On the Difference Between Closest, Furthest, and Orthogonal Pairs: Nearly-Linear vs Barely-Subquadratic Complexity[*]

Ryan Williams[†]

Abstract

Point location problems for n points in d-dimensional Euclidean space (and ℓ_p spaces more generally) have typically had two kinds of running-time solutions:

(Nearly-Linear) less than $d^{\mathrm{poly}(d)} \cdot n \log^{O(d)} n$ time, or

(Barely-Subquadratic) $f(d) \cdot n^{2-1/\Theta(d)}$ time, for various f.

For small d and large n, "nearly-linear" running times are generally feasible, while the "barely-subquadratic" times are generally infeasible, requiring essentially quadratic time. For example, in the Euclidean metric, finding a Closest Pair among n points in $\mathbb{R}^d$ is nearly-linear, solvable in $2^{O(d)} \cdot n \log^{O(1)} n$ time, while the known algorithms for finding a Furthest Pair (the diameter of the point set) are only barely-subquadratic, requiring $\Omega(n^{2-1/\Theta(d)})$ time. Why do these proximity problems have such different time complexities? Is there a barrier to obtaining nearly-linear algorithms for problems which are currently only barely-subquadratic?

We give a novel exact and deterministic self-reduction for the Orthogonal Vectors problem on n vectors in $\{0,1\}^d$ to n vectors in $\mathbb{Z}^{\omega(\log d)}$ that runs in $2^{o(d)}$ time. As a consequence, barely-subquadratic problems such as Euclidean diameter, Euclidean bichromatic closest pair, and incidence detection do not have $O(n^{2-\epsilon})$ time algorithms (in Turing models of computation) for dimensionality $d = \omega(\log \log n)^2$, unless the popular Orthogonal Vectors Conjecture and the Strong Exponential Time Hypothesis are false. That is, while the poly-log-log-dimensional case of Closest Pair is solvable in $n^{1+o(1)}$ time, the poly-log-log-dimensional case of Furthest Pair can encode difficult large-dimensional problems conjectured to require $n^{2-o(1)}$ time.

We also show that the All-Nearest Neighbors problem in $\omega(\log n)$ dimensions requires $n^{2-o(1)}$ time to solve, assuming either of the above conjectures.

1 Introduction

Point proximity and location problems have been core to computer science and computational geometry since Minsky and Papert [MP69] and Knuth's post office problem [Knu73]. In this paper, we study the tasks of finding the closest pair or furthest pair in a point set (i.e., the *diameter*) in low (but not constant) dimensions under natural norms, and incidence problems such as Hopcroft's problem [Mat93, Eri95, Eri96]: given n points in $\mathbb{R}^d$ and n hyperplanes through the origin, does any point lie on any line? (Note this is equivalent to asking whether there are two vectors which are orthogonal, i.e., have inner product 0.) For closest and furthest pair problems, we also consider their *bichromatic* versions where there are n red points, n blue points, and we wish to find a closest (or furthest) red/blue pair. [1] We consider these problems under the ℓ_p metric for $p \in \{1, 2\}$, as well as ℓ_∞. As is standard, we use ℓ_p^d to denote the metric space $(\mathbb{R}^d, \ell_p)$, with the distance functions $||x - y||_p = (\sum_{i=1}^d |x_i - y_i|^p)^{1/p}$ and $||x - y||_\infty = \max_i |x_i - y_i|$.

For large n and modest d (think of d as being at most $\mathrm{poly}(\log \log n)$), some of these tasks look far more difficult to solve than others, for reasons which are still not well-understood (beyond the fact that known techniques do not work). As early as 1976, Bentley and Shamos [BS76] noticed an apparent difference in the difficulties of solving furthest pair and closest pair in ℓ_2 in higher dimensions, and raised it as an important issue to study. Table 1 gives a rough classification of key problems which are known to be "easy" and which seem to be "hard" for large n and modest d.[2]

Note that there are many other core geometry problems with one of the two above runtime types; the above are just some of the core bottlenecks. For example, Hopcroft's problem is a special case of problems such as (batch) point location, which also suffer from the same $n^{2-1/\Theta(d)}$ dependency (see Erickson's work on hardness from Hopcroft's problem [Eri95] for more).

Why do some problems fall on the right side of the table, and can they be moved to the left side? Besides the natural question of understanding the difference between furthest and closest pair, here is another

---

---

[1]We do not consider ℓ_1 and ℓ_2 bichromatic furthest pair explicitly, as it is easy to efficiently reduce between the bichromatic version and the uncolored version. We can reduce from bichromatic to non-bichromatic by adding one extra dimension with large (positive if red, negative if blue) coordinates.

[2]In this paper, we assume a machine model that allows basic arithmetic on entries of vectors and comparisons of points in $\mathbb{Z}^d$ in $\mathrm{poly}(d, \log M)$ time, where M is the largest magnitude of an integer in an input. Such a concrete model is necessary for our hardness results, which are concerned with discrete tasks such as SAT-solving in typical Turing models of computation.

Nearly-Linear ($d^{\mathrm{poly}(d)} \cdot n \log^{O(d)} n$ time)	**Barely-Subquadratic** ($f(d) \cdot n^{2-1/\Theta(d)}$ time)
(Bichrom.) ℓ_∞^d-Furthest Pair [Yao82, GBT84]	ℓ_2^d-Furthest Pair [Yao82, AESW91]
ℓ_2^d-Closest Pair [BS76, KM95, DHKP97]	Bichrom. ℓ_2^d-Closest Pair [AESW91]
ℓ_1^d-Furthest Pair [Yao82, GBT84]	d-dim. Hopcroft's Problem [Cha93, Mat93]
(Bichrom.) ℓ_1^d and ℓ_∞^d-Closest Pair [GBT84, PS85, DHKP97, Cha17]	

Table 1: Nearly-Linear vs Barely-Subquadratic Complexity for Point Problems

motivating example. In 1984, Gabow, Bentley, and Tarjan [GBT84] showed that the ℓ_∞-furthest pair problem (and its bichromatic version) in $\mathbb{R}^d$ is *very* easy, solvable in $\tilde{O}(d \cdot n)$ time. Using this fast algorithm, along with an isometric embedding of ℓ_1^d into $\ell_\infty^{2^d}$, they then solve the (bichromatic or not) furthest pair problem for ℓ_1 in $\tilde{O}(2^d \cdot n)$ time. So computing the ℓ_∞-diameter and ℓ_1-diameter are both "nearly-linear" time problems in $\mathrm{poly}(\log \log n)$ dimensions and higher.

Do similar bounds hold for ℓ_2-furthest pair?

As the above table indicates, the best known algorithms for furthest pair in ℓ_2 (bichromatic or not) still have running time bounds of the form $O(n^{2-1/\Theta(d)})$, which is "barely subquadratic." Is there a fundamental reason why this problem is so much harder in ℓ_2 than in ℓ_1 or in ℓ_∞?

The situation is arguably counter-intuitive, because ℓ_1 and ℓ_∞ are technically more "universal" metrics than ℓ_2, so one might think that problems should be more difficult under the former than the latter. For instance, efficient isometric embeddings of n-point sets from ℓ_2 into ℓ_1 and into ℓ_∞ *are* known in the literature on metric embeddings (see the book [DL97] for references), whereas the converse is not true (see for example [WW75, Chapter 2]). However, these isometric embeddings need $\Omega(n)$ dimensions in the most general cases. There may still be embeddings (perhaps randomized) which map low-dimensional n-point sets in ℓ_2 into sub-exponential-dimensional ℓ_1 (or ℓ_∞). Indeed, in the case of low distortion (where the distances in an embedding are allowed to shrink or grow by small multiplicative amounts) these are well-known, even deterministically in some regimes [LLR94, Ind07, GLR10]. The results of this paper show that "nice" isometric embeddings of ℓ_2 into ℓ_1 would have major implications in fine-grained complexity.

1.1 Strong Difficulty of Proximity Problems in the Euclidean metric
We offer good reasons why furthest pair in ℓ_2 and other barely-subquadratic problems will be difficult to solve as fast as closest pair, even in very low dimensions. We do this by relating ℓ_2-

furthest pair and other "barely subquadratic" problems to the Orthogonal Vectors Conjecture [Wil05, AVW14] and the Strong Exponential Time Hypothesis [IP01, CIP09] in a novel way.

The **OV** problem is:

> *Given n vectors $v_1, \ldots, v_n \in \{0,1\}^d$,*
> *are there i, j such that $\langle v_i, v_j \rangle = 0$?*

Clearly $O(n^2 d)$ time suffices for solving **OV**, and slightly subquadratic-time algorithms are known in the case of small d [AWY15, CW16]. The ORTHOGONAL VECTORS CONJECTURE (OVC) states that there is no **OV** algorithm running in (say) $n^{1.99}$ time for dimensionality $d = \omega(\log n)$.

CONJECTURE 1.1. ([WIL05, AVW14]) *For every $\varepsilon > 0$, there is a $c \geq 1$ such that **OV** cannot be solved in $n^{2-\varepsilon}$ time on instances with $d = c \log n$.*

In other words, OVC states that **OV** requires $n^{2-o(1)}$ time on instances of dimension $\omega(\log n)$. OVC is plausible because it is implied by (and looks much more likely than) the popular Strong Exponential Time Hypothesis [IP01, CIP09] on the time complexity of solving k-SAT [Wil05, WY14].

Straightforward transformations show that OVC implies that both furthest and bichromatic closest pair in $\ell_1^{\omega(\log n)}$ and $\ell_2^{\omega(\log n)}$ require $n^{2-o(1)}$ time [Wil05, AW15]. Also assuming OVC, David, Kartik, and Laekhanukit [DSL16] show that (non-bichromatic) closest pair in $\ell_p^{\omega(\log n)}$ for $p > 2$ and $\ell_\infty^{\omega(\log n)}$ also require $n^{2-o(1)}$ time. It is not so surprising that some proximity search problems in super-log dimensions are hard under OVC, because OVC is a hardness conjecture about a problem in super-log dimensions.

In this paper, we show that OVC implies bichromatic closest pair and furthest pair in ℓ_2 require essentially quadratic time for even *poly-loglog* dimensions, in stark contrast with bichromatic closest pair and furthest pair in both ℓ_1 and ℓ_∞ (which both have $n^{1+o(1)}$-time solutions in this case). Our main technical tool is the following dimensionality reduction for Orthogonal Vectors over $\{0,1\}$ to Orthogonal Vectors over $\mathbb{Z}$, where the input is n vectors $v_1, \ldots, v_n \in \mathbb{Z}^d$:

LEMMA 1.1. (DIMENSIONALITY REDUCTION) *Let ℓ be an integer in $[1, d]$. There is an $n \cdot d^{O(d/\ell)}$-time reduction from* **OV** *for n vectors in $\{0, 1\}^d$ to $d^{O(d/\ell)}$ instances of* **OV** *over $\mathbb{Z}$ for n vectors in $\mathbb{Z}^{\ell+1}$, with vectors of $O((d \log d)/\ell)$-bit entries.*

Applying this lemma, we establish quadratic-time hardness for the barely-subquadratic Hopcroft's problem (which is essentially **OV** over $\mathbb{Z}$), ℓ_2-Furthest Pair, and Bichromatic ℓ_2-Closest Pair in small (poly-log-log) dimensions. It follows that if any one of these three problems became "nearly-linear", then there would have many interesting algorithmic consequences, including new SAT-solving algorithms. For example:

THEOREM 1.1. *Under SETH (or OVC), Hopcroft's problem in $\omega(\log \log n)$ dimensions with vectors of $O(\log n)$-bit entries requires $n^{2-o(1)}$ time.*

THEOREM 1.2. *Under SETH (or OVC), finding a furthest pair in $\omega(\log \log n)^2$ dimensions with vectors of $O(\log n)$-bit entries under the ℓ_2 norm requires $n^{2-o(1)}$ time.*

Therefore, computing the diameter of an n-point set in low-dimensional ℓ_2 is surprisingly more difficult to solve than in the ℓ_1 metric, or in the ℓ_∞ metric. By Gabow-Bentley-Tarjan [GBT84], there are $n^{2-\varepsilon}$-time algorithms for furthest pair under ℓ_1 up to $(1 - \varepsilon) \log n$ dimensions, and under ℓ_∞ up to $n^{1-\varepsilon}$ dimensions. There seems to be an exponential curse of dimensionality in computing the diameter of a point set, going from ℓ_∞ to ℓ_1, and *also* going from ℓ_1 to ℓ_2. Table 1.1 summarizes the consequences for barely-subquadratic problems.

Under the present landscape of fine-grained complexity conjectures, it follows that none of the barely-subquadratic problems we have identified can be made nearly-linear:

COROLLARY 1.1. *Under SETH (or OVC),* **none** *of ℓ_2-Furthest Pair, Bichromatic ℓ_2-Closest Pair, or Hopcroft's problem are solvable in $n^{2-\varepsilon} \cdot \log^{2^{o(\sqrt{d})}} n$ time, for all $\varepsilon > 0$.*

Since the above barely-subquadratic problems have closely-related nearly-linear problems, these results also show that OVC and SETH have consequences for the theory of metric embeddings. For example, since ℓ_∞^d-Furthest Pair can be solved in $\tilde{O}(d \cdot n)$ time, every $n^{1.99}$-time isometric embedding from n points in ℓ_2^d into ℓ_∞ with $d = \omega(\log \log n)^2$ must blow up the dimension doubly-exponentially to $n^{1-o(1)}$ — unless OVC and SETH are false. This is striking when one

remembers that *every n-point metric* can be (efficiently) isometrically embedded into ℓ_∞ with $n - 1$ dimensions (by the classical Frechet embedding).

Unfortunately, the above conditional lower bounds only hold for exact solutions to the problems. Our reductions from **OV** to closest/furthest pair no longer work if we only have $(1 + \varepsilon)$-approximations to the closest/furthest pair (if they did, this paper would be about how OVC is false, thanks to many fast approximation algorithms for these problems [AI17]).

Hardness for All-Nearest Neighbors. The best known algorithms for the ℓ_2-Closest Pair problem are nearly-linear, running in $2^{O(d)} n \log^{O(1)} n$ time. A prominent open problem is whether the exponential dependence on d is necessary: *Does ℓ_2-Closest Pair require $n^{2-o(1)}$ time in $\omega(\log n)$ dimensions?* Could we show hardness under (for example) OVC or SETH?

The question is rather subtle. As mentioned earlier, the related problems of Bichromatic ℓ_2-Closest Pair and ℓ_2-Furthest Pair are easily shown to be **OV**-hard in $\omega(\log n)$ dimensions [Wil05, AW15]. Intuitively speaking, in both of the latter problems, our reductions can "control" the distances between points in such a way that it is easy to encode **OV**. But for ℓ_2-Closest Pair (with no colors), we have much less control, and it is difficult to keep large sets of points far enough apart to successfully encode an **OV** instance [DSL16].

Here we report some progress on this open problem. In the closely-related *All-Nearest Neighbors* problem, the task is to report the ℓ_2-closest pair for all points in the given set. Nearly-linear algorithms are also known for All-Nearest Neighbors, which have essentially the same complexity as ℓ_2-Closest Pair [Cla83, Vai89]. We can show **OV**-hardness for All-Nearest Neighbors:

THEOREM 1.3. *Under OVC, the All-Nearest Neighbors problem in $\ell_2^{\omega(\log n)}$ requires $n^{2-o(1)}$ time, even restricted to vectors with entries from $\{-1, 0, 1\}$.*

The reduction goes through the Set Containment problem (equivalent to **OV**), and uses error-correcting codes to keep one half of the vectors "distant" from each other, and the other half relatively "close" to the first half.

2 Dimensionality Self-Reduction for OV

In this section, we set up the framework for proving hardness for the aforementioned "barely-subquadratic" problems. We begin with the following more general theorem, which will imply the dimensionality reduction lemma.

THEOREM 2.1. *For every d and integer $\ell \in [1, d]$, given two sets of vectors $U, V \subseteq \{0, 1\}^d$, there is a*

Barely-Subquadratic Problem	Lower Bound (Under SETH or OVC)
ℓ_2^d-Furthest Pair [Yao82, AESW91]	$n^{2-o(1)}$ time for $d = \omega(\log\log n)^2$
Bichrom. ℓ_2^d-Closest Pair [AESW91]	$n^{2-o(1)}$ time for $d = \omega(\log\log n)^2$
d-dim. Hopcroft's Problem [Cha93, Mat93]	$n^{2-o(1)}$ time for $d = \omega(\log\log n)$

Table 2: New Lower Bounds For Barely-Subquadratic Problems

deterministic algorithm running in $n \cdot d^{O(d/\ell)}$ time which outputs a list of $t = d^{O(d/\ell)}$ integers $\{k_1, \ldots, k_t\} \subseteq [0, t]$, along with sets $U', V' \subseteq \mathbb{Z}^{\ell+1}$ such that $|U'| = |U|$, $|V'| = |V|$, and all entries in u', v' are $O((d\log d)/\ell)$-bit integers. There is an orthogonal pair $u \in U, v \in V$ if and only if there is a pair $u' \in U', v' \in V'$ such that $\langle u', v' \rangle = k_i$ for some i.

Although it may be difficult to see in hindsight, the proof of Theorem 2.1 is inspired by the Merlin-Arthur communication protocol with $\tilde{O}(\sqrt{d})$ communication for Inner Product, due to Aaronson and Wigderson [AW09]. In that protocol, two parties each hold a d-bit vector, and they wish to determine if their vectors are orthogonal. The protocol shows how a prover can send an $\tilde{O}(\sqrt{d})$-bit message to the two parties, such that the two parties only need to exchange $\tilde{O}(\sqrt{d})$-bits (with $O(\log d)$ public randomness) to determine orthogonality with high probability. They do this by encoding d-bit vectors with $O(\sqrt{d})$-degree bivariate polynomials, and their protocol uses the key good property of low-degree polynomials that we know (they have few roots, so evaluating distinct two polynomials at a random point will yield two distinct values, with decent probability).

In the below proof of Theorem 2.1, there are several major differences. First, we forget one of the variables, and encode our d-bit vectors with ℓ-dimensional vectors whose entries are d/ℓ-degree univariate polynomials. Second, the parameter ℓ allows for a trade-off between the length of the vector and the degrees of the polynomials. (This corresponds to a trade-off between the length of the prover's message and the length of the others' messages, in the Merlin-Arthur protocol.) Third, we do *not* pick random points to evaluate the polynomial on, but rather a single deterministic value. This actually suffices for our purposes.

Proof. Without loss of generality, assume d is a multiple of ℓ, otherwise we can add zeroes at the end of each vector to satisfy this assumption.

First, we will define a univariate polynomial $R_{u,v}(x)$ that captures the inner product of two vectors u and v, then we will give a reduction that uses these polynomials in a crucial way. (Intuitively, $R_{u,v}(x)$ is analogous to Merlin's message in the Aaronson-Wigderson protocol [AW09].) Consider two vectors $u \in U, v \in V$. Di-

vide the d dimensions of both vectors into ℓ contiguous blocks, each of which contains d/ℓ dimensions. Suppose the ith block of u is $[u_{i,1}, \ldots, u_{i,d/\ell}]$ and the ith block of v is $[v_{i,1}, \ldots, v_{i,d/\ell}]$, where all $u_{i,j}, v_{i,j} \in \{0, 1\}$. Construct the polynomials

$$P_{u,i}(x) = \sum_{j=1}^{d/\ell} u_{i,j} \cdot x^{j-1}$$

and

$$Q_{v,i}(x) = \sum_{j=1}^{d/\ell} v_{i,j} \cdot x^{d/\ell - j}.$$

Let $P_u(x)$ be the ℓ-dimensional vector $[P_{u,1}, \ldots, P_{u,\ell}]$ and $Q_v(x)$ be the ℓ-dimensional vector $[Q_{v,1}, \ldots, Q_{v,\ell}]$. Observe that the coefficient of $x^{d/\ell-1}$ in the polynomial $R_{u,v}(x) = \langle P_u(x), Q_v(x) \rangle$ is exactly

$$\sum_{i=1}^{\ell} \sum_{j=1}^{d/\ell} u_{i,j} \cdot v_{i,j} = \langle u, v \rangle.$$

Furthermore, note that for any $u \in U$ and $v \in V$, the polynomial $R_{u,v}(x)$ has degree at most $2d/\ell$, and each of its coefficients are integers in $[0, d]$.

Now we are ready to give the reduction. The idea is to exhaustively try all such polynomials $R_{u,v}(x)$ which would correspond to a zero inner product, and for each one we check for a pair of vectors that yields such a polynomial.

First, enumerate all $t = d^{O(d/\ell)}$ polynomials $R(x)$ of degree at most $2d/\ell$ with coefficients in $[0, d] \cap \mathbb{Z}$ such that the coefficient of $x^{d/\ell-1}$ equals 0.

Set $x_0 := d + 1$. Note that, given the integer value $k = R(x_0) = \sum_{i=0}^{\ell-1} P_{u,i}(x_0) \cdot Q_{v,i}(x_0)$, the polynomial $R(x)$ is uniquely determined, because all of its coefficients are integers in $[0, d]$.

For all $u \in U$ and $v \in V$, compute $u' := P_u(x_0)$ and $v' := Q_v(x_0)$, creating two sets of vectors U' and V' where all vectors have ℓ dimensions, with entries of bit length at most $O((d\log d)/\ell)$.

By enumerating over all such polynomials $R(x)$, we obtain sets U', V', and collection of t integers $\{R(x_0)\}$ each in $O(d\log d)/\ell$ bits, satisfying the conclusion of the theorem. In particular, the vectors $u \in U, v \in V$

satisfy $\langle u, v \rangle = 0$ if and only if there is *some* polynomial $R(x)$ of degree at most $2d/\ell$ with coefficients in $[0, d] \cap \mathbb{Z}$ such that $\langle P_u(x_0), Q_v(x_0) \rangle = R(x_0)$ and the $x^{d/\ell-1}$ coefficient of $R(x)$ equals 0.

Now we prove the dimensionality reduction lemma:

REMINDER OF LEMMA 1.1 *[Dimensionality Reduction for OV] Let $\ell \in [1, d]$. There is an $n \cdot d^{O(d/\ell)}$-time reduction from* **OV** *for n vectors in $\{0, 1\}^d$ to $d^{O(d/\ell)}$ instances of* **OV** *over $\mathbb{Z}$ for n vectors in $\mathbb{Z}^{\ell+1}$, with vectors of $O((d \log d)/\ell)$-bit entries.*

Proof. Given a set S of n (non-zero) vectors in $\{0, 1\}^d$, set $U := S$ and $V := S$ in Theorem 2.1, which produces n vectors U' and V' in $\mathbb{Z}^\ell$ along with a set of $d^{O(d/\ell)}$ numbers T such that S has an orthogonal pair if and only if there is some $u \in U'$, $v \in V'$, and $k \in T$ such that $\langle u, v \rangle = k$.

For every $k \in T$, create new sets of vectors U'_k, V'_k, where every $u \in U'$ is replaced by $u_k := [u, 1]$ in U'_k, and every $v \in V'$ is replaced by $v_k := [v, -k]$ in V'_k. Since all entries in u and v are non-negative, we observe:

1. for all $u \in U'$ and $v \in V'$, $\langle u, v \rangle = k$ if and only if $\langle u_k, v_k \rangle = 0$,

2. for every pair $u_k, u'_k \in U_k$, $\langle u_k, u'_k \rangle \geq 1$, and

3. for every pair $v_k, v'_k \in V_k$, $\langle v_k, v'_k \rangle \geq k^2$.

Consider the set $S_k := U_k \cup V_k \subset \mathbb{Z}^\ell$. By the above three facts, we could only obtain an orthogonal pair of vectors in S_k by taking one vector from U_k and one vector from V_k, and S_k contains an orthogonal pair if and only if there is some $u \in U'$ and $v \in V'$ such that $\langle u, v \rangle = k$. Our reduction calls **OV** on S_k for every $k \in T$, and outputs the relevant orthogonal pair for S if any of the calls return an orthogonal pair for some S_k.

2.1 Consequences Here we show how the above Dimensionality Reduction for OV implies hardness for the barely-subquadratic problems mentioned in the introduction.

REMINDER OF THEOREM 1.1 *[Hopcroft's Problem] Under SETH (or OVC), Hopcroft's problem in $\omega(\log \log n)$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

Proof. Let $c \geq 1$ be an arbitrary constant and let $d := c \log n$. We show how an oracle for Hopcroft's problem in $\omega(\log \log n)$ dimensions, running in $O(n^{2-\delta})$ time for some universal $\delta > 0$, can be used to solve **OV**

for n vectors in d dimensions in $n^{2-\delta+\varepsilon}$ time (regardless of c) for every $\varepsilon > 0$, which would refute the OVC.

Set $\ell := c(\log d)/\alpha = c \log(c \log n)/\alpha$, for a small parameter $\alpha > 0$ to be set later. Applying Lemma 1.1 to a given subset $S \subseteq \{0, 1\}^d$, the reduction runs in time

$$n \cdot (c \log n)^{O(c \log n)/\ell} \leq n \cdot c^{O(\alpha \log n)} \leq n^{1+O(\alpha \log(c))},$$

and produces $n^{O(\alpha \log(c))}$ instances of **OV** with n points in $\mathbb{Z}^{(c \log \log n)/\alpha + O(1)}$, with vectors of $O(\log n)$-bit entries. Setting $\alpha \ll \varepsilon/\log(c)$, the reduction generates $O(n^\varepsilon)$ instances in $\Omega(1/\varepsilon \cdot c \log(c) \cdot \log \log n)$ dimensions, each of which our Hopcroft oracle solves in $n^{2-\delta}$ time, by assumption. This concludes the proof.

REMINDER OF THEOREM 1.2 *[ℓ_2-Furthest Pair] Under SETH (or OVC), finding a ℓ_2-furthest pair in $\omega(\log \log n)^2$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

Proof. Given a fast algorithm for ℓ_2-furthest pair in $\omega(\log \log n)^2$ dimensions, we show how to quickly solve Hopcroft's problem on n points in $\omega(\log \log n)$ dimensions, and appeal to Theorem 1.1.

Let S be a set of n vectors in $\mathbb{Z}^\ell$ with $\ell = \omega(\log \log n)$ and with $O(\log n)$-bit entries. Let $k > 1$ be such that every entry of every vector has magnitude less than n^k. In the following, let $v[i]$ denote the ith component of a vector v.

For every vector $u \in S$, define the $(\ell^2 + 2)$-dimensional vector

$$u' := [u[1] \cdot u[1], u[1] \cdot u[2], \ldots, u[\ell] \cdot u[\ell], 0, n^{2k+1}].$$

That is, the first ℓ^2 components of u' are all possible products of two components of u, followed by the entries 0 and n^{2k+1}. Put each u' in a set U'. Also for every vector $v \in S$, define the $(\ell^2 + 2)$-dimensional vector

$$v' := [v[1] \cdot v[1], v[1] \cdot v[2], \ldots, v[\ell] \cdot v[\ell], n^{2k+1}, 0],$$

and put v' in a set V'. Now observe that:

- for $u'_1, u'_2 \in U'$ coming from some $u_1, u_2 \in S$, $\langle u'_1, u'_2 \rangle = \sum_{i,j \in [\ell]} u_1[i] u_1[j] u_2[i] u_2[j] + n^{4k+2}$.

- for $v'_1, v'_2 \in V'$ coming from some $v_1, v_2 \in S$, $\langle v'_1, v'_2 \rangle = \sum_{i,j \in [\ell]} v_1[i] v_1[j] v_2[i] v_2[j] + n^{4k+2}$.

Note that by our choice of k, $|u_1[i] u_1[j] u_2[i] u_2[j]| \leq n^{4k}$ for all i, j. So all inner products are positive, in both of the above cases. In contrast, for $u' \in U'$ and $v' \in V'$,

$$\langle u', v' \rangle = \sum_{i,j \in [\ell]} u[i] u[j] v[i] v[j]$$

$$= \sum_{i,j} u[i]v[i] \cdot u[j]v[j] = (\langle u, v \rangle)^2.$$

Now, all possible inner products between every $u' \in U'$ and $v' \in V'$ are non-negative, and $\langle u', v' \rangle = 0$ if and only if $\langle u, v \rangle = 0$.

Suppose we normalize all vectors in U' and V', replacing each vector u' and v' by $u'' := u'/||u'||_2$. Since

$$\langle u'', v'' \rangle = \frac{1}{||u'||_2 \cdot ||v'||_2} \langle u, v \rangle,$$

the vector pairs in U', V' with zero inner product are exactly preserved, and all inner products of pairs within U' (and of pairs within V') are still positive. By the law of cosines, for all $u'' \in U'$ and $v'' \in V'$ we have

$$||u'' - v''||_2^2 = ||u''||_2^2 - ||v'||_2^2 - 2\langle u'', v'' \rangle = 2 - 2\langle u'', v'' \rangle.$$

Therefore, taking $S := U' \cup V'$, solving Hopcroft's problem on S is equivalent to finding two vectors with ℓ_2-distance at least $\sqrt{2}$, and this is the maximum possible distance between two vectors in the instance. It follows that solving ℓ_2-furthest pair on these instances will solve Hopcroft's problem on them as well.

CorolLARY 2.1. *Under SETH (or OVC), finding a bichromatic ℓ_2-closest pair in $\omega(\log \log n)^2$ dimensions requires $n^{2-o(1)}$ time, with vectors of $O(\log n)$-bit entries.*

Proof. As before, we begin from the proof of hardness for Hopcroft's problem (Theorem 1.1). The reduction there computes $O(n^\varepsilon)$ instances of Hopcroft's problem on n points in $\Omega(1/\varepsilon \cdot c \log(c) \cdot \log \log n)$ dimensions, for any desired $\varepsilon > 0$. We will slightly modify the proof of Theorem 1.2 for furthest pair to work for bichromatic closest pair.

Let S be a set of n vectors in $\mathbb{Z}^\ell$ with $\ell = \omega(\log \log n)$ and $O(\log n)$-bit entries. We wish to know if two vectors in S are orthogonal.

Let $v[i]$ denote the ith component of a vector v. We define the vectors in U' very similarly to the proof of Theorem 1.2: for all $u \in S$, make the ℓ^2-dimensional vector

$$u' := [u[1] \cdot u[1], u[1] \cdot u[2], \ldots, u[\ell] \cdot u[\ell]].$$

That is, each component of u' is a product of two components of u. Put each u' in a set U' of *red points*. For every vector $v \in S$, define the ℓ^2-dimensional vector

$$v' := [-v[1] \cdot v[1], -v[1] \cdot v[2], \ldots, -v[\ell] \cdot v[\ell]],$$

and put v' in a set V' of *blue points*. Now observe that for every red $u' \in U'$ and every blue $v' \in V'$,

$$\langle u', v' \rangle = -(\langle u, v \rangle)^2.$$

Thus the inner product between a red u' and a blue v' is zero when $\langle u, v \rangle = 0$, and is otherwise negative. If we normalize all vectors in U' and V', those red-blue pairs with zero inner product are preserved, and the rest of the red-blue pairs still have negative inner product. Analogously as in Theorem 1.2, this means that the red-blue pairs with zero inner product have Euclidean distance $\sqrt{2}$, and all other red-blue pairs have distance strictly greater than $\sqrt{2}$. Therefore finding the closest red-blue pair in this ℓ^2-dimensional instance will solve the original instance S of Hopcroft's problem. $\square$

3 Euclidean All-Nearest Neighbors

Here we prove hardness for All-Nearest Neighbors in $\omega(\log n)$ dimensions:

REMINDER OF THEOREM 1.3 *Under OVC, the All-Nearest Neighbors problem in $\ell_2^{\omega(\log n)}$ requires $n^{2-o(1)}$ time, even restricted to vectors with entries from $\{-1, 0, 1\}$.*

It seems plausible that there is a sub-quadratic-time reduction from All-Nearest Neighbors to ℓ_2-Closest Pair (even in high dimensions), so we think of Theorem 1.3 as good evidence that ℓ_2-Closest Pair is also hard for $\omega(\log n)$ dimensions.

Proof. Let $d = O(\log n)$. We begin with the Subset Containment problem: *Given n red subsets of $[d]$ and n blue subsets of $[d]$, is there some red subset that is contained in some blue subset?* It is well-known that this problem is equivalent to **OV** on n vectors in d dimensions [Wil05] (imagine you have a red/blue version of **OV** with vectors in $\{0,1\}^d$, and flip all the bits of the blue vectors; this converts the **OV** instance to a Subset Containment instance).

The main idea of the proof is to use error correcting codes over $\{-1, 1\}$ to keep the red points "far apart" from each other, so that the nearest neighbor of each red point x is a blue point y which is as close to being a superset of x as possible.

Let R be the collection of red sets and let B be the blue sets. We will think of them as vectors in $\{0, 1\}^d$ in the natural way. First, we do a trick which will help control the vector norms. Try all pairs of integers $d_1, d_2 \in [d]$ with $d_1 < d_2$. Take the subset R_{d_1} of R which only contains vectors having exactly d_1 ones, and take the subset B_{d_2} of B which only contains vectors having exactly d_2 ones. We will work with the collections $R' := R_{d_1}$ and $B' := B_{d_2}$ in the following. (The benefit is that we now may assume that all red vectors have the same norm value v_A, and all blue vectors have the same norm value v_B, and it only costs $O(d^2)$ extra calls.)

Let $\varepsilon \in (0, 1/2)$. We say that a *code with distance at least* $(1/2 - \varepsilon)$ is a collection of vectors $S \subseteq \{-1, 1\}^k$ such that for all $u, v \in S$ with $u \neq v$, $\langle u, v \rangle \leq 2\varepsilon k$. Note this condition is equivalent to saying that the Hamming distance between each pair of k-dimensional vectors is at least $(1/2 - \varepsilon)k$. Such codes are known to have polynomial-time constructions. In particular, it was recently shown how to efficiently construct a set S of at least n such vectors, with dimension only $k \leq t = O(\log n)/\varepsilon^{2+o(1)}$ [TS17]. In the following, let S be such a code with

$$\varepsilon := 1/8$$

and the dimension k as a parameter to be set later.

We will add k dimensions to all vectors in R' and B'. For each vector $v_i \in R$, for $i = 1, \ldots, n$, we concatenate the ith codeword from S to the end of it, obtaining a $(d+k)$-dimensional vector v_i'. For each vector $w_i \in B$, we concatenate k zeroes to the end, obtaining a $(d+k)$-dimensional w_i'.

Observe that for all vectors v_i' from R', their ℓ_2-norm squared is

$$(3.1) \qquad ||v_i'||_2^2 = d_1 + \left(\sum_{i=1}^{k} 1^2 \right) = d_1 + k$$

For all vectors w_i' from B', we have $||w_i'||_2^2 = d_2$. Furthermore, observe that for every distinct pair of vectors v_i', v_j' from R', their inner product is at most $(d_1 - 1) + 2\varepsilon k$, because the original vectors v_i and v_j were distinct vectors with exactly d_1 ones (so their inner product is at most $d_1 - 1$), and the inner product of any two distinct codewords is at most $2\varepsilon k$. Therefore for all distinct $v_i', v_j' \in R'$ we have

$$||v_i' - v_j'||_2^2 = ||v_i'||_2^2 + ||v_j'||_2^2 - 2\langle v_i', v_j' \rangle$$
$$= 2(d_1 + k) - 2\langle v_i', v_j' \rangle \qquad \text{(by (3.1))}$$
$$\geq 2(d_1 + k) - 2(d_1 - 1 + 2\varepsilon k)$$
$$= 2k + 2 - 4\varepsilon k = 3k/2 + 2.$$

On the other hand, for a vector v_i' from R' and a vector w_j' from B',

$$||v_i' - w_j'||_2^2 = ||v_i'||_2^2 + ||w_j'||_2^2 - 2\langle v_i', w_j' \rangle$$
$$= d_1 + k + d_2 - 2\langle v_i', w_j' \rangle.$$

Note the inner product $\langle v_i', w_j' \rangle$ is maximized when the original subset v_i (of cardinality d_1) is contained in the subset w_j (of cardinality d_2), in which case $\langle v_i', w_j' \rangle = d_1$. So the minimum possible distance between v_i' and w_j' is

$$||v_i' - w_j'||_2^2 = d_1 + k + d_2 - 2\langle v_i', w_j' \rangle = (d_2 - d_1) + k.$$

Putting it all together, suppose we set k large enough that

$$3k/2 + 2 > d + k$$

(e.g. $k = 2d$ will do). Then, if there is some red set (of cardinality d_1) in R contained in a blue set (of cardinality d_2) in B, then the nearest neighbor of the corresponding point in R' will be a point in B' with distance precisely $(d_2 - d_1) + 4k$ from it. Set $k = \Theta(\log n)$ so that it is at least d, and it is large enough to support at least n distinct codewords with $\varepsilon = 1/8$.

We have reduced **OV** with n vectors in $\{0, 1\}^{c \log n}$ to n points in $\{-1, 0, 1\}^{\Theta(\log n)}$, such that computing all-nearest neighbors in ℓ_2 will determine if the original instance had a red set contained in a blue set. In particular, we can check for every point whether its nearest neighbor corresponds to a set containing it in the original instance, or a set it contains. By the above, there is a red set contained in a blue set if and only if for the cardinalities d_1 and d_2 of these respective sets, the nearest neighbor to some point v in R_{d_1} is a point in B_{d_2} with distance only $(d_2 - d_1) + 4k$ from v.

4 Conclusion

We have given some rigorous explanation for why certain point-location and proximity problems only admit barely-subquadratic time algorithms: they can encode difficult high-dimensional Boolean problems in surprisingly low dimensions. In contrast, the nearly-linear proximity problems seem incapable of such an encoding; moreover, if any of them *were* found to be capable, we would be refuting some major conjectures in fine-grained complexity.

It is likely that many more consequences can be derived than what we have shown here.

- For one example, Backurs and Indyk (personal communication) noticed that our lower bound for bichromatic ℓ_2-Closest Pair implies an inapproximability result for the *fast Gauss transform* [GS91], where we are given a set of n red vectors R and n blue vectors B in $\mathbb{R}^d$, and are asked to compute

$$F(r) = \sum_{b \in B} e^{-||a-b||^2}$$

for every $r \in R$. In particular, they have observed that (under OVC) F cannot be approximated with an additive ε-error in $n^{2-\delta} \cdot \text{poly}(\log(1/\varepsilon), 2^d)$ time, for any fixed $\delta > 0$.

- For another example, a variant of the reduction in Lemma 1.1 (where instead of setting $x := d+1$ in the polynomials, we imagine trying *all* choices for

x from a large-enough field, and we build larger-dimensional Boolean vectors whose inner products model the process of computing inner products among all values of x) was used in recent work with Abboud and Rubenstein [ARW17] to show that finding a vector pair of maximum inner product among a set of n Boolean $n^{o(1)}$-dimensional vectors is hard to non-trivially approximate in sub-quadratic time.

There are many interesting questions to pursue further; here are some particularly compelling ones.

1. **Can the $\omega(\log \log n)$ and $\omega(\log \log n)^2$ dimensionality in our hardness reductions be reduced down to $\omega(1)$ dimensions?** This would demonstrate very tight hardness for solving these problems. Our results currently do not rule out $O(2^{2^d} n)$-time algorithms for any of the problems discussed. The main bottleneck is that in the main reduction (Theorem 2.1 and Lemma 1.1) it seems we have to compute $O(\log n)^{O(\log n)/\ell}$ different instances to go from $O(\log n)$ dimensions down to ℓ dimensions; perhaps there is a more efficient reduction method.

2. All of the nearly-linear problems discussed in this paper actually have $2^{O(d)} \cdot n \log^{O(1)} n$-time algorithms, except for bichromatic ℓ_1 and ℓ_∞ closest pair, for which their best known algorithms have the running time bound $n \cdot \log^{O(d)} n$. **Can stronger hardness be established for these two problems, or can their dependence on d be improved?** So far, prior work [Wil05, DSL16] has only established quadratic-time hardness for these problems when $d = \omega(\log n)$, so it is quite possible that they are in fact solvable in $2^{O(d)} \cdot n \log^{O(1)} n$ time, like the other nearly-linear problems.

3. The All-Nearest Neighbors problem is solvable in $2^{O(d)} \cdot n \log^{O(1)} n$ time in the general case, not just when all vectors are in $\{-1, 0, 1\}$. **Is there is a dimensionality reduction for the special case of $\{-1, 0, 1\}$, similar to Lemma 1.1?** (Please note that this would likely refute OVC and SETH.)

4. **Do any of the "popular conjectures" in fine-grained complexity imply that ℓ_2-Closest Pair requires $n^{2-o(1)}$ time in $\omega(\log n)$ dimensions?**

Acknowledgements

References

[AESW91] Pankaj K. Agarwal, Herbert Edelsbrunner, Otfried Schwarzkopf, and Emo Welzl. Euclidean minimum spanning trees and bichromatic closest pairs. In *Discrete and Computational Geometry*, pages 407–422. ACM, 1991.

[AI17] Alexandr Andoni and Piotr Indyk. Nearest neighbors in high-dimensional spaces. In Jacob E. Goodman, Joseph O'Rourke, and Csaba D. Toth, editors, *Handbook of Discrete and Computational Geometry*. CRC Press, 2017.

[ARW17] Amir Abboud, Aviad Rubenstein, and Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In *FOCS*, page to appear, 2017.

[AVW14] Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *ICALP*, pages 39–51, 2014.

[AW09] Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM TOCT*, 1, 2009.

[AW15] Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. In *FOCS*, 2015.

[AWY15] Amir Abboud, Richard Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *SODA*, pages 218–230, 2015.

[BS76] Jon Louis Bentley and Michael Ian Shamos. Divide-and-conquer in multidimensional space. In *STOC*, pages 220–230. ACM, 1976.

[Cha93] Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete & Computational Geometry*, 9:145–158, 1993.

[Cha17] Timothy M. Chan. Orthogonal Range Searching in Moderate Dimensions: k-d Trees and Range Trees Strike Back. In *SOCG*, volume 77 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:15, 2017.

[CIP09] Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. The complexity of satisfiability of small depth circuits. In *Parameterized and Exact Complexity (IWPEC)*, pages 75–85, 2009.

[Cla83] Kenneth L. Clarkson. Fast algorithms for the all nearest neighbors problem. In *FOCS*, pages 226–232. IEEE, 1983.

[CW16] Timothy M. Chan and Ryan Williams. Deterministic APSP, Orthogonal Vectors, and more: Quickly derandomizing Razborov-Smolensky. In *SODA*, pages 1246–1255, 2016.

[DHKP97] 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.

[DL97] Michel Marie Deza and Monique Laurent. *Geometry of Cuts and Metrics*. Springer, 1997.

[DSL16] Roee David, Karthik C. S., and Bundit Laekhanukit. On the complexity of closest pair via polar-pair of point-sets. *CoRR*, abs/1608.03245, 2016.

[Eri95] Jeff Erickson. On the relative complexities of some geometric problems. In *Proceedings of the 7th Canadian Conference on Computational Geometry*, pages 85–90, 1995.

[Eri96] Jeff Erickson. New lower bounds for Hopcroft's problem. *Discrete & Computational Geometry*, 16(4):389–418, 1996.

[GBT84] Harold N. Gabow, Jon Louis Bentley, and Robert E. Tarjan. Scaling and related techniques for geometry problems. In *STOC*, pages 135–143. ACM, 1984.

[GLR10] Venkatesan Guruswami, James R Lee, and Alexander Razborov. Almost euclidean subspaces of ℓ_1 via expander codes. *Combinatorica*, 30(1):47–68, 2010.

[GS91] Leslie Greengard and John Strain. The fast gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

[Ind07] Piotr Indyk. Uncertainty principles, extractors, and explicit embeddings of ℓ_2 into ℓ_1. In *STOC*, pages 615–620. ACM, 2007.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[KM95] Samir Khuller and Yossi Matias. A simple randomized sieve algorithm for the closest-pair problem. *Information and Computation*, 118(1):34–37, 1995.

[Knu73] Donald Ervin Knuth. *The art of computer programming: sorting and searching*, volume 3. Pearson Education, 1973.

[LLR94] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. In *FOCS*, pages 577–591. IEEE, 1994.

[Mat93] Jirí Matousek. Range searching with efficient hiearchical cutting. *Discrete & Computational Geometry*, 10:157–182, 1993.

[MP69] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. The MIT Press, 1969.

[PS85] Franco P Preparata and Michael Ian Shamos. *Computational Geometry: An Introduction*. Springer, 1985.

[TS17] Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *STOC*, 2017.

[Vai89] Pravin M. Vaidya. An o(n log n) algorithm for the all-nearest-neighbors problem. *Discrete & Computational Geometry*, 4(1):101–115, 1989.

[Wil05] Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

[WW75] J. H. Wells and L. R. Williams. *Embeddings and extensions in analysis*. Springer-Verlag, 1975.

[WY14] Ryan Williams and Huacheng Yu. Finding orthogonal vectors in discrete structures. In *SODA*, pages 1867–1877, 2014.

[Yao82] 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.