Exact Algorithms for the Max-Min Dispersion Problem

Toshihiro Akagi[1], Tetsuya Araki[2], Takashi Horiyama[3], Shin-ichi Nakano[1], Yoshio Okamoto[4], Yota Otachi[5], Toshiki Saitoh[6], Ryuhei Uehara[7], Takeaki Uno[8], and Kunihiro Wasa[8]

[1] Gunma University, Japan
[2] Tokyo Metropolitan University, Japan
[3] Saitama University, Japan
[4] The University of Electro-Communications, Japan
[5] Kumamoto Univesity, Japan
[6] Kyushu Institute of Technology, Japan
[7] JAIST, Japan
[8] National Institute of Informatics, Japan

Ver. Monday 25th December, 2017 08:05

Abstract. Given a set P of n elements, and a function d that assigns a non-negative real number $d(p, q)$ for each pair of elements $p, q \in P$, we want to find a subset $S \subseteq P$ with $|S| = k$ such that the cost $\mathrm{cost}(S) = \min\{d(p, q) \mid p, q \in S\}$ is maximized. This is the max-min k-dispersion problem.
In this paper, exact algorithms for the max-min k-dispersion problem are studied. We first show the max-min k-dispersion problem can be solved in $O(n^{\omega k/3} \log n)$ time. Then, we show two special cases in which we can solve the problem quickly.

Keywords: dispersion problem, algorithm

1 Introduction

The facility location problem and many of its variants have been studied extensively [8, 9]. Typically, we are given a set of locations on which facilities are placed and an integer k, we want to place k facilities on some locations in such a way that a given objective is minimized. In the *dispersion* problem, we want to minimize the interference between the placed facilities. As an example scenario, consider that we are planning to open several chain stores in a city. We wish to locate the stores mutually far away from each other to avoid self-competition. Another example is a case where facilities are mutually obnoxious, such as nuclear power plants and oil storage tanks. See more applications, including *result diversification*, in [21, 23, 16, 6].

More specifically, in the *max-min k-dispersion problem*, we are given a set P of n elements which represent possible locations, and a function d that assigns a non-negative real number $d(p, q)$ for each pair of elements $p, q \in P$. Throughout

this paper, we assume that d is symmetric (i.e., $d(p,q) = d(q,p)$ for all $p, q \in P$, and $d(p,p) = 0$ for all $p \in P$) , but we do not assume that d satisfies the triangle inequality. The value $d(p,q)$ represents the distance between p and q. We are also given an integer k with $k \leq n$. Then, we want to find a subset $S \subseteq P$ with $|S| = k$ such that the cost

$$\mathrm{cost}(S) = \min\{d(p,q) \mid p, q \in S\}$$

is maximized.

The max-min k-dispersion problem was recognized in the early days of research for location theory. At least, Shier [22] wrote and published a paper about k-dispersion on trees in 1977, and related the problem with the k-center problem. The related literature up to the mid 1980's was reviewed by Kuby [15]. Erkut [10] proved the problem is NP-hard even when the triangle inequality is satisfied. A geometric version was studied by Wang and Kuo [25], where points lie in the d-dimensional space, and the distance is Euclidean. Then, they proved the following: when $d = 1$, the problem can be solved in $O(kn)$ time by dynamic programming after $O(n \log n)$-time sorting; when $d = 2$, the problem is NP-hard. The running time for $d = 1$ was recently improved to $O(n \log\log n)$ (after sorting) [2] by sorted matrix search method [13]. (For a good survey for the sorted matrix search method see [1, Section 3.3].)

Ravi et al. [21] proved that the max-min k-dispersion cannot be approximated within any factor in polynomial time, and cannot be approximated within the factor of two in polynomial time when the distance satisfies the triangle inequality, unless P = NP. They also gave a polynomial-time algorithm with approximation ratio two when the triangle inequality is satisfied. Thus, the factor of two is tight.

In the *max-sum k-dispersion problem*, the objective is to maximize the sum of distances between k facilities. The proof by Erkut [10] can easily be adapted to show that the max-sum k-dispersion problem is NP-hard. It is not known whether the problem is still NP-hard on the 2-dimensional Euclidean space. Ravi et al. [21] gave an $O(n \log n + kn)$-time exact algorithm when the points lie on a line, a polynomial-time factor-four approximation algorithm when the triangle inequality is satisfied, and a polynomial-time factor-$(\pi/2 + \epsilon)$ approximation algorithm for the 2-dimensional Euclidean space (note that $\pi/2 \approx 1.571$). The factor of four was improved to two by Birnbaum and Goldman [4] and Hassin et al. [14]. Fekete and Meijer [11] studied the case for the d-dimensional space with the L_1 distance, and gave an $O(n)$-time exact algorithm when k is fixed (after sorting the points by x-coordinates and y-coordinates), and a polynomial-time approximation scheme when k is part of the input. Polynomial-time approximation schemes for the Euclidean distance, or more generally for the negative-type metric were given by Cevallos et al. [5, 6]. For other variations, see [3, 7].

In this paper, exact algorithms for the max-min k-dispersion problem are studied. The main contributions are twofold.

First, we review an intimate relationship with the max-min k-dispersion problem to the maximum independent set problem. A reduction to prove the

NP-hardness of the max-min k-dispersion problem uses the k-independent set problem [21], and we prove the reverse reduction is possible. Namely, we show that if the k-independent set problem can be solved in $T(n, k)$ time, then the max-min k-dispersion problem can be solved in $O((T(n, k) + n^2) \log n)$ time. Note that the k-independent set problem can be solved in $O(n^{\omega k/3})$ time [20], where $\omega < 2.373$ is the matrix multiplication exponent (See Le Gall [17] for the current best bound on ω). Therefore, our result implies that the max-min k-dispersion problem can be solved in $O(n^{\omega k/3} \log n)$ time. We will also discuss some consequences of these reductions.

Then, we turn our attention to two special cases in which we can solve the problem quickly.

We study the case when a set of n points lie on a line, and obtain an $O(n)$-time algorithm after sorting. This is an improvement over the recent $O(n \log \log n)$-time algorithm [2]. In our algorithm, we employ the tree partitioning algorithm by Frederickson [13]. Next, we consider the case when a set of n points lie on a circle on the Euclidean plane, and the distance is measure by the shortest arc length on the circle. Then, we obtain an $O(n)$-time algorithm after sorting.

The remainder of this paper is organized as follows. In Section 2, we consider a relationship between the max-min k-dispersion problem and the k-independent set problem. Section 3 gives an algorithm to solve the dispersion problem when P is a set of points on a line. Section 4 gives an algorithm to solve the dispersion problem when P is a set of points on a circle. Finally Section 5 is a conclusion.

2 Relationship with the maximum independent set problem

2.1 General case

First, we reduce the max-min k-dispersion problem to the k-independent set problem. Here, we remind the definition of the max-min k-dispersion problem. In the max-min k-dispersion problem, we are given a set P of n elements, a function d that assigns a non-negative real number $d(p, q)$ for each pair of elements p, q of P, and an integer k such that $k \leq n$. Then, we want to find a subset $S \subseteq P$ with $|S| = k$ that maximizes $\mathrm{cost}(S) = \min\{d(p, q) \mid p, q \in S\}$.

In k-independent set problem, we are given an undirected graph $G = (V, E)$, and we want to determine whether there exists a subset $S \subseteq V$ of k vertices such that each pair of vertices in S is non-adjacent, and find such a set S if exists.

For our reduction, we consider the following question $\mathrm{Q}(r)$ for a given real number r: does there exist a set of k locations such that the distance of any two locations is at least r.

Observe that the optimal value for the max-min k-dispersion problem is the maximum of r such that the answer to $\mathrm{Q}(r)$ is yes. We also observe that the optimal value lies in the set of possible distances $\{d(p, q) \mid p, q \in P\}$, and the answers to $\mathrm{Q}(r)$ have the following monotonicity: If the answer to $\mathrm{Q}(r)$ is yes, and $r' < r$, then the answer to $\mathrm{Q}(r')$ is also yes. Therefore, if $\mathrm{Q}(r)$ can be answered

correctly for any r, then we can solve the max-min k-dispersion problem by performing binary search over the all possible $O(n^2)$ candidates after sorting the distances $d(p, q)$. The sorting takes $O(n^2 \log n^2) = O(n^2 \log n)$ time.

To answer the question $\mathrm{Q}(r)$ above, we use the k-independent set problem. To this end, we construct the following undirected graph $G(r) = (V, E)$. The vertex set V is identical to P. Two locations $p, q \in P$ are joined by an edge in $G(r)$ if and only if $d(p, q) < r$. Remind that we have the symmetry assumption $d(p, q) = d(q, p)$ for all $p, q \in P$, and thus the undirected graph is well-defined.

Let $S \subseteq V$ be an independent set of size k in $G(r)$. Then, by definition, every pair of two vertices $p, q \in S$ satisfies $d(p, q) \geq r$. This implies that $\mathrm{cost}(S) \geq r$, and the answer to $\mathrm{Q}(r)$ is yes. On the other hand, if the answer to $\mathrm{Q}(r)$ is yes, then there exists a set S of k locations such that $d(p, q) \geq r$ for all $p, q \in S$. This means that S is an independent set in $G(r)$. Therefore, it follows that $G(r)$ has an independent set of size k if and only if the answer to $\mathrm{Q}(r)$ is yes.

Now we analyze the running time. We assume that the k-independent set problem can be solved in $T(n, k)$ time on n-vertex undirected graphs. Sorting the distances takes $O(n^2 \log n)$ time as discussed above. Then, we perform binary search to find the maximum r such that the answer to $\mathrm{Q}(r)$ is yes among the $O(n^2)$ candidates. The number of iterations is $O(\log n^2) = O(\log n)$. For each iteration, we construct the graph $G(r)$, which takes $O(n^2)$ time, and solve the k-independent set problem, which takes $T(n, k)$ time. Therefore, the overall running time is $O(n^2 \log n + (n^2 + T(n, k)) \log n) = O((T(n, k) + n^2) \log n)$.

The current best bound for $T(n, k)$ is $O(n^{\omega k/3})$ [20], where ω is the matrix multiplication exponent. Since $\omega \geq 2$, we obtain the following theorem.

Theorem 1. *The max-min k-dispersion problem can be solved in $O(n^{\omega k/3} \log n)$ time.*

On the other hand, the k-independent set problem can be reduced to the max-min k-dispersion problem as follows [21]. Let $G = (V, E)$ be an undirected graph given as an instance of the k-independent set problem. Then, we construct the following instance of the max-min k-dispersion problem. The set of locations is V. The distance $d(p, q)$ between $p, q \in V$ is defined as follows: $d(p, q) = 1$ if p and q are adjacent in G, and $d(p, q) = 2$ otherwise. Then, G has an independent set S of size k if and only if there exists a set S of locations with $|S| = k$ such that $\mathrm{cost}(S) = 2$.

This reduction does not only prove the NP-hardness of the max-min k-dispersion problem, but also proves the W[1]-hardness of the problem, when k is a parameter, as the k-independent set problem is W[1]-hard when k is a parameter. W[1]-hardness is a concept in parameterized complexity theory. Refer to [12].

In summary, up to a logarithmic-factor overhead in the running time, the max-min k-dispersion problem is equivalent to the k-independent set problem.

2.2 On the Euclidean plane

The discussion above also applies to some special cases. We now look at the case when P is a set of points on the Euclidean plane.

We look at the construction of the graph $G(r)$ more carefully. Remind that in $G(r)$, two vertices $p, q \in P$ are joined by an edge if and only if $d(p, q) < r$. This matches the definition of a *unit disk graph*. Here, we remind the definition of a unit disk graph. A unit disk graph is an undirected graph defined for a set of unit disks. The vertex set is the set of unit disks, and two disks are joined by an edge if and only if the disks intersect. Usually, disks are considered to be closed, but the results below also hold for open disks.

To view $G(r)$ as a unit disk graph, we consider an open disk of radius $r/2$ that has a center at each point $p \in P$. Then, two such disks centered at $p, q \in P$ intersect if and only if $d(p, q) < r$. If we scale the whole picture by the factor of $2/r$, then we obtain $G(r)$ as a unit disk graph.

It is known that the k-independent set problem on unit disk graphs can be solved in $n^{O(\sqrt{k})}$ time [18]. Therefore, from the discussion above, we obtain the following theorem.

Theorem 2. *The max-min k-dispersion problem can be solved in $n^{O(\sqrt{k})}$ time when P lies on the Euclidean plane.*

On the other hand, if the optimal value for the max-min k-dispersion problem is r, then there exists a set of k pairwise disjoint open disks of radius $r/2$ that have their centers in P. This means that if the max-min k-dispersion problem can be solved in $T(n, k)$ time when P is a set of points on the Euclidean plane, then the k-independent set problem on unit disk graphs can be solved in $T(n, k)$ time, too. Since the k-independent set problem on unit disk graphs cannot be solved in $n^{o(\sqrt{k})}$ time under the exponential time hypothesis [19], we can also conclude that the max-min k-dispersion problem on the Euclidean plane cannot be solved in $n^{o(\sqrt{k})}$ time under the exponential time hypothesis. Thus, the running time in Theorem 2 is essentially optimal.

3 Max-min dispersion on a line

In this section we show one can solve the k-dispersion problem in $O(n)$ time if P is a set of points on a line and the order of P on the line is given. The idea of our algorithm is a reduction to the path partitioning problem [13], which can be solved in $O(n)$ time.

Let T be a tree in which each vertex has a non-negative weight w, and k be an integer. The tree k-partitioning problem is to delete $k - 1$ edges in the tree so as to maximize the lightest weight of the remaining subtree. The tree k-partitioning problem can be solved in $O(n)$ time [13], where n is the number of vertices in the tree. If the input tree is a path then it is the *path k-partitioning problem*.

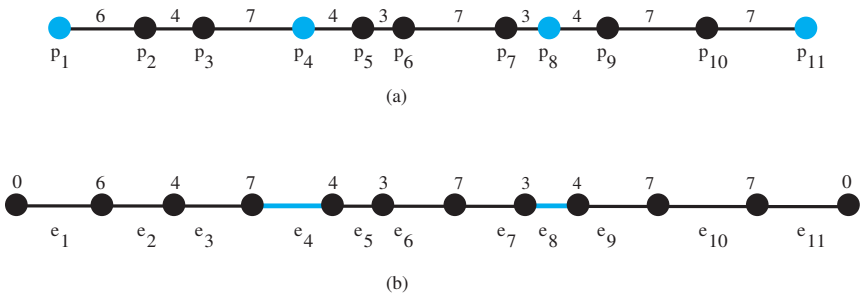

Fig. 1. (a) A 4-dispersion problem on a line, and (b) the corresponding 3 path-partitioning problem on a line.

Given an instance (P, k) of the max-min k-dispersion problem such that P is a set of points on a line and $k \geq 3$, we can transform it to an instance $(P', k-1)$ of the path $(k-1)$-partitioning problem as follows. First, we construct a path $P' = (V', E')$. Assume $P = \{p_1, p_2, \ldots, p_n\}$ and the points appear in this order on the line from left to right. Define $V' = \{p'_0, p'_1, \ldots, p'_n\}$, $E' = \{e_i = (p'_{i-1}, p'_i) \mid p_i \in V\}$, $w(p'_i) = d(p_i, p_{i+1})$ for each $i = 1, 2, \ldots, n-1$, and $w(p'_0) = w(p'_n) = 0$. See an example in Fig. 1. A solution of the max-min 4-dispersion problem in Fig. 1(a) is $\{p_1, p_4, p_8, p_{11}\}$ and its cost is 17. A solution of the path 3-partitioning problem in Fig. 1(b) is $\{e_4, e_8\}$ and its cost is 17. One can observe that a solution of the max-min k-dispersion problem contains $\{p_1, p_n\}$, and if a solution of the max-min k-dispersion problem is $\{p_1, p_n\} \cup \{p_{i_1}, p_{i_2}, \ldots p_{i_{k-2}}\}$, then a solution of the path $k-1$-partitioning problem is $\{e_{i_1}, e_{i_2}, \cdots, e_{i_{k-2}}\}$.

Since one can solve the path k-partitioning problem in $O(n)$ time [13], one can solve the max-min k-dispersion problem in $O(n)$ time.

Theorem 3. *The max-min k-dispersion problem can be solved in $O(n)$ time when P is a set of n points on a line and the order of P on the line is given.*

4 Max-min k-dispersion on a circle

In this section, we show one can solve the k-dispersion problem in $O(n)$ time if P is a set of points on a circle and the order of P on the circle is given. The distance is measured by the arc length of the circle. We assume (1) the length of the circumference, and (2) the length of the clockwise arc from some designated point to each point are given. So, one can compute the central angle corresponding to a given arc in constant time.

First, we design a simple algorithm to solve the max-min 3-dispersion problem which runs $O(n)$ time if P is a set of points on a circle. The outline of the algorithm is as follows. For each point $p_i \in P$, we compute the best three points including p_i, and then output the best three points among them.

Algorithm Find-3-dispersion-on-a-circle(P,k)

```
cost = 0
Ans = ∅
for i = 1 to n do
   find a set S of three points including p_i with maximum cost(S)
   if cost(S) > cost then
      cost = cost(S)
      Ans = S
   end if
end for
Output Ans
```

Now we introduce some definitions. Given $p_i \in P$, let p_i^ℓ and p_i^r be the points on the circle such that p_i, p_i^ℓ, p_i^r are the three corners of the equilateral triangle. Let $A_\ell = (p_i, p_i^\ell)$ be the arc of the circle between p_i and p_i^ℓ with central angle 120°, $A_r = (p_i, p_i^r)$ be the arc of the circle between p_i and p_i^r with central angle 120°, and $A_t = (p_i^\ell, p_i^r)$ be the (open) arc of the circle between p_i^ℓ and p_i^r with central angle 120°.

A set $S = \{p_i, p_\ell, p_r\}$ is of *type-LR* with respect to p_i if $p_\ell \in A_\ell$ and $p_r \in A_r$. Similarly, S is of *type-LL* with respect to p_i if $p_\ell \in A_\ell$ and $p_r \in A_\ell$, is of *type-RR* with respect to p_i if $p_\ell \in A_r$ and $p_r \in A_r$, and is of *type-LT* with respect to p_i if $p_\ell \in A_\ell$ and $p_r \in A_t$, is of *type-TT* with respect to p_i if $p_\ell \in A_t$ and $p_r \in A_t$, etc.

We have the following lemma.

Lemma 1. *When P is a set of points on a circle, an optimal solution S of the max-min 3-dispersion problem is of type-LR or type-TT with respect to some $p_i \in S$.*

Proof. By case analysis. Assume $S = \{p_i, p_\ell, p_r\}$ and p_i, p_ℓ and p_r appear in the clockwise order on the circle. If S is of type-LL with respect p_i, then then S is of type-LR with respect p_ℓ. If S is of type-RR with respect p_i, then then S is of type-LR with respect p_r. If S is of type-LT with respect p_i, then either (1) the arc of the circle between p_ℓ and p_r has the central angle less than 120° and S is of type-LR with respect to p_ℓ, or (2) the arc of the circle between p_ℓ and p_r has the central angle at least 120° and S is of type-TT with respect to p_r. If S is type-TR with respect p_i, then we can prove either (1) S is of type-LR with respect to p_r, or (2) S is of type-TT with respect to p_ℓ. **Q.E.D.**

Lemma 2. *(a) If a solution $S = \{p_i, p_\ell, p_r\}$ is of type-TT with respect to p_i, then p_ℓ is the first point in P on the circle after p_i^ℓ in the clockwise order, and p_r is the first point in P on the circle after p_i^r in the counterclockwise order.*

(b) If a solution $S = \{p_i, p_\ell, p_r\}$ is of type-LR with respect to p_i, then p_ℓ is the first point in P on the circle after p_i^ℓ in the counterclockwise order, and p_r is the first point in P on the circle after p_i^r in the clockwise order.

Proof. (a) Since S is of type-TT, $\min\{d(p_i, p_\ell), d(p_\ell, p_r), d(p_r, p_i)\} = d(p_\ell, p_r)$ holds. Thus, choosing S so as to maximize $d(p_\ell, p_r)$ results in the S with the maximum cost.

(b) Since S is of type-LR, $\min\{d(p_i, p_\ell), d(p_\ell, p_r), d(p_r, p_i)\} \neq d(p_\ell, p_r)$ holds. Thus, choosing S so as to maximize $d(p_i, p_\ell)$ and $d(p_i, p_r)$ results in S with the maximum cost. **Q.E.D.**

We need to compute for each p_i the first point in P on the circle after p_i^ℓ in the clockwise order, and we need $O(n)$ time in total. Similarly we can compute for each p_i the first point in P on the circle after p_i^r in the counterclockwise order, and we need $O(n)$ time in total. We perform this as preprocessing.

Then, for each p_i we need $O(1)$ time to find (1) the set S of three points of type-LR with respect to p_i with maximum cost(S), and (2) the set S of three points of type-TT with respect to p_i with maximum cost(S), and then choose the larger one. This is the set S of three points including p_i with maximum cost(S).

Thus, we have the following theorem.

Theorem 4. *The max-min 3-dispersion problem in $O(n)$ time when P is a set of points on a circle and the order of points on the circle is given.*

Let P be a set of points on a circle and each point has a non-negative weight, and k be an integer. The circle partitioning problem deletes k edges on the circle so as to maximize the lightest weight of the remaining subpath. So, this is the circle version of the path k-partition problem, explained in Section 3. One can solve the circle partition problem in $O(n)$ time [24].

Theorem 5. *One can solve the max-min k-dispersion problem in $O(n)$ time when P is a set of points on a circle and the order of the points on the circle is given.*

The algorithm is rather complicated. However, our algorithm for $k = 3$ is simple to implement.

5 Conclusion

In this paper we have presented some algorithms to solve the dispersion problems.

If P is a set of points on a line and the ordering of the points on the line is given one can solve the dispersion problem in $O(n)$ time.

If P is a set of points on a circle and the ordering of the points on the circle is given one can solve the k-dispersion problem in $O(n)$ time.

One can solve the max-min 3-dispersion problem in $O(n^\omega \log n)$ time, and the max-min k-dispersion problem in $n^{O(\sqrt{k})}$ time.

Can we solve the problem efficiently if P is a set of the corner vertices on a convex polygon?

References

1. P. Agarwal and M. Sharir, Efficient algorithms for geometric optimization, Computing Surveys, 30, pp. 412–458 (1998).
2. T. Akagi and S. Nakano, Dispersion on the line, IPSJ SIG Technical Reports, 2016-AL-158-3 (2016).
3. C. Baur and S. P. Fekete, Approximation of geometric dispersion problems, Proc. of APPROX 1998, pp. 63–75 (1998).
4. B. Birnbaum and K.J.Goldman, An improved analysis for a greedy remote-clique algorithm using factor-revealing LPs, Algorithmica, 50, pp. 42–59 (2009).
5. A. Cevallos, F. Eisenbrand and R. Zenklusen, Max-sum diversity via convex programming, Proc. of SoCG 2016, pp. 26:1–26:14 (2016).
6. A. Cevallos, F. Eisenbrand and R. Zenklusen, Local search for max-sum diversification, Proc. of SODA 2017, pp. 130–142 (2017).
7. B. Chandra and M. M. Halldorsson, Approximation algorithms for dispersion problems, J. of Algorithms, 38, pp. 438-465 (2001).
8. Z. Drezner, Facility location: A Survey of Applications and Methods, Springer (1995).
9. Z. Drezner and H.W. Hamacher, Facility Location: Applications and Theory, Springer (2004).
10. E. Erkut, The discrete p-dispersion problem, European Journal of Operational Research, 46, 48–60 (1990).
11. S. P. Fekete and H. Meijer, Maximum dispersion and geometric maximum weight cliques, Algorithmica, 38, pp. 501-511 (2004).
12. J. Flum and M. Grohe, Parameterized Complexity Theory, Springer (2006).
13. G. Frederickson, Optimal algorithms for tree partitioning, Proc. of SODA 1991, pp. 168–177 (1991).
14. R. Hassin, S. Rubinstein and A. Tamir, Approximation algorithms for maximum dispersion, Operation Research Letters, 21, pp. 133–137 (1997).
15. M. J. Kuby, Programming models for facility dispersion: the p-dispersion and maxisum dispersion problems, Geographical Analysis, 19, pp. 315–329 (1987).
16. T. L. Lei and R. L. Church, On the unified dispersion problem: Efficient formulations and exact algorithms, European Journal of Operational Research, 241, pp. 622-630 (2015).
17. F. Le Gall, Powers of tensors and fast matrix multiplication, Proc. ISSAC 2014, pp. 296–303 (2014).
18. D. Marx and M. Pilipczuk, Optimal parameterized algorithms for planar facility location problems using Voronoi diagrams, Proc. ESA 2015, LNCS 9294, pp. 865–877 (2015).
19. D. Marx and A. Sidiropoulos, The limited blessing of low dimensionality: when $1 - 1/d$ is the best possible exponent for d-dimensional geometric problems, Proc. of SoCG 2014, pp. 67–76, 2014.
20. J. Nešetřil and S. Poljak, On the complexity of the subgraph problem, Commentationes Mathematicae Universitatic Corolinae, 26, pp. 415–419 (1985).
21. 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).
22. D. R. Shier, A min-max theorem for p-center problems on a tree, Transportation Science, 11, pp. 243–252 (1977).
23. 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).

24. K. H. Tsai and D. W. Wang, Optimal algorithms for circle partitioning, Proc. of COCOON 1997, LNCS 1276, pp. 304–310, 2014.
25. D. W. Wang and Y.-S. Kuo, A study on two geometric location problems, Information Processing Letters, 28, pp. 281–286 (1988).