IEICE TRANS. FUNDAMENTALS, VOL.E107–A, NO.9 SEPTEMBER 2024

PAPER *Special Section on Discrete Mathematics and Its Applications*

# Dispersion in a Polygon

**Tetsuya ARAKI**[†a)] ***and*** **Shin-ichi NAKANO**[†b)]**,** ***Members***

**SUMMARY** The dispersion problem is a variant of facility location problems, that has been extensively studied. Given a polygon with $n$ edges on a plane we want to find $k$ points in the polygon so that the minimum pairwise Euclidean distance of the $k$ points is maximized. We call the problem *the $k$-dispersion problem in a polygon*. Intuitively, for an island, we want to locate $k$ drone bases far away from each other in flying distance to avoid congestion in the sky. In this paper, we give a polynomial-time approximation scheme (PTAS) for this problem when $k$ is a constant and $\epsilon < 1$ (where $\epsilon$ is a positive real number). Our proposed algorithm runs in $O(((1/\epsilon)^2 + n/\epsilon)^k)$ time with $1/(1+\epsilon)$ approximation, the first PTAS developed for this problem. Additionally, we consider three variations of the dispersion problem and design a PTAS for each of them.

***key words:*** *facility location problem, algorithm, dispersion problem*

## 1. Introduction

The facility location problem and many of its variants have been studied [17], [18]. Typically, given a set of points on which facilities can be placed and an integer $k$, we want to place $k$ facilities on some points so that a designated objective function is minimized. By contrast in the *dispersion problem*, we want to place facilities so that a designated objective function is maximized.

**Our results** In this paper we consider four dispersion problems on a plane and design a PTAS for each of them for a fixed constant $k$.

A PTAS (Polynomial-Time Approximation Scheme) is a type of approximation algorithm for optimization problems that computes a $(1+\epsilon)$-approximate solution within polynomial time. A PTAS allows for the adjustment of the $\epsilon$ value, enabling the precision of the approximate solution to be set arbitrarily. Therefore, designing a PTAS is important, as it can quickly compute rough approximate solutions or spend longer times computing refined approximate solutions, depending on the requirements.

Given a polygon $P$ with $n$ edges on a plane, we want to find $k$ points in $P$ so that the minimum pairwise Euclidean distance of the $k$ points is maximized. We call the problem *the $k$-dispersion problem in a polygon*. See an example in Fig. 1. Note that the $k$ points may contain a point in a polygon which is not on the boundary, for instance if we choose 5 points in a square, the 5 points consist of the four corner points and the center of the square.

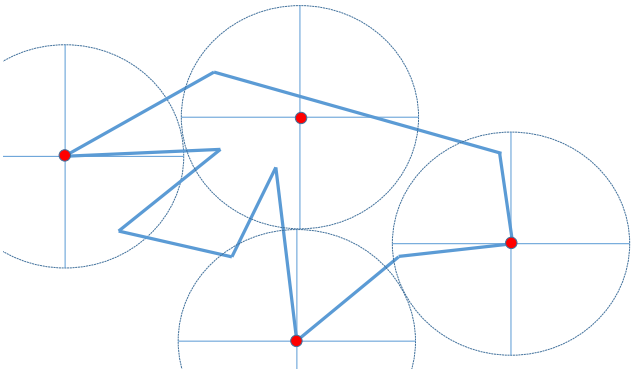


**Fig. 1** A solution of a $k$-dispersion problem in a polygon with $k = 4$.

Intuitively, for an island, we want to locate $k$ drone bases far away from each other to avoid congestion in the sky.

In this paper we give an $O(((1/\epsilon)^2+n/\epsilon)^k)$ time $1/(1+\epsilon)$ approximation algorithm to solve the problem, where $\epsilon < 1$ is a positive number. Thus the problem has a PTAS.

The algorithm first computes a set of (grid) points in $P$ and computes optimal $k$ points among the set of points by a brute force algorithm. By choosing the gap size of the grid suitably, we ensure the approximation ratio.

If $k$ is a part of the inputs, i.e., not a constant, then a similar problem has no PTAS [8].

We consider the following three more problems.

1. Given a polygon with $n$ edges on a plane we want to find $k$ points in the polygon so that the minimum length of paths inside the polygon (where a path is a sequence of straight line segments in the polygon) connecting any two of the $k$ points is maximized. See an example in Fig. 2. Intuitively, for an island, we want to locate $k$ coffee shops far away from each other to avoid self competition for walking customers. We call the problem *the $k$-dispersion problem in a polygon with the geodesic distance*.
2. Given a set of $n$ (connected) straight line segments on a plane we want to find $k$ points on the straight line segments so that the minimum pairwise Euclidean distance of the $k$ points is maximized. See an example in Fig. 3. Intuitively, for some road network, we want to locate $k$ drone bases far away from each other to avoid conges-

Manuscript received September 20, 2023.
Manuscript revised January 29, 2024.
Manuscript publicized March 11, 2024.

[†]Department of Computer Science, Gunma University, Kiryu-shi, 376-8515 Japan.
a) E-mail: tetsuya.araki@gunma-u.ac.jp
b) E-mail: nakano@gunma-u.ac.jp

DOI: 10.1587/transfun.2023DMP0010

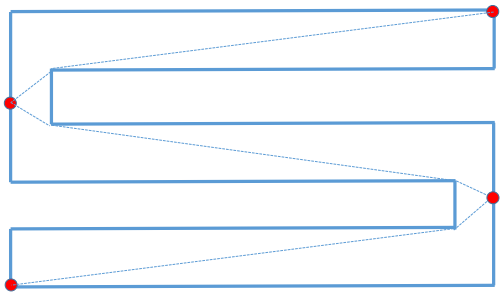


**Fig. 2** A solution of a $k$-dispersion problem in a polygon with $k = 4$ where the minimum length of paths inside the polygon connecting two points among the k points is maximized.

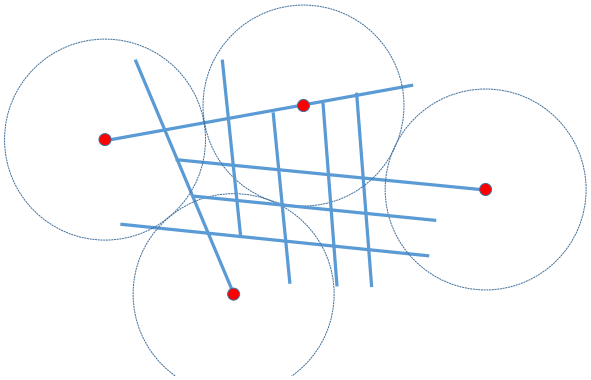


**Fig. 3** A solution of a $k$-dispersion problem in a set of straight line segments with $k = 4$.

tion in the sky so that each base faces a road(s). We call the problem *the k-dispersion problem on straight line segments*.

3. Consider a game space in which the same squares are arranged in a rectangular array of $L$ rows and $L$ columns. The side length of the square is one. For each square $s$ an integer weight $w(s)$ is assigned. We assume that $w(s) \in \{1, 2, \cdots, \mathbb{W}\}$. A point can repeatedly move in a square $s$ either horizontally or vertically at speed distance 1 per $w(s)$ seconds. A point can move the common boundary of two squares $s$ and $s'$ at speed distance 1 per $\min\{w(s), w(s')\}$ seconds. (A point can move in the faster square.) We want to find $k$ points in the game space so that the minimum time to move from one point to other point in the $k$ points is maximized. See an example of the problem and its solution in Fig. 4. Intuitively, for the game space, we want to locate $k$ hunters far away from each other to avoid fighting over prey. We call the problem *the k-dispersion in a game space problem*.

In the well-known k-center problem, research on polygons [7], [27] and line segments [28] has also been conducted. Therefore, considering polygons and line segments in the context of the dispersion problem is a natural extension.

**Related results**

Given a set $C$ of $n$ points, a distance function $d$, and an integer $k$ with $k < n$, the *max-min k-dispersion problem* computes a subset $S \subset C$ with $|S| = k$ such that the cost $cost(S) = \min_{\{u,v\} \subset S}\{d(u, v)\}$ is maximized. Several results are known for the max-min $k$-dispersion problem. The problem is NP-hard [20]. If $d$ is a metric (the distance satisfies the triangle inequality) a polynomial-time approximation algorithm with approximation ratio two is known [29], and it is NP-hard to compute a solution with approximation ratio less than two [29]. An exponential time exact algorithm is known [3]. If $P$ is a set of $n$ points on a line one can solve the problem in $O(kn)$ time by dynamic programming [31], in $O(n \log \log n)$ time by sorted matrix search method [2], and in $O((2k^2)^k n)$ time by the pigeonhole principle [4]. For the max-sum version several results are also known [10], [12]–[14], [21], [25], [29]. For a variety of related problems, see [8], [14]. See more applications, including *result diversification*, in [13], [29], [30].

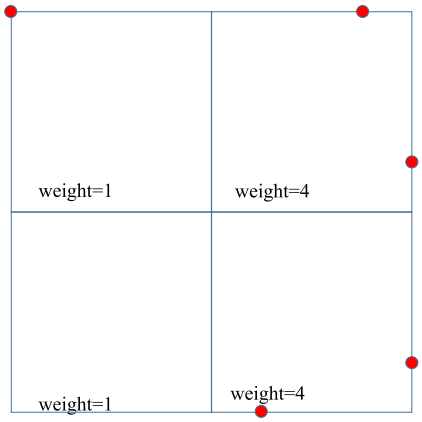



**Fig. 4** An example of the dispersion in a game space problem and its solution with $k = 5$.

Given a set of $n$ disjoint intervals on a line the *max-min dispersion problem on intervals* computes one point from each interval so that the minimum pairwise distance of the $n$ points is maximized. If the disjoint intervals are given in the sorted order on the line, two $O(n)$ time algorithms to solve the problem are known [9], [26]. Given a set of $n$ disjoint intervals on a line and a constant integer $k$ with $k < n$, even if the intervals are given in any (unsorted) order, one can compute $k$ points from the intervals in $O((2k^2)^k n)$ time so that the minimum pairwise distance of the $k$ points is maximized [5].

Given a set of disjoint disks with arbitrary radii, the *dispersion problem on disks* is the problem to compute one point in each disk so that the minimum distance among the points is maximized. The problem is NP-hard, and some approximation algorithms are known [11], [19], [22], also an $O((n/\epsilon^2)^k)$ time $1/(1 + \epsilon)$ approximation algorithm is known [5].

The dispersion problem in a polygon is similar to the following packing problem. Given an integer $k$ and a disk, the disk packing in a disk problem computes the maximum radius of $k$ identical disks which can be packed without overlapping into the given disk. Given an integer $k$, a disk

with radius $r$ and a number $dist$, if we have an algorithm to decide whether one can locate $k$ points in the disk so that the minimum distance among them is $dist$ or more, then, by using the algorithm, we can decide whether one can pack $k$ identical disks with radii $dist/2$ or more into a disk with radius $r + dist/2$. Only the following result is known for the time complexity of the disk packing in a disk problem. It is in EXPTIME, whereas whether it is in PSPACE or not is open [1]. For similar problems the followings are known. It is NP-complete to decide whether a given set of (possibly not identical) disks can be packed into a given square [16, Corollary 7.2]. It is NP-complete to decide whether a given set of (possibly not identical) disks can be packed into a given disk [23, Corollary 6.2].

The remainder of this paper is organized as follows. In Sect. 2 we give an $O(((1/\epsilon)^2 + n/\epsilon)^k)$ time $1/(1+\epsilon)$ approximation algorithm for the $k$-dispersion problem in a polygon with $n$ edges. In Sects. 3–5 we design algorithms for three variant problems. Finally Sect. 6 is a conclusion.

The preliminary version of the paper appeared in [6].

## 2. $k$-Dispersion in a Polygon

Fix a constant integer $k$. Given a polygon $P$ with $n$ edges on a plane, we would like to find $k$ points in $P$ so that the minimum pairwise Euclidean distance of the $k$ points is maximized. Let $\epsilon < 1$ be a positive number. In this section we give an $O(((1/\epsilon)^2 + n/\epsilon)^k)$ time $1/(1+\epsilon)$ approximation algorithm for this problem.

We need some notations. For a set $S$ of points let $cost(S)$ be the minimum pairwise Euclidean distance among the points in $S$. Let $P^*$ be a set of $k$ points in $P$ with the maximum $cost(P^*)$.

Let $W$ be the difference between the $x$-coordinates of a leftmost point and a rightmost point in $P$. Similarly let $H$ be the difference between the $y$-coordinates of a topmost point and a bottommost point in $P$. Without loss of generality we can assume that $W \geq H$.

We have the following lemma.

**Lemma 1.** $cost(P^*) > W/k$

*Proof.* Fix a directed path starting at a leftmost point $p_\ell$ and ending at a rightmost point $p_r$ in $P$. Let $p_0 = p_\ell, p_1, \cdots, p_{k-1} = p_r$ be the points on the directed path such that $p_i$ is the first point having its $x$-coordinate $x(p_\ell) + Wi/(k-1)$ for $i = 0, 1, \cdots, k-1$. Now the distance between any two points of the $k$ points is at least $W/(k-1)$, so more than $W/k$. □

Now we describe the algorithm.

By a standard plane sweep algorithm (see page 1022 of textbook [15]) we can sweep $P$ by a horizontal sweep line from top to bottom in $O(n \log n)$ time, and during the sweep we can maintain the edges of $P$ intersecting the current horizontal sweep line with the left-to-right order of the intersection points with the current horizontal sweep line. During the sweep we construct a set $G$ of points in $P$, as follows. We consider a grid of gap size $W\epsilon/ck$ on $P$, where a constant $c$ is explained later, and stop the horizontal sweep line at each horizontal line on the grid, that is, when the $y$-coordinate of the horizontal sweep line is $W\epsilon i/ck$ for each integer $i$, and append to $G$ the points on the current horizontal sweep line which are (1) the intersection points with the vertical grid lines (the number of such points is at most $(1 + ck/\epsilon)^2$ in total), and (2) the intersection points with edges of $P$ (the number of such points is at most $(1 + ck/\epsilon)n$ in total). Similarly we sweep $P$ by a vertical sweep line from left to right, and append to $G$ the points on the current vertical sweep line which are (3) the intersection points with edges of $P$. Now $|G| \leq (1 + ck/\epsilon)^2 + 2(1 + ck/\epsilon)n$ holds. An example of $G$ is shown in Fig. 5.

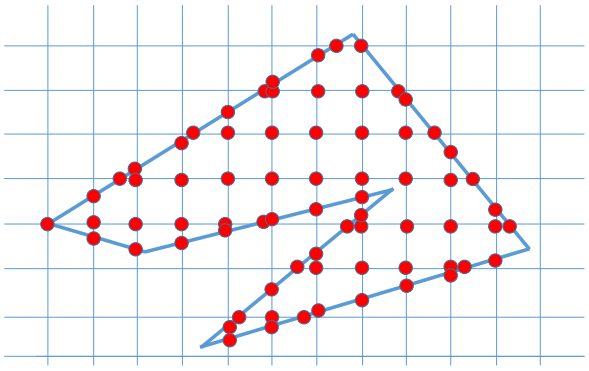


**Fig. 5** An example of the set $G$ of points in $P$.

Let $G(P^*)$ be the set of points derived from $P^*$ by choosing a nearest point in $G$ for each point in $P^*$. We choose $c$ large enough so that [c1] $cost(G(P^*)) > cost(P^*)/(1+\epsilon)$ holds and [c2] no two points in $P^*$ have the common nearest point in $G$. (If two points in $P^*$ have the common nearest point in $G$ then $cost(G(P^*)) = 0$. This case is prohibited because the approximation ratio becomes infinite.)

If we set $c$ as $c \geq 2\sqrt{2}(1+\epsilon)$ then the following holds by Lemma 1.

$$\begin{aligned}
&\frac{1}{c} \leq \frac{1}{2\sqrt{2}(1+\epsilon)} \\
&\frac{2\sqrt{2}}{c}(1+\epsilon) \leq 1 \\
&\frac{2\sqrt{2}}{c}\frac{W(1+\epsilon)}{k} \leq \frac{W}{k} < cost(P^*) \\
&\frac{2\sqrt{2}W\epsilon}{ck} < cost(P^*)\frac{\epsilon}{1+\epsilon} \\
&\qquad = cost(P^*)\frac{(1+\epsilon)-1}{1+\epsilon} \\
&\qquad = cost(P^*) - \frac{1}{1+\epsilon}cost(P^*) \\
&\frac{1}{1+\epsilon}cost(P^*) < cost(P^*) - \frac{2\sqrt{2}W\epsilon}{ck} \leq cost(G(P^*))
\end{aligned}$$

Thus [c1] holds. Note that $cost(P^*) - \frac{2\sqrt{2}W\epsilon}{ck} \leq cost(G(P^*))$ holds, since the distance between a point in $P^*$ and its nearest

point in $G$ is at most $\sqrt{2}\frac{W\epsilon}{ck}$.

If we set $c$ as $c \geq (1+2\sqrt{2})\epsilon$ then the following holds.

$$\frac{1}{(1+2\sqrt{2})\epsilon} \geq \frac{1}{c}$$
$$1 \geq \frac{(1+2\sqrt{2})\epsilon}{c}$$
$$\frac{W}{k} \geq \frac{(1+2\sqrt{2})W\epsilon}{ck}$$
$$\frac{W}{k} - \frac{2\sqrt{2}W\epsilon}{ck} \geq \frac{W\epsilon}{ck}$$

Now the following holds by Lemma 1.

$$cost(G(P^*)) \geq cost(P^*) - \frac{2\sqrt{2}W\epsilon}{ck} > \frac{W}{k} - \frac{2\sqrt{2}W\epsilon}{ck} \geq \frac{W\epsilon}{ck}$$

Note that again $cost(P^*) - \frac{2\sqrt{2}W\epsilon}{ck} \leq cost(G(P^*))$ holds.

Thus, for any two points $p$ and $q$ in $P^*$, let $p'$ and $q'$ be the nearest points in $G$, respectively, then the distance between $p'$ and $q'$ is at least $W\epsilon/ck$, so (c2) holds.

We set $c$ large enough to satisfy the above two conditions.

**Algorithm** Let $G_A$ be the set $G'$ of $k$ points in $G$ maximizing $cost(G')$. If we find a set $G_A$ in $O(|G|^k)$ time by a brute force algorithm we have $cost(G_A) \geq cost(G(P^*)) \geq cost(P^*)/(1+\epsilon)$.

Now we have the following theorem. Note that $k$ is a constant.

**Theorem 1.** *One can compute a set $G_A$ of $k$ points in a given polygon $P$ with $n$ edges with $cost(G_A) \geq cost(P^*)/(1+\epsilon)$ in $O((1/\epsilon^2 + n/\epsilon)^k)$ time, where $P^*$ is a set of $k$ points in $P$ maximizing $cost(P^*)$.*

## 3. $k$-Dispersion Problem in a Polygon with the Geodesic Distance

In the $k$-dispersion problem in a polygon, by replacing the Euclidean distance by the length of a shortest path inside a polygon, we can define the following problem.

Given a polygon $P$ with $n$ edges on a plane, we want to find $k$ points in $P$ so that the minimum length of the shortest paths inside $P$ (where a path is a sequence of straight line segments in the polygon) connecting two points among the $k$ points is maximized.

By preprocessing the polygon in $O(n \log n)$ time, one can compute, for any pair of query points in $P$, the length of the shortest path inside $P$ in $O(\log n)$ time [24]. Thus given a set of $k$ points in $P$ we can compute in $O(k^2 \log n)$ time the minimum length of the shortest paths inside $P$ connecting two points among the $k$ points.

For a set $S$ of points in $P$ let $cost'(S)$ be the minimum length of the shortest paths inside $P$ connecting two points among $S$. Let $P^*$ be a set of $k$ points in $P$ with the maximum $cost'(P^*)$.

We can solve the problem as follows. First by a standard plane sweep algorithm we sweep $P$ by a horizontal sweep line from top to bottom in $O(n \log n)$ time and by a vertical sweep line from left to right in $O(n \log n)$ time, and construct a set of points $G$ in $P$ as the algorithm in Sect. 2. For this problem we additionally append to $G$ the end points of each segment of $P$. So $|G|$ increased by $n$. Now $|G| \leq (1+ck/\epsilon)^2 + (3+2ck/\epsilon)n$ holds. We need to append these points to $G$ to ensure $cost'(P^*) - \frac{2\sqrt{2}W\epsilon}{ck} \leq cost'(G(P^*))$ for this problem. Then by a brute force algorithm compute $G_A$ which is the set $G'$ of $k$ points in $G$ maximizing $cost'(G')$. We can find $G_A$ in $O(|G|^k)$ time.

We again choose $c$ large enough so that [c1"] $cost'(G(P^*)) > cost'(P^*)/(1+\epsilon)$ holds and [c2"] no two points in $P^*$ have the common nearest point in $G$.

We have the following theorem.

**Theorem 2.** *One can compute a set $G_A$ of $k$ points in a given polygon $P$ with $n$ edges with $cost'(G_A) \geq cost'(P^*)/(1+\epsilon)$ in $O((1/\epsilon^2 + n/\epsilon)^k k^2 \log n)$ time, where $d'(u,v)$ is the length of the shortest path inside $P$ connecting two points $u$ and $v$, and $cost'(S) = \min_{\{u,v\} \subset S}\{d'(u,v)\}$ and $P^*$ is a set of $k$ points in $P$ maximizing $cost'(P^*)$.*

*Proof.* One can use the proof of Theorem 1 to prove the Theorem 2. □

## 4. $k$-Dispersion Problem on Straight Line Segments

Given a set $L$ of $n$ straight line segments on a plane, we want to find $k$ points in $L$ so that the minimum pairwise Euclidean distance of the $k$ points is maximized. We assume that $L$ is connected, that is, for any pair of points in $L$, there is a path in $L$ from a point to the other point. We set $W'$ as the difference between the $x$-coordinate of a leftmost end point and the $x$-coordinate of a rightmost end point in $L$. Similarly let $H'$ be the difference between the $y$-coordinate of a topmost end point and the $y$-coordinate of a bottommost end point in $L$. Without loss of generality we can assume that $W' \geq H'$.

Similarly to the previous section, we define that for a set $S$ of points, let $cost(S)$ be the minimum pairwise Euclidean distance among the points in $S$. Let $P^*$ be a set of $k$ points in $L$ with the maximum $cost(P^*)$.

We have the following lemma.

**Lemma 2.** $cost(P^*) > W'/k$

*Proof.* One can use the proof of Lemma 1 to prove the Lemma 2. □

Now we describe the algorithm.

By a standard plane sweep algorithm, we sweep $L$ by a horizontal sweep line from top to bottom in $O(n \log n)$ time. During the sweep we construct a set $G$ of points in $L$, as follows. We consider a grid of gap size $W'\epsilon/ck$ on $L$, where $c$ is a constant, and stop the horizontal sweep line at each horizontal line on the grid, and append to $G$ the points on the

current horizontal sweep line which are (1') the intersection points with straight line segments in $L$ (the number of such points is at most $(1 + ck/\epsilon)n$ in total). Similarly we sweep $L$ by a vertical sweep line from left to right, and append to $G$ the points on the current vertical sweep line which are (2') the intersection points with straight line segments in $L$. Now $|G| \leq 2(1 + ck/\epsilon)n$ holds. Then again, by a brute force algorithm, we compute $G_A$ which is the set $G'$ of $k$ points in $G$ maximizing $cost(G')$.

Now we have the following theorem.

**Theorem 3.** *Given a set $L$ of $n$ straight line segments on a plane, one can compute a set $P_A$ of $k$ points on $L$ with $cost(P_A) \geq cost(P^*)/(1 + \epsilon)$ in $O((n/\epsilon)^k)$ time, where $P^*$ is a set of $k$ points on $L$ with maximum $cost(P^*)$.*

*Proof.* One can use the proof of Theorem 1 to prove the Theorem 3. □

## 5. $k$-Dispersion in a Game Space

In the previous sections, we discussed three dispersion problems with a constant cost of movement, regardless of location. However, in practical, moving through rivers and forests can take longer than moving over plains. Therefore, in this section, we consider the dispersion problems where moving costs differ depending on their individual locations. We can define the following problem.

A game space is composed of identical squares arranged in a rectangular array of $L$ rows and $L$ columns. An integer weight $w(s) \in \{1, 2, \cdots, \mathbb{W}\}$ is assigned to each square, where $\mathbb{W}$ is a constant. A point can repeatedly move in a square $s$ either horizontally or vertically at speed distance 1 per $w(s)$ seconds. We assume that a point can move the common boundary of two squares $s$ and $s'$ at speed distance 1 per $\min\{w(s), w(s')\}$ seconds. The goal is to find $k$ points in the game space so that the minimum time to move from one point to other point in the $k$ points is maximized.

Let $\epsilon < 1$ be a positive number. In this section we give an $O((1/\epsilon)^{2k})$ time $1/(1 + \epsilon)$ approximation algorithm to solve the problem.

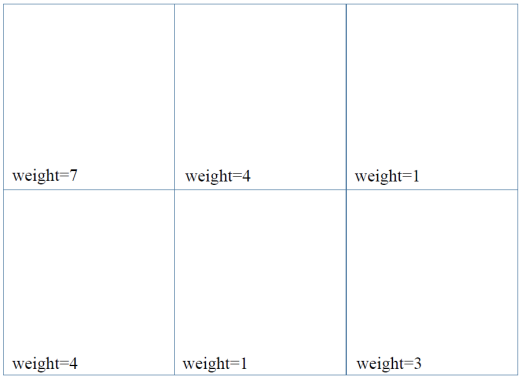
For a set $S$ of points, let $T$ be the set of minimum time to move from a points in $S$ to other points in $S$, and let $cost(S)$ be the minimum in $T$. Let $P^*$ be a set of $k$ points in the game space with the maximum $cost(P^*)$.

We also regard the game space as a graph, in which each vertex is the corner of each square and each edge is one of line segment on the boundary of each square. The weight of each edge is the lighter weight of two incident squares. (If the edge is on the outer face then its weight is the weight of the unique incident square.) An example of the graph of the game space is shown in Fig. 6.

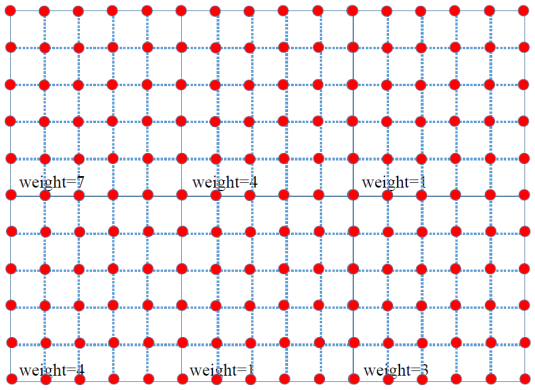
We have the following lemma.

**Lemma 3.** $cost(P^*) > 1/k$

*Proof.* Let $S = \{p_1, p_2, \cdots, p_k\}$ be the $k$ points on the line segment from the lower right corner point to the upper right




(a) A game space of 2 rows and 3 columns.




(b) A graph of the game space (a).

**Fig. 6** An example of the graph of the game space.

corner point of some square $s$ in which they are evenly spaced. Now the minimum time from one point to other point in $S$ is at least $1/(k - 1)$ since $w(s) \geq 1$, then the lemma follows. □

We compute a set $G$ of points as follows. We divide each square $s$ into grids of gap size $\epsilon/(ck\mathbb{W})$, a constant $c$ is explained later, and append to $G$ each corner point on the grids. Now $|G| \leq (ck\mathbb{W}L/\epsilon + 1)^2$ holds. We also regard those grids as a graph. Each point in $|G|$ is a vertex in the graph. The number of edges in this graph is at most $4|G|$. Since $c, k$, and $\mathbb{W}$ are constants, $|G|$ is $O((L/\epsilon)^2)$.

We have the following lemma.

**Lemma 4.** *(a) The time for a point to move from a grid point to the nearest grid point is at most $\frac{\epsilon}{ck\mathbb{W}} / \frac{1}{\mathbb{W}} = \frac{\epsilon}{ck}$.*

*(b) The time for a point to move from a point $p$ to a point among the four grid points surrounding $p$ is at most $2\frac{\epsilon}{ck}$.*

*Proof.* (a) Since the distance to move is $\epsilon/(ck\mathbb{W})$ and the speed is at least $1/\mathbb{W}$.

(b) Since the distance to move is at most $2\epsilon/(ck\mathbb{W})$ and the speed is at least $1/\mathbb{W}$. □

Let $G(P^*)$ be the set of $k$ points derived from $P^*$ by choosing a nearest point in $G$ for each point in $P^*$. We choose $c$ large enough so that [c1] $cost(G(P^*)) > cost(P^*)/(1 + \epsilon)$

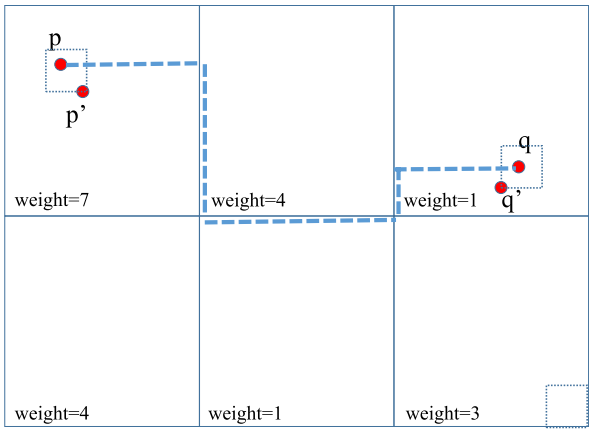



**Fig. 7** Illustration for Lemma 4.

holds and [c2] no two points in $P^*$ have the common nearest point in $G$. (If two points in $P^*$ have the common nearest point in $G$ then $cost(G(P^*)) = 0$. This case is prohibited because the approximation ratio becomes infinite.)

Now, we describe how we set the constant c. If we set $c$ as $c \geq 4(1+\epsilon)$, then the following inequality holds by Lemma 3.

$$\begin{aligned}
&\frac{1}{c} \leq \frac{1}{4(1+\epsilon)} \\
&\frac{4}{c}(1+\epsilon) \leq 1 \\
&\frac{4}{c}\frac{(1+\epsilon)}{k} \leq \frac{1}{k} < cost(P^*) \\
&\frac{4\epsilon}{ck} < cost(P^*)\frac{\epsilon}{1+\epsilon} \\
&\quad = cost(P^*)\frac{(1+\epsilon)-1}{1+\epsilon} \\
&\quad = cost(P^*) - \frac{1}{1+\epsilon}cost(P^*) \\
&\frac{1}{1+\epsilon}cost(P^*) < cost(P^*) - \frac{4\epsilon}{ck} \leq cost(G(P^*))
\end{aligned}$$

Thus [c1] holds. Note that $cost(P^*) - \frac{4\epsilon}{ck} \leq cost(G(P^*))$ holds, since the time for a point to move from a point $p$ in $P^*$ to a point among the four grid points surrounding $p$ is at most $\frac{2\epsilon}{ck}$ by Lemma 4 (b). For a better comprehension, see Fig. 7.

If we set $c$ as $c \geq 5\epsilon$, then the inequality following holds.

$$\begin{aligned}
&\frac{1}{5\epsilon} \geq \frac{1}{c} \\
&1 \geq \frac{5\epsilon}{c} \\
&\frac{1}{k} \geq \frac{5\epsilon}{ck} \\
&\frac{1}{k} - \frac{4\epsilon}{ck} \geq \frac{\epsilon}{ck}
\end{aligned}$$

Now the following holds by Lemma 3.

$$cost(G(P^*)) \geq cost(P^*) - \frac{4\epsilon}{ck} > \frac{1}{k} - \frac{4\epsilon}{ck} \geq \frac{\epsilon}{ck}$$

Note again that $cost(P^*) - \frac{4\epsilon}{ck} \leq cost(G(P^*))$ holds by Lemma 4 (b).

For any two points $p$ and $q$ in $P^*$, let $p'$ and $q'$ be the nearest points in $G$, respectively. Then, the time for a point to move from $p'$ to $q'$ is at least $\epsilon/ck$, that is, they are distinct grid points by Lemma 4 (a), so (c2) holds.

We set $c$ large enough to satisfy the above two conditions.

**Algorithm**

Let $G_A$ be the set $G'$ of $k$ points in $G$ maximizing $cost(G')$. We can find $G_A$ in $O(|G|^k k4|G| \log |G|)$ time by the following algorithm. For every possible $k$ points in $G$, we compute the minimum time for a point to move from a point to other point among the $k$ points, by running a single source shortest path algorithm $k$ times. Since the degree of vertices is at most 4, so the number of edges is at most $4|G|$, then Dijkstra's algorithm finds a solution in $O(4|G| \log |G|)$ time for each and $O(|G|^k k4|G| \log |G|)$ time in total. We have $cost(G_A) \geq cost(G(P^*)) \geq cost(P^*)/(1+\epsilon)$.

Now we have the following theorem.

**Theorem 4.** *One can compute a set $G_A$ of $k$ points in a given game space with $cost(G_A) \geq cost(P^*)/(1+\epsilon)$ in $O((L/\epsilon)^{2k}(L/\epsilon)^2 \log(L/\epsilon))$ time, where $P^*$ is a set of $k$ points in the game space with maximizing $cost(P^*)$.*

## 6. Conclusion

In this paper we designed an algorithm to solve the $k$-dispersion problem in a polygon. For a fixed constant integer $k$, given a polygon with $n$ edges, our algorithm computes a set $G_A$ of $k$ points in the polygon with $cost(G_A) \geq cost(P^*)/(1+\epsilon)$ in $O(((1/\epsilon)^2 + n/\epsilon)^k)$ time, where $P^*$ is an optimal solution. Thus the problem has a PTAS.

Then we have defined three natural dispersion problems. For a constant integer $k$ we can design a PTAS to compute $k$ points in a given polygon so that the minimum length of the shortest paths inside $P$ connecting two points among the $k$ points is maximized, and a PTAS to compute $k$ points on given straight line segments on a plane so that the minimum pairwise Euclidean distance of the $k$ points is maximized, and a PTAS to compute $k$ points on given a game world consisting of square with different travel times for each square so that the minimum time to move from one point to other point in the $k$ points is maximized.

Each algorithm is simple but the first PTAS to solve a natural problem. We hope further improvements continue.

### References

[1] H. Alt, "Computational aspects of packing problems," Bull. EATCS, no.118, 2016.
[2] T. Akagi and S. Nakano, "Dispersion on the line," IPSJ SIG Technical Reports, 2016-AL-158-3, 2016.
[3] T. Akagi, T. Araki, T. Horiyama, S. Nakano, Y. Okamoto, Y. Otachi,

T. Saitoh, R. Uehara, T. Uno, and K. Wasa, "Exact algorithms for the max-min dispersion problem," Proc. FAW 2018, LNCS 10823, pp.263–272, 2018.
[4] T. Araki and S. Nakano, "Max-min dispersion on a line," J. Comb. Optim., vol.44, no.3, pp.1824–1830, 2022.
[5] T. Araki, H. Miyata, and S. Nakano, "Dispersion on intervals," Proc. of CCCG2021, 2021.
[6] T. Araki and S. Nakano, "Away from each other," Proc. WALCOM2023, pp.61–70, 2023.
[7] M. Basappa, R.K. Jallu, and G.K. Das, "Constrained $k$-center problem on a convex polygon," International Journal of Foundations of Computer Science, vol.31, no.2, pp.275–291, 2020.
[8] C. Baur and S.P. Fekete, "Approximation of geometric dispersion problems," Proc. APPROX 1998, pp.63–75, 1998, Also in Algorithmica, vol.30, no.3, pp.451–470, 2001.
[9] T. Biedl, A. Lubiw, A.M. Naredla, P.D. Ralbovsky, and G. Stroud, "Dispersion for intervals: A geometric approach," Prc. SOSA 2021, pp.37–44, 2021.
[10] B. Birnbaum and K.J. Goldman, "An improved analysis for a greedy remote-clique algorithm using factor-revealing LPs," Algorithmica, vol.55, no.1, pp.42–59, 2009.
[11] S. Cabello, "Approximation algorithms for spreading points," Journal of Algorithms, vol.62, no.2, pp.49–73, 2007.
[12] A. Cevallos, F. Eisenbrand, and R. Zenklusen, "Max-sum diversity via convex programming," Proc. SoCG 2016, pp.26:1–26:14, 2016.
[13] A. Cevallos, F. Eisenbrand, and R. Zenklusen, "Local search for max-sum diversification," Proc. SODA 2017, pp.130–142, 2017.
[14] B. Chandra and M.M. Halldorsson, "Approximation algorithms for dispersion problems," J. Algorithms, vol.38, no.2, pp.438–465, 2001.
[15] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, Introduction to Algorithms, MIT Press, 2009.
[16] E.D. Demaine, S.P. Fekete, and R.J. Lang, "Circle packing for origami design is hard," Proc. 5th International Conference on Origami in Science, Mathematics and Education (OSME 2010), pp.609–626, 2010. Also, CoRR abs/1008.1224, 2010.
[17] Z. Drezner, Facility Location: A Survey of Applications and Methods, Springer, 1995.
[18] Z. Drezner and H.W. Hamacher, Facility Location: Applications and Theory, Springer, 2004.
[19] A. Dumitrescu and M. Jiang, "Dispersion in disks," Theory Comput. Syst., vol.51, pp.125–142, 2012.
[20] E. Erkut, "The discrete $p$-dispersion problem," European Journal of Operational Research, vol.46, no.1, pp.48–60, 1990.
[21] S.P. Fekete and H. Meijer, "Maximum dispersion and geometric maximum weight cliques," Algorithmica, vol.38, pp.501–511, 2004.
[22] J. Fiala, J. Kratochvil, and A. Proskurowski, "Systems of distant representatives," Discrete Appl. Math., vol.145, no.2, pp.306–36, 2005.
[23] S.P. Fekete, P. Keldenich, and C. Scheffer, "Packing disks into disks with optimal worst-case density," Proc. 35th International Symposium on Computational Geometry, SoCG 2019, LIPIcs, vol.129, pp.35:1–35:19, 2019.
[24] L.J. Guibas and J. Hershberger, "Optimal shortest path queries in a simple polygon," J. Comput. Syst. Sci., vol.39, no.2, pp.126–152, 1989.
[25] R. Hassin, S. Rubinstein, and A. Tamir, "Approximation algorithms for maximum dispersion," Operation Research Letters, vol.21, no.3, pp.133–137, 1997.
[26] S. Li and H. Wang, "Dispersing points on intervals," Proc. ISAAC 2016, Article no.52, 2016. DOI: 10.4230/LIPIcs.ISAAC.2016.52
[27] E. Oh, S.W. Bae, and H.K Ahn, "Computing a geodesic two-center of points in a simple polygon," Computational Geometry, vol.82, pp.45–59, 2019.
[28] S. Maji and S. Sadhu, "Discrete and mixed two-center problems for line segments," Information Processing Letters, vol.184, 106451, 2024.
[29] S.S. Ravi, D.J. Rosenkrantz, and G.K. Tayi, "Heuristic and special case algorithms for dispersion problems," Operations Research, vol.42, no.2, pp.299–310, 1994.
[30] M. Sydow, "Approximation guarantees for max sum and max min facility dispersion with parameterised triangle inequality and applications in result diversification," Mathematica Applicanda, vol.42, no.2, pp.241–257, 2014.
[31] D.W. Wang and Y.-S. Kuo, "A study on two geometric location problems," Information Processing Letters, vol.28, no.6, pp.281–286, 1988.

**Tetsuya Araki** received the B.E., M.E., and Ph.D. degrees e in Electrical and Electronic Engineering from Kobe University, Kobe, Japan, in 2008, 2010 and 2013, respectively. He was a researcher at National Institute of Informatics from 2013 to 2017, a research assistant professor at Tokyo Metropolitan University from 2017 to 2019. He is currently an assistant professor of Department of Computer Science, Gunma University. His research interests are graph algorithms, social network and data mining.

**Shin-ichi Nakano** received his B.E. and M.E. degrees from Tohoku University, Sendai, Japan, in 1985 and 1987, respectively. In 1987 he joined Seiko Epson Corp. and in 1990 he joined Tohoku University. In 1992, he received Dr. Eng. degree from Tohoku University. Since 1999 he has been a faculty member of Department of Computer Science, Gunma University. His research interests are graph algorithms and graph theory.