Maximum Dispersion and Geometric Maximum Weight Cliques

Sándor P. Fekete[1] and Henk Meijer[2*]

[1] Department of Mathematics, TU Berlin, 10623 Berlin, Germany
fekete@math.tu-berlin.de
[2] Department of Computing and Information Science
Queen's University, Kingston, Ontario K7L 3N6, Canada
henk@cs.queensu.ca

Abstract. We consider geometric instances of the problem of finding a set of k vertices in a complete graph with nonnegative edge weights. In particular, we present algorithmic results for the case where vertices are represented by points in d-dimensional space, and edge weights correspond to rectilinear distances. This problem can be considered as a facility location problem, where the objective is to "disperse" a number of facilities, i.e., select a given number of locations from a discrete set of candidates, such that the average distance between selected locations is maximized. Problems of this type have been considered before, with the best result being an approximation algorithm with performance ratio 2. For the case where k is fixed, we establish a linear-time algorithm that finds an optimal solution. For the case where k is part of the input, we present a polynomial-time approximation scheme.

1 Introduction

A common problem in the area of facility location is the selection of a given number of k locations from a set P of n feasible positions, such that the selected set has optimal distance properties. A natural objective function is the maximization of the average distance between selected points; *dispersion* problems of this type come into play whenever we want to minimize interference between the corresponding facilities. Examples include oil storage tanks, ammunition dumps, nuclear power plants, hazardous waste sites, and fast-food outlets (see [12,4]). In the latter paper the problem is called the *Remote Clique* problem.

Formally, problems of this type can be described as follows: given a graph $G = (V, E)$ with n vertices, and non-negative edge weights $w_{v_1,v_2} = d(v_1, v_2)$. Given $k \in \{2, \ldots, n\}$, find a subset $S \subset V$ with $|S| = k$, such that $w(S) := \sum_{(v_i,v_j)\in E(S)} d(v_i, v_j)$ is maximized. (Here, $E(S)$ denotes the edge set of the subgraph of G induced by the vertex set S.)

From a graph theoretic point of view, this problem has been called a *heaviest subgraph problem*. Being a weighted version of a generalization of the problem of deciding the existence of a k-*clique*, i.e., a complete subgraph with k vertices,

* Research partially supported by NSERC.

K. Jansen and S. Khuller (Eds.): APPROX 2000, LNCS 1913, pp. 132–141, 2000.

the problem is strongly NP-hard [14]. It should be noted that Håstad [9] showed that the problem CLIQUE of maximizing the cardinality of a set of vertices with a maximum possible number of edges is in general hard to approximate within $n^{1-\varepsilon}$. For the heaviest subgraph problem, we want to maximize the number of edges for a set of vertices of given cardinality, so Håstad's result does not imply an immediate performance bound.

Related Work

Over recent years, there have been a number of approximation algorithms for various subproblems of this type. Feige and Seltser [7] have studied the graph problem (i.e., edge weights are 0 or 1) and showed how to find in time $n^{O((1+\log\frac{n}{k})/\varepsilon)}$ a k-set $S \subset V$ with $w(S) \geq (1-\varepsilon)\binom{k}{2}$, provided that a k-clique exists. They also gave evidence that for $k \simeq n^{1/3}$, semidefinite programming fails to distinguish between graphs that have a k-clique, and graphs with densest k-subgraphs having average degree less than $\log n$.

Kortsarz and Peleg [10] describe a polynomial algorithm with performance guarantee $O(n^{0.3885})$ for the general case where edge weights do not have to obey the triangle inequality. A newer algorithm by Feige, Kortsarz, and Peleg [6] gives an approximation ratio of $O(n^{1/3}\log n)$. For the case where $k = \Omega(n)$, Asahiro, Iwama, Tamaki, and Tokuyama [3] give a greedy constant factor approximation, while Srivastav and Wolf [13] use semidefinite programming for improved performance bounds. For the case of dense graphs (i.e., $|E| = \Omega(n^2)$) and $k = \Omega(n)$, Arora, Karger, and Karpinski [1] give a polynomial time approximation scheme. On the other hand, Asahiro, Hassin, and Iwama [2] show that deciding the existence of a "slightly dense" subgraph, i.e., an induced subgraph on k vertices that has at least $\Omega(k^{1+\varepsilon})$ edges, is NP-complete. They also showed it is NP-complete to decide whether a graph with e edges has an induced subgraph on k vertices that has $\frac{ek^2}{n^2}(1+O(n^{\varepsilon-1}))$ edges; the latter is only slightly larger than $\frac{ek^2}{n^2}(1-\frac{n-k}{nk-k})$, which is the the average number of edges in a subgraph with k vertices.

For the case where edge weights fulfill the triangle inequality, Ravi, Rosenkrantz, and Tayi [12] give a heuristic with time complexity $O(n^2)$ and prove that it guarantees a performance bound of 4. (See Tamir [15] with reference to this paper.) Hassin, Rubinstein, and Tamir [8] give a different heuristic with time complexity $O(n^2 + k^2 \log k)$ with performance bound 2. On a related note, see Chandra and Halldórsson [4], who study a number of different remoteness measures for the subset k, including total edge weight $w(S)$. If the graph from which a subset of size k is to be selected is a tree, Tamir [14] shows that an optimal weight subset can be determined in $O(nk)$ time.

In many important cases there is even more known about the set of edge weights than just the validity of triangle inequality. This is the case when the vertex set V corresponds to a point set P in geometric space, and distances between vertices are induced by geometric distances between points. Given the practical motivation for considering the problem, it is quite natural to consider geometric instances of this type. In fact, it was shown by Ravi, Rosenkrantz, and Tayi in [12] that for the case of Euclidean distances in two-dimensional space, it is possible to achieve performance bounds that are arbitrarily close to $\pi/2 \approx 1.57$. For other metrics, however, the best performance guarantee is the factor 2 by [8].

An important application of our problem is data sampling and clustering, where points are to be selected from a large more-dimensional set. Different metric dimensions of a data point describe different metric properties of a corresponding item. Since these properties are not geometrically related, distances are typically not evaluated by Euclidean distances. Instead, some weighted L_1 metric is used. (See Erkut [5].) For data sampling, a set of points is to be selected that has high average distance. For clustering, a given set of points is to be subdivided into k clusters, such that points from the same cluster are close together, while points from different clusters are far apart. If we do the clustering by choosing k center points, and assigning any point to its nearest cluster center, we have to consider the same problem of finding a set of center points with large average distance, which is equivalent to finding a k-clique with maximum total edge weight.

Main Results

In this paper, we consider point sets P in d-dimensional space, where d is some constant. For the most part, distances are measured according to the rectilinear "Manhattan" norm L_1.

Our results include the following:

- A linear time ($O(n)$) algorithm to solve the problem to optimality in case where k is some fixed constant. This is in contrast to the case of Euclidean distances, where there is a well-known lower bound of $\Omega(n \log n)$ in the computation tree model for determining the diameter of a planar point set, i.e., the special case $d = 2$ and $k = 2$ (see [11]).
- A polynomial time approximation scheme for the case where k is not fixed. This method can be applied for arbitrary fixed dimension d. For the case of Euclidean distances in two-dimensional space, it implies a performance bound of $\sqrt{2} + \varepsilon$, for any given $\varepsilon > 0$.

2 Preliminaries

For the most part of this paper, all points are assumed to be points in the plane. Higher-dimensional spaces will be discussed in the end. Distances are measured using the L_1 norm, unless noted otherwise. The x- and y-coordinates of a point p are denoted by x_p and y_p. If p and q are two points in the plane, then the distance between p and q is $d(p, q) = |x_p - x_q| + |y_p - y_q|$. We say that q is above p in direction (a, b) if the angle between the vector in direction (a, b) and the vector $q - p$ is less than $\pi/2$, i.e., if the inner product $\langle q - p, (a, b)\rangle$ of the vector $q - p$ and the vector (a, b) is positive. We say that a point p is maximal in direction (a, b) with respect to a set of points P if no point in P is above p in direction (a, b). For example, if p is an element of a set of points P and p has a maximal y-coordinate, then p is maximal in direction (0,1) with respect to P, and a point p with minimal x-coordinate is maximal in direction (-1,0) with respect to P. If the set P is clear from the context, we simply state that p is maximal in direction (a, b).

The weight of a set of points P is the sum of the distances between all pairs of points in this set, and is denoted by $w(P)$. Similarly, $w(P, Q)$ denotes the total sum of distances between two sets P and Q. For L_1 distances, $w_x(P)$ and $w_x(P, Q)$ denote the sum of x-distances within P, or between P and Q.

3 Cliques of Fixed Size

Let $S = \{s_0, s_1, \ldots, s_{k-1}\}$ be a maximal weight subset of P, where k is a fixed integer greater than 1. We will label the x- and y-coordinates of a point $s \in S$ by some (x_a, y_b) with $0 \leq a < k$ and $0 \leq b < k$ such that $x_0 \leq x_1 \leq \ldots \leq x_{k-1}$ and $y_0 \leq y_1 \leq \ldots \leq y_{k-1}$. So

$$w(S) \;=\; \sum_{0 \leq i < j < k} (x_j - x_i) + \sum_{0 \leq i < j < k} (y_j - y_i).$$

Now we can use local optimality to reduce the family of subsets that we need to consider:

Lemma 1. *There is a maximal weight subset S' of P of cardinality k, such that each point in S' is maximal in direction $(2i + 1 - k, 2j + 1 - k)$ with respect to $P - S'$ for some values of i and j with $0 \leq i, j < k$.*

Proof. Consider an optimal subset $S \subset P$ of cardinality k. Let s_i be a point in S such that there are $k - i - 1$ points $s_\ell = (x_\ell, y_\ell) \in S \setminus \{s_i\}$ with $x_\ell > x_i$ (i.e., "strictly to the right" of p), and i points $s_\ell = (x_\ell, y_\ell) \in S \setminus \{s_i\}$ with $x_\ell \leq x_i$ (i.e., "to the left" of p). Similarly, assume that there are $k-j-1$ points in $S \setminus \{s_i\}$ "strictly above" s_i, and j points in $S \setminus \{s_i\}$ "below" s_i. Now consider replacing s_i by a point $s'_i = s_i + (h_x, h_y)$. This adds h_x to the x-distances between s_i and the i points to the left of s_i, while subtracting no more than h_x from the x-distances between s_i and the $k - i - 1$ points to the right of s_i. In the balance, replacing s_i by s'_i adds at least $(2i - k + 1)h_x$ to the total x-distance. Similarly, we get an addition to the total y-distance of at least $(2j - k + 1)h_y$. If s'_i is above s_i in direction $(2i - k + 1, 2j - k + 1)$, the overall change is positive, and the claim follows.

□

Theorem 1. *Given a constant value for k, a maximum weight subset S of a set of n points P, such that S has cardinality k, can be found in linear time.*

Proof. Consider all directions of the form $(2i+1-k, 2j+1-k)$ with $0 \leq i, j < k$. For each direction (a, b), find $S_k(a, b)$, a set of k points that are maximal in direction (a, b) with respect to $P - S_k(a, b)$. Compute the set $\cup S_k(a, b)$ and try all possible subsets of size k of this set until a subset of maximal weight is found.

Correctness follows from the fact that Lemma 1 implies that $S \subset \cup S_k(a, b)$. Since k is a constant, each set $S_k(a, b)$ can be found in linear time. Since the cardinality of $\cup S_k(a, b)$ is less than or equal to k^3, the result follows.

□

Note that in the above estimate, we did not try to squeeze the constants in the $O(n)$ running time. A closer look shows that for $k = 2$, not more than 2 subsets of P need to be evaluated for possible optimality, for $k = 3$, 8 subsets are sufficient.

4 Cliques of Variable Size

In this section we consider the scenario where k is not fixed, i.e., part of the input. We show that there is a polynomial time approximation scheme (PTAS), i.e., for any fixed positive ε, there is a polynomial approximation algorithm that finds a solution that is within $(1 + \varepsilon)$ of the optimum.

The basic idea is to use a suitable subset of m coordinates that subdivide an optimal solution into subsets of equal cardinality. More precisely, we find (by enumeration) a subdivision of an optimal solution into $m \times m$ rectangular cells C_{ij}, each of which must contain a specific number k_{ij} of selected points. From each cell C_{ij}, the points are selected in a way that guarantees that the total distance to all other cells except for the $m - 1$ cells in the same "horizontal" strip or the $m - 1$ cells in the same "vertical" strip is maximized. As it turns out, this can be done in a way that the total neglected distance within the strips is bounded by a fraction of $(5m - 9)/(2(m - 1)(m - 2))$ of the weight of an optimal solution, yielding the desired approximation property. See Figure 1 for the overall picture.

For ease of presentation we assume that k is a multiple of m and $m > 2$. Approximation algorithms for other values of k can be constructed in a similar fashion and will be treated in the full paper. Consider an optimal solution of k points, denoted by OPT. Furthermore consider a division of the plane by a set of $m + 1$ x-coordinates $\xi_0 \leq \ldots \leq \xi_1 \leq \xi_m$. Let $X_i := \{p = (x, y) \in \Re^2 \mid \xi_i \leq x \leq \xi_{i+1}, 0 \leq i < m\}$ be the vertical strip between coordinates ξ_i and ξ_{i+1}. By enumeration of possible choices of $\xi_0, \ldots, \xi_m$ we may assume that the ξ_i have the property that, for an optimal solution, from each of the m strips X_i precisely k/m points of P are chosen.

In a similar manner, suppose we know $m+1$ y-coordinates $\eta_0 \leq \eta_1 \leq \ldots \leq \eta_m$ such that from each horizontal strip $Y_i := \{p = (x, y) \in \Re^2 \mid \eta_i \leq y \leq \eta_{i+1}, 0 \leq i < m\}$ a subset of k/m points are chosen for an optimal solution.

Let $C_{ij} := X_i \cap Y_j$, and let k_{ij} be the number of points in OPT that are chosen from C_{ij}. Since $\sum_{0 \leq i < m} k_{ij} = \sum_{0 \leq j < m} k_{ij} = k/m$, we may assume by enumeration over the possible partitions of k/m into m pieces that we know all the numbers k_{ij}.

Finally, define the vector $\nabla_{ij} := ((2i+1-m)k/m, (2j+1-m)k/m)$. Now our approximation algorithm is as follows: from each cell C_{ij}, choose the k_{ij} points

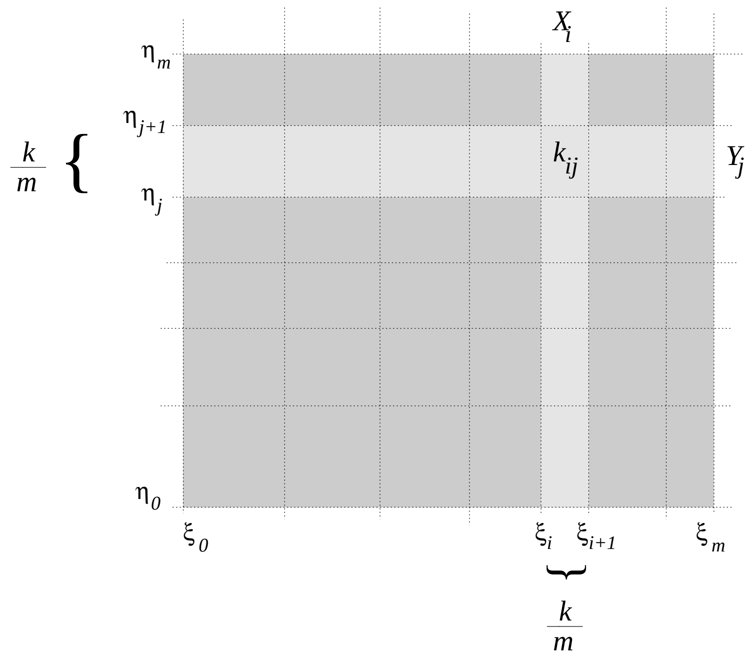

Fig. 1. Subdividing the plane into cells.

that are maximal in direction ∇_{ij}. (Overlap between the selections from different cells is avoided by proceeding in lexicographic order of cells, and choosing the k_{ij} points among the candidates that are still unselected.) Let *HEU* be the point set selected in this way.

It is clear that *HEU* can be computed in polynomial time. We will proceed by a series of lemmas to determine how well $w(HEU)$ approximates $w(OPT)$. In the following, we consider the distances involving points from a particular cell C_{ij}. Let HEU_{ij} be the set of k_{ij} points that are selected from C_{ij} by the heuristic, and let OPT_{ij} be a set of k_{ij} points of an optimal solution that are attributed to C_{ij}. Let $S_{ij} = OPT_{ij} \cap HEU_{ij}$. Furthermore we define $\overline{S}_{ij} = HEU_{ij} \setminus OPT_{ij}$, and $\tilde{S}_{ij} = OPT_{ij} \setminus HEU_{ij}$. Let $HEU_{i\bullet}$, $OPT_{i\bullet}$, $HEU_{\bullet j}$ and $OPT_{\bullet j}$ be the set of k/m points selected from X_i and Y_j by the heuristic and an optimal algorithm respectively. Finally $\overline{HEU}_{i\bullet} := HEU \setminus HEU_{i\bullet}$, $\overline{HEU}_{\bullet j} := HEU \setminus HEU_{\bullet j}$, $\overline{OPT}_{i\bullet} := OPT \setminus OPT_{i\bullet}$ and $\overline{OPT}_{\bullet j} := OPT \setminus OPT_{\bullet j}$.

Lemma 2.

$$\begin{aligned} &w_x(HEU_{ij}, \overline{HEU}_{i\bullet}) + w_y(HEU_{ij}, \overline{HEU}_{\bullet j}) \\ \geq\, &w_x(OPT_{ij}, \overline{OPT}_{i\bullet}) + w_y(OPT_{ij}, \overline{OPT}_{\bullet j}). \end{aligned}$$

Proof. Consider a point $p \in \tilde{S}_{ij}$. Thus, there is a point $p' \in \overline{S}_{ij}$ that was chosen by the heuristic instead of p. Now we can argue like in Lemma 1: Let $h =$

$(h_x, h_y) = p' - p$. When replacing p in OPT by p', we increase the x-distance to the ik/m points "left" of C_{ij} by h_x, while decreasing the x-distance to $(m - i - 1)k/m$ points "right" of C_{ij} by h_x. In the balance, this yields a change of $((2i+1-m)k/m)h_x$. Similarly, we get a change of $((2j+1-m)k/m)h_y$ for the y-coordinates. By definition, we have assumed that the inner product $\langle h, \nabla_{ij}\rangle \geq 0$, so the overall change of distances is nonnegative.

Performing these replacements for all points in $OPT \setminus HEU$, we can transform OPT to HEU, while increasing the sum $w_x(OPT_{ij}, \overline{OPT}_{i\bullet}) + w_y(OPT_{ij}, \overline{OPT}_{\bullet j})$ to the sum $w_x(HEU_{ij}, \overline{HEU}_{i\bullet}) + w_y(HEU_{ij}, \overline{HEU}_{\bullet j})$. □

In the following three lemmas we show that the total difference between the weight of an optimal solution $w(OPT)$ and total value of all the right hand sides (when summed over i) of the inequality in Lemma 2 is a small fraction of $w(OPT)$.

Lemma 3.

$$\sum_{0<i<m-1} w_x(OPT_{i\bullet}) \;\leq\; \frac{w_x(OPT)}{2(m-2)}.$$

Proof. Let $\delta_i = \xi_{i+1} - \xi_i$. Since $i(m-i-1) \geq m-2$ for $0 < i < m-1$, we have for $0 < i < m-1$

$$w_x(OPT_{i\bullet}) \leq \frac{k^2}{2m^2}\delta_i \leq \frac{ik}{m}\frac{(m-i-1)k}{m}\delta_i \;\frac{1}{2(m-2)}.$$

Since OPT has ik/m and $(m-i-1)k/m$ points to the left of ξ_i and right of ξ_{i+1} respectively, we have

$$w_x(OPT) \geq \sum_{0<i<m-1} \frac{ik}{m}\frac{(m-i-1)k}{m}\delta_i$$

so

$$\sum_{0<i<m-1} w_x(OPT_{i\bullet}) \leq \frac{1}{2(m-2)} w_x(OPT).$$

□

Lemma 4. *For $i = 0$ and $i = m-1$ we have*

$$w_x(OPT_{i\bullet}) \;\leq\; \frac{w_x(OPT)}{m-1}$$

Proof. Without loss of generality assume $i = 0$. Let $x_0, x_1, \cdots, x_{(k/m)-1}$ be the x-coordinates of the points $p_0, p_1, \ldots, p_{(k/m)-1}$ in $OPT_{0\bullet}$. So

$$w_x(OPT_{0\bullet}) \;=\; (\frac{k}{m} - 1)(x_{\frac{k}{m}-1} - x_0) + (\frac{k}{m} - 3)(x_{\frac{k}{m}-2} - x_1) + \;\ldots$$

$$\leq (\frac{k}{m} - 1)(\xi_1 - x_0) + (\frac{k}{m} - 3)(\xi_1 - x_1) + \;\ldots$$

$$\leq \frac{k}{m}(\xi_1 - x_0) + \frac{k}{m}(\xi_1 - x_1) + \;\ldots$$

Since $\xi_1 - x_j \leq x - x_j$ where $0 \leq j < k/m$ and x is the x-coordinate of any point in $\overline{OPT_{0\bullet}}$ and since there are $(m-1)k/m$ points in $\overline{OPT_{0\bullet}}$, we have

$$\xi_1 - x_j < \frac{m}{(m-1)k} w_x(p_j, \overline{OPT_{0\bullet}}),$$

so

$$\begin{aligned} w_x(OPT_{0\bullet}) &\leq \frac{k}{m}\frac{m}{(m-1)k} \sum_{0\leq i<\frac{k}{2m}} w_x(p_i, \overline{OPT_{0\bullet}}) \\ &\leq \frac{1}{m-1} \sum_{0\leq i<\frac{k}{m}} w_x(p_i, \overline{OPT_{0\bullet}}) \\ &= \frac{1}{m-1} w_x(OPT_{0\bullet}, \overline{OPT_{0\bullet}}) \\ &\leq \frac{1}{m-1} w_x(OPT). \end{aligned}$$

□

This proves the main properties. Now we only have to combine the above estimates to get an overall performance bound:

Lemma 5.

$$\sum_{0\leq i<m} w_x(OPT_{i\bullet}, \overline{OPT_{i\bullet}}) + \sum_{0\leq j<m} w_y(OPT_{\bullet j}, \overline{OPT_{\bullet j}}) \geq (1 - \frac{5m-9}{2(m-1)(m-2)})w(OPT)).$$

Proof. From Lemmas 3 and 4 we derive that

$$\sum_{0\leq i<m} w_x(OPT_{i\bullet}) \;\leq\; \frac{5m-9}{2(m-1)(m-2)} w_x(OPT)$$

and similarly

$$\sum_{0\leq i<m} w_y(OPT_{\bullet j}) \;\leq\; \frac{5m-9}{2(m-1)(m-2)} w_y(OPT).$$

Since

$$\begin{aligned} w(OPT) &= w_x(OPT) + w_y(OPT) \\ &= \sum_{0\leq i<m} w_x(OPT_{i\bullet}, \overline{OPT_{i\bullet}}) + \sum_{0\leq i<m} w_x(OPT_{i\bullet}) \\ &\quad + \sum_{0\leq j<m} w_y(OPT_{\bullet j}, \overline{OPT_{\bullet j}}) + \sum_{0\leq j<m} w_y(OPT_{\bullet j}), \end{aligned}$$

the result follows. □

Putting together Lemma 2 and the error estimate from Lemma 5, the approximation theorem can now be proven.

Theorem 2. *For any fixed m, HEU can be computed in polynomial time, and*

$$w(HEU) \geq (1 - \frac{5m-9}{2(m-1)(m-2)})w(OPT).$$

Proof. The claim about the running time is clear. Using Lemmas 2 and 5 we derive

$$w(HEU) \geq \sum_{0 \leq i < m} w_x(HEU_{i\bullet}, \overline{HEU}_{i\bullet}) + \sum_{0 \leq j < m} w_y(HEU_{\bullet j}, \overline{HEU}_{\bullet j})$$

$$\geq \sum_{0 \leq i < m} w_x(OPT_{i\bullet}, \overline{OPT}_{i\bullet}) + \sum_{0 \leq j < m} w_y(OPT_{\bullet j}, \overline{OPT}_{\bullet j})$$

$$\geq (1 - \frac{5m-9}{2(m-1)(m-2)})w(OPT).$$

□

5 Implications

It is straightforward to modify our above arguments to point sets under L_1 distances in an arbitrary d-dimensional space, with fixed d.

Theorem 3. *Given a constant value for k and d, the maximal weight subset S of a set of n points in d-dimensional space, such that S has cardinality k, can be found in linear time. If d and ε are constants, but k is not fixed, then there is a polynomial time algorithm that finds a subset whose weight is within $(1+\varepsilon)$ of the optimum.*

Furthermore, we can use the approximation scheme from the previous section to get a $\sqrt{2}(1+\varepsilon)$ approximation factor for the case of Euclidean distances in two-dimensional space, for any $\varepsilon > 0$: In polynomial time, find a k-set S_1 such that $L_1(S)$ is within $(1+\varepsilon)$ of an optimal solution OPT_1 with respect to L_1 distances. Let OPT_2 be an optimal solution with respect to L_2 distances. Then

$$L_2(S) \geq \frac{1}{\sqrt{2}} L_1(S) \geq \frac{1}{\sqrt{2}(1+\varepsilon)} L_1(OPT_1) \geq \frac{1}{\sqrt{2}(1+\varepsilon)} L_1(OPT_2)$$

$$\geq \frac{1}{\sqrt{2}(1+\varepsilon)} L_2(OPT_2),$$

and the claim follows.

6 Conclusions

We have presented algorithms for geometric instances of the maximum weighted k-clique problem. Our results give a dramatic improvement over the previous best approximation factor of 2 that was presented in [12] for the case of general metric spaces. This underlines the observation that geometry can help to get better algorithms for problems from combinatorial optimization.

Furthermore, the algorithms in [12] give better performance for Euclidean metric than for Manhattan distances. We correct this anomaly by showing that among problems involving geometric distances, the rectilinear metric may allow better algorithms than the Euclidean metric.

Acknowledgments

We would like to thank Katja Wolf and Magnús Halldórsson for helpful discussions, and Rafi Hassin for several relevant references.

References

1. S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for NP-hard problems. In *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, pages 284–293, 1995.
2. Y. Asahiro, R. Hassin, and K. Iwama. Complexity of finding dense subgraphs. Manuscript, 2000.
3. Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama. Greedily finding a dense graph. In *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1097 of *Lecture Notes in Computer Science*, pages 136–148. Springer–Verlag, 1996.
4. B. Chandra and M. M. Halldórsson. Facility dispersion and remote subgraphs. In *Proceedings of the 5th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1097 of *Lecture Notes in Computer Science*, pages 53–65. Springer–Verlag, 1996.
5. E. Erkut. The discrete p-dispersion problem. *European Journal of Operations Research*, 46:48–60, 1990.
6. U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, to appear.
7. U. Feige and M. Seltser. On the densest k-subgraph problems. Technical Report CS97-16, `http://www.wisdom.weizmann.ac.il`, 1997.
8. R. Hassin, S. Rubinstein, and A. Tamir. Approximation algorithms for maximum dispersion. *Operations Research Letters*, 21:133–137, 1997.
9. J. Håstad. Clique is hard to approximate within $n^{1-\varepsilon}$. In *Proceedings of the 37th IEEE Annual Symposium on Foundations of Computer Science*, pages 627–636, 1996.
10. G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *Proceedings of the 34th IEEE Annual Symposium on Foundations of Computer Science*, pages 692–701, Palo Alto, CA, 1993.
11. F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer-Verlag, New York, NY, 1985.
12. S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. Heuristic and special case algorithms for dispersion problems. *Operations Research*, 42:299–310, 1994.
13. A. Srivastav and K. Wolf. Finding dense subgraphs with semidefinite programming. In K. Jansen and J. Rolim, editors, *Approximation Algorithms for Combinatorial Optimization (APPROX '98)*, volume 1444 of *Lecture Notes in Computer Science*, pages 181–191, Aalborg, Denmark, 1998. Springer–Verlag.
14. A. Tamir. Obnoxious facility location in graphs. *SIAM Journal on Discrete Mathematics*, 4:550–567, 1991.
15. A. Tamir. Comments on the paper: 'Heuristic and special case algorithms for dispersion problems' by , S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. *Operations Research*, 46:157–158, 1998.