Covering Point Sets with Two Disjoint Disks or Squares*

Sergio Cabello † J. Miguel Díaz-Báñez ‡ Carlos Seara § J. Antoni Sellarès ¶
Jorge Urrutia ‖ Inma Ventura **

September 2007

Abstract

We study the following problem: Given a set of red points and a set of blue points on the plane, find two unit disks C_R and C_B with disjoint interiors such that the number of red points covered by C_R plus the number of blue points covered by C_B is maximized. We give an algorithm to solve this problem in $O(n^{8/3}\log^2 n)$ time, where n denotes the total number of points. We also show that the analogous problem of finding two axis-aligned unit squares S_R and S_B instead of unit disks can be solved in $O(n \log n)$ time, which is optimal. If we do not restrict ourselves to axis-aligned squares, but require that both squares have a common orientation, we give a solution using $O(n^3 \log n)$ time.

1 Introduction

Let R be a set of red points and let B be a set of blue points on the plane. Suppose that we have two circular coins C_R and C_B of the same size. In this paper we study the following problem that we call the *Two-Coin Problem*: Place C_R and C_B on the plane in such a way that the number of red points covered by C_R plus the number of blue points covered by C_B is maximized. We allow C_B and C_R to cover some red (resp. blue) points, but require them to have disjoint interiors. The requirement for disjoint interiors is relevant, for example, in facility location problems where the facilities may interfere negatively with each other, or when their areas of influence are not allowed to overlap. We use n to denote the total number of points.

We point out that if we do not require C_B and C_R to have disjoint interiors, then the problem can be solved independently for each one of the color point sets using well known techniques [5]. Our result also solves the following problem: Given a point set, place two coins with disjoint interiors such that the number of covered points is maximized. To solve this problem we only need to consider each point as an element of R and B.

*A preliminary version of this work appeared at the 21st European Workshop on Computational Geometry [6].

†Department of Mathematics, IMFM, and Department of Mathematics, FMF, University of Ljubljana, Slovenia, sergio.cabello@fmf.uni-lj.si. 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.

‡Departamento de Matemática Aplicada II, Universidad de Sevilla, Spain, dbanez@us.es. Partially supported by grant BFM2003-04062 and MTM2006-03909.

§Departament de Matemàtica Aplicada II, Universitat Politècnica de Catalunya, Spain, carlos.seara@upc.edu. Supported by projects MCYT-FEDER-BFM2003-00368, Gen-Cat-2001SGR00224 and MCYT-HU2002-0010.

¶Institut d'Informàtica i Aplicacions, Universitat de Girona, Spain, sellares@ima.udg.es. Partially supported by project TIN2004-08065-C02-02.

‖Instituto de Matemáticas, Universidad Nacional Autónoma de México, urrutia@matem.unam.mx. Supported by CONACYT of México, Proyecto 37540-A and MTM2006-03909.

**Departamento de Matemáticas, Universidad de Huelva, Spain, iventura@us.es. Partially supported by grant BFM2003-04062 and MTM2006-03909.

A substantially different problem is that of maximizing the number of points covered by two coins whose interiors may intersect, since then one needs to avoid double counting in the intersection of the coins. Work on similar problems has been done by de Berg et al. [3]. Another set of variants can be obtained by considering coins of different shapes. In this direction, we give an $O(n \log n)$ time optimal algorithm for finding two *axis-aligned* square coins S_B and S_R with disjoint interiors such that the number of red points covered by S_R plus the number of blue points covered by S_B is maximized. We also present an $O(n^3 \log n)$ time algorithm for square coins with *arbitrary orientation*, but restrict them to have the same orientation.

Many related problems have been considered in the context of facility location. A natural assumption is that a point is served by a facility if it lies within a given *distance* from it. The metrics used are the Euclidean distance or the L_∞ (box) metric. Many of these problems also arise in operations research [15]. In particular, the problem of placing a unit disk so as to maximize the number of points covered by the disk was first considered by Drezner [7]. Chazelle and Lee [5] provided a quadratic time algorithm to solve this problem. Within the context of locational analysis, models considering semi-obnoxious facilities [16] or cannibalization [17] naturally lead to problems where the optimal solution must consist of disjoint disks.

In pattern recognition and classification problems, a standard method to select prototypes that represent a class is to perform cluster analysis on the training data [8]. The clustering can be obtained by using simple geometric shapes such as disks or squares. In general, the problem we consider fits under the class of constrained clustering [10]. In this context, recent research deals with the following problem: given sets of red and blue points on the plane, maximize the number of blue points covered by a given object while avoiding all the red points. In particular, Aronov and Har-Peled [1] consider this problem when dealing with disks. Segal [21] and Liu and Nadiak [14] consider the case of axis-parallel squares.

Organization. The rest of the paper is organized as follows. In the next section we present an algorithm to solve the Two-Coin Problem. As a main subroutine for our solution, we use a result of Katz and Sharir [13] for representing the incidences between points and congruent annuli. In Section 3.1 we show an asymptotically optimal algorithm for covering with axis-aligned unit squares that have disjoint interiors. We use techniques similar to those in Katz et al. [12]. In Section 3.2 we extend our algorithm for squares with arbitrary, but common orientation. Finally, in Section 4, we discuss extensions of our results to non-congruent shapes and other optimization objectives.

2 The Two-Coin Problem

In this section we present an $O(n^{8/3} \log^2 n)$ time algorithm to solve the Two-Coin Problem. This improves our previous result presented in [6], where an $O(n^3 \log n)$ time algorithm to solve the same problem was presented.

2.1 A Basic Observation

Consider an optimal placement of C_R^* and C_B^*. We show next that we have to consider only solutions that have certain point configurations on their boundaries.

Lemma 1 *If any of the disks C_R^* and C_B^* cover more than one point, then they can be moved to a new optimal placement such that one of the disks has at least two points on its boundary and the other has at least one.*

Proof: We show first that C_R^* and C_B^* can be translated to a position where each of them has at least one point on its boundary. Move the disks C_R^* and C_B^* away from each other in opposite directions along the line joining their centers until each of their respective boundaries meets a point, say p_R and p_B respectively; see Figure 1a–b.

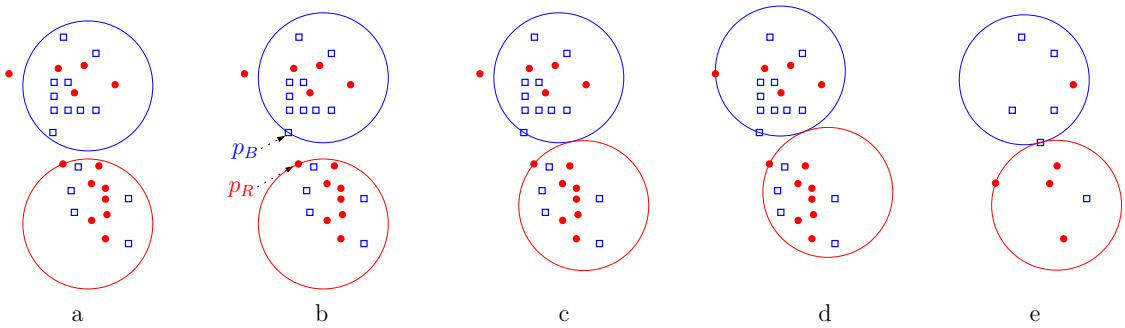

Figure 1: Examples showing some of the steps in the proof of Lemma 1.

Now rotate C_R^* and C_B^* around p_R and p_B respectively and in the clockwise direction keeping their interiors disjoint; they may have to rotate at different speeds if they become tangent (Figure 1b–d). Observe that such rotations always exist unless C_R^* or C_B^* touch both p_R, p_B, in which case our result would be proved (Figure 1e). Suppose then that such rotations are possible. Perform them until some point enters or exists any of the disks, at which point we stop our rotations. At this point we satisfy the conditions of our result (Figure 1d). □

Henceforth, we assume that the optimal solution to the Two-Coin Problem is of size at least 3, as otherwise the problem can be trivially solved.

2.2 The Arrangement of Red and Blue Circles

For each red point $p \in R$ (resp. blue point) we consider the red (resp. blue) unit circle $\mathcal{C}_p$ with center in p. Let $\mathcal{A}$ be the arrangement determined by $\mathcal{C} = \{\mathcal{C}_p;\ p \in R \cup B\}$, and let V, E, and F denote the set of vertices, edges, and faces of $\mathcal{A}$, respectively. Since all the circles have the same radius, $\mathcal{A}$ can be obtained in quadratic time [5, 20].

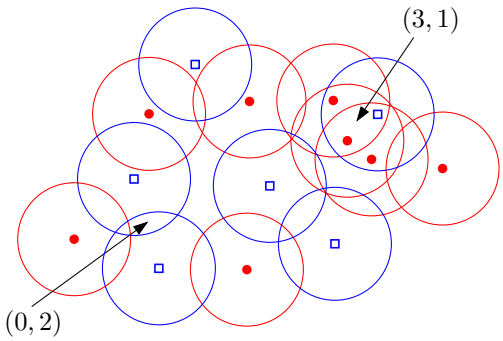

Figure 2: Arrangement $\mathcal{A}$ and the labels $(\mathcal{RF}(f), \mathcal{BF}(f))$ for a pair of cells.

To each face f of the arrangement $\mathcal{A}$, we associate the pair of numbers $(\mathcal{RF}(f), \mathcal{BF}(f))$ such that $\mathcal{RF}(f)$ (resp. $\mathcal{BF}(f)$) is the number of red circles (resp. blue circles) that contain f. Clearly a unit disk with center at a point $x \in f$ covers exactly $\mathcal{RF}(f)$ red and $\mathcal{BF}(f)$ blue points, respectively (Figure 2).

Using standard techniques, we can traverse the dual graph of $\mathcal{A}$ in $O(n^2)$ time. Observe that for any two adjacent faces f and f' separated by an arc of a red circle, $|\mathcal{RF}(f) - \mathcal{RF}(f')| = 1$. Using this fact, we can during a traversal of the faces of $\mathcal{A}$ calculate the values $\mathcal{RF}(f)$, for all $f \in \mathcal{A}$. Similarly we can calculate the values $\mathcal{BF}(f)$, for all $f \in \mathcal{A}$. For an edge $e \in \mathcal{A}$ separating two faces $f, f' \in \mathcal{A}$ we define $\mathcal{RE}(e) = \max\{\mathcal{RF}(f), \mathcal{RF}(f')\}$. For a vertex $v \in \mathcal{A}$ we define $\mathcal{RV}(v)$ to be the maximum $\mathcal{RF}(f)$ over all the faces f that contain v on their boundaries. In a similar way we can define $\mathcal{BE}(e)$ and $\mathcal{BV}(v)$. Clearly all the parameters defined above for the vertices, edges, and faces of $\mathcal{A}$ can be calculated during a traversal of $\mathcal{A}$ in quadratic time.

For technical reasons that will become apparent in Lemma 2, we need to assume that no edge of $\mathcal{A}$ covers more than π of the arc of a circle. Therefore, we subdivide each edge that has length over π into pieces; we can treat the new endpoints that we have introduced as normal vertices of the arrangement.

2.3 The Problem

Observe that a unit circle with center at a vertex of $\mathcal{A}$ (arising as the intersection of two circles) passes through at least two points of $R \cup B$, while a unit circle with center on the interior of an edge of $\mathcal{A}$ passes through exactly one point of $R \cup B$. Therefore by Lemma 1, to solve the Two-Coin Problem, we need to consider only disks with centers on the edges or vertices of $\mathcal{A}$.

Given a unit red circle C_R (resp. blue circle C_B) let $|C_R|$ (resp. $|C_B|$) be the number of red (resp. blue) points lying in C_R (resp. C_B). Let e be an edge in E with endpoints $v_e, v'_e \in V$, and let Q be a point set in the plane. We define $\mathcal{M}_B(e, Q)$ to be the maximum value $|C_B| + |C_R|$ such that the center of C_B belongs to $e \cup \{v_e, v'_e\}$, the center of C_R belongs to Q, and C_B and C_R have disjoint interiors. If all points of Q are such that the interior of the unit circle centered at them intersects the interior of any unit circle with center on $e \cup \{v_e, v'_e\}$, we define $\mathcal{M}_B(e, Q)$ to be 0. $\mathcal{M}_R(e, Q)$ is defined in a similar way.

From Lemma 1, it follows that the optimal solution to the Two-Coin Problem is given by:

$$\max_{e \in E}\{\mathcal{M}_B(e, V), \mathcal{M}_R(e, V)\}.$$

We will restrict ourselves to compute $\max_{e \in E}\{\mathcal{M}_B(e, V)\}$; the other case is symmetric. For simplicity, we will state our formulas under the assumption that $\mathcal{M}_B(e, V) > 0$.

2.4 An Edge

Given a point q on the plane and an edge e of $\mathcal{A}$, let $\Delta(e, q) := \sup\{d(q', q) \mid q' \in e\}$, where $d(\cdot, \cdot)$ denotes the Euclidean distance. Since e is an arc of a circle, it holds that $\Delta(e, q)$ is reached at e or at some of its endpoints, that is $\Delta(e, q) = \max\{d(q', q) \mid q' \in e \cup \{v_e, v'_e\}\}$. Therefore, if $\Delta(e, q) \geq 2$, there is a point $q_e \in e \cup \{v_e, v'_e\}$ such that the unit disks with centers at q and q_e have disjoint interiors.

If q is a point such that $\Delta(e, q) \geq 2$, then

$$\mathcal{M}_B(e, \{q\}) = \mathcal{RV}(q) + \begin{cases} \mathcal{BE}(e) & \text{if } d(v_e, q) < 2,\ d(v'_e, q) < 2. \\ \mathcal{BV}(v_e) & \text{if } d(v_e, q) \geq 2,\ d(v'_e, q) < 2. \\ \mathcal{BV}(v'_e) & \text{if } d(v_e, q) < 2,\ d(v'_e, q) \geq 2. \\ \max\{\mathcal{BV}(v_e), \mathcal{BV}(v'_e)\} & \text{if } d(v_e, q) \geq 2,\ d(v'_e, q) \geq 2. \end{cases}$$

This holds because $\mathcal{BE}(e) \leq \mathcal{BV}(v_e)$ and $\mathcal{BE}(e) \leq \mathcal{BV}(v'_e)$, which implies that, whenever $d(v_e, q) \geq 2$ or $d(v'_e, q) \geq 2$, it is better to place the center of the blue disk at an endpoint of e.

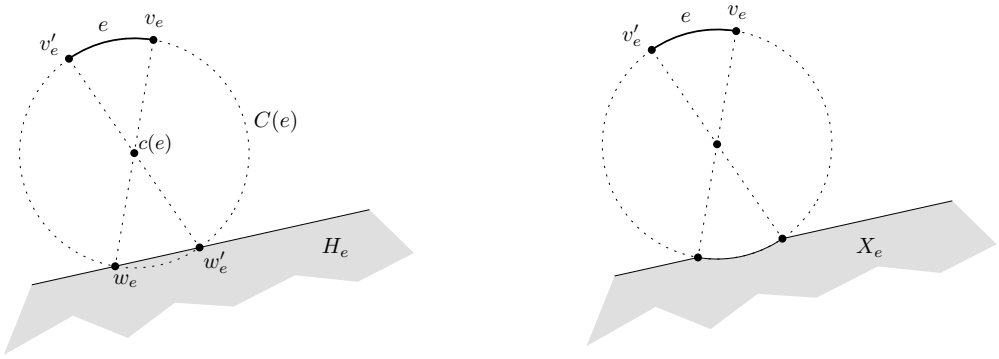

Figure 3: Left: notation concerning an edge e. Right: the region X_e.

Obviously, we have $\mathcal{M}_B(e, V) = \max\{\mathcal{M}_B(e, \{q\}) \mid q \in V,\ \Delta(e, q) \geq 2\}$ and therefore

$$\mathcal{M}_B(e, V) = \max \left\{ \begin{array}{l} \mathcal{BE}(e) + \mathcal{RV}(q) \mid q \in V,\ \Delta(e, q) \geq 2 \\ \mathcal{BV}(v_e) + \mathcal{RV}(q) \mid q \in V,\ d(v_e, q) \geq 2 \\ \mathcal{BV}(v'_e) + \mathcal{RV}(q) \mid q \in V,\ d(v'_e, q) \geq 2 \end{array} \right\}. \tag{1}$$

Given an edge $e \in \mathcal{A}$ we characterize now the set of points q such that $\Delta(e, q) \geq 2$. Let $C(e)$ denote the circle containing e, and let $c(e)$ be the center of $C(e)$. Let w_e and w'_e be the points on $C(e)$ symmetric to v_e and v'_e with respect to $c(e)$, and let H_e be the closed half-plane with w_e, w'_e in its boundary that does not contain v_e. See Figure 3.

The closed annulus centered at a point p with inner radius r and exterior radius r' will be denoted by $Ann(p, r, r')$. For $v \in V$, let $A_v = Ann(v, 2, \infty)$, let $A_e = Ann(c(e), 1, \infty)$, and let $X_e = A_e \cap H_e$.

If e is an edge of $\mathcal{A}$ of length at most π, the following lemma characterizes the set of points q such that $\Delta(e, q) \geq 2$. The proof of this lemma follows immediately from Figures 3 and 4.

Lemma 2 *For any edge $e \in E$ we have*

$$\{q \in \mathbb{R}^2 \mid \Delta(e, q) \geq 2\} = A_{v_e} \cup A_{v'_e} \cup X_e.$$

For a subset $Q \subseteq V$, define $\mathcal{R}(Q) = \max\{\mathcal{RV}(v) \mid v \in Q\}$. Because of Lemma 2, equation (1), and the fact that both $\mathcal{RV}(v_e)$ and $\mathcal{RV}(v'_e) \geq \mathcal{RE}(e)$, it is clear that for any edge $e \in E$ we have

$$\mathcal{M}_B(e, V) = \max \left\{ \begin{array}{l} \mathcal{BE}(e) + \mathcal{R}(V \cap X_e) \\ \mathcal{BV}(v_e) + \mathcal{R}(V \cap A_{v_e}) \\ \mathcal{BV}(v'_e) + \mathcal{R}(V \cap A_{v'_e}) \end{array} \right\}. \tag{2}$$

2.5 Main Result and Discussion

We index the edges E in decreasing order of "quality", that is, let $e_1, \ldots, e_k$ be the elements of E sorted such that $\mathcal{BE}(e_i) \geq \mathcal{BE}(e_{i+1})$, for $1 \leq i < k$. Let $W_i = V \cap X_{e_i}$, for $1 \leq i \leq k$; it holds that $\Delta(e_i, v) \geq 2$ for all $v \in W_i$. Note that if $v \in W_i$ and $v \in W_j$, with $j < i$, then $\mathcal{BE}(e_j) + \mathcal{R}(v) \geq \mathcal{BE}(e_i) + \mathcal{R}(v)$, and there is no need to consider as candidate the blue disk with center in e_i and the red disk with center at v. More generally, if we define $V_i = V \setminus (\bigcup_{j<i} W_j)$, for $1 \leq i \leq k$, then it holds that

$$\max_{e \in E}\{\mathcal{BE}(e) + \mathcal{R}(V \cap X_e)\} = \max_{i=1,\ldots,k}\{\mathcal{BE}(e_i) + \mathcal{R}(V_i \cap X_{e_i})\}. \tag{3}$$

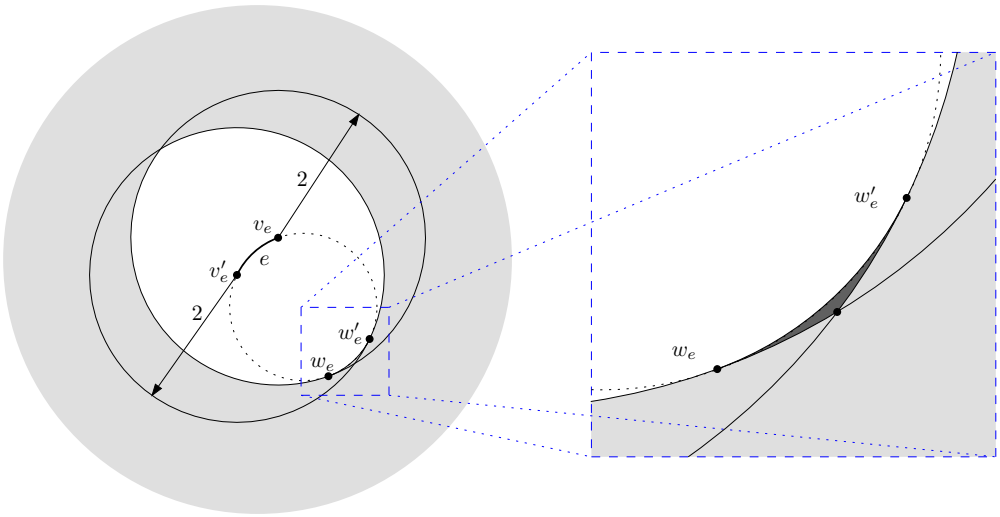

Figure 4: Region of interest in Lemma 2 for edge e, together with a zoom to the most complex part. In the zoom, the region $\{p \in \mathbb{R}^2 \mid \Delta(e,p) \geq 2\} \setminus (A_{v_e} \cup A_{v'_e})$ is in darker grey. This region is covered by X_e.

The main advantage of the formulation on the right side is that the sets $V_i \cap X_{e_i}$, $1 \leq i \leq k$ are pairwise disjoint, which allows a better manipulation. We can use equations (2) and (3) to rewrite our objective as follows

$$\begin{aligned}
\max_{e\in E}\{\mathcal{M}_B(e,V)\} &= \max_{e\in E} \max \left\{ \begin{array}{l} \mathcal{BE}(e) + \mathcal{R}(V \cap X_e) \\ \mathcal{BV}(v_e) + \mathcal{R}(V \cap A_{v_e}) \\ \mathcal{BV}(v'_e) + \mathcal{R}(V \cap A_{v'_e}) \end{array} \right\} \\
&= \max \left\{ \begin{array}{l} \max_{v\in V}\{\mathcal{BV}(v) + \mathcal{R}(V \cap A_v)\} \\ \max_{e\in E}\{\mathcal{BE}(e) + \mathcal{R}(V \cap X_e)\} \end{array} \right\} \\
&= \max \left\{ \begin{array}{l} \max_{v\in V}\{\mathcal{BV}(v) + \mathcal{R}(V \cap A_v)\} \\ \max_{i=1,\ldots,k}\{\mathcal{BE}(e_i) + \mathcal{R}(V_i \cap X_{e_i})\} \end{array} \right\}.
\end{aligned}$$

We next show how to calculate the two values $\max_{v\in V}\{\mathcal{BV}(v)+\mathcal{R}(V\cap A_v)\}$ and $\max_{i=1,\ldots,k}\{\mathcal{BE}(e_i)+\mathcal{R}(V_i \cap X_{e_i})\}$. The approach for this is based on a compact representation of the subset of vertices of V that are inside the annuli $A_v, v \in V$ and $A_e, e \in E$.

Let us recall that a bipartite graph G is a graph whose vertex set can be split into two disjoint subsets X and Y such that all the edges of G join vertices in X to vertices in Y. If for all $x \in X$ and $y \in Y$ the edge xy is in G, then G is called a complete bipartite graph. In this case the sets X and Y are sufficient to represent G, and we will denote G as $G = (X, Y)$. We will rely on the following result by Katz and Sharir regarding the incidences between points and congruent annuli.

Theorem 1 (Katz, Sharir [13]) *Let M be a set of n congruent annuli and P be a set of n points. Then it is possible to compute the set of pairs $\{(A,p) \mid A \in M, p \in P, p \in A\}$ as a collection of edge disjoint bipartite graphs $G_j = (M_j, P_j)$, $M_j \subset M$, $P_j \subset P$, in $O(n^{4/3}\log n)$ time and space, $1 \leq j \leq m$ where m is $O(n^{4/3})$. Moreover $\sum_j(|M_j| + |P_j|) = O(n^{4/3}\log n)$.*

The next result follows from the fact that the size of V is quadratic:

Lemma 3 *We can compute the value $\max_{v\in V}\{\mathcal{BV}(v) + \mathcal{R}(V \cap A_v)\}$ in $O(n^{8/3}\log n)$ time.*

Proof: We have to identify for each $A_v,\ v \in V$, the point $p \in V \cap A_v$ with the largest $\mathcal{RV}(p)$. Let $M = \{A_v \mid v \in V\}$, which consists of $O(n^2)$ annuli. By Theorem 1 we can obtain a compact representation of $\{(A_v, p) \mid A_v \in M,\ p \in V,\ p \in A_v\}$ as a collection of $m = O(n^{8/3})$ edge disjoint bipartite graphs $G_j = (M_j, P_j)$, $M_j \subset M$, $P_j \subset V$, in $O(n^{8/3} \log n)$ time and space.

For each $1 \leq j \leq m$ let $\mathcal{R}_j$ be the largest $\mathcal{RV}(p)$ such that $p \in P_j$. Since $\sum_j |P_j|$ is $O(n^{8/3} \log n)$ the set of all $\mathcal{R}_j$ can be computed in $O(n^{8/3} \log n)$ time, $j = 1, \ldots, m$.

Initialize a variable $\mathcal{R}_v = 0$ for each $v \in V$, which will eventually attain the value $\mathcal{R}(V \cap A_v)$. Next for $i = 1, \ldots, m$ repeat the following: For all $A_v \in M_i$ let $\mathcal{R}_v$ be the maximum between the current value of $\mathcal{R}_v$ and $\mathcal{R}_j$. Clearly at the end of our process for all A_v, the final value of $\mathcal{R}_v$ equals that of the largest $\mathcal{RV}(p)$, $p \in V \cap A_v$, and hence $\mathcal{R}_v = \mathcal{R}(V \cap A_v)$. □

Combining the same technique with semi-dynamic data structures for maintaining the convex hull [4, 11] we obtain the following:

Lemma 4 *We can compute the value* $\max_{i=1,\ldots,k}\{\mathcal{BE}(e_i) + \mathcal{R}(V_i \cap X_{e_i})\}$ *in* $O(n^{8/3} \log^2 n)$ *time.*

Proof: Consider the *multiset* of annuli $M = \{A_e \mid e \in E\}$. By Theorem 1 we can obtain a compact representation of $\{(A_e, p) \mid A_e \in M,\ p \in V,\ p \in A_e\}$ as a collection of $m = O(n^{8/3})$ edge disjoint bipartite graphs $G_j = (M_j, P_j)$, $M_j \subset M$, $P_j \subset V$, in $O(n^{8/3} \log n)$ time and space.

For any edge $e \in E$, let J_e be the set of indices that contain A_e, that is, $J_e = \{j \in \{1, \ldots, m\} \mid A_e \in M_j\}$. For any vertex $v \in V$, let J_v be the set of indices that contain v, that is, $J_v = \{j \in \{1, \ldots, m\} \mid v \in P_j\}$. We can compute the indices J_e, $e \in E$, and the indices $J_v, v \in V$, in $O(n^{8/3} \log n)$ time by scanning the bipartite graphs G_j, $1 \leq j \leq m$. Note that $V \cap A_{e_i} = \bigcup_{j \in J_{e_i}} P_j$, and the union is disjoint because of the properties of Theorem 1.

For each $1 \leq j \leq m$ we store the point set P_j in a semi-dynamic data structure DS_j for maintaining the convex hull [4, 11]: it can be constructed in $O(|P_j| \log |P_j|) = O(|P_j| \log n)$ time, and supports deletions and extreme-point queries (find the point that is extreme in a query direction) in amortized $O(\log |P_j|) = O(\log n)$ time. Since constructing DS_j takes $O(|P_j| \log n)$ time, we spend $\sum_j O(|P_j| \log n) = O(n^{8/3} \log^2 n)$ time.

If at some stage DS_j contains a set of points Q_j, then for any given half-plane H we can construct the set $Q_j \cap H$ and remove it from DS_j in $O((|Q_j \cap H| + 1) \log n)$ time: we query DS_j to find the point $p \in Q_j$ that is extreme in the direction perpendicular to the boundary of the half-plane, and if p is contained in H, then we remove it from DS_j and repeat the process.

Next we compute the sets $V_i \cap X_{e_i}$, $1 \leq i \leq k$, as follows. We proceed in k steps $i = 1, \ldots, k$, and use Q_j for the point set stored in the data structure DS_j, $1 \leq j \leq m$ at any given time. During step i, we compute $Z_i = \bigcup_{j \in J_{e_i}} (Q_j \cap H_{e_i})$, and remove all appearances of elements $v \in Z_i$ from each of the data structures DS_j, $1 \leq j \leq m$. This finishes the description of the algorithm. We claim that $Z_i = V_i \cap X_{e_i}$, $1 \leq i \leq k$. To see this, note that through the process we maintain the invariant that $V_i = \bigcup_j Q_j$ at the beginning of each step i. This clearly holds for $i = 1$. For $i > 1$, the invariant holds because $V \cap A_{e_i} = \bigcup_{j \in J_{e_i}} P_j$, which implies that $V_i \cap A_{e_i} = \bigcup_{j \in J_{e_i}} Q_j$, and therefore $(V_i \cap X_{e_i}) = (V_i \cap A_{e_i} \cap H_{e_i}) = \bigcup_{j \in J_{e_i}} (Q_j \cap H_{e_i})$. As claimed, it follows that $Z_i = V_i \cap X_{e_i}$, $1 \leq i \leq k$, and we can return $\max_{i=1,\ldots,k}\{\mathcal{BE}(e_i) + \mathcal{R}(Z_i)\}$ as the desired value.

The algorithm we have described can be implemented to take $O(n^{8/3} \log^2 n)$ time as follows. In step i, we compute Z_i querying each DS_j, $j \in J_{e_i}$ as discussed above, and we spend $\sum_{j \in J_{e_i}} O((|Q_j \cap H_{e_i}| + 1) \log n) = O((|V_i \cap X_{e_i}| + |J_{e_i}|) \log n)$ time. Here we have used that $V_i \cap A_{e_i} = \bigcup_{j \in J_{e_i}} Q_j$, where the union is disjoint. Using that the sets $V_i \cap X_{e_i}$ are pairwise disjoint, $1 \leq i \leq k$, we can

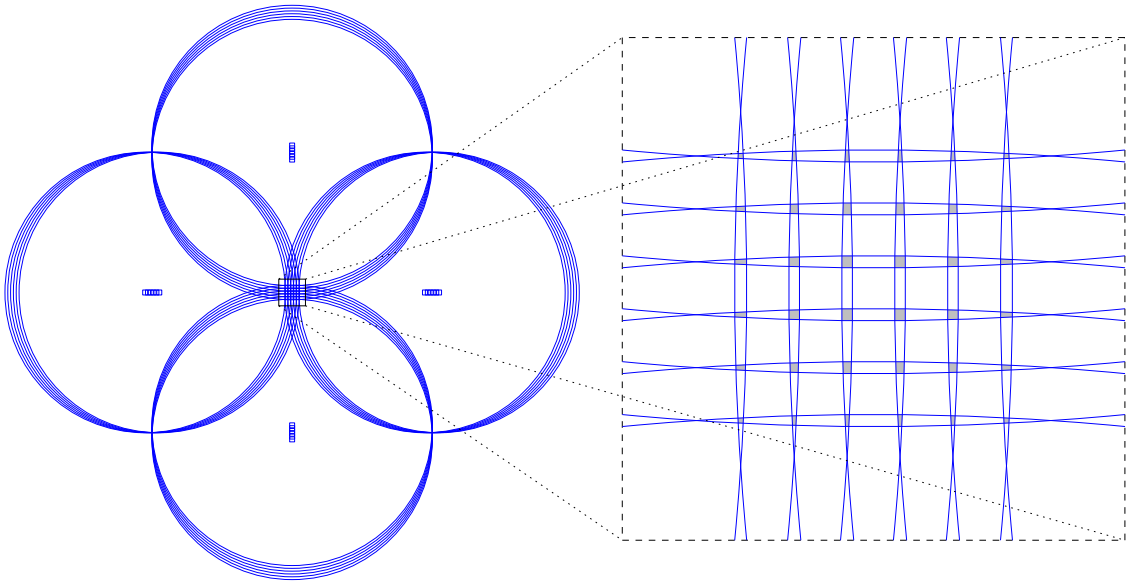

Figure 5: A collection of unit circles and a zoom to a part of the drawing. When the set of unit disks centered at points of B looks like in the figure, placing the center of the blue unit coin C_B in the grey faces of the arrangement would cover most points. A similar scenario but with red circles that is far enough gives rise to $\Theta(n^4)$ optimal solutions to the Two-Coin Problem.

bound the time used to compute $Z_1, \ldots, Z_k$ by

$$\begin{aligned}\sum_{i=1}^{k} O((|V_i \cap X_{e_i}| + |J_{e_i}|)\log n) &= \sum_{i=1}^{k} O(|V_i \cap X_{e_i}|\log n) + \sum_{e\in E} O(|J_e|\log n)\\ &= O(|V|\log n) + O(n^{8/3}\log^2 n)\\ &= O(n^{8/3}\log^2 n).\end{aligned}$$

The deletion of $v \in Z_i$ has to be done in DS_j, $j \in J_v$, and we spend $O(|J_v|)$ time plus the time needed to perform the deletions. Since each point $v \in V$ is deleted at most once during the whole algorithm, and deleting a point from DS_j takes amortized $O(\log n)$ time, it follows that all deletions take at most $\sum_{v\in V} O(|J_v|\log n) = O(n^{8/3}\log^2 n)$ time. □

From the preceding discussion and Lemmas 3 and 4, we conclude our main result.

Theorem 2 *Given a set of red points and a set of blue points on the plane, we can find in $O(n^{8/3}\log^2 n)$ time positions for two interior-disjoint unit disks C_R and C_B such that the number of red points covered by C_R plus the number of blue points covered by C_B is maximized.*

Observe that there may be $\Theta(n^4)$ optimal solutions to our problem. An example showing this bound can be obtained as follows. Given n, construct an instance with $n/2$ blue points that has $\Theta(n^2)$ optimal solutions for the problem of maximum coverage unit disk problem, make a copy and color it red, and, finally, place the red and blue copies far apart; see Figure 5 for an example. It follows that there are $\Theta(n^2)$ optimal placements for the blue color, and the same for the red color. Since the optimal placements for red and blue do not interact, we have $\Theta(n^4)$ optimal solutions to our problem.

Finally, note that the problem we are considering is 3SUM-hard [9]: the 3SUM-hardness result by Aronov and Har-Peled [1] regarding depth in arrangements of disks can be easily adapted to our problem.

3 Two Disjoint Squares

In this section we consider the case when our coins are two axis-parallel unit squares S_R and S_B. As in the previous section, we restrict ourselves to squares of the same size, i.e., unit squares. This condition can be easily removed, leaving the results unchanged. We will present a solution that requires $O(n \log n)$ time. This improves the result presented in a previous version of this work [6], where an algorithm using $O(n^2)$ time was given. We also present an algorithm for the same problem but removing the condition that the squares must be axis-aligned.

3.1 Two Disjoint Axis-Parallel Unit Squares

Let S_R^* and S_B^* be an optimal solution. Observe first that since S_R^* and S_B^* are isothetic, there is a horizontal or vertical line h that separates them. Without loss of generality, assume that h is horizontal, and that S_R^* is below and S_B^* is above the line h (Figure 6).

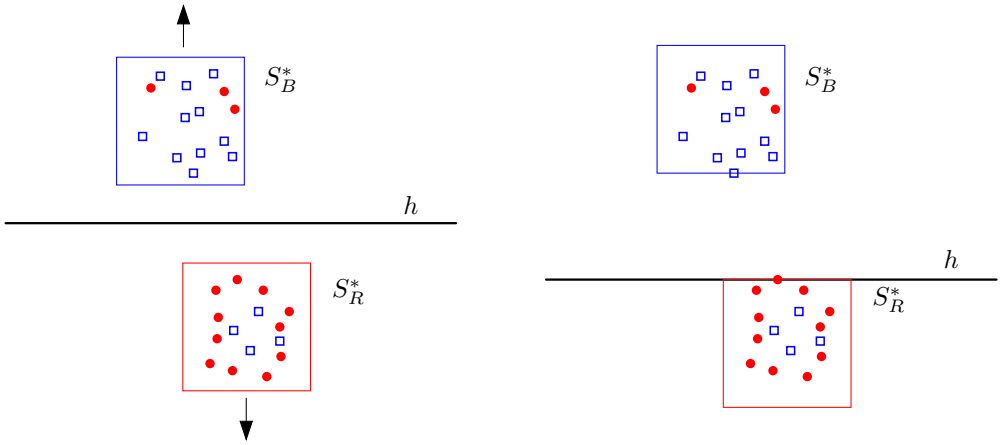

Figure 6: Assumptions on how the optimal solution looks like when the optimal squares have a separating horizontal line and the red square is below it.

Assume that S_R^* has a red point on its top edge, otherwise we can slide it down until its top side meets a red point. Similarly we can assume that S_B^* contains a blue point on its bottom edge. We can also assume that the line h contains a red point, for otherwise we can simply slide it down until it hits an element of R. Using these observations, we will outline a process to find an optimal pair S_R^* and S_B^* in $O(n \log n)$ time. We consider the following subproblem:

Problem 1 *For each point $p \in R \cup B$, find the unit square below the horizontal line h_p passing through p which contains the maximum number $\mathcal{R}(p)$ of red points.*

To solve this problem, we will use segment trees in a similar way to that used by Katz et al. [12] to solve the following problem:

Theorem 3 (Katz et al. [12]) *Given a set P of n weighted points within an axis-parallel rectangle $\mathcal{R}$, and another axis-parallel rectangle $\mathcal{Q}$ which is smaller than $\mathcal{R}$, it is possible to place $\mathcal{Q}$ within $\mathcal{R}$ in $O(n \log n)$ time, such that the sum of the weights of the points covered by $\mathcal{Q}$ is minimized.*

To solve Problem 1 we sweep a horizontal slab $\mathcal{S}$ of width 1 from bottom to top; see Figure 7. During the sweep, we maintain a segment tree structure whose root stores the coordinates of the corners of a red unit square within the slab that contains the maximum number of red points. The events when the segment tree has to be updated are:

1. a point of R hits the bottom boundary line of $\mathcal{S}$, or
2. a point of R hits the top boundary line of $\mathcal{S}$.

Each of these events can be handled in the segment tree in $O(\log n)$ time, using the same approach as Katz et al. [12]. Since there are $2n$ events, we need $O(n \log n)$ time overall.

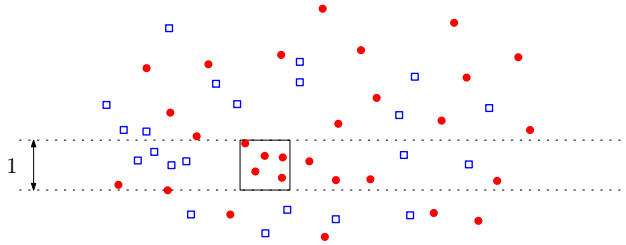

Figure 7: Sweeping the slab. The root of the segment tree keeps track of a best red square within the slab.

For a point $p \in R \cup B$, let $\mathcal{RS}_p$ be a square below the horizontal line h_p passing trough p that contains the maximum number $\mathcal{R}(p)$ of red points. Observe that during the sweep of $\mathcal{S}$ from bottom to top, we can maintain the position of $\mathcal{RS}_p$; each time we reach a point of $R \cup B$ we update the segment tree, if needed, and compare the best square within the slab with the overall best square until then. We conclude the following result.

Lemma 5 *In $O(n \log n)$ time we can find, for all points $p \in R \cup B$, the square $\mathcal{RS}_p$ that contains the largest possible number $\mathcal{R}(p)$ of red points below h_p.*

In a similar way we can determine for each point $p \in R \cup B$ the position of the square $\mathcal{BS}_p$ above h_p that contains the largest number $\mathcal{B}(p)$ of blue points. To solve our problem, all we need to do is to find the point $p \in R \cup B$ that maximizes $\mathcal{R}(p) + \mathcal{B}(p)$. Clearly this can be done in linear time. Thus we have:

Theorem 4 *Given a set of red points and a set of blue points on the plane, we can find in $O(n \log n)$ time positions for two axis-parallel unit-squares S_R and S_B with disjoint interiors such that the number of red points covered by S_R plus the number of blue points covered by S_B is maximized.*

Now we show that the algorithm above is asymptotically optimal by proving an $\Omega(n \log n)$ time lower bound. This lower bound holds in the algebraic tree model of computation, and we will show it by reduction from the following problem.

The ϵ-distance problem: *Given n real numbers $x_1, x_2, \ldots, x_n$ and a positive real number $\epsilon > 0$, determine whether there exists a permutation σ of $(1, 2, \ldots, n)$ such that $x_{\sigma(1)} \leq x_{\sigma(2)} \leq \cdots \leq x_{\sigma(n)}$, and $x_{\sigma(i+1)} > x_{\sigma(i)} + \epsilon$, for all i, $1 \leq i \leq n-1$.*

Theorem 5 [19] *The ϵ-distance problem requires $\Omega(n \log n)$ time under the algebraic computation tree model.*

Proof: The ϵ-distance problem is equivalent to the problem of deciding if a given $(x_1, \ldots, x_n)$ belongs to the set

$$W = \{(x_1, x_2, \ldots, x_n) \in \mathbb{R}^n \mid |x_i - x_j| > \epsilon \text{ for all } i \neq j\}.$$

It is easy to see that W consists of $n!$ connected components, and therefore the lower bound follows from Ben-Or's theorem [2, 18]. □

Theorem 6 *Finding positions for two axis-parallel unit-squares S_R and S_B with disjoint interiors such that the number of red points covered by S_R plus the number of blue points covered by S_B is maximized requires $\Omega(n \log n)$ time in the algebraic computation tree model.*

Proof: Let $S = \{x_1, x_2, \ldots, x_n\}$ and a positive real number $\epsilon > 0$ be an instance for the ϵ-distance problem, and assume w.l.o.g. that $x_n = \max(S)$. Let $R = \{(x_1, 0), (x_2, 0), \ldots, (x_n, 0)\}$, and let $B = \{(b_1, 0), (b_2, 0), \ldots, (b_n, 0)\}$, where $b_i = (x_n + 4\epsilon) + x_i$, $i = 1, \ldots, n$. See Figure 8.

Let us now solve the Two-Coin Problem for $R \cup B$, using ϵ-squares instead of unit-squares.

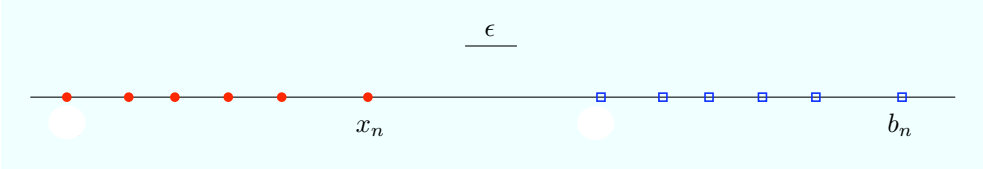

Figure 8: Red and blue points in the reduction for Theorem 6.

By construction, S_R^* and S_B^* cover only red (resp. blue) points, and both of them cover exactly the same number of points. It is clear that the answer to the ϵ-distance problem for S is affirmative if and only if the number of points covered by S_R^* and by S_B^* is exactly one. □

3.2 Two Disjoint Arbitrarily Oriented Squares

We now consider a generalized version of the two axis parallel squares problem. We want to determine two disjoint unit squares S_B and S_R such that they have *the same orientation* such that the number of red points covered by S_R plus the number of blue points covered by S_B is maximized.

Our purpose is to determine a finite number of candidate orientations and then, for each of them, apply the algorithm described in the previous subsection. In order to determine candidate orientations, we will use the following result.

Lemma 6 *The optimal squares S_R^* and S_B^* can be moved to a new position in the plane such that one of the two squares either:* (i) *has at least three points on its edges, two of them on two adjacent edges; or* (ii) *has at least two points on its edges, one of them on a corner.*

Proof: Since S_R^* and S_B^* have common orientation and disjoint interior there is a line h, parallel to two edges of each square, that separates them. Without loss of generality, assume that h is horizontal and that S_R^* is above and S_B^* below h.

Observe that since S_R^* (S_B^*) is an optimal solution, while translating and rotating S_R^* (S_B^*) the first red (blue) point met by an edge of S_R^* (S_B^*) is a point belonging to $R \cap S_R^*$ ($B \cap S_B^*$), so the number of red (blue) points covered by S_R^* (S_B^*) does not change.

Slide S_R^* up until its bottom edge meets a red point r_1, and S_B^* down until its top edge meets a blue point b_1. Assume that b_1 is located to the left or on the vertical line through r_1. Now we slide S_B^* to its left until its right edge meets a blue point b_2 and S_R^* to its right until its left edge meets a red point r_2. Observe that it can occur that $r_2 = r_1$ or/and $b_2 = b_1$.

Finally, we simultaneously apply a motion to S_R^* and S_B^* as follows. Maintaining their common orientation, we rotate S_R^* and S_B^* in the clockwise direction while requiring the bottom and left edge of S_R^* to pass through r_1 and r_2 and the top and right edge of S_B^* to pass through b_1 and b_2 at all times. During the motion the bottom-left (top-right) corner of S_R^* (S_B^*) moves along an arc of the circle of diameter r_1r_2 (b_1b_2). If $r_2 = r_1$ ($b_2 = b_1$) the motion of S_R^* (S_B^*) is a rotation around the bottom-left (top-right) corner of S_R^* (S_B^*). The motion terminates as soon as: i) an edge of S_R^* (S_B^*) meets one more red (blue) point r_3 (b_3); ii) a corner of S_R^* (S_B^*) meets r_1 or r_2 (b_1 or b_2).

In the case that b_1 is located to the right of the vertical line through r_1 we proceed similarly, but rotating S_R^* and S_B^* in counterclockwise direction. In any case the rotation angle is at most $\pi/2$ radians and, by the choice of the orientation of the rotation, during the movement S_R^* and S_B^* remain disjoint. □

Let us see that conditions (i) and (ii) of Lemma 6 determine a finite number of candidate orientations. Figure 9 shows the four cases than can arise according to Lemma 6. Squares of cases a) and b) correspond to condition (i) and squares of cases c) and d) correspond to condition (ii).

case *a*) case *b*) case *c*) case *d*)

Figure 9: Basic cases.

In the four cases, points p and q determine the orientation of the square. Next we explain how to determine this orientation. Cases a) and c): points p and q are on the same edge of a unit square. Consequently $d(p,q) \leq 1$. The line passing through p and q determines the orientation of the unit square. Cases b) and d): points p and q are on opposed edges of a unit square. Consequently, it holds that $1 \leq d(p,q) \leq \sqrt{2}$. The parallel lines passing through p and q at distance 1 determine the orientation of the unit square. Observe that when $d(p,q) = 1$, the orientation obtained in cases a-c) and b-d) coincides. From this discussion, we have the following result.

Lemma 7 *Candidate orientations for optimal squares S_R and S_B can be obtained from pairs of points p, q of the same color such that:*

(1) $d(p,q) \leq 1$ *and the candidate orientation is determined by the line passing through p and q;*

(2) $1 < d(p,q) \leq \sqrt{2}$ *and the candidate orientation is determined by the two parallel lines passing through p and q at distance* 1.

We can now solve the arbitrarily oriented squares problem as follows. First, compute the $O(n^2)$ candidate orientations determined by Lemma 7, and then apply to each candidate orientation the $O(n \log n)$ time algorithm from Theorem 4 to solve the two disjoint axis-parallel unit-squares problem. This leads to the following result.

Theorem 7 *Given a set of red points and a set of blue points on the plane, we can find in $O(n^3 \log n)$ time positions for two unit-squares S_R and S_B with arbitrary but common orientation and disjoint interiors such that the number of red points covered by S_R plus the number of blue points covered by S_B is maximized.*

4 Conclusions

We have presented algorithms for the following problem: Given a set of red points and a set of blue points, find two unit disks (or unit squares) C_R and C_B with disjoint interiors that maximize the number of red points covered by C_R plus the number of blue points covered by C_B. As noted in the introduction, the same algorithms can be used to solve the monochromatic version of the problem, where we only have one point set and we want to maximize the number of points covered by two interior-disjoint unit disks (or squares).

A natural extension of the problem consists in assigning weight to the points, and then optimizing the sum of the weights of the red points covered by the red disk plus the sum of the weight of the blue points covered by the blue disk. Our algorithms can be easily modified to handle this generalization.

Our algorithms can also be extended to handle coins of different sizes, or when they are rectangles of a given shape. The presentation for unit-squares directly carries out to rectangles. For differently-sized disks, the relevant region for an edge can also be expressed as the union of two congruent annuli and the intersection of an annulus and a halfplane. The rest of the presentation carries out in the same way.

We note that our algorithms can be easily modified for any optimization function that is monotone on the number of red points covered by C_R and the number of blue points covered by C_B, that is, any function that increases whenever C_R or C_B cover more red or blue points respectively. For example, we could be interested in disks (or rectangles) C_R, C_B maximizing the value $\min\{|R \cap C_R|, |B \cap C_B|\}$. The same time bounds we have obtained here apply for this problem.

Finally we left as a research problem that of finding two disjoint squares S_B and S_R with arbitrary, but not necessarily equal orientations, such that the number of blue points covered by S_B plus the number of red points covered by S_R is maximized.

Acknowledgements

These problems studied here 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 other workshop participants for helpful comments.

References

[1] B. Aronov and S. Har-Peled. On approximating the depth and related problems. *Proceedings 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2005, pp. 886–894.

[2] M. Ben-Or. Lower bounds for algebraic computation trees. *Proceedings of the fifteenth Annual ACM Symposium on Theory of Computing*, 1983, pp. 80–86

[3] M. de Berg, S. Cabello, and S. Har-Peled. Covering many or few points with unit disks. In *WAOA 2006*, *LNCS* 4368, pp. 55–68.

[4] G. S. Brodal and R. Jacob. Dynamic Planar Convex Hull. *Proceedings of the 43rd Annual Symposium on Foundations of Computer Science*, 2002, pp. 617–626.

[5] B. Chazelle and D. T. Lee. On a circle placement problem. *Computing*, Vol. 36, 1986, pp. 1–16.

[6] J. M. Díaz-Báñez, C. Seara, J. A. Sellarès, J. Urrutia, and I. Ventura. Covering point sets with two convex objects. In *21st European Workshop on Computational Geometry*, 2005.

[7] Z. Drezner. On a modified one-center model. *Management Science*, 27, 1981, pp. 848–851.

[8] R. Duda, P. Hart, and D. Stork. *Pattern Classification. John Wiley and Sons, Inc.*, New York, 2001.

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

[10] J. A. Hartigan. *Clustering Algorithms. Wiley*, New York, 1975.

[11] J. Hershberger and S. Suri. Applications of a semi-dynamic convex hull algorithm. *BIT*, 32, 1990, pp. 249–267.

[12] M. J. Katz, K. Kedem, and M. Segal. Improved algorithms for placing undesirable facilities. *Computers & Operations Research*, 29, 2002, pp. 1859–1872.

[13] M. J. Katz and M. Sharir. An expander-based approach to geometric optimization. *SIAM J. Computing*, 26, 1997, pp. 1384–1408.

[14] Y. Liu and M. Nadiak. Planar case of the maximum box and related problems. *Proceeding of the Canadian Conference on Computational Geometry*, 2003, pp. 11–13.

[15] R. F. Love, J. G. Morris, and G. O. Wesolowsky. *Facilities Location. North-Holland*, Amsterdam, Chapter 3.3, 1988.

[16] Y. Ohsawa, F. Plastria, and K. Tamura. Euclidean push-pull partial covering problems. *Computers & Operations Research*, 33 (12), 2006, pp. 3566–3582.

[17] F. Plastria. Avoiding cannibalisation and/or competitor reaction in planar single facility location. *Journal of the Operational Research Society of Japan*, 48 (2), 2005, pp. 148–157.

[18] F. P. Preparata and M. I. Shamos. *Computacional Geometry, An Introduction. Springer-Verlag*, 1988.

[19] V. Sacristán. Lower bounds for some geometric problems. *Technical Report MA2-IR-98-0034*, Universitat Politècnica de Catalunya, 1998.

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

[21] M. Segal. Planar maximum box problem. *Journal of Mathematical Modelling and Algorithms*, 3, 2004, pp. 31–38.