.16 *[Hardness of Bichromatic ℓ_2-closest Pair in $c^{\log^* n}$ Dimension] Assuming SETH (or OVC), there is a constant c such that Bichromatic ℓ_2-Closest Pair in $c^{\log^* n}$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

5 NP · UPP communication protocol and Exact Hardness for $\mathbb{Z}$-Max-IP

We note that the inapproximability results for (Boolean) Max-IP is established via a connection to the MA communication complexity protocol of Set-Disjointness [5]. In the light of this, in this section we view our reduction from OV to $\mathbb{Z}$-Max-IP (Lemma 1.17 and Theorem 4.3) in the perspective of communication complexity.

We observe that in fact, our reduction can be understood as an NP · UPP communication protocol for Set Disjointness. Moreover, we show that if we can get a slightly better NP · UPP communication protocol for Set-Disjointness, then we would be able to prove $\mathbb{Z}$-Max-IP is hard even for $\omega(1)$ dimensions (and also ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair).

5.1 NP · UPP Communication Protocol for Set-Disjointness

First, we rephrase the results of Lemma 1.17 and Theorem 4.3 in a more convenience way for our use here.

▶ **Lemma 5.1** (Rephrasing of Lemma 1.17 and Theorem 4.3). *Let $1 \le \ell \le d$, and $m = \ell^{O(6^{\log^* d} \cdot (d/\ell))}$. There exists a family of functions*

$$\psi^i_{\mathsf{Alice}}, \psi^i_{\mathsf{Bob}} : \{0,1\}^d \to \mathbb{R}^{(\ell+1)^2}$$

for $i \in [m]$ such that:

- *when $x \cdot y = 0$, there is an i such that $\psi^i_{\mathsf{Alice}}(x) \cdot \psi^i_{\mathsf{Bob}}(y) \ge 0$;*
- *when $x \cdot y > 0$, for all i $\psi^i_{\mathsf{Alice}}(x) \cdot \psi^i_{\mathsf{Bob}}(y) < 0$;*
- *all $\psi^i_{\mathsf{Alice}}(x)$ and $\psi^i_{\mathsf{Bob}}(y)$ can be computed in $\mathrm{poly}(d)$ time.*

We also need the standard connection between UPP communication protocols and sign-rank [60] (see also Chapter 4.11 of [48]).

▶ **Lemma 5.2** (Equivalence of sign-rank and UPP communication protocol [60]). *The following statements are equivalent:*

- *There is a d-cost UPP communication protocol for a problem $F : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, where $\mathcal{X}$ and $\mathcal{Y}$ are input sets of Alice and Bob respectively.*
- *There are mappings $\psi^{\mathcal{X}} : \mathcal{X} \to \mathbb{R}^{2^d}$ and $\psi^{\mathcal{Y}} : \mathcal{Y} \to \mathbb{R}^{2^d}$ such that for all $(x,y) \in \mathcal{X} \times \mathcal{Y}$:*
  - *if $F(x,y) = 1$, $\psi^{\mathcal{X}}(x) \cdot \psi^{\mathcal{Y}}(y) \ge 0$;*
  - *otherwise, $\psi^{\mathcal{X}}(x) \cdot \psi^{\mathcal{Y}}(y) < 0$.*

From the above lemmas, we immediately get the needed communication protocol and prove Theorem 1.21 (restated below for convenience).

Reminder of Theorem 1.21 *For all* $1 \le \alpha \le n$*, there is an*

$$\left(\alpha \cdot 6^{\log^* n} \cdot (n/2^\alpha), O(\alpha)\right)\text{-computational-efficient}$$

$\mathsf{NP} \cdot \mathsf{UPP}$ *communication protocol for* DISJ_n.

Proof Sketch. We set $\alpha = \log \ell$ here. Given the function families $\{\psi^i_{\mathsf{Alice}}\}, \{\psi^i_{\mathsf{Bob}}\}$ from Lemma 5.1, Merlin just sends the index $i \in [m]$ and the rest follows from Lemma 5.2. ◀

5.2 Slightly Better Protocols Imply Hardness in $\omega(1)$ Dimensions

Finally, we show that if we have a slightly better $\mathsf{NP} \cdot \mathsf{UPP}$ protocol for Set-Disjointness, then we can show $\mathbb{Z}$-Max-IP requires $n^{2-o(1)}$ time even for $\omega(1)$ dimensions (and so do ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair). We restate Theorem 1.22 here for convenience.

Reminder of Theorem 1.22 *Assuming SETH (or OVC), if there is an increasing and unbounded function* f *such that for all* $1 \le \alpha \le n$*, there is a*

$$(n/f(\alpha), \alpha)\text{-computational-efficient}$$

$\mathsf{NP} \cdot \mathsf{UPP}$ *communication protocol for* DISJ_n*, then* $\mathbb{Z}$*-Max-IP*$_{n,\omega(1)}$ *requires* $n^{2-o(1)}$ *time with vectors of* polylog(n)*-bit entries. The same holds for* ℓ_2*-Furthest Pair and Bichromatic* ℓ_2*-Closest Pair.*

Proof. Suppose otherwise, there is an algorithm $\mathbb{A}$ for $\mathbb{Z}$-Max-IP$_{n,d}$ running in $n^{2-\varepsilon_1}$ time for all constant d and for a constant $\varepsilon_1 > 0$ (note for the sake of Lemma 4.5 and Lemma 4.6, we only need to consider $\mathbb{Z}$-Max-IP here).

Now, let c be an arbitrary constant, we are going to construct an algorithm for $\mathsf{OV}_{n,c\log n}$ in $n^{2-\Omega(1)}$ time, which contradicts OVC.

Let $\varepsilon = \varepsilon_1/2$, and α be the first number such that $c/f(\alpha) < \varepsilon$, note that α is also a constant. Consider the $(c \log n/f(\alpha), \alpha)$-computational-efficient $\mathsf{NP} \cdot \mathsf{UPP}$ protocol Π for $\mathsf{DISJ}_{c\log n}$, and let A, B be the two sets in the $\mathsf{OV}_{n,c\log n}$ instance. Our algorithm via reduction works as follows:

- There are 2^α possible messages in $\{0,1\}^\alpha$, let $m_1, m_2, \ldots, m_{2^\alpha}$ be an enumeration of them.
- We first enumerate all possible advice strings from Merlin in Π, there are $2^{c\log n/f(\alpha)} \le 2^{\varepsilon \cdot \log n} = n^\varepsilon$ such strings, let $\phi \in \{0,1\}^{\varepsilon \cdot \log n}$ be such an advice string.
  - For each $x \in A$, let $\psi_{\mathsf{Alice}}(x) \in \mathbb{R}^{2^\alpha}$ be the probabilities that Alice accepts each message from Bob. That is, $\psi_{\mathsf{Alice}}(x)_i$ is the probability that Alice accepts the message m_i, given its input x and the advice ϕ.
  - Similarly, for each $y \in B$, let $\psi_{\mathsf{Bob}}(y) \in \mathbb{R}^{2^\alpha}$ be the probabilities that Bob sends each message. That is, $\psi_{\mathsf{Bob}}(y)_i$ is the probability that Bob sends the message m_i, give its input y and the advice ϕ.
  - Then, for each $x \in A$ and $y \in B$, $\psi_{\mathsf{Alice}}(x) \cdot \psi_{\mathsf{Bob}}(y)$ is precisely the probability that Alice accepts at the end when Alice and Bob holds x and y correspondingly and the advice is ϕ. Now we let A_ϕ be the set of all the $\psi_{\mathsf{Alice}}(x)$'s, and B_ϕ be the set of all the $\psi_{\mathsf{Bob}}(y)$'s.

- If there is a ϕ such that $\mathsf{OPT}(A_\phi, B_\phi) \geq 1/2$, then we output yes, and otherwise output no.

From the definition of Π, it is straightforward to see that the above algorithm solves $\mathsf{OV}_{n,c\cdot\log n}$. Moreover, notice that from the computational-efficient property of Π, the reduction itself works in $n^{1+\varepsilon} \cdot \operatorname{polylog}(n)$ time, and all the vectors in A_ϕ's and B_ϕ's have at most $\operatorname{polylog}(n)$ bit precision, which means $\mathsf{OPT}(A_\phi, B_\phi)$ can be solved by a call to $\mathbb{Z}$-Max-$\mathsf{IP}_{n,2^\alpha}$ with vectors of $\operatorname{polylog}(n)$-bit entries.

Hence, the final running time for the above algorithm is bounded by $n^\varepsilon \cdot n^{2-\varepsilon_1} = n^{2-\varepsilon}$ (2^α is still a constant), which contradicts the OVC. ◀

6 Improved MA Protocols

In this section we prove Theorem 1.24 (restated below for convenience).

Reminder of Theorem 1.24 *There is an* MA *protocol for DISJ$_n$ and IP$_n$ with communication complexity*

$$O\left(\sqrt{n \log n \log\log n}\right).$$

To prove Theorem 1.24, we need the following intermediate problem.

▶ **Definition 6.1** (The Inner Product Modulo p Problem (IP_n^p)). Let p and n be two positive integers, in IP_n^p, Alice and Bob are given two vectors X and Y in $\{0,1\}^n$, and they want to compute $X \cdot Y \pmod{p}$.

Note that IP_n and IP_n^p are not Boolean functions, so we need to generalize the definition of an MA protocol. In an MA protocol for IP_n, Merlin sends the answer directly to Alice together with a proof to convince Alice and Bob. The correctness condition becomes that for the right answer $X \cdot Y$, Merlin has a proof such that Alice and Bob will accept with high probability (like 2/3). And the soundness condition becomes that for the wrong answers, every proof from Merlin will be rejected with high probability.

We are going to use the following MA protocol for IP_n^p, which is a slight adaption from the protocol in [64].

▶ **Lemma 6.2** (Implicit in Theorem 3.1 of [64]). *For a sufficiently large prime q and integers T and n, there is an*

$$\left(O\left(n/T \cdot \log q\right), \log n + O(1), O\left(T \cdot \log q\right), 1/2\right)\text{-}\mathit{efficient}$$

MA *protocol for* IP_n^q.

Proof Sketch. The only adaption is that we just use the field $\mathbb{F}_{q^2}$ with respect to the given prime q. (In the original protocol it is required that $q \geq T$.) ◀

Now we ready to prove Theorem 1.24.

Proof of Theorem 1.24. Since a IP_n protocol trivially implies a DISJ_n protocol, we only need to consider IP_n in the following.

Now, let x be the number such that $x^x = n$, for convenience we are going to pretend that x is an integer. It is easy to see that $x = \Theta(\log n / \log\log n)$. Then we pick $10x$

distinct primes $p_1, p_2, \ldots, p_{10x}$ in $[x+1, x^2]$ (we can assume that n is large enough to make x satisfy the requirement of Lemma 2.4). Let T be a parameter, we use Π_{p_i} to denote the $\Big(O\left(n/T \cdot \log p_i\right), \log n + O(1), O\left(T \cdot \log p_i\right), 1/2\Big)$-efficient MA protocol for $\mathsf{IP}_n^{p_i}$.

Our protocol for IP_n works as follows:

- Merlin sends Alice all the advice strings from the protocols $\Pi_{p_1}, \Pi_{p_2}, \ldots, \Pi_{p_{10x}}$, together with a presumed inner product $0 \le z \le n$.
- Note that Π_{p_i} contains the presumed value of $X \cdot Y \pmod{p_i}$, Alice first checks whether z is consistent with all these Π_{p_i}'s, and rejects immediately if it does not.
- Alice and Bob jointly toss $O(\log(10x))$ coins, to pick a uniform random number $i^\star \in [10x]$, and then they simulate $\Pi_{p_{i^\star}}$. That is, they pretend they are the Alice and Bob in the protocol $\Pi_{p_{i^\star}}$ with the advice from Merlin in $\Pi_{p_{i^\star}}$ (which Alice does have).

Correctness. Let $X, Y \in \{0,1\}^n$ be the vectors of Alice and Bob. If $X \cdot Y = z$, then by the definition of these protocols Π_{p_i}'s, Alice always accepts with the correct advice from Merlin.

Otherwise, let $d = X \cdot Y \neq z$, we are going to analyze the probability that we pick a "good" $p_{i^\star}$ such that $p_{i^\star}$ does not divide $|d - z|$. Since $p_i > x$ for all p_i's and $x^x > n \ge |d - z|$, $|d - z|$ cannot be a multiplier of more than x primes in p_i's.

Therefore, with probability at least 0.9, our pick of $p_{i^\star}$ is good. And in this case, from the definition of the protocols Π_{p_i}'s, Alice and Bob would reject afterward with probability at least 1/2, as $d \pmod{p_{i^\star}}$ differs from $z \pmod{p_{i^\star}}$. In summary, when $X \cdot Y \neq z$, Alice rejects with probability at least $0.9/2 = 0.45$, which finishes the proof for the correctness.

Complexity. Now, note that the total advice length is

$$O\left(n/T \cdot \sum_{i=1}^{10x} \log p_i\right) = O\left(n/T \cdot \log \prod_{i=1}^{10x} x^2\right) = O\left(n/T \cdot \log x^{20x}\right) = O\left(n/T \cdot \log n\right).$$

And the communication complexity between Alice and Bob is bounded by

$$O\left(T \cdot \log x^2\right) = O\left(T \cdot \log\log n\right).$$

Setting $T = \sqrt{n \log n / \log\log n}$ balances the above two quantities, and we obtain the needed MA-protocol for DISJ_n. ◀

7 Future Works

We end our paper by discussing a few interesting research directions.

1. The most important open question from this paper is that can we further improve the dimensionality reduction for OV? It is certainly weird to consider $2^{O(\log^* n)}$ to be the right answer for the limit of the dimensionality reduction. This term seems more like a product of the nature of our recursive construction and not the problem itself. We conjecture that there should be an $\omega(1)$ dimensional reduction with a more direct construction.
   One possible direction is to combine the original polynomial-based construction from [72] together with our new number theoretical one. These two approaches seem completely different, hence a clever combination of them may solve our problem.
2. In order to prove $\omega(1)$ dimensional hardness for ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair, we can also bypass the OV dimensionality reduction things by proving $\omega(1)$ dimensional hardness for $\mathbb{Z}$-Max-IP directly. One possible way to approach this question is to start from the $\mathsf{NP} \cdot \mathsf{UPP}$ communication protocol connection as in Section 5

(apply Theorem 1.22), and (potentially) draw some connections from some known UPP communication protocols.

3. We have seen an efficient reduction from $\mathbb{Z}$-OV to $\mathbb{Z}$-$\mathsf{Max\text{-}IP}$ which only blows up the dimension quadratically, is there a similar reduction from $\mathbb{Z}$-$\mathsf{Max\text{-}IP}$ back to $\mathbb{Z}$-OV? Are $\mathbb{Z}$-$\mathsf{Max\text{-}IP}$ and $\mathbb{Z}$-OV equivalent?
4. By making use of the new AG-code based MA protocols, we can shave a $\widetilde{O}(\sqrt{\log n})$ factor from the communication complexity, can we obtain an $O(\sqrt{n})$ MA communication protocol matching the lower bound for DISJ_n? It seems new ideas are required.

   Since our MA protocol works for both DISJ and IP, and IP does seems to be a harder problem. It may be better to find an MA protocol only works for DISJ. It is worth noting that an $O(\sqrt{n})$ AMA communication protocol for DISJ is given by [64], which doesn't work for IP.
5. Can the dependence on ε in the algorithms from Theorem 1.5 be further improved? Is it possible to apply ideas in the $n^{2-1/\widetilde{\Omega}(\sqrt{c})}$ algorithm for $\mathsf{Max\text{-}IP}_{n,c\log n}$ from [13]?
6. For the complexity of 2-multiplicative-approximation to $\mathsf{Max\text{-}IP}_{n,c\log n}$, Theorem 1.5 implies that there is an algorithm running in $n^{2-1/O(\log c)}$ time, the same as the best algorithm for $\mathsf{OV}_{n,c\log n}$ [9]. Is this just a coincidence? Or are there some connections between these two problems?
7. We obtain a connection between hardness of $\mathbb{Z}$-$\mathsf{Max\text{-}IP}$ and $\mathsf{NP} \cdot \mathsf{UPP}$ communication protocols for Set-Disjointness. Can we get similar connections from other $\mathsf{NP} \cdot \mathcal{C}$ type communication protocols for Set-Disjointness? Some candidates include $\mathsf{NP} \cdot \mathsf{SBP}$ and $\mathsf{NP} \cdot \mathsf{promiseBQP}$ (QCMA).

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *TOCT*, 1(1):2:1–2:54, 2009. doi:10.1145/1490270.1490272.

2 Amir Abboud and Arturs Backurs. Towards hardness of approximation for polynomial time problems. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 67. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

3 Amir Abboud and Søren Dahlgaard. Popular conjectures as a barrier for dynamic planar graph algorithms. In *Proceedings of the IEEE 57th Annual Symposium on Foundations of Computer Science*, pages 477–486, 2016.

4 Amir Abboud and Aviad Rubinstein. Fast and deterministic constant factor approximation algorithms for lcs imply new circuit lower bounds. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 94. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

5 Amir Abboud, Aviad Rubinstein, and R. Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 25–36. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.12.

6 Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *Proc. of the 55th FOCS*, pages 434–443, 2014.

7 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *Proc. of the 41st ICALP*, pages 39–51, 2014.

8 Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching triangles and basing hardness on an extremely popular conjecture. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 41–50. ACM, 2015.

9 Amir Abboud, Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 218–230. Society for Industrial and Applied Mathematics, 2015.

10 Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *Proceedings of the IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 434–443, 2014.

11 Pankaj K Agarwal, Herbert Edelsbrunner, Otfried Schwarzkopf, and Emo Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. *Discrete & Computational Geometry*, 6(3):407–422, 1991.

12 Thomas Dybdahl Ahle, Rasmus Pagh, Ilya Razenshteyn, and Francesco Silvestri. On the complexity of inner product similarity join. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 151–164. ACM, 2016.

13 Josh Alman, Timothy M Chan, and Ryan Williams. Polynomial representations of threshold functions and algorithmic applications. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 467–476. IEEE, 2016.

14 Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. In *Proc. of the 56th FOCS*, pages 136–150. IEEE, 2015.

15 Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Proc. of the 47th FOCS*, pages 459–468. IEEE, 2006.

16 Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. Practical and optimal lsh for angular distance. In *Advances in Neural Information Processing Systems*, pages 1225–1233, 2015.

17 Alexandr Andoni, Piotr Indyk, Huy L Nguyen, and Ilya Razenshteyn. Beyond locality-sensitive hashing. In *Proc. of the 25th SODA*, pages 1018–1028. SIAM, 2014.

18 Alexandr Andoni and Ilya Razenshteyn. Optimal data-dependent hashing for approximate near neighbors. In *Proc. of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 793–801. ACM, 2015.

19 Tom M. Apostol. *Introduction to analytic number theory*. Springer Science & Business Media, 2013.

20 Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. URL: `http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264`.

21 Arturs Backurs and Piotr Indyk. Edit Distance Cannot Be Computed in Strongly Subquadratic Time (unless SETH is false). In *Proc. of the 47th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 51–58, 2015.

22 Arturs Backurs and Piotr Indyk. Which regular expression patterns are hard to match? In *Proc. of the 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 457–466, 2016.

23 Jon Louis Bentley and Michael Ian Shamos. Divide-and-conquer in multidimensional space. In *Proceedings of the eighth annual ACM symposium on Theory of computing*, pages 220–230. ACM, 1976.

24 Karl Bringman and Marvin Künnemann. Multivariate fine-grained complexity of longest common subsequence. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1216–1235. SIAM, 2018.

25 Karl Bringmann. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *Proc. of the 55th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 661–670, 2014.

26 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A dichotomy for regular expression membership testing. *arXiv preprint arXiv:1611.00918*, 2016.

27 Harry Buhrman, Richard Cleve, Ronald De Wolf, and Christof Zalka. Bounds for small-error and zero-error quantum algorithms. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 358–368. IEEE, 1999.

28 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 63–68. ACM, 1998.

29 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In *IWPEC*, volume 5917, pages 75–85. Springer, 2009.

30 Timothy M Chan. A (slightly) faster algorithm for klee's measure problem. In *Proceedings of the twenty-fourth annual symposium on Computational geometry*, pages 94–100. ACM, 2008.

31 Tobias Christiani. A framework for similarity search with space-time tradeoffs using locality-sensitive filtering. In *Proc. of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 31–46. SIAM, 2017.

32 Tobias Christiani and Rasmus Pagh. Set similarity search beyond minhash. *arXiv preprint arXiv:1612.07710*, 2016.

33 Don Coppersmith. Rapid multiplication of rectangular matrices. *SIAM Journal on Computing*, 11(3):467–471, 1982.

34 Svyatoslav Covanov and Emmanuel Thomé. Fast integer multiplication using generalized fermat primes. *arXiv preprint arXiv:1502.02800*, 2015.

35 Roee David, CS Karthik, and Bundit Laekhanukit. On the complexity of closest pair via polar-pair of point-sets. *CoRR, abs/1608.03245*, 2016.

36 Ronald de Wolf. A note on quantum algorithms and the minimal degree of epsilon-error polynomials for symmetric functions. *arXiv preprint arXiv:0802.1816*, 2008.

37 Martin Dietzfelbinger, Torben Hagerup, Jyrki Katajainen, and Martti Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1):19–51, 1997.

38 Martin Fürer. Faster integer multiplication. *SIAM Journal on Computing*, 39(3):979–1005, 2009.

39 Francois Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1029–1046. SIAM, 2018.

40 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and R. Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. In *Proc. of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2162–2181, 2017.

41 Isaac Goldstein, Tsvi Kopelowitz, Moshe Lewenstein, and Ely Porat. Conditional lower bounds for space/time tradeoffs. In Faith Ellen, Antonina Kolokolova, and Jörg-Rüdiger Sack, editors, *Algorithms and Data Structures*, pages 421–436, Cham, 2017. Springer International Publishing.

42 Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219. ACM, 1996.

43 David Harvey, Joris Van Der Hoeven, and Grégoire Lecerf. Even faster integer multiplication. *Journal of Complexity*, 36:1–30, 2016.

44 Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-

vector multiplication conjecture. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, pages 21–30, 2015.

45 Monika Henzinger, Andrea Lincoln, Stefan Neumann, and Virginia Vassilevska Williams. Conditional hardness for sensitivity problems. *arXiv preprint arXiv:1703.01638*, 2017.

46 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

47 Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proc. of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

48 Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

49 Matti Karppa, Petteri Kaski, and Jukka Kohonen. A faster subquadratic algorithm for finding outlier correlations. In *Proc. of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1288–1305. Society for Industrial and Applied Mathematics, 2016.

50 C.S. Karthik, Bundit Laekhanukit, and Pasin Manurangsi. On the parameterized complexity of approximating dominating set. *arXiv preprint arXiv:1711.11029*, 2017.

51 Samir Khuller and Yossi Matias. A simple randomized sieve algorithm for the closest-pair problem. *Information and Computation*, 118(1):34–37, 1995.

52 Hartmut Klauck. Rectangle size bounds and threshold covers in communication complexity. In *Computational Complexity, 2003. Proceedings. 18th IEEE Annual Conference on*, pages 118–134. IEEE, 2003.

53 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In *Proceedings of the 27th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1272–1287, 2016.

54 Robert Krauthgamer and Ohad Trabelsi. Conditional lower bounds for all-pairs max-flow. *arXiv preprint arXiv:1702.05805*, 2017.

55 Jiří Matoušek. Efficient partition trees. *Discrete & Computational Geometry*, 8(3):315–334, 1992.

56 Jiří Matoušek. Range searching with efficient hierarchical cuttings. *Discrete & Computational Geometry*, 10(2):157–182, 1993.

57 Behnam Neyshabur and Nathan Srebro. On symmetric and asymmetric lshs for inner product search. In *Proc. of the 32nd International Conference on Machine Learning, ICML*, pages 1926–1934, 2015.

58 Mihai Patrascu. Towards polynomial lower bounds for dynamic problems. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 603–610, 2010.

59 Mihai Pătraşcu and Ryan Williams. On the possibility of faster sat algorithms. In *Proc. of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1065–1075. SIAM, 2010.

60 Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986.

61 Ali Rahimi, Benjamin Recht, et al. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5, 2007.

62 Parikshit Ram and Alexander G Gray. Maximum inner-product search using cone trees. In *Proc. of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 931–939. ACM, 2012.

63 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Proc. of the 45th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 515–524, 2013.

64 Aviad Rubinstein. Hardness of approximate nearest neighbor search. In *STOC*, page To appear, 2018.

65 Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). In *Advances in Neural Information Processing Systems*, pages 2321–2329, 2014.

66 Anshumali Shrivastava and Ping Li. Asymmetric minwise hashing for indexing binary inner products and set containment. In *Proc. of the 24th International Conference on World Wide Web*, pages 981–991. ACM, 2015.

67 Christina Teflioudi and Rainer Gemulla. Exact and approximate maximum inner product search with lemp. *ACM Transactions on Database Systems (TODS)*, 42(1):5, 2016.

68 Gregory Valiant. Finding correlations in subquadratic time, with applications to learning parities and the closest pair problem. *Journal of the ACM (JACM)*, 62(2):13, 2015.

69 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *To appear in the proceedings of the ICM*, 2018.

70 R. Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2–3):357–365, 2005.

71 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 664–673. ACM, 2014.

72 Ryan Williams. On the difference between closest, furthest, and orthogonal pairs: Nearly-linear vs barely-subquadratic complexity. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1207–1215. SIAM, 2018. doi:10.1137/1.9781611975031.78.

73 Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1867–1877. SIAM, 2014.

74 Andrew Chi-Chih Yao. On constructing minimum spanning trees in k-dimensional spaces and related problems. *SIAM Journal on Computing*, 11(4):721–736, 1982.

A A Dimensionality Reduction for Max-IP

In fact, tracing the proof of Theorem 4.1, we observe that it is possible to compute the inner product $x \cdot y$ itself from $\psi_{b,\ell}(x) \cdot \psi_{b,\ell}(y)$, that is:

▶ **Corollary A.1.** *Let b, ℓ be two sufficiently large integers. There is a reduction $\psi_{b,\ell} : \{0,1\}^{b\cdot\ell} \to \mathbb{Z}^\ell$ and $b \cdot \ell + 1$ sets $V^0_{b,\ell}, V^1_{b,\ell}, \dots, V^{b\cdot\ell}_{b,\ell} \subseteq \mathbb{Z}$, such that for every $x, y \in \{0,1\}^{b\cdot\ell}$,*

$$x \cdot y = k \Leftrightarrow \psi_{b,\ell}(x) \cdot \psi_{b,\ell}(y) \in V^k_{b,\ell} \quad \text{for all } 0 \le k \le b \cdot \ell,$$

and

$$0 \le \psi_{b,\ell}(x)_i < \ell^{6^{\log^*(b)} \cdot b}$$

for all possible x and $i \in [\ell]$. Moreover, the computation of $\psi_{b,\ell}(x)$ takes $\mathrm{poly}(b \cdot \ell)$ time, and the sets $V^k_{b,\ell}$'s can be constructed in $O\left(\ell^{O(6^{\log^*(b)} \cdot b)} \cdot \mathrm{poly}(b \cdot \ell)\right)$ time.

Together with Theorem 4.3, it proves Corollary 4.4 (restated below).

Reminder of Corollary 4.4 *Let $1 \le \ell \le d$. There is an*

$$O\left(n \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))} \cdot \mathrm{poly}(d)\right) \text{-time}$$

reduction from Max-IP$_{n,d}$ *to* $d \cdot \ell^{O(6^{\log^* d} \cdot (d/\ell))}$ *instances of* $\mathbb{Z}$-Max-IP$_{n,(\ell+1)^2}$*, with vectors of entries with bit-length* $O\left(d/\ell \cdot \log \ell \cdot 6^{\log^* d}\right)$.

Proof Sketch. Let $b = d/\ell$ (assume ℓ divides d here for simplicity), A and B be the sets in the given Max-IP$_{n,d}$ instance, we proceed similarly as the case for OV.

We first enumerate a number k from 0 to d, for each k we construct the set $V_{b,\ell}^{k}$ as specified in Corollary A.1. Then there is $(x, y) \in A \times B$ such that $x \cdot y = k$ if and only if there is $(x, y) \in A \times B$ such that $\psi_{b,\ell}(x) \cdot \psi_{b,\ell}(y) \in V_{b,\ell}^{k}$. Using exactly the same reduction as in Lemma 1.17, we can in turn reduce this into $\ell^{O(6^{\log^*(b)} \cdot b)}$ instances of $\mathbb{Z}$-OV$_{n,\ell+1}$.

Applying Theorem 4.3, with evaluation of $(d+1) \cdot \ell^{O(6^{\log^*(b)} \cdot b)}$ $\mathbb{Z}$-Max-IP$_{n,(\ell+1)^2}$ instances, we can determine whether there is $(x, y) \in A \times B$ such that $x \cdot y = k$ for every k, from which we can compute the answer to the Max-IP$_{n,d}$ instance. ◀

B Nonuniform to Uniform Transformation for Dimensionality Reduction for OV

In this section we discuss the transformation from nonuniform construction to uniform one for dimensionality reduction for OV. In order to state our result formally, we need to introduce some definitions.

▶ **Definition B.1** (Nonuniform Reduction)**.** Let $b, \ell, \kappa \in \mathbb{N}$. We say a function $\varphi : \{0,1\}^{b \cdot \ell} \to \mathbb{Z}^{\ell}$ together with a set $V \subseteq \mathbb{Z}$ is a (b, ℓ, κ)-reduction, if the following holds:

- For every $x, y \in \{0,1\}^{b \cdot \ell}$,
$$x \cdot y = 0 \Leftrightarrow \varphi(x) \cdot \varphi(y) \in V.$$
- For every x and $i \in [\ell]$,
$$0 \leq \varphi(x)_i < \ell^{\kappa \cdot b}.$$

Similarly, let τ be an increasing function, we say a function family $\{\varphi_{b,\ell}\}_{b,\ell}$ together with a set family $\{V_{b,\ell}\}_{b,\ell}$ is a τ-reduction family, if for every b and ℓ, $(\varphi_{b,\ell}, V_{b,\ell})$ is a $(b, \ell, \tau(b))$-reduction.

Moreover, if for all b and all $\ell \leq \log\log\log b$, there is an algorithm $\mathbb{A}$ which computes $\varphi_{b,\ell}(x)$ in $\text{poly}(b)$ time given b, ℓ and $x \in \{0,1\}^{b \cdot \ell}$, and constructs the set $V_{b,\ell}$ in $O\left(\ell^{O(\tau(b) \cdot b)} \cdot \text{poly}(b)\right)$ time given b and ℓ, then we call $(\varphi_{b,\ell}, V_{b,\ell})$ a uniform-τ-reduction family.

▶ Remark B.2. The reason we assume ℓ to be small is that in our applications we only care about very small ℓ, and that greatly simplifies the notation. From Theorem 4.1, there is a uniform-$\left(6^{\log^* b}\right)$-reduction family, and a better uniform-reduction family implies better hardness for $\mathbb{Z}$-OV and other related problems as well (Lemma 1.17, Theorem 4.3, Lemma 4.6 and Lemma 4.5).

Now we are ready to state our nonuniform to uniform transformation result formally.

▶ **Theorem B.3.** *Letting τ be an increasing function such that $\tau(n) = O(\log\log\log n)$ and supposing there is a τ-reduction family, then there is a uniform-$O(\tau)$-reduction family.*

Proof Sketch. The construction in Theorem 4.1 is recursive, it constructs the reduction $\psi_{b,\ell}$ from a much smaller reduction $\psi_{b_{\mathsf{micro}},\ell}$, where $b_{\mathsf{micro}} \le \log b$. In the original construction, it takes $\log^* b$ recursions to make the problem sufficiently small so that a direct construction can be used. Here we only apply the reduction thrice. First let us abstract the following lemma from the proof of Theorem 4.1.

▶ **Lemma B.4** (Implicit in Theorem 4.1). *Letting $b, \ell, b_{\mathsf{micro}}, \kappa \in \mathbb{N}$ and supposing $\ell^{\kappa \cdot b_{\mathsf{micro}}} = b$ and there is a $(b_{\mathsf{micro}}, \ell, \kappa)$-reduction (φ, V'), the following holds:*

- *There is a $(b, \ell, 6 \cdot \kappa)$-reduction (ψ, V).*
- *Given (φ, V'), for all $x \in \{0,1\}^{b \cdot \ell}$, $\psi(x)$ can be computed in $\mathrm{poly}(b \cdot \ell)$, and V can be constructed in $O\left(\ell^{O(\kappa \cdot b)} \cdot \mathrm{poly}(b \cdot \ell)\right)$ time.*

Now, let $b, \ell \in \mathbb{N}$, we are going to construct our reduction as follows.

Let b_1 be the number such that

$$\ell^{\tau(b) \cdot 6^2 \cdot b_1} = b,$$

and similarly we set b_2 and b_3 so that

$$\ell^{\tau(b) \cdot 6 \cdot b_2} = b_1 \quad \text{and} \quad \ell^{\tau(b) \cdot b_3} = b_2.$$

We can calculate from above that $b_3 \le \log \log \log b$.

From the assumption that there is a τ-reduction, there is a $(b_3, \ell, \tau(b_3))$-reduction $(\varphi_{b_3,\ell}, V_{b_3,\ell})$, which is also a $(b_3, \ell, \tau(b))$-reduction, as τ is increasing. Note that we can assume $\ell \le \log \log \log b$ and $\tau(b) \le \log \log \log b$ from assumption. Now we simply use a brute force algorithm to find $(\varphi_{b_3,\ell}, V_{b_3,\ell})$. There are

$$\ell^{\tau(b) \cdot b_3 \cdot \ell \cdot 2^{b_3 \cdot \ell}} = b^{o(1)}$$

possible functions from $\{0,1\}^{b_3 \cdot \ell} \to \{0, \dots \ell^{\tau(b_3) \cdot b_3} - 1\}^{\ell}$. Given such a function φ, one can check in $\mathrm{poly}(2^{b_3 \cdot \ell}) = b^{o(1)}$ time that whether one can construct a corresponding set V to obtain our $(b_3, \ell, \tau(b))$-reduction.

Applying Lemma B.4 thrice, one obtain a $(b, \ell, O(\tau(b)))$-reduction (ψ, V). And since $\varphi_{b_3,\ell}$ can be found in $b^{o(1)}$ time, together with Lemma B.4, we obtain a uniform-τ-reduction family. ◀

Finally, we give a direct corollary of Theorem B.3 that the existence of an $O(1)$-reduction family implies hardness of $\mathbb{Z}$-OV, $\mathbb{Z}$-Max-IP, ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair in $\omega(1)$ dimensions.

▶ **Corollary B.5.** *If there is an $O(1)$-reduction family, then for every $\varepsilon > 0$, there exists a $c \ge 1$ such that $\mathbb{Z}$-OV, $\mathbb{Z}$-Max-IP, ℓ_2-Furthest Pair and Bichromatic ℓ_2-Closest Pair in c dimensions with $O(\log n)$-bit entries require $n^{2-\varepsilon}$ time.*

Proof Sketch. Note that since its hardness implies the harnesses of other three, we only need to consider $\mathbb{Z}$-OV here.

From Theorem B.3 and the assumption, there exists a uniform-$O(1)$-reduction. Proceeding similar as in Lemma 1.17 with the uniform-$O(1)$-reduction, we obtain a better dimensionality self reduction from OV to $\mathbb{Z}$-OV. Then exactly the same argument as in Theorem 1.18 with different parameters gives us the lower bound required. ◀

C Hardness of Approximate {-1,1}-Max-IP via Approximate Polynomial for OR

We first show that making use of the $O(\sqrt{n})$-degree approximate polynomial for OR [27, 36], OV can be reduced to approximating $\{-1,1\}$-Max-IP.

▶ **Theorem C.1.** *Letting $\varepsilon \in (0,1)$, there is an algorithm reducing an $\mathsf{OV}_{n,d}$ instance with sets A, B to a $\{-1,1\}$-Max-IP_{n,d_1} instance with sets $\widetilde{A}$ and $\widetilde{B}$, such that:*

- $d_1 = \binom{d}{\leq O\left(\sqrt{d \log 1/\varepsilon}\right)}^3 \cdot 2^{O\left(\sqrt{d \log 1/\varepsilon}\right)} \cdot \varepsilon^{-1}$*, in which the notation* $\binom{n}{\leq m}$ *denotes* $\sum_{i=0}^{m} \binom{n}{i}$.
- *There is an integer $T > \varepsilon^{-1}$ such that if there is an $(a,b) \in A \times B$ such that $a \cdot b = 0$, then $\mathit{OPT}(\widetilde{A}, \widetilde{B}) \geq T$.*
- *Otherwise, $|\mathit{OPT}(\widetilde{A}, \widetilde{B})| \leq T \cdot \varepsilon$.*
- *Moreover, the reduction takes $n \cdot \mathrm{poly}(d_1)$ time.*

We remark here that the above reduction fails to achieve a characterization: setting $\varepsilon = 1/2$ and $d = c \log n$ for an arbitrary constant c, we have $d_1 = 2^{\widetilde{O}(\sqrt{\log n})}$, much larger than $\log n$. Another interesting difference between the above theorem and Lemma 3.3 (the reduction from OV to approximating Max-IP) is that Lemma 3.3 reduces one OV instance to many Max-IP instances, while the above reduction only reduces it to one $\{-1,1\}$-Max-IP instance.

Proof of Theorem C.1.

Construction and Analysis of Polynomial $P_\varepsilon(z)$. By [27, 36], there is a polynomial $P_\varepsilon : \{0,1\}^d \to \mathbb{R}$ such that:

- P_ε is of degree $D = O\left(\sqrt{d \log 1/\varepsilon}\right)$.
- For every $z \in \{0,1\}^d$, $P_\varepsilon(z) \in [0,1]$.
- Given $z \in \{0,1\}^d$, if $\mathsf{OR}(z) = 0$, then $P_\varepsilon(z) \geq 1 - \varepsilon$, otherwise $P_\varepsilon(z) \leq \varepsilon$.
- P_ε can be constructed in time polynomial in its description size.

Now, let us analyze P_ε further. For a set $S \subseteq [d]$, let $\chi_S : \{0,1\}^d \to \mathbb{R}$ be $\chi_S(z) := \prod_{i \in S} (-1)^{z_i}$. Then we can write P_ε as:

$$P_\varepsilon := \sum_{S \subseteq [d], |S| \leq D} \chi_S \cdot \langle \chi_S, P_\varepsilon \rangle,$$

where $\langle \chi_S, P_\varepsilon \rangle$ is the inner product of χ_S and P_ε, defined as $\langle \chi_S, P_\varepsilon \rangle := \mathbb{E}_{x \in \{0,1\}^d} \chi_S(x) \cdot P_\varepsilon(x)$.

Let $c_S = \langle \chi_S, P_\varepsilon \rangle$, from the definition it is easy to see that $c_S \in [-1,1]$.

Discretization of Polynomial P_ε. Note that $P_\varepsilon(z)$ has real coefficients, we need to turn it into another polynomial with integer coefficients first.

Let $M := \binom{d}{\leq D}$, consider the following polynomial $\widehat{P}_\varepsilon$:

$$\widehat{P}_\varepsilon := \sum_{S \subseteq [d], |S| \leq D} \lfloor c_S \cdot 2M/\varepsilon \rfloor \cdot \chi_S.$$

We can see that $|\widehat{P}_\varepsilon(z)/(2M/\varepsilon) - P_\varepsilon(z)| \le \varepsilon$ for every $z \in \{0,1\}^d$, and we let $\hat{c}_S := \lfloor c_S \cdot M \cdot 2/\varepsilon \rfloor$ for convenience.

Simplification of Polynomial $\widehat{P}_\varepsilon$. $\widehat{P}_\varepsilon(z)$ is expressed over the basis χ_S's, we need to turn it into a polynomial over standard basis.

For each $S \subseteq [d]$, consider χ_S, it can also be written as:

$$\chi_S(z) = \prod_{i \in S} (-1)^{z_i} := \prod_{i \in S} (1 - 2z_i) = \sum_{T \subseteq S} (-2)^{|T|} z_T,$$

where $z_T := \prod_{i \in T} z_i$. Plugging it into the expression of $\widehat{P}_\varepsilon$, we have

$$\widehat{P}_\varepsilon(z) := \sum_{T \subseteq [d], |T| \le D} \left(\sum_{S \subseteq [d], |S| \le D, T \subseteq S} \hat{c}_S \right) \cdot (-2)^{|T|} z_T.$$

Set

$$\tilde{c}_T := \left(\sum_{S \subseteq [d], |S| \le D, T \subseteq S} \hat{c}_S \right) \cdot (-2)^{|T|},$$

the above simplifies to

$$\widehat{P}_\varepsilon(z) := \sum_{T \subseteq [d], |T| \le D} \tilde{c}_T \cdot z_T.$$

Properties of Polynomial $\widehat{P}_\varepsilon$. Let us summarize some properties of $\widehat{P}_\varepsilon$ for now. First we need a bound on $|\tilde{c}_T|$, we can see $|\hat{c}_S| \le M \cdot 2/\varepsilon$, and by a simple calculation we have

$$|\tilde{c}_T| \le M^2 \cdot 2^D \cdot 2/\varepsilon.$$

Let $B = M^2 \cdot 2^D \cdot 2/\varepsilon$ for convenience. For $x, y \in \{0,1\}^d$, consider $\widehat{P}_\varepsilon(x, y) := \widehat{P}_\varepsilon(x_1 y_1, x_2 y_2, \dots, x_d y_d)$ (that is, plugging in $z_i = x_i y_i$), we have

$$\widehat{P}_\varepsilon(x, y) := \sum_{T \subseteq [d], |T| \le D} \tilde{c}_T \cdot x_T \cdot y_T,$$

where $x_T := \prod_{i \in T} x_i$ and y_T is defined similarly. Moreover, we have

- If $x \cdot y = 0$, then $\widehat{P}_\varepsilon(x, y) \ge (2M/\varepsilon) \cdot (1 - 2\varepsilon)$.
- If $x \cdot y \ne 0$, then $|\widehat{P}_\varepsilon(x, y)| \le (2M/\varepsilon) \cdot 2\varepsilon$.

The Reduction. Now, let us construct the reduction, we begin with some notations. For two vectors a, b, we use $a \circ b$ to denote their concatenation. For a vector a and a real x, we use $a \cdot x$ to denote the vector resulting from multiplying each coordinate of a by x. Let $\operatorname{sgn}(x)$ be the sign function that outputs 1 when $x > 0$, -1 when $x < 0$, and 0 when $x = 0$. For $x \in \{-B, -B+1, \dots, B\}$, we use $e_x \in \{-1, 0, 1\}^B$ to denote the vector whose first $|x|$ elements are $\operatorname{sgn}(x)$ and the rest are zeros. We also use $\mathbf{1}$ to denote the all-1 vector with length B.

Let $T_1, T_2, \dots, T_M$ be an enumeration of all subsets $T \subseteq [d]$ such that $|T| \le D$, we define

$$\varphi_x(x) := \circ_{i=1}^M (e_{\tilde{c}_{T_i}} \cdot x_{T_i}) \text{ and } \varphi_y(y) := \circ_{i=1}^M (\mathbf{1} \cdot y_{T_i}).$$

And we have

$$\varphi_x(x) \cdot \varphi_y(y) = \sum_{i=1}^{M} (e_{\tilde{c}_{T_i}} \cdot \mathbf{1}) \cdot (x_{T_i} \cdot y_{T_i}) = \sum_{i=1}^{M} \tilde{c}_{T_i} \cdot x_{T_i} \cdot y_{T_i} = \widehat{P}_\varepsilon(x,y).$$

To move from $\{-1,0,1\}$ to $\{-1,1\}$, we use the following carefully designed reductions $\psi_x, \psi_y : \{-1,0,1\} \to \{-1,1\}^2$, such that

$$\psi_x(-1) = \psi_y(-1) = (-1,-1), \quad \psi_x(0) = (-1,1),$$

$$\psi_y(0) := (1,-1), \quad \text{and} \quad \psi_x(1) = \psi_y(1) = (1,1).$$

It is easy to check that for $x, y \in \{-1,0,1\}$, we have $\psi_x(x) \cdot \psi_y(y) = 2 \cdot (x \cdot y)$.

Hence, composing the above two reductions, we get our desired reductions $\phi_x = \psi_x^{\otimes(B \cdot M)} \circ \varphi_x$ and $\phi_y = \psi_y^{\otimes(B \cdot M)} \circ \varphi_y$ such that for $x, y \in \{0,1\}^d$, $\phi_x(x), \phi_y(y) \in \{-1,1\}^{2B \cdot M}$ and $\phi_x(x) \cdot \phi_y(y) = 2 \cdot \widehat{P}_\varepsilon(x,y)$.

Finally, given an $\mathsf{OV}_{n,d}$ instance with two sets A and B, we construct two sets $\widetilde{A}$ and $\widetilde{B}$, such that $\widetilde{A}$ consists of all $\phi_x(x)$'s for $x \in A$, and $\widetilde{B}$ consists of all $\phi_y(y)$'s for $y \in B$.

Then we can see $\widetilde{A}$ and $\widetilde{B}$ consist of n vectors from $\{-1,1\}^{d_1}$, where

$$d_1 = 2B \cdot M = M^3 \cdot 2^D \cdot 2/\varepsilon = \binom{d}{\leq O\left(\sqrt{d \log 1/\varepsilon}\right)}^3 \cdot 2^{O\left(\sqrt{d \log 1/\varepsilon}\right)} \cdot \varepsilon^{-1}$$

as stated.

It is not hard to see the above reduction takes $n \cdot \text{poly}(d_1)$ time. Moreover, if there is a $(x,y) \in A \times B$ such that $x \cdot y = 0$, then $\mathsf{OPT}(\widetilde{A}, \widetilde{B}) \geq (4M/\varepsilon) \cdot (1 - 2\varepsilon)$, otherwise, $\mathsf{OPT}(\widetilde{A}, \widetilde{B}) \leq (4M/\varepsilon) \cdot 2\varepsilon$. Setting ε above to be $1/3$ times the ε in the statement finishes the proof. ◀

With Theorem C.1, we are ready to prove our hardness results on $\{-1,1\}$-$\mathsf{Max\text{-}IP}$.

▶ **Theorem C.2.** *Assume SETH (or OVC). Letting $\alpha : \mathbb{N} \to \mathbb{R}$ be any function of n such that $\alpha(n) = n^{o(1)}$, there is another function β satisfying $\beta(n) = n^{o(1)}$ and an integer $T > \alpha$ (β and T depend on α), such that there is no $n^{2-\Omega(1)}$-time algorithm for $\{-1,1\}$-$\mathsf{Max\text{-}IP}_{n,\beta(n)}$ distinguishing the following two cases:*

- *$\mathsf{OPT}(A,B) \geq T$ (A and B are the sets in the $\{-1,1\}$-$\mathsf{Max\text{-}IP}$ instance).*
- *$|\mathsf{OPT}(A,B)| \leq T/\alpha(n)$.*

Proof. Letting $\alpha = n^{o(1)}$ and $k = \log \alpha / \log n$, we have $k = o(1)$. Setting $d = c \log n$ where c is an arbitrary constant and $\varepsilon = \alpha^{-1}$ in Theorem C.1, we have that an $\mathsf{OV}_{c \log n}$ reduces to a certain $\alpha(n)$-approximation to a $\{-1,1\}$-$\mathsf{Max\text{-}IP}_{n,d_1}$ instance with sets A and B, where

$$d_1 = \binom{c \log n}{\leq O(\sqrt{ck} \log n)}^3 \cdot 2^{O(\sqrt{ck} \log n)} \leq \left(\frac{\sqrt{c}}{\sqrt{k}}\right)^{O(\sqrt{ck} \log n)} \cdot 2^{O(\sqrt{ck} \log n)} = n^{O(\log(c/k) \cdot \sqrt{ck})}.$$

Now set $\beta = n^{k^{1/3}}$ and T be the integer specified by Theorem C.1, since $k = o(1)$, $\beta = n^{o(1)}$. Suppose otherwise there is an $n^{2-\Omega(1)}$-time algorithm for distinguishing whether $\mathsf{OPT}(A,B) \geq T$ or $|\mathsf{OPT}(A,B)| \leq T/\alpha(n)$. Then for any constant c, $O(\log(c/k)\sqrt{ck}) \leq k^{1/3}$ for sufficiently large n, which means $d_1 \leq \beta(n)$ for a sufficiently large n, and there is an $n^{2-\Omega(1)}$-time algorithm for $\mathsf{OV}_{c \log n}$ by Theorem C.1, contradiction to OVC. ◀

D A Proof of Lemma 3.3

Finally, we present a proof of Lemma 3.3, which is implicit in [64].

We need the following efficient MA protocol for Set-Disjointness from [64], which is also used in [50].[15]

▶ **Lemma D.1** (Theorem 3.2 of [64]). *For every α and m, there is an $(m/\alpha, \log_2 m, \mathrm{poly}(\alpha), 1/2)$-efficient* MA *protocol for* DISJ_m.

We want to reduce the error probability while keeping the number of total random coins relatively low. To achieves this, we can use an expander graph (Theorem 2.7) to prove the following theorem.

▶ **Lemma D.2.** *For every α, m and $\varepsilon < 1/2$, there is an $(m/\alpha, \log_2 m + O(\log \varepsilon^{-1}), \mathrm{poly}(\alpha) \cdot \log \varepsilon^{-1}, \varepsilon)$-efficient* MA *protocol for* DISJ_m.

Proof. Let c_1 and $\mathcal{F} : \{0,1\}^{\log m + c_1 \cdot \log \varepsilon^{-1}} \to [m]^{c_1 \cdot \log \varepsilon^{-1}}$ be the corresponding constant and function as in Theorem 2.7, and let Π denote the $(m/\alpha, \log_2 m, \mathrm{poly}(\alpha), 1/2)$-efficient MA protocol for DISJ_m in Lemma D.1. Set $q = c_1 \cdot \log \varepsilon^{-1}$ and our new protocol Π_{new} works as follows:

- Merlin still sends the same advice to Alice as in Π.
- Alice and Bob jointly toss $r = \log m + q$ coins to get a string $w \in \{0,1\}^r$. Then we let $w_1, w_2, \ldots, w_q$ be the sequence corresponding to $\mathcal{F}(w)$, each of them can be interpreted as $\log m$ bits.
- Bob sends Alice q messages, the i-th message m_i corresponds to Bob's message in Π when the random bits is w_i.
- After that, Alice decides whether to accept or not as follows:
  - If for every $i \in [q]$, Alice would accept Bob's message m_i with random bits w_i in Π, then Alice accepts.
  - Otherwise, Alice rejects.

It is easy to verify that the advice length, message length and number of random coins satisfy our requirements.

For the error probability, note that when these two sets are disjoint, the same advice in Π leads to acceptance of Alice. Otherwise, suppose the advice from Merlin is either wrong or these two sets are intersecting, then half of the random bits in $\{0,1\}^{\log m}$ leads to the rejection of Alice in Π. Hence, from Theorem 2.7, with probability at least $1 - \varepsilon$, at least one of the random bits w_i's would lead to the rejection of Alice, which completes the proof. ◀

Finally, we prove Lemma 3.3 (restated below).

Reminder of Lemma 3.3 *There is a universal constant c_1 such that, for every integer c, reals $\varepsilon \in (0,1]$ and $\tau \geq 2$, $\mathsf{OV}_{n, c\log n}$ can be reduced to n^{ε} $\mathsf{Max}\text{-}\mathsf{IP}_{n,d}$ instances (A_i, B_i) for $i \in [n^{\varepsilon}]$, such that:*

- $d = \tau^{\mathrm{poly}(c/\varepsilon)} \cdot \log n$.
- *Letting $T = c \log n \cdot \tau^{c_1}$, if there is $a \in A$ and $b \in B$ such that $a \cdot b = 0$, then there exists an i such that $\mathsf{OPT}(A_i, B_i) \geq T$.*
- *Otherwise, for all i we must have $\mathsf{OPT}(A_i, B_i) \leq T/\tau$.*

[15] The protocol in [50] also works for the k-party number-in-hand model.

Proof. The reduction follows exactly the same as in [5], we recap here for completeness.

Set $\alpha = c/\varepsilon$, $m = c{\cdot}\log n$ and $\varepsilon = 1/\tau$, and let Π be the $(m/\alpha, \log_2 m + O(\log \varepsilon^{-1}), \text{poly}(\alpha) \cdot \log \varepsilon^{-1}, \varepsilon)$-efficient MA protocol for Set-Disjointness as in Lemma D.2.

Now, we first enumerate all of $2^{m/\alpha} = 2^{\varepsilon \cdot \log n} = n^{\varepsilon}$ possible advice strings, and create an Max-IP instance for each of the advice strings.

For a fix advice $\psi \in \{0,1\}^{\varepsilon \cdot \log n}$, we create an Max-IP instance with sets A_ψ and B_ψ as follows. We use $a \circ b$ to denote the concatenation of the strings a and b.

Let $r = \log_2 m + c_1 \cdot \log \varepsilon^{-1}$, where c_1 is the constant hidden in the big O notation in Lemma D.2, and $\ell = \text{poly}(\alpha) \cdot \log \varepsilon^{-1}$. Let $m_1, m_2, \ldots, m_{2^\ell}$ be an enumeration of all strings in $\{0,1\}^\ell$.

- For each $a \in A$, and for each string $w \in \{0,1\}^r$, we create a vector $a^w \in \{0,1\}^{2^\ell}$, such that a^w_i indicates that given advice ψ and randomness w, whether Alice accepts message m_i or not (1 for acceptance, 0 for rejection). Let the concatenation of all these a^w's be a_ψ. Then A_ψ is the set of all these a_ψ's for $a \in A$.
- For each $b \in B$, and for each string $w \in \{0,1\}^r$, we create a vector $b^w \in \{0,1\}^{2^\ell}$, such that $b^w_i = 1$ if Bob sends the message m_i given advice ψ and randomness w, and $= 0$ otherwise. Let the concatenation of all these b^w's be b_ψ. Then B_ψ is the set of all these b_ψ's for $b \in B$.

We can see that for $a \in A$ and $b \in B$, $a_\psi \cdot b_\psi$ is precisely the number of random coins leading Alice to accept the message from Bob given advice ψ when Alice and Bob holds a and b correspondingly. Therefore, let $T = 2^r = c \log n \cdot \tau^{c_1}$, from the properties of the protocol Π, we can see that:

- If there is $a \in A$ and $b \in B$ such that $a \cdot b = 0$, then there is $\psi \in \{0,1\}^{\varepsilon \cdot \log n}$ such that $a_\psi \cdot b_\psi \geq T$.
- Otherwise, for all $a \in A$, $b \in B$ and advice $\psi\{0,1\}^{\varepsilon \cdot \log n}$, $a_\psi \cdot b_\psi \leq T/\tau$.

And this completes the proof. ◀