JCSE Journal of Computing Science and Engineering

Regular Paper

Journal of Computing Science and Engineering,
Vol. 18, No. 3, September 2024, pp. 135-143

Covering Points in the Plane by Two Rectangular Annuli

Sang Won Bae*
Division of Artificial Intelligence and Computer Science, Kyonggi University, Suwon, Korea
swbae@kgu.ac.kr

Abstract

Given a set P of n points in the plane, we study the problem of covering P by an optimal pair of two disjoint rectangular annuli. The optimality is determined by a prescribed cost function that depends on the widths of the resulting rectangular annuli, such as the maximum or the sum of the widths of the two annuli. In this paper, we present the first $O(n \log n)$-time algorithms for a wide range of cost functions, including the min-max and min-sum versions of the problem. We also show the matching lower bound of $\Omega(n \log n)$, in particular, for the min-sum problem.

Category: Algorithms and Complexity

Keywords: Annulus; Rectangular annulus; Geometric location; Computational geometry; Algorithm

1. INTRODUCTION

In the curve fitting problem, we are given a set P of input points and a target family C of curves, and want to find an optimal curve that best fits P. One possible solution to the curve fitting problem, when C consists of closed convex curves, can be found by solving the minimum-width annulus problem.

An annulus informally means a ring-shaped region in the plane, often described by two concentric circles. By choosing C as various families of generalized closed convex curves, one can think of generalized annuli, such as rectangular or square annuli. The minimum-width annulus problem asks to find an annulus of a certain shape, described by C, of minimum width that encloses a given set P of points in the plane. Among other shapes, the case of circular annuli has been first studied with an application to the roundness problem [1–3]. The first sub-quadratic $O(n^{\frac{8}{5}+\varepsilon})$-time algorithm was presented by Agarwal et al. [4], and soon improved to $O(n^{\frac{3}{2}+\varepsilon})$ time by Agarwal and Sharir [5]. Linear-time approximation schemes are also known by Agarwal et al. [6] and by Chan [7].

The minimum-width axis-parallel rectangular annulus problem was first considered by Abellanas et al. [8] who presented an $O(n)$-time algorithm. Gluchshenko et al. [9] presented an $O(n \log n)$-time algorithm that computes a minimum-width axis-parallel square annulus, and proved a matching lower bound. It becomes more difficult when considering rectangular or square annuli that minimize width over all orientations. For rectangular annuli in arbitrary orientation, an $O(n^2 \log n)$-time algorithm that finds a minimum-width rectangular annulus was presented by Mukherjee et al. [10]. For the square case, the first $O(n^3 \log n)$-time algorithm that computes a minimum-width square annulus over all orientations was presented by the author [11]. This algorithm was soon improved to $O(n^3)$ [12] and to $O(n^2 \log n)$ time [13], subsequently.

There are even more results on the minimum-width annulus problems and their generalizations. Bae [14] and Ahn et al. [15] considered the minimum-width annulus problem with outliers: given a set P of n points and a nonnegative integer $k \geq 0$, find a minimum-width annulus

Open Access http://dx.doi.org/10.5626/JCSE.2024.18.3.135 http://jcse.kiise.org

Received 20 August 2024; **Accepted** 6 September 2024
*Corresponding Author

 pISSN: 1976-4677 eISSN: 2093-8020

of a specific shape that encloses at least $n - k$ points in P. The minimum-area rectangular and square annulus problems [12] and the case of parallelogram annuli [16–18] have also been studied. High-dimensional extensions have also been discussed in the literature. Mukherjee et al. [10] showed that the minimum-width cuboidal shell enclosing P can be computed in linear time as well. The author of [19, 20] studied the minimum-width cubic and hypercubic shell problem, which is a high-dimensional extension of the square annulus problem. The problem of finding the minimum-width cuboidal shell with outliers was also considered [21].

All the previous results mentioned above have considered the problems of enclosing points P by a *single* annulus, or its high-dimensional extension, such as a cubic or cuboidal shell. In this paper, for the first time, we address the problem of computing an optimal *pair* of two rectangular annuli whose union encloses given points P (Fig. 1). While the optimality is determined by a prescribed cost function, the most natural and popular would be minimizing the larger width of the two resulting annuli or minimizing the sum of the two widths, namely, the *min-max two rectangular annuli problem* and the *min-sum two rectangular annuli problem*, respectively. Our results are summarized as follows:

(1) We show that both the min-max and the min-sum two rectangular annuli problems can be solved in $O(n \log n)$ time.
(2) We also consider any general cost function f and the *min-cost two rectangular annuli problem*, in which we want to minimize the cost of two resulting annuli with respect to f. This generalizes both the min-max and the min-sum variants. We show the same $O(n \log n)$-time algorithms work for this general problem.
(3) We discuss the lower bound of the problem and prove that the min-sum two rectangular annuli problem requires $\Omega(n \log n)$ time for any algorithm in the algebraic decision tree model. The same lower bound is implied for the min-cost two rectangular annuli problem.

All these upper and lower bounds are the first nontrivial results to the two rectangular annuli problems, to the author's best knowledge.

The rest of the paper is organized as follows: Section II gives precise definitions of necessary concepts and our problems, and essential geometric observations that simplify the problems. Based on those observations, we present an $O(n \log n)$-time algorithm for the min-max problem in Section III. Section IV is devoted to describe our $O(n \log n)$-time algorithm for the min-cost two rectangular annuli problem with a general cost function. Finally, we conclude the paper in Section V with discussions about the lower bound of the problem and open questions.

II. PRELIMINARIES AND OBSERVATIONS

A standard coordinate system is assumed in the plane $\mathbb{R}^2$, having the *horizontal* x-axis and the *vertical* y-axis.

A. Rectangular Annuli

In this paper, we are only interested in axis-parallel rectangles. Hereafter, any rectangle we discuss is assumed to be axis-parallel, unless stated otherwise. Consider a rectangle R in the plane $\mathbb{R}^2$. We call the intersection point of its two diagonals the *center* of R. The *height* and the *width* of R are the lengths of its vertical side and horizontal side, respectively. The (*inward*) *offset* of R by $\delta \geq 0$, or simply the *δ-offset* of R, is a rectangle obtained by sliding the four sides of R inwards by distance δ. If R is of height h and width w, then the offset of R by $\delta = \frac{1}{2}\min\{h, w\}$ is degenerated to a line segment or a point, called the *base* of R. Note that the base of R is either vertical if $h > w$ or horizontal if $h < w$ (Fig. 2(a)).

For any positive $\delta \leq \frac{1}{2}\min\{h, w\}$, consider the δ-offset R' of R. Then, the closed region A between R and R', including its boundary, is called a *rectangular annulus* with the *outer rectangle* R and the *inner rectangle* R' (Fig. 2(b)). The distance δ between the sides of R and R' is called the *width* of the annulus. Let $w(A)$ denote the

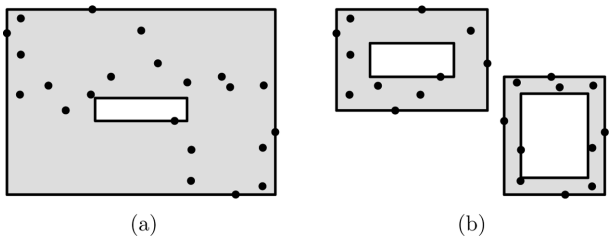

Fig. 1. (a) A minimum-width rectangular annulus enclosing given points P and (b) a pair of two rectangular annuli enclosing the same set P of points that minimizes the larger width of the two annuli.

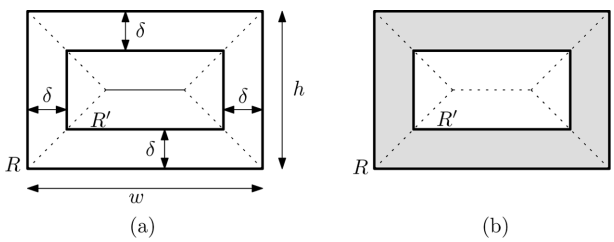

Fig. 2. (a) The δ-offset R' of rectangle R. The solid horizontal segment in the middle is the base of R. (b) The rectangular annulus of width δ whose outer rectangle is R and inner rectangle is its δ-offset R'.

width of annulus A.

Let P be a given set of n points in the plane $\mathbb{R}^2$. The *smallest enclosing rectangle* for P, denoted by $R(P)$, is uniquely determined by the topmost, leftmost, bottommost, and rightmost points of P. Abellanas et al. [8] proved the following, while there can be infinitely many rectangular annuli of minimum width that enclose P.

LEMMA 1 (Abellanas et al. [8]). *There exists a minimum-width rectangular annulus enclosing P such that the outer rectangle of A is R(P), the smallest rectangle enclosing P. Such a rectangular annulus is unique and can be computed in O(n) time*.

We let $A(P)$ be the annulus enclosing P as described above, so $A(P)$ is of minimum width among those enclosing P and its outer rectangle is $R(P)$. Note that the minimum-width rectangular annulus shown in Fig. 1(a) is indeed $A(P)$ whose outer rectangle is $R(P)$ for the set P depicted in the figure.

B. Problem Definition

Now, we present a precise definition of our problems. Given a set P of n points in the plane $\mathbb{R}^2$,

- the *min-max two rectangular annuli problem* asks to find a pair (A_1, A_2) of two disjoint rectangular annuli such that $P \subset A_1 \cup A_2$ and the maximum of their widths, $\max\{w(A_1), w(A_2)\}$, is minimized, and
- the *min-sum two rectangular annuli problem* asks to find a pair (A_1, A_2) of two disjoint rectangular annuli such that $P \subset A_1 \cup A_2$ and the sum of their widths, $w(A_1) + w(A_2)$, is minimized.

We even generalize the problems by adopting a *cost function* $f: \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ that satisfies the following conditions:

(i) $f(w_1, w_2) = f(w_2, w_1)$ for any $w_1, w_2 \geq 0$.
(ii) $f(w_1, w_2) \leq f(w_1', w_2')$ if $w_1 \leq w_1'$ and $w_2 \leq w_2'$, for any $w_1, w_2, w_1', w_2' \geq 0$.
(iii) The value of $f(w_1, w_2)$ can be computed in $O(1)$ time for any given $w_1, w_2 \geq 0$.

Then, in the *min-cost two rectangular annuli problem*, we are additionally given a cost function f as described above and our goal is to find a pair (A_1, A_2) of two disjoint rectangular annuli such that P is enclosed by $A_1 \cup A_2$ and $f(w(A_1), w(A_2)) \leq f(w(A_1'), w(A_2'))$ for all pairs (A_1', A_2') of annuli with $P \subset A_1' \cup A_2'$.

Note that the min-cost problem generalizes the min-max and min-sum variants by setting $f(w_1, w_2) = \max\{w_1, w_2\}$ and $f(w_1, w_2) = w_1 + w_2$, respectively.

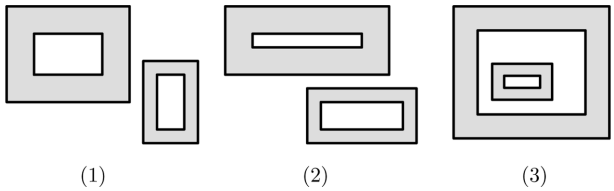

Fig. 3. Three cases of a pair of two disjoint rectangular annuli: (1) the horizontally separated case, (2) the vertically separated case, and (3) the nested case.

C. Characterization of Optimal Solutions

Consider any pair (A_1, A_2) of two disjoint rectangular annuli. It is not difficult to observe that such a pair (A_1, A_2) always falls into one of the following three cases:

(1) (*Horizontally separated*) There is a vertical line that separates A_1 and A_2.
(2) (*Vertically separated*) There is a horizontal line that separates A_1 and A_2.
(3) (*Nested*) Either A_2 is contained in the interior of the inner rectangle of A_1 or A_1 is contained in the interior of the inner rectangle of A_2.

Fig. 3 shows these three cases. This implies that any optimal solutions to our two rectangular annuli problems also must fall into one of the three cases.

Now, we define the following subsets of P and their corresponding costs:

(1) For each $0 \leq i \leq n$, let $L_i \subseteq P$ be the set of i leftmost points from P and $\overline{L}_i := P \setminus L_i$. Note that $L_0 = \emptyset$ and $L_n = P$. Ties are broken arbitrarily, yet consistently so that it holds $L_i \subset L_{i+1}$. Define

$$f_H(i) := f(w(A(L_i)), w(A(\overline{L}_i))).$$

(2) For each $0 \leq i \leq n$, let $T_i \subseteq P$ be the set of i topmost points from P and $\overline{T}_i := P \setminus T_i$. Ties are broken arbitrarily, yet consistently so that it holds $T_i \subset T_{i+1}$. Define

$$f_V(i) := f(w(A(T_i)), w(A(\overline{T}_i))).$$

(3) For $p \in P$, define $\delta_p \geq 0$ to be such that p lies on the boundary of the δ_p-offset of $R(P)$. For $0 \leq i \leq n$, let $D_i \subseteq P$ be the set of i points in P of smallest δ_p-values, and $\overline{D}_i := P \setminus D_i$. Ties are broken arbitrarily, yet consistently so that it holds $D_i \subset D_{i+1}$. Note that D_4 consists of the four points that lie on the boundary of $R(P)$ as their δ_p-values are all zero. Define

$$f_N(i) := f(w(A(D_i)), w(A(\overline{D}_i))).$$

We then observe the following, which is the key for our algorithms.

LEMMA 2. *There exists an optimal pair* (A_1^*, A_2^*) *of two disjoint rectangular annuli with respect to a cost function f such that either*

(1) $A_1^* = A(L_i)$ *and* $A_2^* = A(\overline{L}_i)$ *for some i,*
(2) $A_1^* = A(T_i)$ *and* $A_2^* = A(\overline{T}_i)$ *for some i, or*
(3) $A_1^* = A(D_i)$ *and* $A_2^* = A(\overline{D}_i)$ *for some i.*

Therefore, it holds that

$$f(w(A_1), w(A_2)) = \min_{0 \le i \le n} \{f_H(i), f_V(i), f_N(i)\}.$$

Proof. Let (A_1, A_2) be any optimal pair of two disjoint rectangular annuli such that $P \subset A_1 \cup A_2$ and $f(w(A_1), w(A_2))$ is minimized. Let $P_1 := P \cap A_1$ and $P_2 := P \cap A_2$. Note that we have $P = P_1 \cup P_2$ and $P_1 \cap P_2 = \emptyset$, as A_1 and A_2 are disjoint. As discussed above, since A_1 and A_2 are disjoint, they are either (1) horizontally separated, (2) vertically separated, or (3) nested. We consider each case separately.

First, suppose that A_1 and A_2 are horizontally separated, so there is a vertical line that separates A_1 and A_2. Without loss of generality, assume that A_1 is to the left of A_2. Then, we observe that $P_1 = L_i$, where $i = |P_1|$, and $P_2 = \overline{L}_i$. Now, define $A_1^* := A(L_i)$ and $A_2^* := A(\overline{L}_i)$. By definition and Lemma 1, we have $w(A_1^*) \le w(A_1)$ and $w(A_2^*) \le w(A_2)$. Hence,

$$f(w((A_1^*), w(A_2^*)) \le f(w(A_1), w(A_2))$$

by condition (ii) of cost function f. This implies the optimality of (A_1^*, A_2^*).

Second, suppose that A_1 and A_2 are vertically separated, so there is a horizontal line that separates A_1 and A_2. Without loss of generality, assume that A_1 is above A_2. Then, we observe that $P_1 = T_i$, where $i = |P_1|$, and $P_2 = \overline{T}_i$. Now, define $A_1^* := A(T_i)$ and $A_2^* := A(\overline{T}_i)$. By Lemma 1, we have $w(A_1^*) \le w(A_1)$ and $w(A_2^*) \le w(A_2)$. Hence,

$$f(w(A_1^*), w(A_2^*)) \le f(w(A_1), w(A_2)),$$

by condition (ii) of cost function f. This implies the optimality of (A_1^*, A_2^*).

Finally, suppose that A_1 and A_2 are nested. Without loss of generality, we assume that A_2 is contained in the interior of the inner rectangle of A_1. In this case, we define $A_1^* := A(P_1)$ and $A_2^* := A(P \setminus A_1^*)$. Observe that A_1^* encloses P_1 and possibly some points from P_2, that is, $P_1 \subseteq P \cap A_1^*$, while we still have $w(A_1^*) \le w(A_1)$ and $w(A_2^*) \le w(A_2)$ by definition and Lemma 1. So, (A_1^*, A_2^*) is also an optimal pair, similarly as above. Now, observe that the outer rectangle of A_1^* is the smallest enclosing rectangle $R(P)$ for P and its inner rectangle is determined by some point $q \in P$, which is the δ_q-offset of $R(P)$. Hence, we have $P \cap A_1^* = D_i$ and $P \cap A_2^* = \overline{D}_i$, where i is the rank of q such that $q \in D_i$ and $q \notin D_{i-1}$. □

Note that Lemma 2 dramatically simplifies the problem to computing at most $O(n)$ cost values: $f_H(i), f_V(i), f_N(i)$ for $0 \le i \le n$.

III. ALGORITHM FOR THE UNIMODAL CASE

In this section, we solve a special case of the min-cost two rectangular annuli problem, where the given cost function f induces some favorable properties.

Before getting into the main discussion, we mention that the exact values of $f_H(i)$, $f_V(i)$, and $f_N(i)$ can be computed in $O(n)$ time by Lemma 1.

LEMMA 3. *Given a set P of n points, a cost function f, and an integer i with* $0 \le i \le n$, *the exact values of* $f_H(i)$, $f_V(i)$, *and* $f_N(i)$ *can be computed in* $O(n)$ *time.*

Proof. First, we discuss how to compute $f_H(i)$. Recall that $f_H(i) = f(w(A(L_i)), w(A(\overline{L}_i)))$ and L_i is the set of i leftmost points from P. To specify L_i, we apply the linear-time selection algorithm [22] to find the i-th leftmost point in P and then collect the i points from P that have x-coordinates at most the i-th point. This also gives us the set $\overline{L}_i = P \setminus L_i$. We then apply Lemma 1 to compute $A(L_i)$ and $A(\overline{L}_i)$ together with their widths. This way, we can compute the value of $f_H(i)$ in $O(n)$ time.

The case of $f_V(i)$ is almost identical to the above case of $f_H(i)$.

Lastly, consider $f_N(i)$. Recall that $f_N(i) = f(w(A(D_i)), w(A(\overline{D}_i)))$ and D_i is the set of i closest points from the boundary of $R(P)$. To specify D_i, compute $R(P)$ and the δ_p-values for all $p \in P$, and store the δ_p-values into an array. We then apply the selection algorithm [22] as above to pick the i-th smallest value in the array. From this, we can specify the subsets D_i and $\overline{D}_i$ in $O(n)$ time. Finally, we apply Lemma 1 to compute $A(D_i)$ and $A(\overline{D}_i)$ as above. The total time complexity is bounded by $O(n)$ as claimed.

Lemma 3 implies an $O(n^2)$-time algorithm since we are done by computing all values of $f_H(i), f_V(i)$, and $f_N(i)$ for $0 \le i \le n$ by Lemma 2. In the following, we show how to improve it to $O(n \log n)$ time.

Specifically, consider the min-max problem, in which $f(w_1, w_1) = \max\{w_1, w_2\}$. In this case, we have a nice property among the values of $f_H(i), f_V(i)$, and $f_N(i)$, called the *unimodality*. More precisely, we have:

LEMMA 4. *Suppose* $f(w_1, w_1) = \max\{w_1, w_2\}$, *and let*

$g \in \{f_H, f_V, f_N\}$. Then, there exists a unique integer m with $0 \le m \le n$ such that $g(i)$ is non-increasing for $i \le m$ and is $g(i)$ is non-decreasing for $i \ge m$.

Proof. Recall $f_H(i) = \max\{w(A(L_i)), w(A(\overline{L}_i))\}$. The claimed property is easily seen from the fact that $w(A(L_i))$ is non-decreasing and $w(A(\overline{L}_i))$ is non-increasing as i increases from 0 to n.

The other two functions $f_V(i)$ and $f_N(i)$ are analogous. □

The unimodality of f_H, f_V, and f_N enables us to apply a ternary search to find the minimum value of each of them. More precisely, let $g \in \{f_H, f_V, f_N\}$ be any of the three functions. Our goal is to find the minimum of g, that is, $\min\{g(0), g(1), \ldots, g(n)\}$. Set $a := \lfloor n/3 \rfloor$ and $b := \lfloor 2n/3 \rfloor$. Evaluate $g(a)$ and $g(b)$ using Lemma 3 in $O(n)$ time. If $g(a) > g(b)$, then we know that the minimum of g cannot be attained at $i \le a$ by the unimodality, so we discard the range $[0, a]$ and recurse our search in the range $[a + 1, n]$; if $g(a) < g(b)$, then we discard the range $[b, n]$ by the same reason and recurse our search in $[0, b - 1]$; if $g(a) = g(b)$, then we can discard the ranges $[0, a - 1]$ and $[b + 1, n]$, and recurse our search in $[a, b]$. In either case, the search space is reduced to a fraction of at most 2/3 after spending $O(n)$ time. Hence, it takes only $O(n \log n)$ time to find the minimum of g. So, we conclude our first algorithmic result.

THEOREM 1. *Given a set P of n points in the plane, the min-max two rectangular annuli problem can be solved in $O(n \log n)$ time.*

Though the above algorithm is described for the min-max case, it is not difficult to see that the same algorithm also works for any cost function f if the three functions f_H, f_V, f_N are unimodal.

COROLLARY 1. *Given a set P of n points in the plane and a cost function f satisfying conditions (i)–(iii), suppose that the three functions f_H, f_V, and f_N are unimodal. Then, the min-cost two rectangular annuli problem can be solved in $O(n \log n)$ time.*

IV. ALGORITHM FOR THE GENERAL CASE

We then consider any general cost function f satisfying conditions (i)–(iii). The unimodality allows us a very simple $O(n \log n)$-time algorithm as discussed above, but this is not always the case. One such example is the min-sum two rectangular annuli problem. If $f(w_1, w_2) = w_1 + w_2$, then each of the three functions f_H, f_V, f_N is not necessarily unimodal.

In general, it seems there is no better way but computing all values of $f_H(i)$, $f_V(i)$, $f_N(i)$ for $0 \le i \le n$. To beat the linear-time algorithm described in Lemma 3, we show that evaluating each one can be done in logarithmic time using known geometric data structures.

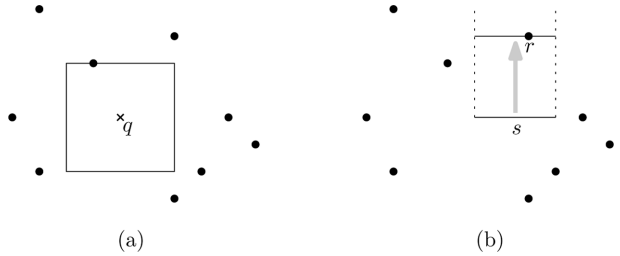

Fig. 4. Illustrations to (a) the L_∞ nearest neighbor query for a query point q and (b) the segment dragging query for a query segment $\boldsymbol{s}$ when dragged upwards for the same set P of points. For the answer to the queries, (a) the largest empty square centered at q and (b) the point $r \in P$ first hit by the dragged segment are returned.

The essential subproblem here is to compute the minimum-width rectangular annulus and its width for a certain subset of P. More precisely, consider the following query problem: given a query rectangle R, find the minimum-width rectangular annulus enclosing $P \cap R$ whose outer rectangle is R. Note that this is equivalent to finding the largest offset R' of R whose interior contains no point of P. So, let us call this problem the *largest empty offset query problem*.

A. Largest Empty Offset Queries

In order to efficiently process largest empty offset queries, we make use of data structures that support the following queries:

- (L_∞ *nearest neighbor queries*) Given a query point $q \in \mathbb{R}^2$, find a nearest neighbor of q from P under the L_∞ distance or, equivalently, find the largest axis-parallel empty square centered at q that contains no points of P in its interior.

- (*Segment dragging queries*) Given a horizontal segment s, find the first point from P hit by $\boldsymbol{s}$ when $\boldsymbol{s}$ is being dragged vertically upwards or downwards. If there is no such point hit by $\boldsymbol{s}$, report that there is no such point. Symmetrically, for a vertical segment $\boldsymbol{s}$, we drag $\boldsymbol{s}$ horizontally leftwards or rightwards to find the first point from P hit by $\boldsymbol{s}$.

Fig. 4 shows this process. These two types of queries are known to be handled in $O(\log n)$ time after preprocessing P in $O(n \log n)$ time using $O(n)$ space. For L_∞ nearest neighbor queries, we build the L_∞ *Voronoi diagram* of P with an optimal point location structure. This can be done in $O(n \log n)$ time using $O(n)$ space [23]. Chazelle [24] presented a data structure for segment dragging queries that requires $O(n \log n)$ preprocessing

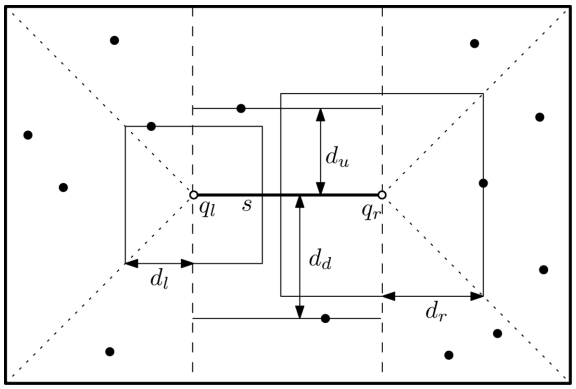

Fig. 5. Illustration to the algorithm processing a largest empty offset query for a query rectangle *R*. Black dots depict points in *P* in the interior of *R*. In this case, d_l is the minimum among the others d_l, d_u, d_d, and the $(h/2 - d_l)$-offset of *R* should be the answer to the query.

time and $O(n)$ space.

Assume that these data structures have already been built for the two kinds of queries. Now, we describe how to handle a largest empty offset query for a given query rectangle *R*. Let *h* and *w* be the height and width of *R*, and assume without loss of generality that $h \leq w$; the other case where $h > w$ can be handled symmetrically. Let $\boldsymbol{s}$ be the base of *R*, which is the $(h/2)$-offset of *R*. Note that $\boldsymbol{s}$ is a horizontal segment as $h \leq w$. Let q_l and q_r be its left and right endpoints, respectively. We query q_l and q_r for the L_∞ nearest neighbor and then query $\boldsymbol{s}$ for the segment dragging query, twice: upwards and downwards. By the two L_∞ nearest neighbor queries, we obtain two squares centered at q_l and q_r, respectively; let d_l and d_r be half the side lengths of the two squares, respectively. By the two segment dragging queries, upwards and downwards, we obtain two points above and below $\boldsymbol{s}$; let d_u and d_d be the distances to these points from $\boldsymbol{s}$, respectively. After specifying the values of d_l, d_r, d_u, d_d, we take their minimum

$$d := \min\{d_l, d_r, d_u, d_d, h/2\}$$

and answer the largest empty offset query by reporting the $(h/2 - d)$-offset R' of *R*.

We claim that the $(h/2 - d)$-offset R' of *R* is indeed the largest offset of *R* whose interior contains no point of *P* (Fig. 5).

LEMMA 5. *After preprocessing P in $O(n \log n)$ time and $O(n)$ space, a largest empty offset query can be correctly answered in $O(\log n)$ time.*

Proof. The complexity for preprocessing and handling a query has been discussed above. Thus, in this proof, we focus on proving the correctness of the above algorithm.

From the description of the algorithm, the answer to a largest empty offset query is determined as the $(h/2 - d)$-offset of the query rectangle *R*, when the height *h* of *R* is at most its width, and *d* is chosen as the minimum of the five values d_l, d_r, d_u, d_d, $h/2$. Hence, there are five cases according to which of the five is the minimum, determining *d*.

First, suppose $d = d_l$, so *d* is determined by the L_∞ nearest neighbor query for q_l (Fig. 5). Let S_l be the resulting square centered at q_l and p_l be the L_∞ nearest neighbor of q_l, that is, p_l lies on the boundary of S_l. We observe that p_l does not lie on the right side of S_l, since, if so, p_l lies above or below the base $\boldsymbol{s}$ of *R* and the distance from $\boldsymbol{s}$ to p_l, which is either d_u or d_d, must be strictly smaller than the L_∞ distance d_l from q_l to p_l, a contradiction. Now, consider the $(h/2 - d_l)$-offset R' of *R*. Then, p_l lies on the boundary of R', more precisely, either on its left, top, or bottom side, so there is no empty offset of *R* larger than R'. The emptiness of the interior of R' can be verified as follows. Since $d_l \leq d_u$ and $d_l \leq d_d$, there is no point of *P* in the rectangle swept by dragging $\boldsymbol{s}$ upwards and downwards by distance d_l. Further, the interior of S_l is empty and the interior of the square with side length $2d_l$ and center q_r is also empty since $d_l \leq d_r$. The union of those empty regions is equal to R', so the interior of R' is empty. This completes the proof for the case where $d = d_l$. The case where $d = d_r$ can also be handled symmetrically.

Second, suppose $d = d_u$, so *d* is determined by the segment dragging query with $\boldsymbol{s}$ dragged upwards. Let R' be the $(h/2 - d_u)$-offset of *R*. Note that there is a point in *P* on the top side of R', so there is no empty offset of *R* larger than R'. Since $d_u \leq d_d$, there is no point of *P* in the rectangle swept by dragging s downwards by distance d_u. Also, since $d_u \leq d_l$ and $d_u \leq d_r$, the two squares centered at q_l and q_r with side length $2d_u$ do not contain any point of *P* in their interior as well. Similarly as above, since the union of those empty regions cover the rectangle R', we confirm that the interior of R' is empty. This completes the proof for the case where $d = d_u$. The case where $d = d_d$ can also be handled symmetrically.

Finally, suppose $d = h/2$, so we have that the four values d_l, d_r, d_u, d_d are all at least $h/2$. From the discussions above, it is not difficult to see that this implies that the interior of *R* does not contain any point of *P*. Hence, in this case, the answer to the largest empty offset query should be *R* itself, that is, the 0-offset of *R*. □

B. Overall Algorithm

Now, we turn back to the original problem and describe the overall algorithm. We start by preprocessing *P* and building necessary data structures for largest empty offset queries. Then, we compute $f_H(i)$, $f_V(i)$, and $f_N(i)$ for all $0 \leq i \leq n$ by largest empty offset queries. For this purpose, we need to compute the corresponding outer rectangles in advance.

First, we discuss how to compute $f_H(i)$. Recall that

$f_H(i) = f(w(A(L_i)), w(A(\overline{L}_i))$. To apply largest empty offset queries, we compute and store $R(L_i)$ and $R(\overline{L}_i)$ for all i. For this purpose, we sort P in the x-coordinates and scan P in this order. The subsets L_i are specified during this scan and at the same time we maintain the four extreme points of L_i, that is, the leftmost, rightmost, topmost, and bottommost points of L_i. These four extreme points determine the smallest enclosing rectangle $R(L_i)$, so we store them in an array. The subsets $\overline{L}_i$ and their corresponding rectangles $R(\overline{L}_i)$ can be obtained and stored by another scan of P in the reversed direction. It is obvious that this process takes $O(n)$ time after sorting. We are then able to evaluate $f_H(i)$ for all i in $O(n \log n)$ total time as $A(L_i)$ and $A(\overline{L}_i)$ for each i can be obtained by largest empty offset queries for $R(L_i)$ and $R(\overline{L}_i)$.

The case of $f_V(i)$ can be handled almost the same way as above, so we omit the details.

Lastly, what remains is to compute $f_N(i)$. Recall that $f_N(i) = f(w(A(D_i)), w(A(\overline{D}_i))$. In this case, note that $R(D_i)$ is always the same as $R(P)$, so $A(D_i)$ can be found directly without a largest empty offset query. We start by sorting the points of P in the increasing order of their δ_p-values, and scan them in this order. The subsets D_i are specified during this scan and, at the same time, we also obtain $A(D_i)$ as its outer rectangle is $R(P)$ and inner rectangle is determined as the δ_q-offset of $R(P)$, where q is the i-th point of P, that is, $q \in D_i \setminus D_{i-1}$. To find out $A(\overline{D}_i)$, we scan the sorted list in the reversed direction. During this reversed scan, we maintain the four extreme points of $\overline{D}_i$, that is, its leftmost, rightmost, topmost, and bottommost points. From these four points, we can compute and store $R(\overline{D}_i)$ in an array for all i in $O(n)$ time after sorting. To obtain $A(\overline{D}_i)$, we apply a largest empty offset query for $R(\overline{D}_i)$. This way, we can evaluate $f_N(i)$ for all i. The total time spent is $O(n \log n)$.

Summarizing, we obtain our main theorem.

THEOREM 2. *Given a set P of n points in the plane and a cost function f satisfying conditions (i)–(iii), the min-cost two rectangular annuli problem can be solved in $O(n \log n)$ time using $O(n)$ space.*

The min-sum two rectangular annuli problem is a special case of Theorem 2.

COROLLARY 2. *Given a set P of n points in the plane, the min-sum two rectangular annuli problem can be solved in $O(n \log n)$ time using $O(n)$ space.*

V. CONCLUDING REMARKS

We have discussed the two rectangular annuli problems and presented the first algorithms for the problem that run in $O(n \log n)$ time for general cost functions.

One natural question is about the computational complexity of the problem. We partially answer the question by showing that the *maximum gap problem* is reducible in linear time to the min-sum two rectangular annuli problem. Given a set S of n real numbers, the *maximum gap* of S, denoted by $g(S)$, is the maximum difference between two consecutive numbers of S when they are sorted. It is well known that the problem of computing the maximum gap $g(S)$ of a given set S has $\Omega(n \log n)$ lower bound in the algebraic decision tree model [25].

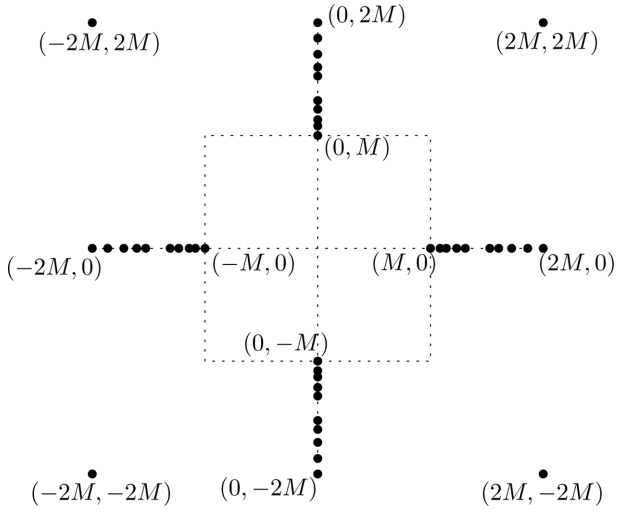

Fig. 6. Our construction of a set P consisting of $4n + 4$ points.

THEOREM 3. *Any algorithm for the min-sum two rectangular annuli problem requires $\Omega(n \log n)$ time in the algebraic decision tree model.*

Proof. We transform a given instance of the maximum gap problem to an instance of the min-sum two rectangular annuli problem. Let S be a given set of n real numbers, and we want to compute the maximum gap $g(S)$. Without loss of generality, we assume that $0 \in S$ is the minimum element and let $M \geq 0$ be its maximum element. Note that this assumption can be easily achieved by finding out the minimum element of S and subtract it from every one in S in $O(n)$ time, and also that M can be computed in $O(n)$ time. We then construct a set P of $4n + 4$ points in the plane $\mathbb{R}^2$ as follows:

$$P := \{(0, M+a), (0, -M-a), (M+a, 0), (-M-a, 0) \mid a \in S\}$$
$$\cup \{(2M, 2M), (-2M, 2M), (2M, -2M), (-2M, -2M)\}$$

Fig. 6 shows our construction of P. We claim that the optimal solution to the min-sum two rectangular annuli problem for P directly implies the exact value of $g(S)$. More precisely, the sum of the widths of the optimal pair of annuli will be shown to be exactly $M - g(S)$.

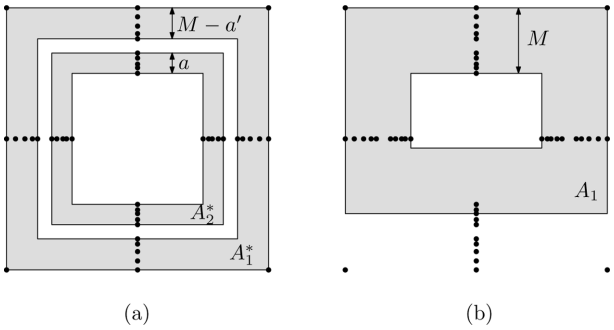

Fig. 7. (a) The nested pair (A_1^*, A_2^*) with $w(A_1^*) + w(A_2^*) = M - g(S)$. (b) For any vertically separated pair (A_1, A_2), the width of A_1 or of A_2 is at least M.

First, we show that

$$\min_i f_N(i) = M - g(S).$$

Consider any nested pair of two rectangular annuli that encloses P. Let $a < a' \in S$ be two consecutive elements in S that define the maximum gap, that is, $g(S) = a' - a$. Then, there exists a nested pair (A_1^*, A_2^*) of annuli such that: the outer rectangle of A_1^* is $R(P)$ and its inner rectangle is the offset of $R(P)$ whose boundary contains four points $(0, M + a')$, $(0, -M - a')$, $(M + a', 0)$, $(-M - a', 0) \in P$; the outer rectangle of A_2^* is the offset of $R(P)$ whose boundary contains four points $(0, M + a)$, $(0, -M - a)$, $(M + a, 0)$, $(-M - a, 0) \in P$ and its inner rectangle is defined by four points $(0, M)$, $(0, -M)$, $(M, 0)$, $(-M, 0) \in P$ (Fig. 7(a)). Then, the sum of widths is exactly

$$w(A) + w(A_2^*) = M - a' + a = M - g(S),$$

and it is not difficult to see that this is the minimum among $f_N(i)$ for all $0 \le i \le 4n + 4$.

Next, we argue that the nested pair (A_1^*, A_2^*) is indeed the optimal solution, that is,

$$\min_i \{f_H(i), f_V(i), f_N(i)\} = M - g(S)\}.$$

To see this, consider any vertically separated pair (A_1, A_2) of annuli enclosing P such that $A_1 = A(T_i)$ and $A_2 = A(\overline{T_i})$ for some i. Since the set P is symmetric due to our construction, we do not need to repeatedly discuss horizontally separated pairs. Again by the symmetric configuration of our construction P, we can assume that $i \ge 2n + 2$. So, T_i contains some points lying on the x-axis, and also contains two topmost points $(-2M, 2M)$, $(2M, 2M) \in P$. Hence, the outer rectangle $R(T_i)$ of A_1 has width $4M$ and height at least $2M$. Since T_i also contains point $(0, M) \in P$, the inner rectangle of A_1 must be placed below the point $(0, M)$ and all other points of the form $(0, M + a) \in P$ for all $a \in S$ (Fig. 7(b)). This implies that the width of A_1 is at least M, so we have

$$w(A_1) + w(A_2) \ge M \ge M - g(S),$$

as claimed.

Therefore, by Lemma 2, the nested pair (A_1^*, A_2^*) of two rectangular annuli is an optimal solution to the min-sum two rectangular annuli problem for P, and the linear-time reduction is complete. □

Since the min-sum problem is a special case of the general min-cost problem, we obtain the same lower bound as well.

COROLLARY 3. *Any algorithm for the min-cost two rectangular annuli problem requires* $\Omega(n \log n)$ *time in the algebraic decision tree model, when a cost function f is part of input*.

Note that the above reduction does not work for the min-max problem or for the unimodal case of the min-cost problem. We finish the paper by leaving the following open question: What is the true computational complexity of the min-max two rectangular annuli problem? Can one solve the min-max problem faster than $O(n \log n)$ time, or prove its matching lower bound $\Omega(n \log n)$?

CONFLICT OF INTEREST

The authors have declared that no competing interests exist.

ACKNOWLEDGEMENTS

This work was supported by Kyonggi University Research Grant 2023.

REFERENCES

1. A. D. Wainstein, "A non-monotonous placement problem in the plane," in *Software Systems for Solving Optimal Planning Problems: 9th All-Union Symposium Abstracts, Minsk, BSSR, USSR*, 1986, pp. 70-71.
2. H. Ebara, N. Fukuyama, H. Nakano, and Y. Nakanishi, "Roundness algorithms using the Voronoi diagrams," in *Abstracts of the 1st Canadian Conference on Computational Geometry (CCCG)*, Montreal, Canada, 1989, pp. 41-41.
3. U. Roy and X. Zhang, "Establishment of a pair of concentric circles with the minimum radial separation for assessing roundness error," *Computer-Aided Design*, vol. 24, no. 3, pp. 161-168, 1992. https://doi.org/10.1016/0010-4485(92)90035-9
4. P. K. Agarwal, M. Sharir, and S. Toledo, "Applications of parametric searching in geometric optimization," *Journal of Algorithms*, vol. 17, no. 3, pp. 292-318, 1994. https://doi.org/10.1006/jagm.1994.1038
5. P. K. Agarwal and M. Sharir, "Efficient randomized

algorithms for some geometric optimization problems," *Discrete & Computational Geometry*, vol. 16, pp. 317-337, 1996. https://doi.org/10.1007/BF02712871
6. P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, "Approximating extent measures of points," *Journal of the ACM (JACM)*, vol. 51, no. 4, pp. 606-635, 2004. https://doi.org/10.1145/1008731.1008736
7. T. M. Chan, "Approximating the diameter, width, smallest enclosing cylinder, and minimum-width annulus," *International Journal of Computational Geometry & Applications*, vol. 12, no. 1-2, pp. 67-85, 2002. https://doi.org/10.1142/S0218195902000748
8. M. Abellanas, F. Hurtado, C. Icking, L. Ma, B. Palop, and P. A. Ramos, "Best fitting rectangles," in *Proceedings of the 19th European Workshop on Computational Geometry (EuroCG)*, Bonn, Germany, 2003, pp. 1-18.
9. O. N. Gluchshenko, H. W. Hamacher, and A. Tamir, "An optimal O(nlogn) algorithm for finding an enclosing planar rectilinear annulus of minimum width," *Operations Research Letters*, vol. 37, no. 3, pp. 168-170, 2009. https://doi.org/10.1016/j.orl.2009.02.007
10. J. Mukherjee, P. R. S. Mahapatra, A. Karmakar, and S. Das, "Minimum-width rectangular annulus," *Theoretical Computer Science*, vol. 508, pp. 74-80, 2013. https://doi.org/10.1016/j.tcs.2012.02.041
11. S. W. Bae, "Computing a minimum-width square annulus in arbitrary orientation," *Theoretical Computer Science*, vol. 718, pp. 2-13, 2018. https://doi.org/10.1016/j.tcs.2016.11.010
12. S. W. Bae, "On the minimum-area rectangular and square annulus problem," *Computational Geometry*, vol. 92, article no. 101697, 2021. https://doi.org/10.1016/j.comgeo.2020.101697
13. S. W. Bae and S. D. Yoon, "Empty squares in arbitrary orientation among points," *Algorithmica*, vol. 85, no. 1, pp. 29-74, 2023. https://doi.org/10.1007/s00453-022-01002-1
14. S. W. Bae, "Computing a minimum-width square or rectangular annulus with outliers," *Computational Geometry*, vol. 76, pp. 33-45, 2019. https://doi.org/10.1016/j.comgeo.2018.08.002
15. H. K. Ahn, T. Ahn, S. W. Bae, J. Choi, M. Kim, E. Oh, C. S. Shin, and S. D. Yoon, "Minimum-width annulus with outliers: circular, square, and rectangular cases," *Information Processing Letters*, vol. 145, pp. 16-23, 2019. https://doi.org/10.1016/j.ipl.2019.01.004
16. S. W. Bae, "Minimum-width double-strip and parallelogram annulus," *Theoretical Computer Science*, vol. 833, pp. 133-146, 2020. https://doi.org/10.1016/j.tcs.2020.05.045
17. S. W. Bae, "Minimum-width parallelogram annulus with given angles," *Journal of Computing Science and Engineering*, vol. 15, no. 2, pp. 78-83, 2021. https://doi.org/10.5626/JCSE.2021.15.2.78
18. S. W. Bae, "On the minimum-area parallelogram annulus problem," *Symmetry*, vol. 14, no. 2, article no. 359, 2022. https://doi.org/10.3390/sym14020359
19. S. W. Bae, "Computing a minimum-width cubic and hypercubic shell," *Operations Research Letters*, vol. 47, no. 5, pp. 398-405, 2019. https://doi.org/10.1016/j.orl.2019.07.006
20. S. W. Bae, "An optimal algorithm for the minimum-width cubic shell problem," *Operations Research Letters*, vol. 51, no. 5, pp. 477-482, 2023. https://doi.org/10.1016/j.orl.2023.07.002
21. S. W. Bae, "Minimum-width cuboidal shells with outliers," *Journal of Computing Science and Engineering*, vol. 14, no. 1, pp. 1-8, 2000. https://doi.org/10.5626/JCSE.2020.14.1.1
22. M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest, and R. E. Tarjan, "Time bounds for selection," *J. Comput. Syst. Sci.*, vol. 7, no. 4, pp. 448-461, 1973. https://doi.org/10.1016/S0022-0000(73)80033-9
23. D. T. Lee, "Two-dimensional Voronoi diagrams in the Lp-metric," *Journal of the ACM (JACM)*, vol. 27, no. 4, pp. 604-618, 1980. https://doi.org/10.1145/322217.322219
24. B. Chazelle, "An algorithm for segment-dragging and its implementation," *Algorithmica*, vol. 3, pp. 205-221, 1988. https://doi.org/10.1007/BF01762115
25. F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction*. New York, NY: Springer, 1985.

Sang Won Bae https://orcid.org/0000-0002-8802-4247

Sang Won Bae received his Ph.D. in 2008 at Department of Computer Science, Korea Advanced Institute of Science and Technology, Daejeon, Korea. At present, he is working as a professor at Kyonggi University, Suwon, Korea. Research interests include algorithms design and analysis in computational geometry, discrete and combinatorial geometry, graph theory, and their algorithmic applications to other disciplines.