Away from Rivals

Kazuyuki Amano *   Shin-ichi Nakano †

Abstract

Let P be a set of n points, and $d(p,q)$ be the distance between a pair of points p,q in P. We assume the distance is symmetric and satisfies the triangle inequality. 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 $\min_{p \in S}\{cost_2(p,S)\}$.

In this paper we give a simple $1/8-$approximation algorithm for the 2-dispersion problem.

1 Introduction

Many facility location problems compute locations minimizing some cost or distance [4, 5]. While in this paper we consider a dispersion problem which computes locations maximizing some cost or distance [1, 2, 3, 6, 9, 10, 11].

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 [2], and diversity maximization has became an important concept in information retrieval.

A typical dispersion problem is as follows. Given a set P of points 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 [9].

In this paper we consider a recently proposed related problem called the 2-dispersion problem [7, 8]. 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 among the cost 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. We call the problem 2-dispersion problem. In [7, 8] 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 1/8-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. In Section 4 we consider more general problem called c-dispersion problem. Finally Section 4 is a conclusion.

2 Definitions

Let P be a set of n points, and $d(p,q)$ be the distance between a pair of points p,q in P. We assume that the distance is symmetric and satisfies the triangle inequality, meaning $d(p,q) = d(q,p)$ and $d(p,q) + d(q,r) \geq d(p,r)$.

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.

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 $cost_2(S) \geq cost_2(S^*)/8$, namely the approximation ratio of our algorithm is at least 1/8.

*Department of Computer Science, Gunma University, amano@cs.gunma-u.ac.jp

†Department of Computer Science, Gunma University, nakano@cs.gunma-u.ac.jp

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 $cost_2(S_3)$
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

Let D_p be the disk with center at p and the radius $r^* = cost_2(S^*)/4$. Let $D^* = \{D_p | p \in S^*\}$. 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 $p_1, p_2, p_3 \in S^*$. Now $d(p_1, p_2) < 2r^*$ and $d(p_1, p_3) < 2r^*$ hold, then $cost_2(p_1, S^*) < d(p_1, p_2) + d(p_1, p_3) < 4r^* = cost_2(S^*)$, a contradiction. □

Lemma 2 *For each $i = 3, 4, \ldots, k$, $cost_2(p_i, S_{i-1}) \geq r^*$ holds.*

Proof. Clearly the claim holds for $i = 3$. Assume $j - 1 < k$ 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^* 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 $r^* = cost_2(S^*)/4$.

Then the distance from p^* to the 2nd nearest point in S_{j-1} is at least r^* so $cost_2(p^*, S_{j-1}) \geq r^*$. Since the algorithm choose p_j in a greedy manner, $cost_2(p_j, S_{j-1})$ is also at least r^*. Thus $cost_2(p_j, S_{j-1}) \geq r^*$ holds.

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

We now count the number N of distinct pairs (p^*, q) with (1) $p^* \in S^*$, (2) $q \in S_{j-1}$ and (3) $d(p^*, q) < r^*$.

By Lemma 1 each D_q with $q \in S_{j-1}$ contains at most two points in S^*. Thus $N \leq 2(j-1) < 2k$. Since Case 1 does not occur, each D_{p^*} with $p^* \in S^*$ contains two or more points in S_{j-1}, so $N \geq 2k$. A contradiction.

Thus Case 2 never occurs. □

Lemma 3 *For each $i = 3, 4, \ldots, k$, $cost_2(S_i) \geq r^*/2$ holds.*

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

To prove $cost_2(S_j) \geq r^*/2$ we only need to show for any three points u, v, w in S_j, $d(u, v) + d(u, w) \geq r^*/2$. We have the following four cases.

If none of u, v, w is p_j, then $d(u, v) + d(u, w) \geq r^*/2$ is clearly held as it was held in S_{j-1}.

If u is p_j, then by Lemma 2 $d(p_j, v) + d(p_j, w) \geq cost_2(p_j, S_{j-1}) \geq r^*$. Thus $d(u, v) + d(u, w) \geq r^*/2$ holds.

If v is p_j, assume for a contradiction that $d(u, p_j) + d(u, w) < r^*/2$. Then clearly $d(u, p_j) = d(p_j, u) < r^*/2$ and by the triangle inequality $d(p_j, w) \leq d(p_j, u) + d(u, w) = d(u, p_j) + d(u, w) < r^*/2$. Then $cost_2(p_j, S_{j-1}) \leq d(p_j, u) + d(p_j, w) < r^*$, contradiction to Lemma 2. Thus if v is p_j then $d(u, p_j) + d(u, w) \geq r^*/2$ holds.

If w is p_j, then we can prove the claim in a similar manner to the case v is p_j. □

Since $S_k = S$, we have the following theorem.

Theorem 4 $cost_2(S) \geq cost_2(S^*)/8$.

Thus the approximation ratio of **Algorithm 1** is at least 1/8.

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 $d(q_1, r) = d(q_2, s) = d(q_3, q_4) = \epsilon$, where ϵ is small enough.

Note that $cost_2(S) = cost_2(q_3, S) \leq 0.25 + \epsilon$ while $cost_2(S^*) = 1$ 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 Generalization

The 2-dispersion problem can be naturally generalized to the c-dispersion problem as follows.

For a point $p \in P$ and a subset $S \subset P$ with $|S| \geq c+1$, the c-dispersion cost $cost_c(p, S)$ of $p \in S$ with respect to S is the sum of the distances from p to the nearest c points in $S \setminus \{p\}$. The c-dispersion cost $cost_c(S)$ of $S \subset P$ with $|S| \geq c+1$ is $\min_{p \in S}\{cost_c(p, S)\}$. Given P, d and an integer $k \geq c+1$, the c-dispersion problem is the problem to find the subset S of P with $|S| = k$ such that the c-dispersion cost $cost_c(S)$ is maximized.

We can naturally generalize our greedy algorithm in Section 3 to the algorithm to solve the c-dispersion problem. See **Algorithm 2**.

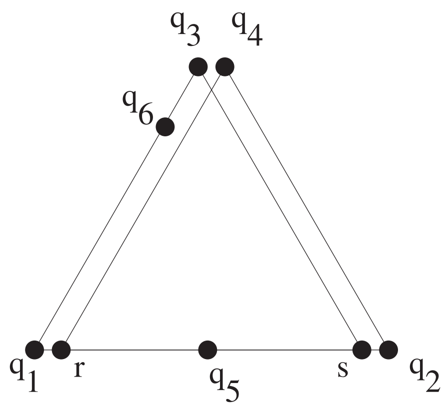

Figure 1: An example of a solution $S = \{q_1, q_2, \ldots, q_6\}$ with approximation ratio 1/4.

Algorithm 2 greedy-c(P, d, k)

compute $S_{c+1} \subset P$ consisting of the $c+1$ points $p_1, p_2, \ldots, p_{c+1}$ with maximum cost $cost_c(S_c)$
for $i = c+2$ to k **do**
  find a point $p_i \in P \setminus S_{i-1}$ such that $cost_c(p_i, S_{i-1})$ is maximized
  $S_i = S_{i-1} \cup \{p_i\}$
end for
output S

Let S^* be the optimal solution for a given c-dispersion problem, and $S \subset P$ the solution obtained by the greedy algorithm above. We now consider the approximation ratio of the solution obtained by the greedy algorithm.

Let D_p be the disk with center at p and the radius $r^{**} = cost_c(S^*)/(2c)$. Let $D^{**} = \{D_p | p \in S^*\}$. We have the following three lemmas.

Lemma 5 *For any $p \in P$, D_p properly contains at most c points in S^*.*

Proof. Assume for a contradiction that D_p properly contains $c+1$ points, say $q_1, q_2, \ldots, q_{c+1} \in S^*$. Now $d(q_{c+1}, q_t) < 2r^{**}$ holds for each $t = 1, 2, \ldots, c$. Then $cost_2(q_{c+1}, S^*) < d(q_{c+1}, q_1) + d(q_{c+1}, q_2) + \cdots + d(q_{c+1}, q_c) < 2cr^{**} = cost_c(S^*)$, a contradiction. □

Lemma 6 *For each $i = c+1, c+2, \ldots, k$, $cost_c(p_i, S_{i-1}) \geq r^{**}$ holds.*

Proof. Clearly the claim holds for $i = c+1$. Assume $j-1 < k$ and the claim holds for each $i = c+1, c+2, \ldots, j-1$. Now we consider for $i = j$. We have the following two cases.
Case 1: There is a point p^* in S^* such that D_{p^*} properly contains at most $c-1$ point in S_{j-1}.

Then the distance from p^* to the c-th nearest point in S_{j-1} is at least r^{**} so $cost_c(p^*, S_{j-1}) \geq r^{**}$. Since the algorithm choose p_j in a greedy manner, $cost_c(p_j, S_{j-1})$ is also at least r^{**}. Thus $cost_c(p_j, S_{j-1}) \geq r^{**}$ holds.
Case 2: Otherwise.

We now count the number N of distinct pairs (p^*, q) with (1) $p^* \in S^*$, (2) $q \in S_{j-1}$ and (3) $d(p^*, q) < r^{**}$.

By Lemma 5 each D_q with $q \in S_{j-1}$ contains at most c points in S^*. Thus $N \leq c(j-1) < ck$. Since Case 1 does not occur, each D_{p^*} with $p^* \in S^*$ contains c or more points in S_{j-1}, so $N \geq ck$. A contradiction.

Thus Case 2 never occurs. □

Lemma 7 *For each $i = c+1, c+2, \ldots, k$, $cost_c(S_i) \geq r^{**}/c$ holds.*

Proof. Clearly the claim holds for $i = c+1$. Assume that $j-1 < k$ and the claim holds for each $i = c+1, c+2, \ldots, j-1$. Now we consider for $i = j$.

For any point u in S_j we show $cost_c(u, S_j) \geq r^{**}/c$ holds. We have three cases. Let $S(u)$ be the set of point in $S_j \setminus \{u\}$ consisting of the nearest c points to u.

If $p_j \notin \{u\} \cup S(u)$, then clearly $cost_c(u, S_j) \geq r^{**}/c$ holds, since $cost_c(u, S_{j-1}) \geq r^{**}/c$ holds.

If $p_j = u$, then by Lemma 6 $cost_c(u, S_j) \geq r^{**}$ holds, so $cost_c(u, S_j) \geq r^{**}/c$ holds.

If $p_j \in S(u)$, then assume for a contradiction that $cost_c(u, S_j) < r^{**}/c$. Let $S(u) = \{q_1, q_2, \ldots, q_c\}$ and $q_x = p_j$. Then clearly $d(u, p_j) < cost_c(u, S_j) < r^{**}/c$ and by the triangle inequality for each $t \neq x$ $d(p_j, q_t) \leq d(p_j, u) + d(u, q_t) = cost_c(u, S_j) < r^{**}/c$. Then $cost_c(p_j, S_j) \leq d(p_j, q_1) + d(p_j, q_2) + \cdots + d(p_j, q_c) < r^{**}$, contradiction to Lemma 6. □

Since $S_k = S$, we have the following theorem.

Theorem 8 $cost_c(S) \geq cost_c(S^*)/(2c^2)$.

5 Conclusion

In this paper we have presented a simple $1/8-$approximation algorithm to solve the 2-dispersion problem. The running time of the algorithm is $O(n^3)$. Similarly we have presented a simple $1/(2c^2)-$approximation algorithm to solve the c-dispersion problem. The running time of the algorithm is $O(n^{c+1})$.

References

[1] C. Baur and S.P. Fekete, Approximation of Geometric Dispersion Problems, Pro. of APPROX '98, Pages 63-75 (1998).

[2] A. Cevallos, F. Eisenbrand and R. Zenklusen, Local search for max-sum diversification, Proc. of SODA '17, pp.130-142 (2017).

[3] B. Chandra and M. M. Halldorsson, Approximation Algorithms for Dispersion Problems, J. of Algorithms, 38, pp.438-465 (2001).

[4] Z. Drezner, Facility Location: A Survey of Applications and Methods, Springer (1995).

[5] Z. Drezner and H.W. Hamacher, Facility Location: Applications and Theory, Springer (2004).

[6] R. Hassin, S. Rubinstein and A. Tamir, Approximation Algorithms for Maximum Dispersion, Operation Research Letters, 21, pp.133-137 (1997).

[7] T. L. Lei, R. L. Church, A unified model for dispersing facilities, Geographical Analysis, 45, pp.401-418 (2013).

[8] T. L. Lei, R. L. Church, On the unified dispersion problem: Efficient formulations and exact algorithms, European Journal of Operational Research, 241, pp.622-630 (2015).

[9] S. S. Ravi, D. J. Rosenkrantz and G. K. Tayi, Heuristic and Special Case Algorithms for Dispersion Problems, Operations Research, 42, pp.299-310 (1994).

[10] M. Sydow, Approximation Guarantees for Max Sum and Max Min Facility Dispersion with Parameterised Triangle Inequality and Applications in Result Diversification, Mathematica Applicanda, 42, pp.241-257 (2014).

[11] D. W. Wang and Yue-Sun Kuo, A study on Two Geometric Location Problems, Information Processing Letters, 28, pp.281-286 (1988).