Facility location problems in the plane based on reverse nearest neighbor queries

S. Cabello* J. M. Díaz-Báñez† S. Langerman‡ C. Seara§
I. Ventura¶

15th October 2007

Abstract

The Reverse Nearest Neighbor (RNN) problem is to find all points in a given data set whose nearest neighbor is a given query point. Given a set of blue points and a set of red points, the bichromatic version of the RNN problem, for a query blue point, is to find all the red points whose blue nearest neighbour is the given query point. In this paper, we introduce and investigate new optimization problems in the plane according to the Bichromatic Reverse Nearest Neighbor (BRNN) query. We give efficient algorithms to compute a new blue point such that: (1) the number of associated red points is maximum (MAXCOV criterion); (2) the maximum distance to the associated red points is minimum (MINMAX criterion); (3) the minimum distance to the associated red points is maximum (MAXMIN criterion). These problems arise in competitive location where competing facilities are established. Our solutions use techniques from computational geometry, such as the concept of depth in an arrangement of disks or upper envelope of surface patches in three dimensions.

Keywords: Reverse Nearest Neighbor; Competitive Location; Computational Geometry.

*Department of Mathematics, Institute for Mathematics, Physics and Mechanics, Slovenia. Partially supported by the European Community Sixth Framework Programme under a Marie Curie Intra-European Fellowship, and by the Slovenian Research Agency, project J1-7218. sergio.cabello@imfm.uni-lj.si

†Departamento de Matemática Aplicada II, Universidad de Sevilla. Partially supported by project BFM2003-04062. dbanez@us.es

‡Chercheur qualifié du FNRS, Department d'Informatique, Université Libre de Bruxelles. stefan.langerman@ulb.ac.be

§Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya. Partially supported by projects MCYT-FEDER-BFM2003-00368, Gen-Cat-2005SGR00692, and MCYT HU2002-0010. carlos.seara@upc.edu

¶Departamento de Matemáticas, Universidad de Huelva. Partially supported by project BFM2003-04062. iventura@us.es

1 Introduction

In the *Nearest Neighbor problem* (NN), the objects in the database that are nearer to a given query object than any other objects in the database have to be found. In the conceptually inverse problem, *Reverse Nearest Neighbor problem* (RNN), objects that have the query object as their nearest neighbor have to be found. Reverse Nearest Neighbors queries have emerged as an important class of queries for spatial and other types of databases. The concept has been introduced by Korn et al. [19, 20], where a large number of applications in marketing and decision support system are given. See [30] for a recent survey on the current state-of-art and open geometric problems in another application area.

The RNN problem itself has several variants, namely, the monochromatic, bichromatic, static or dynamic versions. In the monochromatic case, all points have the same color. In the bichromatic case, the point set consists of both red and blue points and the problem is to find the points of the other color for which a query point is a bichromatic nearest neighbor. In the static version of the problem, the distances between the points in the set remain unchanged whereas in the dynamic problem they may change. Some related work includes [6, 22, 23, 27, 29]. High-dimensional instances of RNN and BRNN have been hardly considered in the past, in contrast with the NN problem, and it is striking to see how thin the literature on (B)RNN is, as compared to the literature on NN. In particular, little is known about the approximability of (B)RNN in Euclidean spaces of arbitrary dimensions. This shows that the planar instances of (B)RNN are still relevant at the present time.

This paper considers the planar static bichromatic variant in which the data points are of two categories. In particular, we define *RNN facility location problems* in a two dimensional space, in which some points are designated as facilities and others as customers. In this setting, a *reverse nearest neighbor query* asks for the set of customers affected by the opening of a new facility at some point, assuming all customers choose their nearest facility (Figure 1). We point out here that we pick the name "reverse" from the data mining community and the concept is different than the "inverse" or "reverse" as used sometimes in the operational research field, where the goal is to modified the underlying space to improve the efficiency [32].

We study optimization problems that arise when considering various optimization criteria: maximizing the number of potential customers for the new facility (MAXCOV criterion); minimizing the maximum distance to the associated clients (MINMAX criterion); and maximizing the minimum distance to the associated clients (MAXMIN criterion). The MAXCOV and MINMAX criteria deal with the location of an *attractive* facility (bars, discos, hospitals, schools, supermarkets, fixed wireless base stations, etc), while the MAXMIN criterion seeks the best location for a new *obnoxious* facility (rubbish dumps, chemical plants, etc). Notice that these problems can be interpreted as the location of a new facility in a competitive environment. Competitive facility location address the placement of sites by competing market players. Typically, the expected income the new facility will generate depends on the market share it captures. Competitive location models have been studied in several disciplines such as geography, economics, marketing and operations research. Comprehensive surveys of competitive facility location models can be found in [14, 15, 24, 31]. A continuous analogue to the MAXCOV problem was considered in [8, 10], where the problem of placing a new facility in a location that maximizes the area of the corresponding Voronoi region is considered. Observe that the MAXCOV criterion can also be seen as a greedy step in a discrete version of the Voronoi game [2].

Finally, as already pointed out above, the applications of the problems under considerations are also related to various fields that lie beyond the scope of facility location problems, and, that are also useful for advanced database applications.

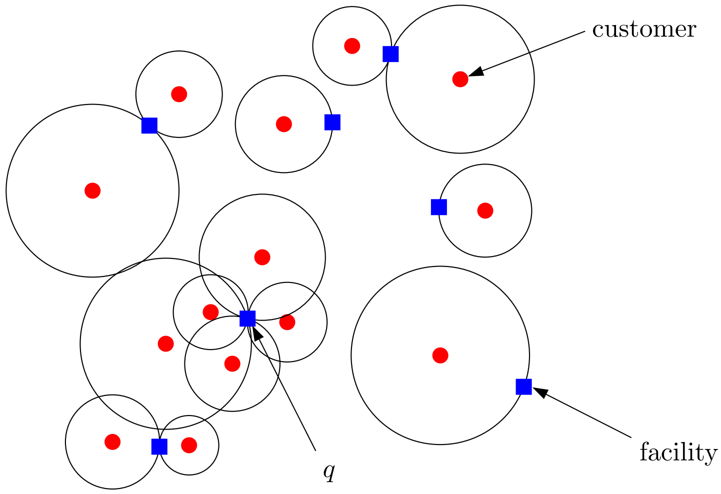

Figure 1: The bichromatic RNN query. $BRNN(q)$ has five points.

An outline of the paper is as follows. In Section 2 we state the optimization problems. In Section 3 we propose exact and approximate algorithms for the MAXCOV problem and we prove its 3SUM hardness. An $O(n^{2+\epsilon})$-time algorithm for the MINMAX and the MAXMIN problems is described in Section 4. In Section 5 we also consider several variants of the problems which include the combination of criteria, the use of the L_1 and L_∞-metrics and the reverse farthest neighbor version. Finally, concluding remarks of the paper are put forward in Section 6.

2 Problem statement

In the sequel, unless otherwise specified, we use the L_2 metric and $d(p, q)$ denotes the Euclidean distance between the points p and q. Let $S = \{p_1, \ldots, p_N\}$ be a set of points in the plane. Given a point b in the plane, the *reverse nearest neighbor set* of b is defined as

$$\text{RNN}(b) = \{p_i \in S : d(p_i, b) \leq d(p_i, p_j), \forall p_j \in S \setminus \{p_i\}\}.$$

For the bichromatic case, assume we have a nonempty set $R = \{r_1, \ldots, r_n\}$ of n red points (clients) and a nonempty set $B = \{b_1, \ldots, b_m\}$ of m blue points (facilities), with $n \geq m \geq 2$. Given a new query blue point $b \notin B$, the *bichromatic reverse nearest neighbor set* is defined as

$$\text{BRNN}(b) = \{r_i \in R : d(r_i, b) \leq d(r_i, b_j), \forall b_j \in B\}.$$

Notice that the monochromatic and bichromatic settings differ on, for example, the size of the output of the queries, as it is stated in the following result.

Lemma 1. [28] *For any query point, the set* RNN(b) *has at most* 6 *points, but the size of* BRNN(b) *may be arbitrarily large.*

Note that for any blue point $b \notin B$ we have $0 \leq |\,\text{BRNN}(b)| \leq n$. Notice also that if $r_i \in \text{BRNN}(b)$, then (by definition) the open disk centered at r_i and radius $d(r_i, b)$ is empty of blue points. We formalize the optimization problems as follows.

The MAXCOV **problem.** *Given a bichromatic point set $S = R \cup B$, compute the value*

$$\text{MAXCOV}(S) = \max\{|\,\text{BRNN}(b)| : b \in \mathbb{R}^2 \setminus B\},$$

that is, compute the maximum number of points that BRNN(b) *may have for a new point $b \notin B$, and find a witness placement* b_0 *such that* $|\,\text{BRNN}(b_0)| = \text{MAXCOV}(S)$.

In the MAXCOV problem, we may also be interested in computing the locus $\mathcal{L}_S$ of all points b satisfying $|\,\text{BRNN}(b)| = \text{MAXCOV}(S)$. More generally, for any positive integer k, we may be interested in the level set $L(k) = \{b \in \mathbb{R}^2 : |\,\text{BRNN}(b)| \geq k\}$. Observe that $L(\text{MAXCOV}(S)) = \mathcal{L}_S$, and $L(1) = \{b \in \mathbb{R}^2 : \text{BRNN}(b) \neq \emptyset\}$.

The MINMAX **problem.** *Given a bichromatic point set $S = R \cup B$ and a region $X \subseteq L(1)$, compute the value*

$$\text{MINMAX}(S) = \min_{b \in X} \max\{d(b, x) : x \in \text{BRNN}(b)\},$$

and find a witness placement $b_0 \in X$ *such that* $\max\{d(b_0, x) : x \in \text{BRNN}(b_0)\} = \text{MINMAX}(S)$.

The MAXMIN **problem.** *Given a bichromatic point set $S = R \cup B$ and a region $X \subseteq L(1)$, compute the value*

$$\text{MAXMIN}(S) = \max_{b \in X} \min\{d(b, x) : x \in \text{BRNN}(b)\},$$

and find a witness placement $b_0 \in X$ *such that* $\min\{d(b_0, x) : x \in \text{BRNN}(b_0)\} = \text{MAXMIN}(S)$.

Notice that for both the MINMAX and MAXMIN problems we add the additional constraint that the new point b has to be placed in a given region X with $X \subseteq L(1)$, as otherwise we could always place b such that BRNN$(b) = \emptyset$. We assume that X is a region bounded by O(n) pieces, each with constant description complexity. The region X has to be bounded for the MAXMIN problem to be well-defined, and this condition is guaranteed by the fact that $X \subseteq L(1)$, which is always bounded. Typically, we would consider X to be a level set $L(k)$ for some value k. Although for some values k, the level set $L(k)$ may have quadratic complexity in n, we will see that we can handle this type of sets within the same asymptotic bounds.

Note that the MAXCOV and MAXMIN/MINMAX criteria are of completely different nature: while in the MAXCOV criterion our goal is to maximize the number of points in a set, which is a discrete measure, in the MAXMIN/MINMAX criteria we optimize a distance, which a is continuous measure. This different nature is reflected in the solutions that we present.

3 The MAXCOV problem

In this section we provide exact and approximate algorithms for the MAXCOV problem, as well as a hardness result for the exact problem.

3.1 Exact solution

For every red point $r_i \in R$, we denote by $b(r_i)$ the nearest blue point. Let R_i be the *red disk* with radius $d(r_i, b(r_i))$ centered at point r_i. The set of n disks $\{R_1, \ldots, R_n\}$ can be computed in $O((n+m)\log m) = O(n \log m)$ time as follows: compute the Voronoi diagram of B and preprocess it for point location; after $O(m \log m)$ time, a point location query can be replied in $O(\log m)$ time [5]. By locating each $r_i \in R$ in the Voronoi diagram, we get the points $b(r_1), \ldots, b(r_n)$ in $O(n \log m)$, which is sufficient information to construct the set of disks $\{R_1, \ldots, R_n\}$.

Let $\mathcal{A}$ be the arrangement produced by the set of n red disks $\{R_1, \ldots, R_n\}$. The idea of the algorithm is to associate a label l_c to each cell c of $\mathcal{A}$ with the number of disks from $\{R_1, \ldots, R_n\}$ that contain it, and then look for the cells in $\mathcal{A}$ with maximum label. Indeed, if a cell c has label k, it means that a blue point b inside this cell c is contained in exactly k red disks, which means that the point b is the closest point of the k red points corresponding to the red disks. Observe that if we do not assume general position, the cell with greatest label may be a vertex of $\mathcal{A}$, such as the vertex b in Figure 2.

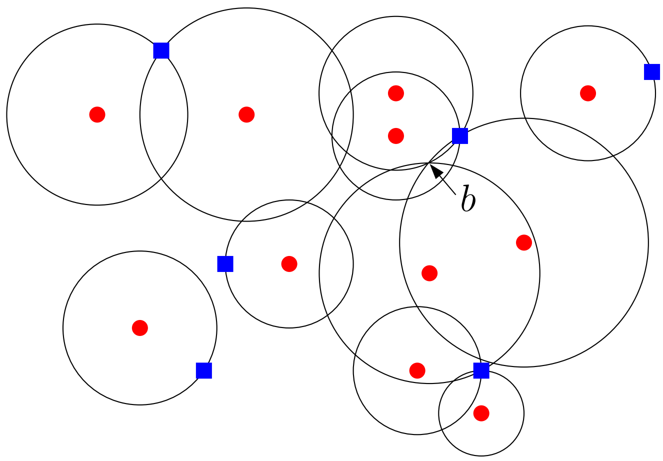

Figure 2: Arrangement of red circles $R_1, \ldots, R_n$.

The arrangement $\mathcal{A}$, together with the labels l_c for each cell $c \in \mathcal{A}$ can be constructed in $O(n^2 \log n)$ time using a standard sweep-line algorithm as Bentley and Ottmann [7]. Computing the arrangement determined by a set of curve segments in the plane is a classical problem in computational geometry. A slightly faster construction of the arrangement $\mathcal{A}$ with $O(n^2)$ expected running time is proposed in [12, 26]. More recently, a deterministic algorithm that use a divide-and-conquer approach to achieve an optimal running time $O(n^2)$ has been described in [3].

As we are dealing with the planar case, the computation of an arrangement of circles is of acceptable complexity. Resorting to an arrangement of circles is reminiscent of the approach of [19], which reduces the RNN problem to point location among balls.

Once we have computed the arrangement $\mathcal{A}$ induced by the disks $\{R_1, \ldots, R_n\}$, we can construct the dual graph G of the arrangement containing a node for each cell $c \in \mathcal{A}$ and an edge between two cells whenever their closures intersect. If two faces $c, c' \in \mathcal{A}$ are adjacent in G, it is easy to compute the label $l_{c'}$ from the label l_c. Therefore, making a traversal in the dual graph G, we can compute the labels l_c for all faces $c \in \mathcal{A}$. With this information, it is possible to compute l_c for all the edges and vertices $c \in \mathcal{A}$. Special care has to be paid if the arrangement is degenerate, that is, if some disks in $\{R_1, \ldots, R_n\}$ are tangent; details are standard and omitted. After computing l_c for all cells $c \in \mathcal{A}$, we can find the value $\text{MAXCOV}(S)$ using that $\text{MAXCOV}(S) = \max\{l_c \mid c \in \mathcal{A}\}$ and report the locus $\mathcal{L}_S$ of all optimal placement using that $\mathcal{L}_S = \bigcup_{\{c \in \mathcal{A} : l_c = \text{MAXCOV}(S)\}} c$. We summarize.

Theorem 1. *The value* $\text{MAXCOV}(S)$ *and the set of all optimal placements* $\mathcal{L}_S$ *can be computed in* $O(n^2)$ *worst-case running time.*

Notice that we can also construct any of the level sets $L(k)$ in the same running time. However, observe that the level set $L(1)$ is exactly the union of the n disks $R_1, \ldots, R_n$, which can be described in linear space and constructed in near-linear time [18]. Once we have a level set $L(k)$ under the MAXCOV criterion, we may be interested in one that optimizes the MAXMIN or MINMAX criteria. We will show how to deal with this in Subsection 5.1.

3.2 Approximation algorithm

We have given above a quadratic running-time algorithm for solving the MAXCOV problem, and we show below that solving the MAXCOV problem is actually 3SUM hard [16], implying that a sub-quadratic algorithm is unlikely to exist. In some applications, it may be that a quadratic time algorithm is not affordable and we would be satisfied with an approximation algorithm that places a new facility that is suboptimal, that is, the number of clients it acquires may be smaller than that of the optimal placement, but the running time of the algorithm is close to linear.

We have seen above that computing the value $\text{MAXCOV}(S)$ is equivalent to finding the *maximum depth* in the arrangement of disks $\mathcal{A}$, that is, it can be reduced to finding a point in the plane having the largest number of covering disks. It also follows from the discussion above that if we find a point b whose depth in $\mathcal{A}$ is d, then it follows that $|\text{BRNN}(b)| = d$, and so $\text{MAXCOV}(S) \geq d$. A probabilistic algorithm to find a point that $(1-\varepsilon)$-approximates the maximum depth in an arrangement of n disks is given by Aronov and Har-Peled [4], and it readily leads to the following result.

Theorem 2. *Given a parameter* $\varepsilon > 0$*, we can find in* $O(n\varepsilon^{-2} \log n)$ *expected time a placement that, with high probability, is a* $(1-\varepsilon)$*-approximation to the value* $\text{MAXCOV}(S)$*.*

Proof. In Subsection 3.1 we showed how to compute the set of n red disks $\{R_1, \ldots, R_n\}$ in $O(n \log m)$ time. Then we use the probabilistic algorithm by Aronov and Har-Peled [4] to approximate the maximum depth in a family of pseudo-disks. □

3.3 Complexity of MAXCOV

The hardness of the problem changes substantially from $m = 1$ to $m = 2$. We show below that for $m = 2$, the problem is 3SUM hard [16], and therefore the problem is at least as hard as many other problems for which no sub-quadratic algorithm is known. On the other hand, for $m = 1$, the problem can be solved in $O(n \log n)$ time, and this is asymptotically optimal in the algebraic decision tree model of computation (see Theorem 4).

Theorem 3. *For $m \geq 2$, computing the value* MAXCOV(S) *is 3SUM hard.*

Proof. The present proof is similar to the one used in [4] for showing the 3SUM hardness of computing the maximum depth in an arrangement of disks. In this paper, the authors used a well-known 3SUM hard problem in the reduction: given a set of lines in the plane with integer coefficients, decide whether any three of the lines have a point in common [16]. We show how to reduce this problem to the problem of computing the value MAXCOV(S). In contrast to that problem, where the input is a collection of disks, we have to reduce the problem to an instance of MAXCOV, whose input is a set of red and blue points. Since not all collections of disks can arise from a $MAXCOV$ problem, and furthermore we want a set of 2 blue points, the original reduction does not apply directly.

Given a set L of n lines with integer coefficients and distinct slopes. See Figure 3 for the following construction. We first find an axis-parallel rectangle Q enclosing all the vertices of the arrangement of lines $\mathcal{A}(L)$. A rectangle Q can be computed in $O(n \log n)$ time noting that the leftmost, rightmost, topmost, and bottommost intersection points are defined by lines with (circularly) consecutive slopes.

Let d be the diameter of Q and q the center of Q. Because the coefficients of the lines are integers, we can compute in linear time a value Δ such that all lines not incident to a vertex of $\mathcal{A}(L)$ are at a distance at least Δ from that vertex.

Let us assume that q lies at $(0, 0)$, and consider the points $b^+ = (0, \beta), b^- = (0, -\beta)$ for some value β to be fixed shortly. We then represent each line $\ell \in L$ by using two red points according to the following construction: let p_ℓ and p'_ℓ be the intersection points between ℓ and the boundary of Q, let D^+_ℓ and D^-_ℓ be the respective disks with boundary through b^+, p_ℓ, p'_ℓ and b^-, p_ℓ, p'_ℓ, and let r^+_ℓ and r^-_ℓ be the respective centers of D^+_ℓ and D^-_ℓ. We can assume that the radii of the disks are large enough, as compared to the dimensions of Q in order to make sure that the bounding circles of these two disks intersect the boundary of Q in only two points, namely p_ℓ and p'_ℓ. Let $B = \{b^+, b^-\}$, let R be the set of $2n$ red points $\{r^+_\ell, r^-_\ell \mid \ell \in L\}$, and let $S = R \cup B$.

For each line $\ell \in L$, the point r^+_ℓ is above the x-axis, while r^-_ℓ is below the x-axis. Therefore, b^+ is the blue point closest to r^+_ℓ and b^- is the blue point closest to r^-_ℓ.

It is possible to choose β sufficient large, so that $D^+_\ell \cap D^-_\ell$ is contained in a strip of width Δ around ℓ. This ensures that a vertex of the arrangement $\mathcal{A}(L)$ is contained in $D^+_\ell \cap D^-_\ell$ if and only if it is incident to ℓ. Elementary trigonometry shows that the value $\beta = \Delta + \Delta^{-1} \cdot d$ is large enough, and therefore the construction only uses numbers that are polynomially bounded.

A new blue point b will capture a red point r^+_ℓ (or r^-_ℓ) if and only if it is contained in D^+_ℓ (or D^-_ℓ, respectively). Every point inside of Q is contained either in D^+_ℓ or D^-_ℓ for every ℓ, and so every point in Q is contained in at least n disks, and no point outside of Q is

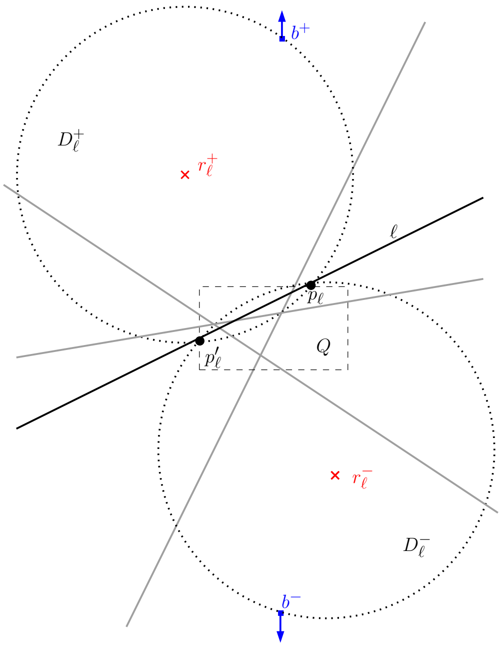

Figure 3: Construction in the 3SUM hardness proof.

contained in more than n disks. Furthermore there is a point in Q contained in at least $n+3$ disks (i.e., point $b \in Q$ is a witness such that $\text{MAXCOV}(S) \geq n+3$) if and only if three lines of L intersect in a common point. The overall reduction takes $O(n \log n)$ time. □

Theorem 4. *The value* MAXCOV(S) *for a set S of n red points and one blue point can be computed in $O(n \log n)$ time, and this is asymptotically optimal under the algebraic decision tree model.*

Proof. Let b be the only blue point and assume that there are not three points on a line. We find an open half-plane H_b with b on its boundary and that contains as many red points as possible. This can be done in $O(n \log n)$ time, by sorting the red points radially from b and performing a rotational sweep of a half-plane with b on its boundary. We then place a new blue point b' close enough to b such that b' captures all the points in $R \cap H_b$. It is obvious that this is an optimal solution, and we have found it in $O(n \log n)$ time.

Next we prove a lower bound. From the discussion in Section 3.1, it is clear that it is sufficient to show an $\Omega(n \log n)$ lower bound for the problem of finding the depth of an arrangement of n disks passing through a common point. Consider the *uniform gap* problem in a quadrant of the unit circle: Given n points $\{p_1, \ldots, p_n\}$ in a quadrant of the unit circle $\mathbb{S}^1 = \{(x, y) \in \mathbb{R}^2 \mid x^2 + y^2 = 1\}$, and a value $\varepsilon > 0$, decide if there is a permutation $\sigma : \{1, \ldots, n\} \to \{1, \ldots, n\}$ such that $d(p_{\sigma(i)}, p_{\sigma(i+1)}) = \varepsilon$ for all $i \in \{1, \ldots, n-1\}$, where the distance $d(\cdot, \cdot)$ refers to the Euclidean distance. This problem has a lower bound of $\Omega(n \log n)$ time in the algebraic decision tree model [21, 25].

Given an instance $P = \{p_1, \ldots, p_n\}$, ε for the uniform gap problem, we make the following reduction to our problem; see Figure 4. For each i, let q_i, q'_i be the points on $\mathbb{S}^1$ at distance ε from p_i, let ℓ_i, ℓ'_i be the lines bisecting the segments $\overline{p_i q_i}$ and $\overline{p_i q'_i}$, and let D_i and D'_i be the

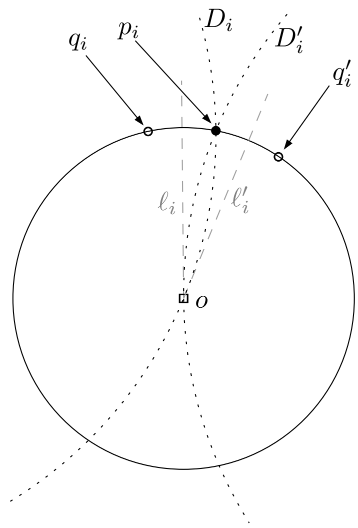

Figure 4: Reduction in Theorem 4.

disks that have o, p_i on their boundary and are tangent to ℓ_i and ℓ'_i, respectively. Note that $D_i \cap D'_i$ lies in one of the wedges defined by ℓ_i and ℓ'_i.

Let o be the blue point, and let the centers of the disks D_i be the set of $2n$ red points for our instance of the MAXCOV(S) problem. Let $\mathcal{D}$ be the set of $2n$ disks $\{D_i, D'_i \mid p_i \in P\}$. The set $\mathcal{D}$ can be constructed in linear time; we next show how computing the depth of the arrangement $\mathcal{D}$ gives the answer to the uniform gap problem. If the answer to the instance P, ε is *yes*, then all the regions $D_1 \cap D'_1 \setminus \{o\}, \ldots, D_n \cap D'_n \setminus \{o\}$ are disjoint, and the maximum depth of $\mathcal{D}$ is $n+1$ (the point o has depth $2n$, but since o is a blue point, we cannot place another blue point there). In contrast, if there are indices i, j such that $d(p_i, p_j) < \varepsilon$, then $D_i \cap D'_i \cap D_j \cap D'_j \setminus \{o\} \neq \emptyset$, and therefore the depth of the arrangement is, at least, $n+2$. Finally, we are left with the case when the answer to the gap problem is *no* because in all permutations a pair of consecutive points are at distance larger than ε. This case can be ruled out from the beginning by finding the leftmost and the rightmost points in P (which are well defined because P is in one quadrant) and checking that they are at the appropriate distance. □

4 The MINMAX and MAXMIN problems

We are given a bichromatic set $S = B \cup R$ formed by a set of m blue points B (facilities) and a set of n red points R (clients), $n \geq m \geq 2$, and a constraint region $X \subseteq L(1)$.

The MINMAX problem

According to the MINMAX criterion we are interested in finding a new blue point $p \in X$ such that the maximum distance to the points in BRNN(p) is minimized. Consider the cost function $Cost : L(1) \to \mathbb{R}$ that measures for each point $p \in L(1)$ the cost, according to the MINMAX criterion, of placing the new blue point, or facility, at p; it follows that

$Cost(p) = \max\{d(p, x) : x \in BRNN(p)\}$. Consider the graph of the function $Cost$ in $3D$. Next, we are going to give a combinatorial description of this graph.

Embed the plane containing R, B in the plane $z = 0$ in 3-space, that is, consider the point sets R, B as embedded in the xy-plane in 3D. For a "client" point $r_i = (x_i, y_i) \in R$, consider the (solid) cylinder

$$Cyl_i = \{(x, y, z) \in \mathbb{R}^3 \mid (x - x_i)^2 + (y - y_i)^2 \leq (d(r_i, b(r_i)))^2\},$$

which is the vertical, solid cylinder through the disk centered at r_i with radius $d(r_i, b(r_i))$, and consider the (surface) cone

$$Con_i = \{(x, y, z) \in \mathbb{R}^3 \mid (x - x_i)^2 + (y - y_i)^2 = z^2, z \geq 0\}$$

with apex at point $(x_i, y_i, 0) \in R$. See Figure 5 left for an example. Finally, let Σ_i be the portion of the surface Con_i contained in Cyl_i. Observe that Σ_i is a surface patch with constant complexity. See Figure 5 right for an example.

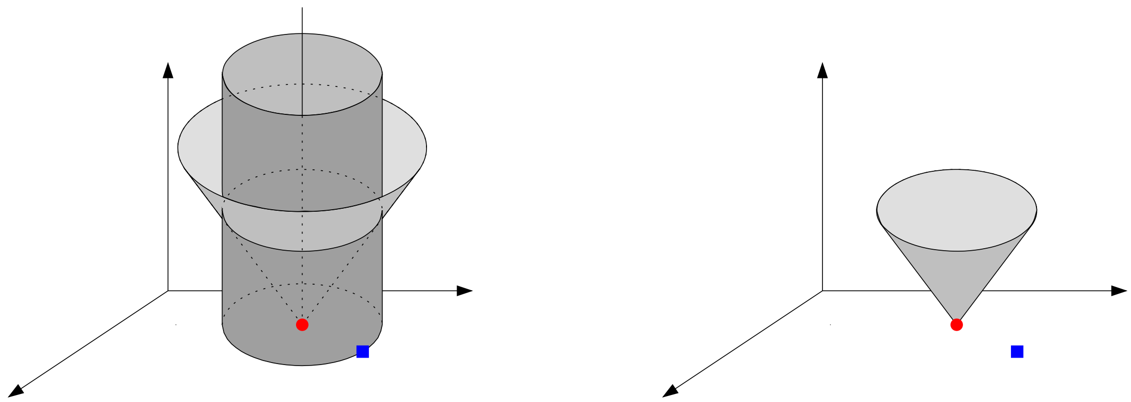

Figure 5: Left: solid cylinder Cyl_i and cone Con_i associated to the point $r_i \in R$. Right: surface patch Σ_i associated with $r_i \in R$.

The reason for considering Σ_i for each point r_i is the following: $\rho = (x, y, t) \in \mathbb{R}^3$ is a point vertically above (resp. below) Σ_i if and only if $r_i \in BRNN(x, y)$ and $d((x, y), r_i) \leq t$ (resp. $d((x, y), r_i) \geq t$). To see the validity of this claim, observe that ρ has a vertical above/below relation with Σ_i if and only if $\rho \in Cyl_i$. Moreover, by the way the cone Con_i is defined, it holds that $\rho = (x, y, t)$ is above Con_i if and only if $d((x, y), r_i) \leq t$. A similar analysis applies to a point ρ below Σ_i and the claim follows.

Let U be the upper envelope of the surfaces $\Sigma_1, \ldots, \Sigma_n$. Using the discussion above we readily obtain the following property.

Lemma 2. *The upper envelope U is the graph of the function Cost.*

We are interested in finding a point $p \in X$ that minimizes $Cost$, and therefore the problem reduces to finding the lowest point in the envelope U restricted to the region X. Let U_X be the portion of U defined over X. If X has complexity $O(n)$ we can argue that U_X

has complexity $O(n^{2+\varepsilon})$ as follows, where the complexity of an envelope U_X is defined as its number of vertices, edges, and faces. For each boundary arc $a \in X$, we consider a vertical wall $W_a = a \times \mathbb{R}$ in 3D. Since X has $O(n)$ complexity, we have $O(n)$ surfaces of the type W_a.

The upper envelope U_W of the surfaces $\Sigma_1, \ldots, \Sigma_n$ together with the walls W_a for arcs a in the boundary of X can be computed and described in $O(n^{2+\varepsilon})$ time, for any fixed $\varepsilon > 0$ [1]. However, since we have introduced the vertical walls W_a, the domain of each patch of U_W is either fully contained in X or fully outside X. It follows that the restriction U_X of U to X can be constructed in $O(n^{2+\varepsilon})$ time.

It remains to find the lower point of U_X. Observe that this point does not necessarily have to be a vertex. However, finding the lower point of U_X can be done by checking each component of U_X, that is, each vertex, edge, and face. For a vertex and an edge in U_X, the lower point can be found in constant time, while for each face in U_X we can find the minimum in time proportional to its complexity. Since the complexity of U_X is $O(n^{2+\varepsilon})$, we conclude the following.

Theorem 5. *The* MINMAX *problem can be solved in $O(n^{2+\varepsilon})$ time, for any fixed $\varepsilon > 0$.*

The MAXMIN **problem**

Using the same approach, the MAXMIN problem can be solved by computing the lower envelope L of $\Sigma_1, \ldots, \Sigma_n$, considering its restriction L_X to a given set X, and finding the highest point in L_X. The same analysis that we have carried out applies to this case, and we obtain the following result.

Theorem 6. *The* MAXMIN *problem can be solved in $O(n^{2+\varepsilon})$ time, for any fixed $\varepsilon > 0$.*

5 Extensions

In this section we consider some extensions of the problems above. First, we combine the MINMAX or MAXMIN criteria with the MAXCOV criteria. Second, we solve the same problems as above under the L_1 and L_∞-metrics. Finally, we consider a different rule to associate clients to facilities, namely, the furthest neighbor rule.

5.1 MINMAX **and** MAXMIN **criteria for optimal** MAXCOV **solutions**

In Subsection 3.1 we have shown that the locus $L(k)$ of all placements achieving k clients can be found in near-quadratic time. Here we describe how to find the best location b within $L(k)$ according to the MINMAX criterion. The MAXMIN criterion can be handled similarly.

Theorem 7. *According to the* MINMAX *criterion, the best location in the set of placements in a level set $L(k)$ can be computed in $O(n^{2+\varepsilon})$ time, for any fixed $\varepsilon > 0$.*

Proof. We use a combination of ideas from Subsection 3.1 and Section 4. Like in Section 4, let U be the upper envelope of the surface patches $\Sigma_1, \ldots, \Sigma_n$. We are interested in finding the lower point of U restricted to the locus $L(k)$, for some value k. Recall that for each point

r_i the circle R_i is centered at r_i and has radius $d(r_i, b(r_i))$. Observe that each cell of $L(k)$ is a cell in the arrangement $\mathcal{A}$ of disks $R_1, \ldots, R_n$. Let U_k be the restriction of the upper envelope U to the set $L(k)$. We next argue that U_k has complexity $O(n^{2+\varepsilon})$ and can be constructed in $O(n^{2+\varepsilon})$ time. For each disk R_i, consider the (surface) cylinder $C_i = R_i \times \mathbb{R}$ in $\mathbb{R}^3$. The upper envelope U' of the surfaces $\Sigma_1, \ldots, \Sigma_n, C_1, \ldots, C_n$ has complexity $O(n^{2+\varepsilon})$ and can be constructed in $O(n^{2+\varepsilon})$ time [1]. Moreover, because we have included $C_1, \ldots, C_n$ in the set of surfaces, the domain of each patch of U' is contained in a cell in the arrangement $\mathcal{A}$. In particular, the restriction of U_k to a cell of $c \in L(k)$ is the same as the restriction of U' to the same cell. We conclude that the envelope U_k has complexity $O(n^{2+\varepsilon})$, and we can find the lower point in U_k using $O(n^{2+\varepsilon})$ time by checking each component of U_k independently. □

Clearly, by finding the highest point of the corresponding lower envelope, similar result applies if we replace the MINMAX criterion by the MAXMIN criterion. Details are omitted.

5.2 The problems under the L_1 and L_∞-metrics

The distance function between facilities and clients depends on the kind of applications. Euclidean distance is appropriate when facilities and clients are spatially located. However, it is also common in location theory to use other distances [11]. In the following, we show how to apply the same techniques for the problems under the L_1 and L_∞ metrics.

Consider the L_∞ metric. For the MAXCOV criterion, the ideas described in Subsection 3.1 directly apply, but they yield better running times. As above, let R_i be the disk (square) with radius $d_\infty(r_i, b(r_i))$ centered at point r_i, and define the arrangement $\mathcal{A}$ induced by $\{R_1, \ldots, R_n\}$. We have to compute the maximum depth of $\mathcal{A}$. Although $\mathcal{A}$ may have quadratic complexity, the maximum depth in an arrangement of n rectangles can be found in $O(n \log n)$ time. This corresponds to a maximum clique in the intersection graph of rectangles [17]. Alternatively, we may use a sweep-line algorithm maintaining a segment tree describing the depth of the line in the arrangement [9]. Since the same argument applies to the L_1 metric, this leads to the following result.

Theorem 8. *In the L_∞ and L_1 metrics, we can compute the value* MAXCOV(S) *and a witness placement in $O(n \log n)$ worst-case running time.*

Observe that the description of all the optimal placements may take $\Omega(n^2)$, since it may consist of the union of many cells from $\mathcal{A}$. Of course, the 3SUM-hardness proof does not carry to the L_∞ or L_1 metric, and there is no need to consider approximation algorithms.

Theorem 9. *In the L_∞ and L_1 metrics, the* MINMAX *problem can be solved in $O(n^2\alpha(n))$ time.*

Proof. For the MINMAX criterion, the same ideas as described for the L_2 metric apply. For each point r_i, we consider the square cylinders $Cyl_i = R_i \times \mathbb{R}$, and the polyhedral cones Con_i such that its section at $z = t$ corresponds a square centered at r_i and side length $2t$. Notice that Σ_i is a surface consisting of 4 triangles, that is, 4 piece-wise linear patches. As above, we want to compute the upper envelope of these linear patches, which can be done in $O(n^2\alpha(n))$ time [13]. The rest of the analysis carries out like before, and we obtain the following improved bound. □

5.3 The reverse farthest neighbor problem

In above Sections we considered the notion of "influence" of a data point on a database as introduced in [19]. In many decision support situations the notion of the "influence set" of a data point is given in terms of geographical proximity or similarity and the distance between vectors is taken as a measure of dissimilarity. If we base the influence set on dissimilarity rather than similarity, the farthest neighbor rather than nearest neighbor can be considered. In [19, 30], finding the set of all reverse farthest neighbors for a query point under the L_2 distance has been proposed as open problem in the monochromatic version. We study here the bichromatic version for the MAXCOV optimization problem. We define the influence set of a blue point b to be the set of all red points r such that b is further from r with respect to any other blue point. More formally, the *bichromatic reverse farthest neighbor set* is defined as

$$\mathrm{BRFN}(b) = \{r_i \in R : d(r_i, b) \geq d(r_i, b_j), \forall b_j \in B\}.$$

We would like to locate a new obnoxious facility and, in order to minimize the risk of this location, maximize the number of clients far away from the new undesirable facility. In this case, a suitable criterion is the MAXCOV as above, but using the farthest neighbor rule. We formalize the new optimization problem as follows.

The farthest MAXCOV **problem.** *Given a bichromatic point set $S = R \cup B$ and a region $X \subset \mathbb{R}^2$, compute the value $\mathrm{MAXCOV}(S) = \max\{|\,\mathrm{BRFN}(b)| : b \in X \setminus B\}$, that is, compute the maximum number of points that $\mathrm{BRFN}(b)$ may have for a new point $b \in X \setminus B$, and find a witness placement $b_0 \in X \setminus B$ such that $|\,\mathrm{BRFN}(b_0)| = \mathrm{MAXCOV}(S)$.*

Notice now that for this problem we also consider the additional constraint that the new point b has to be placed in a given, *bounded* region X, as otherwise, we could always place b to the infinity and the problem is trivially solved. See Figure 6 for an example.

An algorithm similar to the one of Section 3.1 can be applied. For every red point $r_i \in R$, we denote by $b(r_i)$ a *farthest* blue point. Let R_i be the *red disk* with radius $d(r_i, b(r_i))$ centered at point r_i. The set of n disks $\{R_1, \ldots, R_n\}$ can be computed in $O(n \log m) = O(n \log n)$ by using the *farthest* Voronoi diagram of B and preprocessing it for point location [5]. The main observation now is that for any query b, the reverse farthest neighbors r_i are those for which the circles R_i do *not* include b. Therefore, given the arrangement $\mathcal{A}_F$ produced by the set of n red disks $\{R_1, \ldots, R_n\}$, the problem reduces to compute, for each cell $c \in \mathcal{A}_F$, the number of red circles that do not contain the cell c. This value can be obtained observing that, if a cell c has depth k, then we can attach to c the label $l_c = n - k$. In this way, we obtain the solution in $O(n^2)$ worst-case running time.

However, the following result shows that we only need to search for an optimal solution in the boundary of X.

Lemma 3. *If the constraint region X is bounded, there exists a witness point b_0 on the boundary of X that attains $|\,\mathrm{BRFN}(b_0)| = \mathrm{MAXCOV}(S)$.*

Proof. Note that all the blue points B are contained in each of the disks R_i by the definition of the disks $R_1, \ldots, R_n$ are defined. Therefore, all the disks $R_1, \ldots, R_n$ have a common intersection that contains B. Let p_R be any point in $R_1 \cap \cdots \cap R_n$.

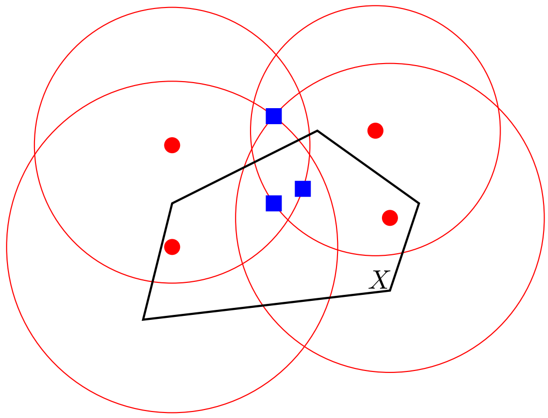

Figure 6: Arrangement $\mathcal{A}_F$ and the constraint region X.

Let c be a cell of $\mathcal{A}_F \cap X$ that has minimum depth, among the cells of $\mathcal{A}_F \cap X$. We claim that c intersects the boundary of X, which proves the statement. Indeed, consider a point $p_c \in c \subseteq X$, and consider a straight walk from p_c in the direction of the vector $\overrightarrow{p_R p_c}$. Because $p_R \in R_1 \cap \cdots \cap R_n$, the ray from p_R to p_c can only exit disks and the depth can only decrease during this walk. Hence the minimum depth is attained when the walk reaches the boundary of X. □

As mentioned before, if X is unbounded, the problem can be trivially solved. When X is bounded, Lemma 3 implies that the search can be restricted to the boundary ∂X of X, which is a one-dimensional space. If the boundary of X has a constant description complexity, the region $\partial X \cap R_i$ has $O(1)$ connected components, for any disk R_i. In this case, we can easily construct the restriction of $\mathcal{A}_F$ to ∂X in $O(n \log n)$ time. Finally, note that we did not explicitly use the L_2 metric, and therefore, the approach also works for the L_∞ and L_1 metrics. We summarize.

Theorem 10. *Let $X \subset \mathbb{R}^2$ be a region with constant description complexity. In the L_1, L_2, and L_∞ metrics, one can solve in $O(n \log n)$ time the furthest* MAXCOV *problem in the constraint region X for a set of n red points and m blue points, $m \leq n$.*

6 Concluding remarks

Given a query blue point, the bichromatic reverse nearest neighbor problem is to find all red points for which the query point is a nearest blue neighbor under some given distance metric. Such queries repeatedly arise when designing efficient algorithms in a variety of areas. In this paper, we introduced and efficiently solved some optimization problems with a direct interpretation in the area of Competitive Facility Location. In particular, we studied three problems (MAXCOV, MINMAX, and MAXMIN) for L_2, L_1 and L_∞ metrics.

The facility location problems usually consider weights measuring the importance of the sites (clients). The MAXCOV problem can be solved analogously in the weighted case. We may also consider to have multiplicative weights for the MINMAX problem, i.e., each point

r_i gets a weight w_i and we want to minimize the maximum $w_i d(r_i, b)$ where $r_i \in \text{BRNN}(b)$. In this case, we only have to change the slope of the cones that we constructed, and the results go through.

We also considered other variations of the problems that arise by combining different criteria, and also the problem related to the farthest neighbor rule, instead of the nearest neighbor rule. For this version, an $O(n \log n)$-time algorithm has been proposed for the MAXCOV criterion. However, it is still an open problem if it is possible to process the input in a data structure (within $O(n \log n)$ time) such that the reverse farthest neighbor set for a query point can be answered in $O(\log n)$ time for the L_2 metric.

Finally, there are several natural problems for further research by considering other optimization problems, like for example, minimizing or maximizing the average or the sum of the distances to BRNN(b).

We recall that our methods and analyses were designed for the planar case exclusively. Adapting them to a higher-dimensional setting, even three dimensions, is a challenge.

Acknowledgements

We are grateful to anonymous referees for many useful comments. These problems were posed and partially solved during the *Second Spanish Workshop on Geometric Optimization*, July 5–10, 2004, El Rocío, Huelva, Spain. The authors would like to thank the Ayuntamiento de Almonte for their support and the other workshop participants for helpful comments.

References

[1] P. K. Agarwal, O. Schwarzkopf, and M. Sharir. The overlay of lower envelopes and its applications. *Discrete Computational Geometry*, 15, 1996, pp. 1–13.

[2] H.-K. Ahn, S.-W. Cheng, O. Cheong, M. Golin, and R. van Oostrum. Competitive facility location: the Voronoi game. *Theoretical Computer Science*, 310, 2004, pp. 457–467.

[3] N. M. Amato, M. T. Goodrich, and E. A. Ramos. Computing the arrangement of curve segments: Divide-and-conquer algorithms via sampling. *Proc. 11th ACM-SIAM Sympos. Discrete Algorithms*, 2000, 705–706.

[4] B. Aronov and S. Har-Peled. On approximating the depth and related problems. *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, 2005.

[5] F. Aurenhammer and R. Klein. Voronoi diagrams. *In J.-R- Sack and J. Urrutia, editors, Handbook of Computational Geometry, Elsevier Science Publishers B. V. North-Holland, Amsterdam*, 2000, pp. 210–290.

[6] R. Benetis, C. S. Jensen, G. Karčiauskas, and S. Šaltenis. Nearest neighbor and reverse nearest neighbor queries for moving objects. *Symposium on Database Engineering & Applications*, 2002, pp. 44–53.

[7] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Trans. Comput.*, C-28, 1979, pp. 643–647.

[8] O. Cheong, A. Efrat, and S. Har-Peled. On finding a guard that sees most and a shop that sells most. *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, 2004, pp. 1098–1107.

[9] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry, Algorithms and Applications.* Springer, 1997.

[10] F. Dehne, R. Klein, and R. Seidel. Maximizing a Voronoi Region: The convex case. *Proceedings of ISAAC, 2002, pp. 624–634.*

[11] Z. Drezner and H. W. Hamacher. *Facility location: applications and theory*, Springer, 2002.

[12] H. Edelsbrunner, L. Guibas, J. Pach, R. Pollack, R. Seidel, and M. Sharir. Arrangements of curves in the plane–topology, combinatorics, and algorithms. *Theoretical Computer Science*, 92, 1992, pp. 319–336.

[13] H. Edelsbrunner, L. Guibas, and M. Sharir. The upper envelope of piecewise linear functions: algorithms and applications. *Discrete Computational Geometry*, 4, 1989, pp. 311–336.

[14] H. A. Eiselt and G. Laporte. Competitive spatial models. *European Journal of Operational Research*, 39, 1989, pp. 231–242.

[15] H. A. Eiselt, G. Laporte, and J. F. Thisse. Competitive location models: A framework and bibliography. *Transportation Science*, 27, 1993, pp. 44–54.

[16] A. Gajentaan and M. H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Computational Geometry Theory and Applications*, 5, 1995, pp. 165–185.

[17] H. Imai and T. Asano. Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane. *J. Algorithms*, 4, 1983, pp. 310–323.

[18] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of Jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete Computational Geometry*, 1, 1986, pp. 59–71.

[19] F. Korn and S. Muthukrishnan. Influence sets based on reverse nearest neighbor queries. *In W. Chen, J. Naughton and P. A. Bernstein editors, Proceedings of the ACM SIGMOD International Conference on Management of Data*, Vol. 29.2 of SIGMOD Record, 2000, pp. 201–212.

[20] F. Korn, S. Muthukrishnan, and D. Srivastava. Reverse nearest neighbor aggregates over data streams. *Proceedings of the 28th VLDB Conference*, Hong Kong, China, 2002.

[21] D. T. Lee and Y. F. Wu. Geometric complexity of some location problems. *Algorithmica*, 1, 1986, pp. 193–211.

[22] K.-I. Lin and M. Nolen. Applying bulk insertion techniques for dynamic reverse nearest neighbor problems. *Seventh International Database Engineering and Applications Symposium*, 2003.

[23] A. Maheshwari, J. Vahrenhold, and N. Zeh. On reverse nearest nighbor queries. *Proceedings of the 14th Canadian Conference on Computational Geometry*, 2002.

[24] F. Plastria. Static competitive location: an overview of optimisation approaches. *European Journal of Operational Research*, 129, 2001, pp. 461–470.

[25] V. Sacristán. Lower bounds for some geometric problems. *Technical Report MA2-IR-98-0034*, 1998. Available at `http://www-ma2.upc.es/~vera/recerca.html`

[26] M. Sharir and P. K. Agarwal. *Davenport-Schinzel sequences and their geometric applications*, Cambridge University Press, 1995.

[27] A. Singh, H. Ferhatosmanoglu and A. Aman Tosun. High dimensional reverse nearest neighbor queries. *Proceedings of the twelfth International Conference on Information and Knowledge Management*, New Orleans, 2003, pp. 91–98.

[28] M. Smid. Closest point problems in computational geometry. *In J.-R. Sack and J. Urrutia editors, Handbook on Computational Geometry, Elsevier Science*, pp. 877–936, 1997.

[29] Y. Tao, D. Papadias, and X. Lian. Reverse kNN search in arbitrary dimensionality. *Proccedings of the 30th VLDB Conference*, Toronto, Canada, 2004.

[30] G.T. Toussaint. Geometric proximity graphs for improving nearest neighbor methods in instance-based learning and data mining. *International Journal of Computational Geometry and Applications*, 15, 2005, pp. 101–150.

[31] Q. Wang, R. Batta, and C. M. Rump. Algorithms for a facility location problem with stochastic customer demand and immobile servers. *Journal Annals of Operations Research*, Springer Issue, Vol. 111, No. 1-4, 2002, pp. 17–34

[32] J. Zhang, Z. Liu, and Z. Ma. Some reverse location problems. *European Journal of Operational Research*, 124, 2000, pp. 77–88.