NAKHODA

Journal of Algorithms and Computation

journal homepage: http://jac.ut.ac.ir

Efficient Approximation Algorithms for Point-set Diameter in Higher Dimensions

Mahdi Imanparast[*1], Seyed Naser Hashemi[†2] and Ali Mohades [‡3]

[1,2,3]Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran.
[1]Department of Computer Science, University of Bojnord, Bojnord, Iran.

ABSTRACT

We study the problem of computing the diameter of a set of n points in d-dimensional Euclidean space for a fixed dimension d, and propose a new $(1+\varepsilon)$-approximation algorithm with $O(n + 1/\varepsilon^{d-1})$ time and $O(n)$ space, where $0 < \varepsilon \leqslant 1$. We also show that the proposed algorithm can be modified to a $(1+O(\varepsilon))$-approximation algorithm with $O(n + 1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ running time. Our proposed algorithms are different with the previous algorithms in terms of computational technique and data structures. These results provide some improvements in comparison with existing algorithms in terms of simplicity and data structure.

Keyword: Diameter, Point-set, Approximation algorithms, Higher dimensions.

AMS subject Classification: 68U05, 68W25.

ARTICLE INFO

Article history:
Received 16, March 2019
Received in revised form 22, October 2019
Accepted 25 November 2019
Available online 31, December 2019

1 Introduction

Given a finite set $\mathcal{S}$ of n points, the diameter of $\mathcal{S}$, denoted by $D(\mathcal{S})$ is the maximum distance between two points of $\mathcal{S}$. Namely, we want to find a diametrical pair p and q such that $D(\mathcal{S}) = \max_{\forall p,q \in \mathcal{S}}(||p-q||)$. Computing the diameter of a set of points has a large history, and it may be required in various fields such as database, data mining,

*Corresponding author: M. Imanparast. Email: m.imanparast@aut.ac.ir
†nhashemi@aut.ac.ir
‡mohaddes@aut.ac.ir

and vision. A trivial brute-force algorithm for this problem is as follows: compute the distance between each pair of points and then choose the maximum distance. Since computing the distance takes constant time, this algorithm takes $O(n^2)$ time, but this is too slow for large-scale datasets that occur in the fields. Hence, we need a faster algorithm which may be exact or is an approximation. By reducting from the set disjointness problem, it can be shown that computing the diameter of n points in $\mathbb{R}^d$ requires $\Omega(n \log n)$ operations in the algebraic computation-tree model [16]. It is shown by Yao that it is possible to compute the diameter in sub-quadratic time in each dimension [21]. There are well-known solutions in two and three dimensions. In the plane, this problem can be computed in optimal time $O(n \log n)$, but in three dimensions, it is more difficult. Clarkson and Shor [6] present an $O(n \log n)$-time randomized algorithm. Their algorithm needs to compute the intersection of n balls (with the same radius) in $\mathbb{R}^3$. It may be slower than the brute-force algorithm for the most practical datasets. Moreover, it is not an efficient method for higher dimensions because the intersection of n balls with the same radius has a large size. Some deterministic algorithms with running time $O(n \log^3 n)$ [2, 17] and $O(n \log^2 n)$ [18, 5] are found for solving this problem. Finally, Ramos [19, 20] introduced an optimal deterministic $O(n \log n)$-time algorithm in $\mathbb{R}^3$. Cheong et al. [10] present an $O(n \log^2 n)$ randomized algorithm that computes the all-pairs farthest neighbors for n points on the convex position in $\mathbb{R}^3$.

In the absence of fast algorithms, many attempts have been done to approximate the diameter in low and high dimensions. A 2-approximation algorithm with $O(dn)$ time in d dimensions can be found easily by selecting a point x of $\mathcal{S}$ and then finding the farthest point y from it by brute-force manner. The first non-trivial approximation algorithm for the diameter is presented by Egecioglu and Kalantari [11] that approximates the diameter with factor $\sqrt{3}$ and operations cost $O(dn)$ in d dimensions. They also present an iterative algorithm with $t \leq n$ iterations and the cost $O(dn)$ for each iteration that has approximation factor $\sqrt{5 - 2\sqrt{3}}$. Agarwal et al. [1] present a $(1 + \varepsilon)$-approximation algorithm in $\mathbb{R}^d$ with $O(n/\varepsilon^{(d-1)/2})$ running time by projection to directions. Barequet and Har Peled [4] present a $\sqrt{d}$-approximate diameter method with $O(dn)$ time. They also describe a $(1 + \varepsilon)$-approximate approach for computing the diameter with $O(n + 1/\varepsilon^{2d})$ time in $\mathbb{R}^d$. They show that the running time can be improved to $O(n + 1/\varepsilon^{2(d-1)})$. Similarly, Har Peled [14] presents an approach which for the most inputs is able to compute very fast the exact diameter, or an approximation. Although, in the worst case, the algorithm running time is still quadratic, and it is sensitive to the hardness of the input. His algorithm is able to return a pair of points p and q such that $||p - q|| \geq (1 - \varepsilon)D$, for each value $\varepsilon > 0$ in each dimension with $O((n + 1/\varepsilon^{2d}) \log 1/\varepsilon)$ running time. He shows that with a complicated analysis, this running time can be reduced to $O((n + 1/\varepsilon^{3(d-1)/2}) \log 1/\varepsilon)$. Simultaneously, Maladain and Boissonnat [15] present an exact algorithm for the diameter by computing the double normals in each dimension, but their algorithm is not worst-case optimal. They also show that with having double normals, a $\sqrt{3}$-approximation of the diameter in each dimension is provided. Moreover, Finocchiaro and Pellegrini [12] describe an algorithm that finds in $O(dn \log n + n^2)$ time with high probability a $(1 + \varepsilon)$-approximation for the diameter of a set of n points in d-dimensional Euclidean space. Chan [7] observes that a combination of two approaches in [1] and [4] yields a $(1 + \varepsilon)$-approximation

Table 1: A summary on the complexity of some utilizable non-constant approximation algorithm for the diameter of a point set in d-dimensional Euclidean space.

Reference	**Approximation Factor**	**Running Time**
[1]	$1+\varepsilon$	$O(\frac{n}{\varepsilon^{(d-1)/2}})$
[4]	$1+\varepsilon$	$O(n+1/\varepsilon^{2(d-1)})$
[14]	$1+\varepsilon$	$O((n+1/\varepsilon^{2d})\log\frac{1}{\varepsilon})$
[7]	$1+\varepsilon$	$O(n+1/\varepsilon^{\frac{3(d-1)}{2}})$
Ours	$1+\varepsilon$	$O(n+1/\varepsilon^{d-1})$
[7]	$1+O(\varepsilon)$	$O(n+1/\varepsilon^{d-\frac{1}{2}})$
[8]	$1+O(\varepsilon)$	$O(n+1/\varepsilon^{d-\frac{3}{2}})$
[9]	$1+O(\varepsilon)$	$O(n/\sqrt{\varepsilon}(\log\frac{1}{\varepsilon})^{O(1)}+1/\varepsilon^{\frac{d}{2}+1}(\log\frac{1}{\varepsilon})^{O(1)})$
[3]	$1+O(\varepsilon)$	$O(n\log\frac{1}{\varepsilon}+1/\varepsilon^{\frac{(d-1)}{2}+\alpha})$
[13]	$1+O(\varepsilon)$	$O(n+1/\varepsilon^{\frac{(d-1)}{2}})$
Ours	$1+O(\varepsilon)$	$O(n+1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$

algorithm with $O(n+1/\varepsilon^{3(d-1)/2})$ time and a $(1+O(\varepsilon))$-approximation algorithm with $O(n+1/\varepsilon^{d-\frac{1}{2}})$ time. He also introduces a core-set theorem, and shows that using this theorem, a $(1+O(\varepsilon))$-approximation for the diameter in $O(n+1/\varepsilon^{d-\frac{3}{2}})$ time can be found [8]. Recently, Chan [9] has proposed an approximation algorithm with $O((n/\sqrt{\varepsilon}+1/\varepsilon^{\frac{d}{2}+1})(\log\frac{1}{\varepsilon})^{O(1)})$ time by applying the Chebyshev polynomials for the diameter in low constant dimensions, and Arya et al. [3] show that by applying an efficient decomposition of a convex body using a hierarchy of Macbeath regions, it is possible to compute an approximation for the diameter of a point set in $O(n\log\frac{1}{\varepsilon}+1/\varepsilon^{\frac{(d-1)}{2}+\alpha})$ time, where α is an arbitrarily small positive constant. Finally, authors in [13] provide a new approximation algorithm with the running time $O(n+1/\varepsilon^{\frac{(d-1)}{2}})$. Table 1 provides a summary on some non-constant approximation algorithm for the point-set diameter.

1.1 Our results

In this paper, we propose a new $(1+\varepsilon)$-approximation algorithm for computing the diameter of a set $\mathcal{S}$ of n points in $\mathbb{R}^d$ with $O(n+1/\varepsilon^{d-1})$ time and $O(n)$ space, where $0<\varepsilon\leqslant 1$. Moreover, we show that the proposed algorithm can be modified to a $(1+O(\varepsilon))$-approximation algorithm with $O(n+1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time and $O(n)$ space. As stated above, two new results have been recently presented for the diameter problem in [9] and [3]. It should be noted that our algorithms are completely different in terms of computational technique. The polynomial technique provided by Chan [9] is based on using Chebyshev polynomials and discrete upper envelope subroutine [8], and

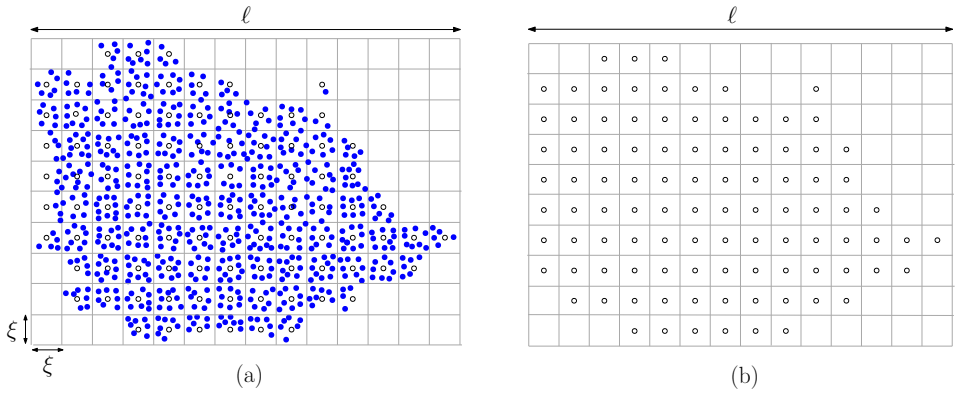

Figure 1: (a) A set of points in $\mathbb{R}^2$ and an ξ-gird. Initial points are shown by blue points and their corresponding central cell-points are shown by circle points. (b) Rounded point set $\hat{\mathcal{S}}$.

the method presented by Arya et al. [3] requires the use of complex data structures to approximately answer queries for polytope membership, directional width, and nearest-neighbor. While our algorithms in comparison with these algorithms are simpler in terms of understanding and data structure. The remainder of this paper is organized as follows: in section 2, we describe our proposed algorithms. Subsection 2.1 includes our analysis over the first algorithm. Subsection 2.2 presents a modified version of the proposed algorithm. In section 3, we draw our conclusion.

2 The proposed algorithm

In this section, we describe our new approximation algorithm to compute the diameter of a set $\mathcal{S}$ of n points in $\mathbb{R}^d$. In order to follow our algorithm, we first find extreme points in each coordinate and compute the axis-parallel bounding box of $\mathcal{S}$, which is denoted by $B(\mathcal{S})$. We use the largest length side ℓ of $B(\mathcal{S})$ to impose grids on the point set. In fact, we first decompose $B(\mathcal{S})$ to a grid of regular hypercubes with side length ξ, where $\xi = \varepsilon\ell/2\sqrt{d}$. We call each hypercube a cell. Then, each point of $\mathcal{S}$ is rounded to its corresponding central cell-point. Figure 1 shows an example of the rounding process for a point set in $\mathbb{R}^2$.

In the following, we impose again a ξ_1-grid to $B(\mathcal{S})$ for $\xi_1 = \sqrt{\varepsilon}\ell/2\sqrt{d}$. Then, we round each point of the rounded point set $\hat{\mathcal{S}}$ to its nearest grid-point in this new grid that results in a point set $\hat{\mathcal{S}}_1$. Let, $\mathcal{B}_\delta(p)$ be a hypercube with side length δ and central-point p. We restrict our search for finding diametrical pairs of the first rounded point set $\hat{\mathcal{S}}$ into two hypercubes $\mathcal{B}_{2\xi_1}(\hat{p}_1)$ and $\mathcal{B}_{2\xi_1}(\hat{q}_1)$ corresponding to two diametrical pair $\hat{p}_1$ and $\hat{q}_1$ in the point set $\hat{\mathcal{S}}_1$. Let us use two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$ for maintaining points of the rounded point set $\hat{\mathcal{S}}$, which are inside two hypercubes $\mathcal{B}_{2\xi_1}(\hat{p}_1)$ and $\mathcal{B}_{2\xi_1}(\hat{q}_1)$, respectively. See Figure 2. Then, it is sufficient to find a diameter between points of $\hat{\mathcal{S}}$, which are inside two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$. We use notation $Diam(\mathcal{B}_1, \mathcal{B}_2)$ for the process of computing the diameter of point set $\mathcal{B}_1 \cup \mathcal{B}_2$. Altogether, we can present the following algorithm.

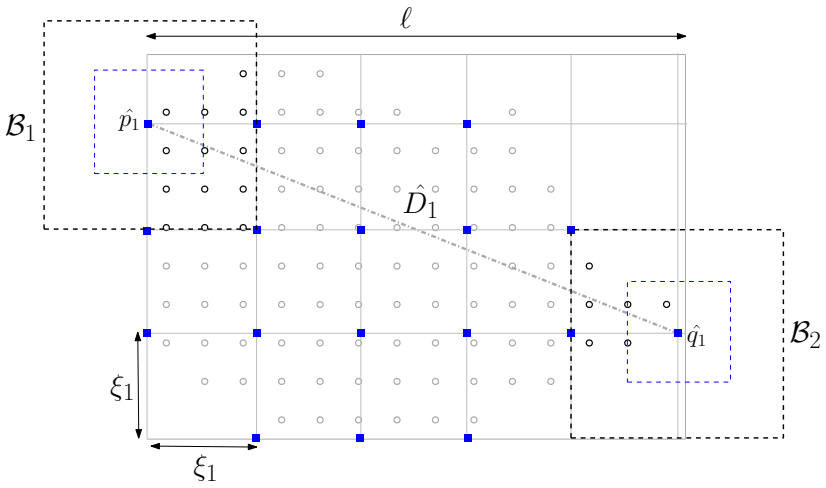

Figure 2: Points of the set $\hat{\mathcal{S}}$ are shown by circle points and their corresponding nearest grid-points in set $\hat{\mathcal{S}}_1$ are shown by blue square points. The searching domain for finding the diameter of the point set $\hat{\mathcal{S}}$ is reduced into two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$.

Algorithm 1: APPROXIMATE DIAMETER $(\mathcal{S}, \varepsilon)$

Input: a set $\mathcal{S}$ of n points in $\mathbb{R}^d$ and an error parameter ε.

Output: approximate diameter $\tilde{D}$.

1: Compute the axis-parallel bounding box $B(\mathcal{S})$ for a point set $\mathcal{S}$.
2: $\ell \leftarrow$ Find the length of the largest side in $B(\mathcal{S})$.
3: Set $\xi \leftarrow \varepsilon\ell/2\sqrt{d}$ and $\xi_1 \leftarrow \sqrt{\varepsilon}\ell/2\sqrt{d}$.
4: $\hat{\mathcal{S}} \leftarrow$ Round each point of $\mathcal{S}$ to its central-cell point in a ξ-grid.
5: $\hat{\mathcal{S}}_1 \leftarrow$ Round each point of $\hat{\mathcal{S}}$ to its nearest grid-point in a ξ_1-grid.
6: $\hat{D}_1 \leftarrow$ Compute the diameter of the point set $\hat{\mathcal{S}}_1$ by brute-force manner, and simultaneously, a list of the diametrical pair $(\hat{p}_1, \hat{q}_1)$, such that $\hat{D}_1 = ||\hat{p}_1 - \hat{q}_1||$.
7: Find points of $\hat{\mathcal{S}}$ which are in two hypercubes $\mathcal{B}_1 = \mathcal{B}_{2\xi_1}(\hat{p}_1)$ and $\mathcal{B}_2 = \mathcal{B}_{2\xi_1}(\hat{q}_1)$ for each diametrical pair $(\hat{p}_1, \hat{q}_1)$.
8: $\hat{D} \leftarrow$ Compute $Diam(\mathcal{B}_1, \mathcal{B}_2)$, corresponding to each diametrical pair $(\hat{p}_1, \hat{q}_1)$ by brute-force manner and return the maximum value between them.
9: $\tilde{D} \leftarrow \hat{D} + \varepsilon\ell/2$.
10: Output $\tilde{D}$.

Note that we only need $O(dn)$ time to round points to their central-cell points. We need $O(d)$ time for computing the central cell-point for each point. Thus, we can round all point of a set of n points to their central-cell points in $O(dn)$ time. Similarly, rounding n points to their nearest grid-point can be done in $O(dn)$ time.

2.1 Analysis

In this subsection, we analyze the proposed algorithm.

Theorem 1. *Algorithm 1 computes an approximate diameter for a set $\mathcal{S}$ of n points in $\mathbb{R}^d$ in $O(n + 1/\varepsilon^{d-1})$ time and $O(n)$ space, where $0 < \varepsilon \leqslant 1$.*

Proof. Finding the extreme points in all coordinates and finding the largest side of $B(\mathcal{S})$ can be done in $O(dn)$ time. The rounding step takes $O(d)$ time for each point, and for all of them takes $O(dn)$ time, but for computing the diameter over the

rounded point set $\hat{\mathcal{S}}_1$ we need to know the number of points in the set $\hat{\mathcal{S}}_1$. We know that the largest side of the bounding box $B(\mathcal{S})$ has length ℓ and the side length of each cell in ξ_1-grid is $\xi_1 = \sqrt{\varepsilon}\ell/2\sqrt{d}$. On the other hand, the volume of a hypercube of side length L in d-dimensional space is L^d. Since, corresponding to each point in the point set $\hat{\mathcal{S}}_1$, we can take a hypercube of side length ξ_1. Therefore, the number of grid-points in an imposed ξ_1-grid to the bounding box $B(\mathcal{S})$ is at most

$$\frac{(\ell+\xi_1)^d}{(\xi_1)^d} = \left(\frac{\ell}{\sqrt{\varepsilon}\ell/2\sqrt{d}} + 1\right)^d = \left(\frac{2\sqrt{d}}{\sqrt{\varepsilon}} + 1\right)^d = O\left(\frac{(2\sqrt{d})^d}{\varepsilon^{\frac{d}{2}}}\right). \tag{1}$$

So, the number of points in $\hat{\mathcal{S}}_1$ is at most $O((2\sqrt{d})^d/\varepsilon^{\frac{d}{2}})$. Hence, by the brute-force quadratic algorithm, we need $O((2\sqrt{d})^d/\varepsilon^{\frac{d}{2}})^2) = O((2\sqrt{d})^{2d}/\varepsilon^d)$ time for computing all distances between grid-points of the set $\hat{\mathcal{S}}_1$, and its diametrical pair list. Then, for a diametrical pair $(\hat{p}_1, \hat{q}_1)$ in point set $\hat{\mathcal{S}}_1$, we compute two sets $\mathcal{B}_1$ and $\mathcal{B}_2$. They include points of $\hat{\mathcal{S}}$ which are inside two hypercubes $\mathcal{B}_{2\xi_1}(\hat{p}_1)$ and $\mathcal{B}_{2\xi_1}(\hat{q}_1)$, respectively. This work takes $O(dn)$ time. In addition, for computing the diameter of point set $\mathcal{B}_1 \cup \mathcal{B}_2$, we need to know number of points in each of them. On the other hand, the number of points in two sets $\mathcal{B}_1$ or $\mathcal{B}_2$ is at most

$$\frac{Vol(\mathcal{B}_{2\xi_1})}{Vol(\mathcal{B}_\xi)} = \frac{(2\sqrt{\varepsilon}\ell/2\sqrt{d})^d}{(\varepsilon\ell/2\sqrt{d})^d} = \frac{(2\sqrt{\varepsilon})^d}{\varepsilon^d} = \frac{(2)^d}{\varepsilon^{\frac{d}{2}}}. \tag{2}$$

Hence, for computing $Diam(\mathcal{B}_1, \mathcal{B}_2)$, we need $O(((2)^d/\varepsilon^{\frac{d}{2}})^2) = O((2)^{2d}/\varepsilon^d)$ time by brute-force manner, but we might have more than one diametrical pair $(\mathcal{B}_1, \mathcal{B}_2)$. Since the point set $\hat{\mathcal{S}}_1$ is a set of grid-points, so we could have in the worst-case $O(2^d)$ different diametrical pairs $(\mathcal{B}_1, \mathcal{B}_2)$ in point set $\hat{\mathcal{S}}_1$. This means that this step takes at most $O(2^d \cdot (2)^{2d}/\varepsilon^d) = O((2\sqrt{2})^{2d}/\varepsilon^d)$ time.

Now, we can present the complexity of our algorithm as follows:

$$\begin{aligned} T_d(n) &= O(dn) + O(dn) + O\left(\frac{(2\sqrt{d})^{2d}}{\varepsilon^d}\right) + O(2^d dn) + O\left(\frac{(2\sqrt{2})^{2d}}{\varepsilon^d}\right), \\ &\leqslant O(2^d dn) + O\left(\frac{(2\sqrt{d})^{2d}}{\varepsilon^d}\right), \\ &= O\left(2^d dn + \frac{(2\sqrt{d})^{2d}}{\varepsilon^d}\right). \end{aligned} \tag{3}$$

Since d is fixed, we have:

$$T_d(n) = O(n + \frac{1}{\varepsilon^d}). \tag{4}$$

We can also reduce the running time of the Algorithm 1 by discarding some internal points which do not have any potential to be the diametrical pairs in rounded point set $\hat{\mathcal{S}}_1$, and similarly, in two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$. This can be done by considering all the points which are same in their $(d-1)$ coordinates and keep only highest and lowest. Then, the number of points in point set $\hat{\mathcal{S}}_1$, and two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$ can be reduced to $O(1/\varepsilon^{\frac{d}{2}-\frac{1}{2}})$. So, using the brute-force quadratic algorithm, we

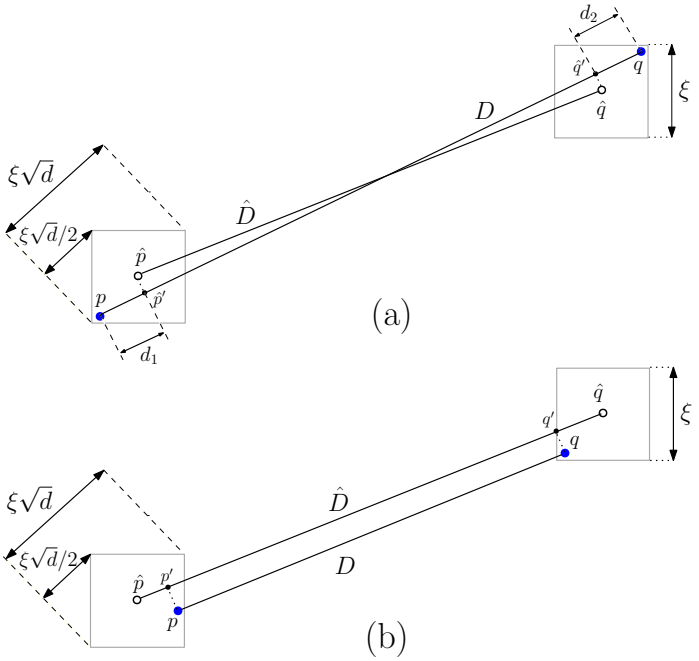

Figure 3: Two cases in proof of the Theorem 2. Two central-cell points $\hat{p}$ and $\hat{q}$ are used for computing the approximate diameter $\hat{D}$. Their corresponding true points are p and q.

need $O((1/\varepsilon^{\frac{d}{2}-\frac{1}{2}})^2)$ time to find the diametrical pairs. Hence, this gives us the total running time $O(n + 1/\varepsilon^{d-1})$. About the required space, we only need $O(n)$ space for storing required points sets. So, this completes the proof. □

Now, we explain the details of the approximation.

Theorem 2. *Algorithm 1 computes an approximate diameter $\tilde{D}$ such that: $D \leqslant \tilde{D} \leqslant (1+\varepsilon)D$, where $0 < \varepsilon \leqslant 1$.*

Proof. In line 7 of the Algorithm 1, we compute two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$, for each diametrical pair $(\hat{p}_1, \hat{q}_1)$ in the point set $\hat{\mathcal{S}}_1$. We know that a grid-point $\hat{p}_1$ in point set $\hat{\mathcal{S}}_1$ is formed from points of the set $\hat{\mathcal{S}}$ which are inside hypercube $B_{\xi_1}(\hat{p}_1)$. We use a hypercube $\mathcal{B}_1$ of side length $2\xi_1$ to make sure that we do not lost any candidate diametrical pair of the first rounded point set $\hat{\mathcal{S}}$ around a diametrical point $\hat{p}_1$, (see Figure 2). In the next step, we should find the diametrical pair $(\hat{p}, \hat{q}) \in \hat{\mathcal{S}}$ for points which are inside two point sets $\mathcal{B}_1$ and $\mathcal{B}_2$. Hence, it is remained to show that the diameter, which is computed by two points $\hat{p}$ and $\hat{q}$, is a $(1+\varepsilon)$-approximation of the true diameter. Let $\hat{p}$ and $\hat{q}$ be two central-cell points of the rounded point set $\hat{\mathcal{S}}$ which are used in line 8 of the Algorithm 1 for computing the approximate diameter $\hat{D}$. Then, we have two cases, either two true points p and q are in farthest distance from each other in their corresponding cells (Figure 3 (a)), or they are in nearest distance from each other (Figure 3 (b)). It is obvious that the other cases are between these two cases.

For first case (Figure 3 (a)), we can rotate line $\hat{p}\hat{q}$ such that two points $\hat{p}$ and $\hat{q}$ are projected on line pq. Let these two projected points be $\hat{p}'$ and $\hat{q}'$, and we set

$d_1 = ||p - \hat{p}'||$ and $d_2 = ||q - \hat{q}'||$. We know that the side length of each cell in a grid which is used for point set $\hat{\mathcal{S}}$ is ξ. So, the hypercube (cell) diagonal is $\xi\sqrt{d}$. From Figure 3 (a) it can be found that $d_1 < \xi\sqrt{d}/2$ and $d_2 < \xi\sqrt{d}/2$. Therefore, we have

$$\begin{aligned} D &= \hat{D} + d_1 + d_2, \\ D &\leqslant \hat{D} + \xi\sqrt{d}/2 + \xi\sqrt{d}/2, \\ D &\leqslant \hat{D} + \xi\sqrt{d}, \\ D - \xi\sqrt{d} &\leqslant \hat{D}. \end{aligned} \tag{5}$$

Similarly, for the second case (Figure 3 (b)), we can project two points p and q on line $\hat{p}\hat{q}$. Let these two projected points be p' and q'. We know that $c_1 = ||\hat{p} - p'|| < \xi\sqrt{d}/2$ and $c_2 = ||\hat{q} - q'|| < \xi\sqrt{d}/2$. Therefore, we have

$$\begin{aligned} \hat{D} &= D + c_1 + c_2, \\ \hat{D} &\leqslant D + \xi\sqrt{d}/2 + \xi\sqrt{d}/2, \\ \hat{D} &\leqslant D + \xi\sqrt{d}. \end{aligned} \tag{6}$$

Then, from (5) and (6) we can result:

$$D - \xi\sqrt{d} \leqslant \hat{D} \leqslant D + \xi\sqrt{d}. \tag{7}$$

Since we know that $\xi = \varepsilon\ell/2\sqrt{d}$, we have:

$$D - \varepsilon\ell/2 \leqslant \hat{D} \leqslant D + \varepsilon\ell/2. \tag{8}$$

Now, we can simplify (8) as following:

$$D \leqslant \hat{D} + \varepsilon\ell/2 \leqslant D + \varepsilon\ell. \tag{9}$$

We know that $\ell \leqslant D$. For this reason we can result:

$$D \leqslant \hat{D} + \varepsilon\ell/2 \leqslant (1 + \varepsilon)D. \tag{10}$$

Finally, if we assume that $\tilde{D} = \hat{D} + \varepsilon\ell/2$, we have:

$$D \leqslant \tilde{D} \leqslant (1 + \varepsilon)D. \tag{11}$$

Therefore, the theorem is proven. □

2.2 The modified algorithm

In this subsection, we present a modified version of our proposed algorithm by combining it with a recursive approach due to Chan [7]. Hence, we first explain Chan's recursive approach and then use it in a phase of our proposed algorithm. As mentioned before, Agarwal et al. [1] proposed a $(1+\varepsilon)$-approximation algorithm for computing the diameter of a set of n points in $\mathbb{R}^d$ with $O(n/\varepsilon^{(d-1)/2})$ running time by projecting on directions. In fact, they found a small set of directions which can approximate well all directions. This can be done by forming unit vectors which start from origin to grid-points of a uniform grid on a unit sphere [1], or to grid-points on the boundary of a box [8]. These sets of directions have cardinality $O(1/\varepsilon^{(d-1)/2})$. The following observation explains how we can find these directions on the boundary of a box.

Observation 3. *([8]) Consider a box B which includes origin o such that the boundary of this box (∂B) be in distance at least 1 from origin. For a $\sqrt{\varepsilon/2}$-grid on ∂B and for each vector $\vec{x}$, there is a grid point $\boldsymbol{a}$ on ∂B such that the angle between two vectors $\vec{a}$ and $\vec{x}$ is at most $arccos(1-\varepsilon/8) \leqslant \sqrt{\varepsilon}$.*

Proof. By scaling, we may assume that $x \in \partial B$. Since, in a $\sqrt{\varepsilon/2}$-grid the diagonal of each cell is $\sqrt{\varepsilon}$, so there is a grid point a such that: $||a-x|| \leqslant \sqrt{\varepsilon}/2$. Then, we have:

$$\begin{aligned} 2a \cdot x &\geqslant ||a||^2 + ||x||^2 - (\sqrt{\varepsilon})^2/4, \\ &\geqslant 2||a||||x|| - \varepsilon/4, \\ &\geqslant 2||a||||x||(1-\varepsilon/8). \end{aligned} \tag{12}$$

This results the observation, because: $cos\angle(a,x) = a \cdot x/||a||||x||$. □

This observation explains that grid-points on the boundary of a box (∂B) form a set V_d of $O(1/\varepsilon^{(d-1)/2})$ numbers of unit vectors in $\mathbb{R}^d$ such that for each $x \in \mathbb{R}^d$, there is a vector $a \in V_d$ from origin o to a grid-point a on ∂B, where the angle between two vectors x and a is at most $\sqrt{\varepsilon}$. On the other hand, according to observation 3, there is a vector $a \in V_d$ such that if α be the angle between two vectors x and a, then $\alpha \leqslant arccos(1-\varepsilon/8)$, and so $cos\alpha \geqslant (1-\varepsilon/8)$. If x' be the projection of the vector x on the vector a, then:

$$\begin{aligned} ||x|| &= \frac{||x'||}{cos\alpha} \\ &\leqslant ||x'|| \frac{1}{(1-\frac{\varepsilon}{8})} \\ &\leqslant ||x'||(1+\frac{\varepsilon}{8}+\frac{\varepsilon^2}{8^2}+\frac{\varepsilon^3}{8^3}+\cdots) \\ &\leqslant ||x'||(1+\varepsilon). \end{aligned} \tag{13}$$

So, we have:

$$||x'|| \leqslant ||x|| \leqslant (1+\varepsilon)||x'||. \tag{14}$$

This means that if pair (p,q) be the diametrical pair of a point set, then there is a vector $a \in V_d$ such that the angle between two vectors pq and a is at most $\sqrt{\varepsilon}$. See Figure 4. Then, pair (p',q') which is the projection of pair (p,q) on the vector a, is a $(1+\varepsilon)$-approximation of the true diametrical pair (p,q), and we have:

$$||p'-q'|| \leqslant ||p-q|| \leqslant (1+\varepsilon)||p'-q'||. \tag{15}$$

In other words, we can project point set $\mathcal{S}$ on $O(1/\varepsilon^{(d-1)/2})$ directions for all $a \in V_d$, and compute a $(1+\varepsilon)$-approximate of the diameter by finding maximum diameter between all directions. We project n points on $|V_d| = O(1/\varepsilon^{(d-1)/2})$ directions. In addition, computing the extreme points on each direction $a \in V_d$ takes $O(n)$ time. Consequently, Agarwal et al. [1] algorithm computes a $(1+\varepsilon)$-approximate of the diameter in $O(n/\varepsilon^{(d-1)/2})$ time. Chan [7] proposes that if we reduce number of points from n to $O(1/\varepsilon^{d-1})$ by rounding to a grid and then apply Agarwal et al. [1] method on this rounded point set, we need $O((1/\varepsilon^{d-1})/\varepsilon^{(d-1)/2}) = O(1/\varepsilon^{3(d-1)/2})$

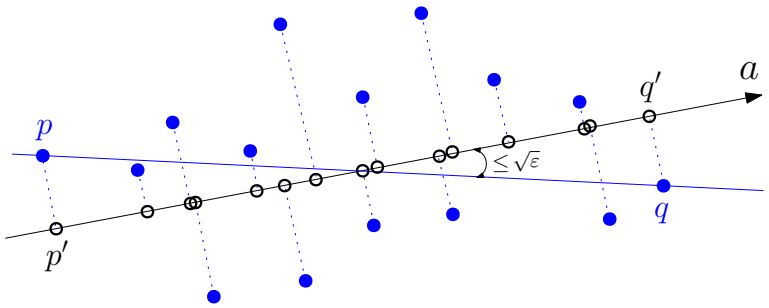

Figure 4: Projecting a point set on a direction a. True diametrical pair (p, q) and its projected diametrical pair (p', q').

time to compute the maximum diameter over all $O(1/\varepsilon^{(d-1)/2})$ directions. Taking into account $O(n)$ time for rounding to a grid, this new approach takes $O(n + 1/\varepsilon^{3(d-1)/2})$ time for computing a $(1+\varepsilon)$-approximation for the diameter of a set of n points. Moreover, Chan [7] observed that the bottleneck of this approach is the large number of projection operations. Hence, he proposes that we can project points on a set of $O(1/\sqrt{\varepsilon})$ 2-dimensional unit vectors instead of $O(1/\varepsilon^{(d-1)/2})$ d-dimensional unit vectors to reduce the problem to $O(1/\sqrt{\varepsilon})$ numbers of $(d-1)$-dimensional subproblems which can be solved recursively. Then, according to relation (14), for a vector $x \in \mathbb{R}^2$, there is a vector a such that:

$$||x'|| \leqslant ||x|| \leqslant (1+\varepsilon)||x'||, \quad x \in \mathbb{R}^2. \tag{16}$$

Since a is a unit vector ($||a|| = 1$), therefore, $||x'|| = (a \cdot x)/||a|| = a \cdot x$. Hence, we can rewrite the previous relation as following:

$$(a \cdot x)^2 \leqslant ||x||^2 \leqslant (1+\varepsilon)^2 (a \cdot x)^2, \quad x \in \mathbb{R}^2, \quad a \in V_2, \tag{17}$$

or

$$(a_1x_1 + a_2x_2)^2 \leqslant (x_1^2 + x_2^2) \leqslant (1+\varepsilon)^2 (a_1x_1 + a_2x_2)^2, \quad a \in V_2. \tag{18}$$

when x_i is the ith coordinate for a point $x \in \mathbb{R}^d$. We can expand (18) to:

$$(a_1x_1+a_2x_2)^2+\cdots+x_d^2 \leqslant (x_1^2+x_2^2+\cdots+x_d^2) \leqslant (1+\varepsilon)^2((a_1x_1+a_2x_2)^2+\cdots+x_d^2). \tag{19}$$

Now, define the projection $\pi_a : \mathbb{R}^d \to \mathbb{R}^{d-1} : \pi_a(x) = (a_1x_1 + a_2x_2, x_3, \cdots, x_d) \in \mathbb{R}^{d-1}$. Then, we can rewrite relation (19) for each vector $x \in \mathbb{R}^d$ as following:

$$||\pi_a(x)||^2 \leqslant ||x||^2 \leqslant (1+\varepsilon)^2 ||\pi_a(x)||^2, \ \ a \in V_2. \tag{20}$$

So, since $||\pi_a(p-q)|| = ||\pi_a(p)|| - ||\pi_a(q)||$ we have for diametrical pair (p, q):

$$||\pi_a(p-q)|| \leqslant ||p-q|| \leqslant (1+\varepsilon)||\pi_a(p-q)||, \ \ a \in V_2. \tag{21}$$

Therefore, for finding a $(1 + O(\varepsilon))$-approximation for the diameter of point set $P \subseteq \mathbb{R}^d$, it is sufficient that we approximate recursively the diameter of a projected point set $\pi_a(P) \subset \mathbb{R}^{d-1}$ over each of the vectors $a \in V_2$. Then, the maximum diametrical pair computed in the recursive calls is a $(1 + O(\varepsilon))$–approximation to

the diametrical pair. Now, let us reduce the number of points from n to $O(1/\varepsilon^{d-1})$ by rounding to a grid. Let we denote the required time for computing the diameter of m points in d-dimensional space with $t_d(m)$, then for a rounded point set on a grid with $m = O(1/\varepsilon^{d-1})$ points, this approach breaks the problem into $O(1/\sqrt{\varepsilon})$ subproblems in a $(d-1)$ dimension. Hence, we have a recurrence $t_d(m) = O(m + 1/\sqrt{\varepsilon}t_{d-1}(O(1/\varepsilon^{d-1})))$. By assuming $E = 1/\varepsilon$, we can rewrite the recurrence as:

$$t_d(m) = O(m + E^{\frac{1}{2}}t_{d-1}(O(E^{d-1}))). \tag{22}$$

This can be solved to: $t_d(m) = O(m + E^{d-\frac{1}{2}})$. In this case, $m = O(1/\varepsilon^{d-1})$, so, this recursive takes $O(1/\varepsilon^{d-\frac{1}{2}})$ time. Taking into account $O(n)$ time, we spent for rounding to a grid at the first, Chan's recursive approach computes a $(1 + O(\varepsilon))$-approximation for the diameter of a set of n points in $O(n + 1/\varepsilon^{d-\frac{1}{2}})$ time [7].
In the following, we use Chan's recursive approach in a phase of our proposed algorithm and present a modified version of it with running time $O(n + 1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$.

Algorithm 2: APPROXIMATE DIAMETER 2 $(\mathcal{S}, \varepsilon)$

Input: a set $\mathcal{S}$ of n points in $\mathbb{R}^d$ and an error parameter ε.

Output: approximate diameter $\tilde{D}$.

1: Compute the axis-parallel bounding box $B(\mathcal{S})$ for a point set $\mathcal{S}$.
2: $\ell \leftarrow$ Find the length of the largest side in $B(\mathcal{S})$.
3: Set $\xi \leftarrow \varepsilon\ell/2\sqrt{d}$ and $\xi_2 \leftarrow \varepsilon^{\frac{1}{3}}\ell/2\sqrt{d}$.
4: $\hat{\mathcal{S}} \leftarrow$ Round each point of $\mathcal{S}$ to its central-cell point in a ξ-grid.
5: $\hat{\mathcal{S}}_1 \leftarrow$ Round each point of $\hat{\mathcal{S}}$ to its nearest grid-point in a ξ_2-grid.
6: $\hat{D}_1 \leftarrow$ Compute the diameter of the point set $\hat{\mathcal{S}}_1$ by brute-force, and simultaneously, a list of the diametrical pair $(\hat{p}_1, \hat{q}_1)$, such that $\hat{D}_1 = ||\hat{p}_1 - \hat{q}_1||$.
7: Find points of $\hat{\mathcal{S}}$ which are in two hypercubes $\mathcal{B}_1 = \mathcal{B}_{2\xi_2}(\hat{p}_1)$ and $\mathcal{B}_2 = \mathcal{B}_{2\xi_2}(\hat{q}_1)$ for each diametrical pair $(\hat{p}_1, \hat{q}_1)$.
8: $\tilde{D} \leftarrow$ Compute $Diam(\mathcal{B}_1, \mathcal{B}_2)$, corresponding to each diametrical pair $(\hat{p}_1, \hat{q}_1)$ by using Chan's [7] recursive approach and return the maximum value $||p' - q'||$ over all of them.
9: Output $\tilde{D}$.

Now we will analyze the running time and approximation factor of the Algorithm 2.

Theorem 4. *A $(1 + O(\varepsilon))$-approximation for the diameter of a set of n points in d-dimensional Euclidean space can be computed in $O(n + 1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time, where $0 < \varepsilon \leqslant 1$.*

Proof. As it can be seen, lines 1 to 5 of the Algorithm 2 are the same as the Algorithm 1. We do rounding to grids twice and reach to a point set $\hat{\mathcal{S}}_1$ in $O(n)$ time. In this case, the number of points in rounded points set $\hat{\mathcal{S}}_1$ is at most:

$$\frac{(\ell + \epsilon_2)^d}{(\epsilon_2)^d} = \left(\frac{\ell}{\varepsilon^{\frac{1}{3}}\ell/2\sqrt{d}} + 1\right)^d = \left(\frac{2\sqrt{d}}{\varepsilon^{\frac{1}{3}}} + 1\right)^d = O\left(\frac{(2\sqrt{d})^d}{\varepsilon^{\frac{d}{3}}}\right). \tag{23}$$

This can be reduced to $O((2\sqrt{d})^d/\varepsilon^{\frac{d}{3}-\frac{1}{3}})$, by keeping only highest and lowest points which are the same in their $(d-1)$ coordinates. So, for finding all diametrical pairs of the point set $\hat{\mathcal{S}}_1$, we can use the quadratic brute-force algorithm with $O((2\sqrt{d})^d/\varepsilon^{\frac{d}{3}-\frac{1}{3}})^2) = O((2\sqrt{d})^{2d}/\varepsilon^{\frac{2d}{3}-\frac{2}{3}})$ time. Then, for each diametrical pair

$(\hat{p}_1, \hat{q}_1) \in \hat{\mathcal{S}}_1$, we compute two sets $\mathcal{B}_1$ and $\mathcal{B}_2$ which include points of set $\hat{\mathcal{S}}$ which are inside two hypercubes $\mathcal{B}_{2\xi_2}(\hat{p}_1)$ and $\mathcal{B}_{2\xi_2}(\hat{q}_1)$, respectively. Moreover, the number of points in two sets $\mathcal{B}_1$ or $\mathcal{B}_2$ is at most

$$\frac{Vol(\mathcal{B}_{2\xi_2})}{Vol(\mathcal{B}_{\xi})} = \frac{(2\varepsilon^{\frac{1}{3}}\ell/2\sqrt{d})^d}{(\varepsilon\ell/2\sqrt{d})^d} = \frac{(2\varepsilon^{\frac{1}{3}})^d}{\varepsilon^d} = \frac{(2)^d}{\varepsilon^{\frac{2d}{3}}}. \tag{24}$$

This can be reduced to $O((2)^d/\varepsilon^{\frac{2d}{3}-1})$, by keeping only highest and lowest points which are the same in their $(d-1)$ coordinates. Now, for computing $Diam(\mathcal{B}_1, \mathcal{B}_2)$, we use Chan's [7] recursive approach instead of using the quadratic brute-force algorithm on the point set $\mathcal{B}_1 \cup \mathcal{B}_2$. On the other hand, computing the diameter on a set of $O(1/\varepsilon^{\frac{2d}{3}-1})$ points using Chan's recursive approach takes the following recurrence based on relation (22): $t_d(m) = O(m + 1/\sqrt{\varepsilon}t_{d-1}(O(1/\varepsilon^{\frac{2d}{3}-1})))$. By assuming $E = 1/\varepsilon$, we can rewrite the recurrence as:

$$t_d(m) = O(m + E^{\frac{1}{2}}t_{d-1}(O(E^{\frac{2d}{3}-1}))). \tag{25}$$

This can be solved to: $t_d(m) = O(m + E^{\frac{2d}{3}-\frac{1}{2}})$. In this case, $m = O(E^{\frac{2d}{3}-1})$, so, this recursive takes $O(E^{\frac{2d}{3}-1} + E^{\frac{2d}{3}-\frac{1}{2}}) = O(1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time. Also, if we have more than one diametrical pair $(\hat{p}_1, \hat{q}_1)$ in point set $\hat{\mathcal{S}}_1$, then this step takes at most $O((2^d)(2)^d/\varepsilon^{\frac{2d}{3}-\frac{1}{2}}) = O(2^{2d}/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time. Therefore, we can write the complexity time of the algorithm as following:

$$\begin{aligned} T_d(n) &= O(dn) + O(dn) + O\left(\frac{(2\sqrt{d})^{2d}}{\varepsilon^{\frac{2d}{3}-\frac{2}{3}}}\right) + O(2^d dn) + O\left(\frac{2^{2d}}{\varepsilon^{\frac{2d}{3}-\frac{1}{2}}}\right), \\ &\leqslant O(2^d dn) + O\left(\frac{(2\sqrt{d})^{2d}}{\varepsilon^{\frac{2d}{3}-\frac{1}{2}}}\right), \\ &= O\left(2^d dn + \frac{(2\sqrt{d})^{2d}}{\varepsilon^{\frac{2d}{3}-\frac{1}{2}}}\right). \end{aligned} \tag{26}$$

Since d is fixed, we have:

$$T_d(n) = O\left(n + \frac{1}{\varepsilon^{\frac{2d}{3}-\frac{1}{2}}}\right). \tag{27}$$

In addition, Chan's recursive approach in line 8 of the Algorithm 2 returns a diametrical pair (p', q') which is a $(1 + O(\varepsilon))$-approximation for the diametrical pair $(\hat{p}, \hat{q}) \in \hat{\mathcal{S}}$. This means that:

$$||p' - q'|| \leqslant ||\hat{p} - \hat{q}|| \leqslant (1 + O(\varepsilon))||p' - q'||. \tag{28}$$

Moreover, the diametrical pair $(\hat{p}, \hat{q})$ is an approximation of the true diametrical pair $(p, q) \in \mathcal{S}$, and according to relation (10), we have:

$$||p - q|| \leqslant ||\hat{p} - \hat{q}|| + \varepsilon\ell/2 \leqslant (1 + \varepsilon)||p - q||. \tag{29}$$

Hence, from (28) and (29) we can result:

$$\begin{aligned} ||p-q|| &\leqslant ||\hat{p}-\hat{q}|| + \varepsilon\ell/2, \\ &\leqslant ||\hat{p}-\hat{q}|| + \varepsilon||\hat{p}-\hat{q}||, \\ &\leqslant (1+\varepsilon)||\hat{p}-\hat{q}||, \\ &\leqslant (1+\varepsilon)((1+O(\varepsilon))||p'-q'||), \\ &\leqslant (1+O(\varepsilon))||p'-q'||. \end{aligned} \tag{30}$$

So, Algorithm 2 finds a $(1+O(\varepsilon))$-approximation of the diameter of a point set $\mathcal{S}$ of n points in $O(n+1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time. Therefore, this completes the proof. □

3 Conclusion

We have presented a new $(1+\varepsilon)$-approximation algorithm to compute the diameter of a point set $\mathcal{S}$ of n points in $\mathbb{R}^d$ for a fixed dimension d in $O(n+1/\varepsilon^{d-1})$ time, where $0 < \varepsilon \leqslant 1$. Moreover, we show that the proposed algorithm can be modified to a $(1+O(\varepsilon))$-approximation algorithm with $O(n+1/\varepsilon^{\frac{2d}{3}-\frac{1}{2}})$ time. Our proposed algorithms provide some improvements in comparison with existing algorithms in terms of simplicity, understanding and data structure.

References

[1] Agarwal, P. K., Matousek, J. , Suri, S., Farthest neighbors maximum spanning trees and related problems in higher dimensions. Computational Geometry: Theory and Applications, 1 (1992), 189–201.

[2] Amato, N. M., Goodrich, M. T., Ramos, E. A., Parallel algorithms for higher dimensional convex hulls. In Proceedings of the 35th annual Symposium on Foundations of Computer Science, (1994), 683-694.

[3] Arya, S., da Fonseca, G. D., Mount, D. M., Near-optimal ε-kernel construction and related problems. In Proceedings of the 33rd International Symposium on Computational Geometry (SoCG'17), (2017), pages 10:1–15.

[4] Barequet, G., Har-Peled, S., Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. Journal of Algorithms, 38 (2001), 91–109.

[5] Bespamyatnikh, S., An efficient algorithm for the three-dimensional diameter problem. Discrete and Computational Geometry, 25, 2 (2001), 235–255.

[6] Clarkson, K. L., Shor, P. W., Applications of random sampling in computational geometry. Discrete and Computational Geometry, 4 (1989), 387–421.

[7] Chan, T. M., Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus. International Journal of Computational Geometry and Applications, (2002), 67-85.

[8] Chan, T. M., Faster core-set constructions and data stream algorithms in fixed dimensions. Computational Geometry: Theory and Applications, 35 (2006), 20–35.

[9] Chan, T. M., Applications of Chebyshev polynomials to low-dimensional computational geometry. In Proceedings of the 33rd International Symposium on Computational Geometry (SoCG'17), (2017), pages 26:1–15.

[10] Cheong, O, Shin, C. S., Vigneron A., Computing farthest neighbors on a convex polytope. In Proceedings of the 7th Annual International Computational and Combinatoric Conference, (2001), 159–169.

[11] Egecioglu, O., Kalantari B., Approximating the diameter of a set of points in the Euclidean space. Information Processing Letters, 32, 4 (1989), 205–211.

[12] Finocchiaro, D. V., Pellegrini, M., On computing the diameter of a point set in high dimensional Euclidean space. Theoretical Computer Science, 287 (2002), 501–514.

[13] Imanparast, M., Hashemi, S. N., Better Approximation Algorithm for Point-set Diameter. IAENG International Journal of Computer Science, 46, 4 (2019), 594–598.

[14] Har-Peled, S., A practical approach for computing the diameter of a point set. In Proceedings of the 17th annual Symposium on Computational Geometry (SoCG'01), (2001), 177-186.

[15] Malandain, G. , Boissonnat, J. D., Computing the diameter of a point set. International Journal of Computational Geometry and Applications, 12, 6 (2002), 489–509.

[16] Preparata, F. P., Shamos, M. I., Computational Geometry: an Introduction. New York, Springer-Verlag, (1985), 176–182.

[17] Ramos, E. A., Intersection of unit-balls and diameter of a point set in $\mathbb{R}^3$. Computational Geometry: Theory and Applications, 8 (1997), 57–65.

[18] Ramos, E. A., Construction of 1-d lower envelopes and applications. In Proceedings of the 13th annual ACM Symposium on Computational Geometry (SoCG'97), (1997), 57–66.

[19] Ramos, E. A., Deterministic algorithms for 3-D diameter and some 2-D lower envelopes. In Proceedings of the 16th annual Symposium on Computational Geometry (SoCG'00), (2000).

[20] Ramos E. A., An optimal deterministic algorithm for computing the diameter of a three-dimensional point set. Discrete and Computational Geometry, 26 (2001), 245–265.

[21] Yao, A. C., On constructing minimum spanning trees in k-dimensional spaces and related problems. SIAM Journal on Computing, 11 (1982), 721–736.