An Approximation Algorithm for the 2-Dispersion Problem

SUMMARY Let $P$ be a set of points on the plane, and $d(p, q)$ be the distance between a pair of points $p, q$ in $P$. For a point $p \in P$ and a subset $S \subset P$ with $|S| \geq 3$, the 2-dispersion cost, denoted by $cost_2(p, S)$, of $p$ with respect to $S$ is the sum of (1) the distance from $p$ to the nearest point in $S \setminus \{p\}$ and (2) the distance from $p$ to the second nearest point in $S \setminus \{p\}$. The 2-dispersion cost $cost_2(S)$ of $S \subset P$ with $|S| \geq 3$ is $\min_{p \in S} cost_2(p, S)$. Given a set $P$ of $n$ points and an integer $k$ we wish to compute a point subset $S$ of $P$ with maximum $cost_2(S)$. In this paper we give a simple $1/(4 \sqrt{3})$ approximation algorithm for the problem.

key words: dispersion problem, approximation algorithm

1. Introduction

Many facility location problems compute locations minimizing some cost or distance [5], [6]. While in this paper we consider a dispersion problem which computes locations maximizing some cost or distance [2]–[4], [7], [10]–[12].

Dispersion problems has an important application for information retrieval. It is desirable to find a small subset of a large data set, so that the small subset have a certain diversity. Such a small subset may be a good sample to overview the large data set [3], and diversity maximization has become an important concept in information retrieval.

A typical dispersion problem is as follows. Given a set $P$ of points on the plane and an integer $k$, find $k$ points subset $S$ of $P$ maximizing a designated cost. If the cost is the minimum distance between a pair of points in $S$ then it is called the max-min dispersion problem, and if the cost is the sum of the distances between all pair of points in $S$ then it is called the max-sum dispersion problem. Unfortunately both problems are NP-hard, even the distance satisfies the triangle inequality [10].

In this paper we consider a recently proposed related problem called the 2-dispersion problem [8], [9]. We give a simple approximation algorithm for the 2-dispersion problem, where the cost of a point in $S$ is the sum of the distances to the nearest two points in $S$, and the cost of $S$ is the minimum cost among the costs of points in $S$. Intuitively we wish to locate our $k$ chain stores so that each store is located far away from the nearest two “rival” stores to avoid self-competition. In [8], [9] more general variants, including max-min and max-sum dispersion problems are studied.

In this paper we give a simple approximation algorithm for the 2-dispersion problem defined above. Our algorithm computes a $1/(4 \sqrt{3})$-approximate solution for the 2-dispersion problem. This is the first approximation algorithm for the 2-dispersion problem.

The remainder of the paper is organized as follows. Section 2 gives some definitions. Section 3 gives our simple approximation algorithm for the 2-dispersion problem. Finally Sect. 4 is a conclusion.

A preliminary version of the paper with approximation ratio $1/8$ has been presented at [1].

2. Definitions

Let $P$ be a set of $n$ points on a plane, and $d(p, q)$ be the distance between a pair of points $p, q$ in $P$. We assume that the distance is the Euclidean distance.

For a point $p \in P$ and a subset $S \subset P$ with $|S| \geq 3$, the 2-dispersion cost $cost_2(p, S)$ of $p$ with respect to $S$ is the sum of (1) the distance from $p$ to the nearest point in $S \setminus \{p\}$ and (2) the distance from $p$ to the second nearest point in $S \setminus \{p\}$. The 2-dispersion cost $cost_2(S)$ of $S \subset P$ with $|S| \geq 3$ is $\min_{p \in S} cost_2(p, S)$.

Given $P, d$ and an integer $k \geq 3$, the 2-dispersion problem is the problem to find the subset $S$ of $P$ with $|S| = k$ such that the 2-dispersion cost $cost_2(S)$ is maximized.

3. Greedy Algorithm

Now we give an approximation algorithm to solve the 2-dispersion problem. See Algorithm 1. The algorithm is a simple greedy algorithm.

Algorithm 1 greedy($P, d, k$)

compute $S_3 \subset P$ consisting of the three points $p_1, p_2, p_3$ with maximum $cost_3(S_3)$

for $i = 4$ to $k$ do
find a point $p_i \in P \setminus S_{i-1}$ such that $cost_2(p_i, S_{i-1})$ is maximized $S_i = S_{i-1} \cup \{p_i\}$
end for
output $S$

Now we consider the approximation ratio of the solution obtained by the algorithm.

Let $S^* \subset P$ be the optimal solution for a given 2-dispersion problem, and $S \subset P$ the solution obtained by
the algorithm above. We are going to show \( \text{cost}_2(S) \geq \text{cost}_2(S^*/(4 \sqrt{3})) \), namely the approximation ratio of our algorithm is at least \( 1/(4 \sqrt{3}) \).

Let \( D_p \) be the disk with center at \( p \) and the radius \( r^* = \text{cost}_2(S^*/(2 \sqrt{3})) \). We have the following three lemmas.

**Lemma 1** For any \( p \in P \), \( D_p \) properly contains at most two points in \( S^* \).

**Proof** Assume for a contradiction that \( D_p \) properly contains three points in \( S^* \). If the three points are locating at the corner points of the maximum equilateral triangle properly contained in \( D_p \), then the 2-dimension cost of the three points is maximized, and the cost is less than \( 2 \sqrt{3}r^* \). Thus for any solution \( S \subset P \) containing the three points \( \text{cost}_2(S) < 2 \sqrt{3}r^* = \text{cost}_2(S^*) \), a contradiction. \( \Box \)

**Lemma 2** For each \( i = 3, 4, \ldots, k \), \( \text{cost}_2(p_i, S_{i-1}) \geq r^* \) holds.

**Proof** For convenience we regard \( S = \{p_1, p_2\} \). Clearly the claim holds for \( i = 3 \). Assume \( j \leq k \) and the claim holds for each \( i = 3, 4, \ldots, j-1 \). Now we consider for \( i = j \). We have the following two cases.

**Case 1:** There is a point \( p^* \) in \( S^* \setminus S_{j-1} \) such that \( D_{p^*} \) properly contains at most one point in \( S_{j-1} \). Note that \( D_{p^*} \) is the disk with center at \( p^* \) and the radius \( r^* = \text{cost}_2(S^*/(2 \sqrt{3})) \).

Then the distance from \( p^* \) to the second nearest point in \( S_{j-1} \) is at least \( r^* \) so \( \text{cost}_2(p^*, S_{j-1}) \geq r^* \). Since the algorithm chooses \( p_j \) in a greedy manner, \( \text{cost}_2(p_j, S_{j-1}) \) is also at least \( r^* \). Thus \( \text{cost}_2(p_j, S_{j-1}) \geq r^* \) holds.

**Case 2:** Otherwise. (For each point \( p^* \) in \( S^* \setminus S_{j-1} \), \( D_{p^*} \) contains at least two points in \( S_{j-1} \).)

1. Construct directed graph \( G = (V, A) \) so that (1) \( V = S^* \cup S_{j-1} \) and (2) \( A = \{(u, v)u \in S^*, v \in S_{j-1}, u \neq v, \text{and} d(u, v) < r^*\} \).

Now we count the number \(|A|\) of direct edges. Let \( x_{ab} \) be the number of vertices in \( S^* \setminus S_{j-1} \) having exactly a incoming edges and \( b \) outgoing edges. Note that each \( s \in S^* \cap S_{j-1} \) has at most one incoming edges, since otherwise \( D_s \) has three points in \( S^* \), a contradiction to Lemma 1. Thus \( x_{ab} = 0 \) if \( a \geq 2 \).

Since Case 1 does not occur each vertex in \( S^* \setminus S_{j-1} \) has two or more outgoing edges, however each vertex in \( S^* \cap S_{j-1} \) may have one or less outgoing edge.

Thus by counting outgoing edges for each vertex in \( S^* \cap S_{j-1} \) with exactly one or zero incoming edge and exactly one or zero outgoing edge separately, \(|A| \geq 2(2k - x_{00} - x_{01} - x_{10} - x_{11}) + x_{01} + x_{11} \).

By Lemma 1 each vertex in \( S_{j-1} \) has at most two incoming edges, so we have \(|A| \leq 2(2j - x_{00} - x_{01} - x_{10} - x_{11} + x_{01} + x_{11}) \).

If \( x_{10} = 0 \) then the two counting above contradict each other, since \( j \leq k \).

Otherwise, there is a vertex, say \( t \), in \( S^* \setminus S_{j-1} \) with exactly one incoming edge, say \( (s, t) \), and no outgoing edge.

Now \( s \in S^* \setminus S_{j-1} \) holds, since otherwise \( s \in S^* \cap S_{j-1} \) holds and there is direct edge \((t, s)\), a contradiction. Then \( s \) has another outgoing edge \((s, t') \) except \((s, t)\).

Now \( t' \in S_{j-1} \setminus S^* \) holds, since otherwise \( D_t \) has three points in \( S^* \) corresponding to \( s, t, t' \), a contradiction to Lemma 1.

Now \( t' \) has exactly one incoming edge \((s, t)\), since otherwise \( t' \) has another incoming edge, say \((s', t)\), except \((s, t') \) then \( \text{cost}_2(s, S_{j-1}) \leq d(s, t) + d(s', t) \leq 3r^* = 3\text{cost}_2(S^*/(2 \sqrt{3})) \), a contradiction. Thus \( s, t, t' \) have the following two cases.

Thus Case 2 never occurs. \( \Box \)

**Lemma 3** For each \( i = 3, 4, \ldots, k \), \( \text{cost}_2(S_i) \geq r^*/2 \) holds.

**Proof** Clearly the claim holds for \( i = 3 \). Assume that \( j \leq k \) and the claim holds for each \( i = 3, 4, \ldots, j-1 \). Now we consider for \( i = j \).

We have \(|A'| \geq 2(2k - 2c - x_{00} - x_{01} - x_{10} - x_{11}) \), a contradiction to Lemma 1.

Thus Case 2 never occurs. \( \Box \)

**Theorem 1** \( \text{cost}_2(S) \geq \text{cost}_2(S^*/(4 \sqrt{3})) \).

Thus the approximation ratio of Algorithm 1 is at least \( 1/(4 \sqrt{3}) \).

Is the approximation ratio above best possible? We now provide an example for which our algorithm computes a solution with approximation ratio asymptotically 1/4. See an example in Fig. 1. \( P = \{q_1, q_2, q_3, q_4, q_5, q_6, r, s\} \) and \( k = 6 \) for which our algorithm computes a solution \( S = \{q_1, q_2, \ldots, q_6\} \), where the points are chosen in this order. The distances between points are as follows, \( d(q_1, q_2) = d(q_2, q_3) = d(q_3, q_1) = 1 \). \( q_5 \) is the midpoint between \( q_1 \) and \( q_2, \), \( q_6 \) is on the line segment between \( q_1 \) and \( q_3 \) and \( d(q_1, q_6) = 0.75 \) and \( d(q_3, q_6) = 0.25 \). Finally we set \( d(q_1, r) = d(q_2, r) = d(q_3, q_4) = \epsilon \), where \( \epsilon \) is small enough.
Fig. 1 An example of a solution $S = \{q_1, q_2, \ldots, q_6\}$ with approximation ratio 1/4.

Note that $\text{cost}_2(S) = \text{cost}_2(q_3, S) \leq 0.25 + \epsilon$ while $\text{cost}_2(S^*) = 1$ for $S^* = \{q_1, q_2, q_3, q_4, r, s\}$. Thus the approximation ratio is 1/4.

Thus we still have a chance to improve the approximation ratio of our simple greedy algorithm, or we can find an example of $P$ for which our algorithm generates a solution with approximation ratio smaller than 1/4.

4. Conclusion

In this paper we have presented a simple $1/(4\sqrt{3})$-approximation algorithm to solve the 2-dispersion problem. The running time of the algorithm is $O(n^3)$.

References


Kazuyuki Amano received his M.E. and D.E. degrees in information sciences from Tohoku University in 1993 and 1996, respectively. Presently, he is a professor in Department of Computer Science, Gunma University. His research interests include the theory of computation and combinatorics. He enjoys writing computer programs to solve puzzles and problems of discrete mathematics.

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. Since 1999 he has been a faculty member of Department of Computer Science, Faculty of Engineering, Gunma University. His research interests are graph algorithms and graph theory. He is a member of IPSJ and ACM.