Algorithms for Halfplane Coverage and Related Problems

Haitao Wang
Kahlert School of Computing, University of Utah, Salt Lake City, UT, USA

Jie Xue
Department of Computer Science, New York University Shanghai, China

Abstract

Given in the plane a set of points and a set of halfplanes, we consider the problem of computing a smallest subset of halfplanes whose union covers all points. In this paper, we present an $O(n^{4/3}\log^{5/3} n\log^{O(1)}\log n)$-time algorithm for the problem, where n is the total number of all points and halfplanes. This improves the previously best algorithm of $n^{10/3}2^{O(\log^* n)}$ time by roughly a quadratic factor. For the special case where all halfplanes are lower ones, our algorithm runs in $O(n\log n)$ time, which improves the previously best algorithm of $n^{4/3}2^{O(\log^* n)}$ time and matches an $\Omega(n\log n)$ lower bound. Further, our techniques can be extended to solve a star-shaped polygon coverage problem in $O(n\log n)$ time, which in turn leads to an $O(n\log n)$-time algorithm for computing an instance-optimal ϵ-kernel of a set of n points in the plane. Agarwal and Har-Peled presented an $O(nk\log n)$-time algorithm for this problem in SoCG 2023, where k is the size of the ϵ-kernel; they also raised an open question whether the problem can be solved in $O(n\log n)$ time. Our result thus answers the open question affirmatively.

2012 ACM Subject Classification Theory of computation → Computational geometry; Theory of computation → Design and analysis of algorithms

Keywords and phrases halfplane coverage, circular coverage, star-shaped polygon coverage, ϵ-kernels

Digital Object Identifier 10.4230/LIPIcs.SoCG.2024.79

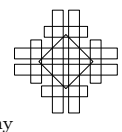
Related Version *Full Version*: https://arxiv.org/abs/2402.16323

Funding *Haitao Wang*: Supported in part by NSF under Grant CCF-2300356.

1 Introduction

Let P be a set of points and H a set of halfplanes in the plane. In the *halfplane coverage* problem, the goal is to find a smallest subset of halfplanes in H whose union covers all points in P. The problem was studied in the literature before. Har-Peled and and Lee [18] first proposed an $O(n^5)$-time algorithm for the problem, where $n = |P| + |H|$. Later, Pedersen and Wang [29] provided an improved algorithm of $O(n^4\log n)$ time. Very recently, Liu and Wang [24] solved the problem in $n^{10/3}2^{O(\log^* n)}$ time. In this paper, we present a new algorithm for the problem of $O(n^{4/3}\cdot\log^{5/3} n\log^{O(1)}\log n)$ time, which improves the best known bound [24] by roughly a quadratic factor.

In the *lower-only* version where all halfplanes in H are lower-open (i.e., of the form $y \le ax + b$), our new algorithm runs in $O(n\log n)$ time. The lower-only halfplane coverage problem has also been studied before. Chan and Grant's techniques [10] solved the problem in $O(n^4\log n)$ time. Pedersen and Wang [29] derived an $O(n^2\log n)$-time algorithm. The recent work of Liu and Wang [24] proposed an improved solution of $n^{4/3}2^{O(\log^* n)}$ time, which was the best known result prior to this paper. We also show that $\Omega(n\log n)$ is a lower bound for the problem under the algebraic decision tree model; therefore, our algorithm is optimal.



Interestingly, our new techniques for the halfplane coverage problem can also be applied to efficiently compute *instance-optimal* kernels in the plane. For a set Q of n points in the plane and a parameter $\epsilon \in (0, 1)$, a subset $Q' \subseteq Q$ is an ϵ-kernel of Q if the projection of the convex hull of Q' approximates that of Q within $(1 - \epsilon)$ factor in every direction. Given Q and ϵ, the problem is to compute a smallest ϵ-kernel for Q, called an *instance-optimal* ϵ-kernel. Very recently, Agarwal and Har-Peled [1] gave an $O(nk \log n)$-time algorithm for the problem in SoCG 2023, where k is the size of the instance-optimal ϵ-kernel. They raised as an open question whether the problem can be solved in $O(n \log n)$ time. We provide an $O(n \log n)$-time algorithm based on our techniques for halfplane coverage, and hence answer the open question affirmatively.

To see how the problem of computing instance-optimal ϵ-kernels is related to halfplane coverage, we recall the approach of Agarwal and Har-Peled [1]. In this approach, the task of finding instance-optimal ϵ-kernel is first reduced in $O(n \log n)$ time to the following *star-shaped polygon coverage* problem. Given a star-shaped polygon R of n vertices with respect to a center point o (i.e., the segment $\overline{op} \subseteq R$ for every point $p \in R$) and a set of n halfplanes that do not contain o, the problem is to compute a smallest subset of halfplanes whose union covers the boundary of R (after the problem reduction, the size of a smallest subset is exactly equal to k, the size of an instance-optimal ϵ-kernel in the original problem). Agarwal and Har-Peled [1] then solved the star-shaped polygon coverage problem in $O(nk \log n)$ time. By extending our techniques for the halfplane coverage problem, we present an improved algorithm of $O(n \log n)$ time for the star-shaped polygon coverage problem, which in turn solves the problem of computing instance-optimal ϵ-kernels in $O(n \log n)$ time. In addition, we prove that $\Omega(n \log n)$ is a lower bound for the star-shaped polygon coverage problem under the algebraic decision tree model, and thus our algorithm is optimal.

Besides computing kernels, our techniques may find other applications as well. One remarkable example is the computation of Hausdorff approximation [1, 19].

- Agarwal and Har-Peled [1] proposed an $O(nk \log n)$ time algorithm for computing an optimal Hausdorff approximation for a set of n points in the plane, where k is the size of the optimal solution. In this algorithm, they followed the similar techniques to the above instance-optimal ϵ-kernel problem. Specifically, they reduced the Hausdorff approximation problem to a problem of covering the boundary of a star-shaped region Q of curved boundary in the plane. Using our new techniques, the optimal Hausdorff approximation problem can now also be solved in $O(n \log n)$ time.
- Har-Peled and Raichel [19] considered a "dual" Hausdorff approximation problem for a set of n points in the plane and a parameter $k \in \{1, \ldots, n\}$. They gave a randomized algorithm whose runtime is bounded by $O(\sqrt{k}(n \log n)^{3/2} + kn \log^2 n)$ with high probability. Their algorithm utilizes the above $O(nk \log n)$-time algorithm in [1] for the optimal Hausdorff approximation problem as a black box. Using our new $O(n \log n)$-time algorithm instead and follow the same analysis as in [19], the dual Hausdorff approximation problem can now be solved in $O((n \log n)^{3/2})$ time with high probability.

1.1 Other related work

The halfplane coverage problem belongs to the domain of geometric set cover, which in turn is a special case of the (general) set cover problem. For most types of geometric objects, the geometric set cover problem is NP-hard. One example that has received much attention in the literature is the disk coverage problem, i.e., given a set of disks and a set of points in the plane, the problem is to find a smallest subset of disks that together

cover all points. The problem is NP-hard even if all disks have the same radius [16], while many approximation algorithms have been proposed [22, 27, 28]. On the other hand, the problem becomes polynomial-time solvable if all disk centers lie on the same line or if the disk centers and the points are separated by a line [5, 9, 13, 24, 29]. Another well-studied case of geometric set cover is the rectangle coverage problem where the given geometric objects are (axis-parallel) rectangles. This problem is also NP-hard, even when the rectangles are slabs, unit squares, or with certain constraints [10, 17]. Agarwal and Pan [3] proposed an $O(\log \log k)$-approximation algorithm for rectangle coverage with near-linear running time, where k is the optimum. Finally, geometric set cover with other objective functions have also been studied, e.g., minimizing the membership or the ply of the solution [6, 8, 14, 26].

For the ϵ-kernel problem, Agarwal, Har-Peled, and Varadarajan proved in their seminal paper [2] that an ϵ-kernel of size $O(\epsilon^{-(d-1)/2})$ exists for a set of n points in the d-dimensional space $\mathbb{R}^d$. However, an ϵ-kernel could be much smaller in practice [31]. Therefore, it is well-motivated to study algorithms for computing instance-optimal ϵ-kernels. The problem unfortunately becomes NP-hard in $\mathbb{R}^3$ [1]. Agarwal and Har-Peled [1] thus studied the exact algorithms for the 2D case.

1.2 Our approach

For the lower-only halfplane coverage problem, we show that the problem can be reduced to an *interval coverage* problem: Given a set P' of points and a set S of intervals on the x-axis, compute a smallest subset of intervals whose union covers all points. The problem can be easily solved in $O(|P'| + |S|)$ time by a greedy algorithm after sorting. However, the issue with this approach is that while $|P'| = n$, we may have $|S| = \Omega(n^2)$ after the problem reduction. More specifically, points of P' are defined by points of P and intervals of S are defined by halfplances of H. While each point of P defines a single point in P' (thus $|P'| = n$), each halfplane could define $\Theta(n)$ intervals of S (thus $|S| = \Omega(n^2)$). As such, the total time of the algorithm could be $\Omega(n^2)$. To improve the algorithm, our crucial observation is that it suffices to use one particular interval of S for each halfplane. Consequently, we only need to use a subset $\widehat{S} \subseteq S$ of size at most n such that a smallest subset of $\widehat{S}$ for covering P' is also an optimal solution for covering P' with S. In this way, the lower-only halfplane coverage problem is solved in $O(n \log n)$ time.

For the star-shaped polygon coverage, we extend the above idea. We reduce it to a *circle coverage* problem: Given a set S of arcs on a circle C, compute a smallest subset of arcs whose union covers the entire circle C. The problem can be solved in $O(|S| \log |S|)$ time [1, 21]. As above, the issue is that $|S|$ could be $\Omega(n^2)$ (more specifically, arcs of S are defined by halfplanes of H and each halfplane may define $\Theta(n)$ arcs). Indeed, this is the main obstacle that prevents Agarwal and Har-Peled [1] from obtaining an $O(n \log n)$ time algorithm. Our new and crucial observation is that, as above, it is sufficient to use only one particular arc for each halfplane. Consequently, we only need to use a subset $\widehat{S} \subseteq S$ of size at most n such that a smallest subset of arcs of $\widehat{S}$ covering C is also an optimal solution for covering C with S. In this way, the star-shaped polygon coverage problem is solved in $O(n \log n)$ time.

To solve the general halfplane coverage problem, we consider two cases depending on whether H has three halfplanes whose union is $\mathbb{R}^2$. In the case where H does not have such three halfplanes, by Helly's theorem, the common intersection of the complements of all halfplanes of H is not empty; let o be a point in the common intersection. Then, no halfplane of H contains o. With the help of o, we can compute in $O(n \log n)$ time a smallest subset of halfplanes to cover all points, by extending the idea for solving the lower-only halfplane coverage problem. In the case where H has three halfplanes whose union is $\mathbb{R}^2$, the optimal

solution size τ^* is at most three. If $\tau^* = 3$, then it suffices to find three halfplanes from H whose union is $\mathbb{R}^2$, which can be done in linear time [18]. If $\tau^* = 1$, then this case can be easily solved in $O(n \log n)$ time using halfplane range emptiness queries. For the remaining case $\tau^* = 2$, we wish to find two halfplanes from H whose union covers all points of P. Although this looks like a special case, it turns out this is the bottleneck case of the entire problem, which is surprising (and perhaps also interesting). Our algorithm for this case runs in $O(n^{4/3} \log^{5/3} n \log^{O(1)} \log n)$ time, which takes significantly more time than all other parts of the overall algorithm (all other parts together take $O(n \log n)$ time). Although we do not have a proof, we feel that $\Omega(n^{4/3})$ might be a lower bound for this subproblem (and thus the entire problem), at least under a somewhat restricted computational model [15].

Remark. After the submission of this paper, Liu and Wang [23] gave a new implementation of their original algorithm in [24] for the lower-only halfplane coverage problem; the new implementation runs in $O(n \log n)$ time. Although their algorithm also reduces the problem to an interval coverage problem, it is fundamentally different from ours. For instance, their algorithm first identifies some "prunable" halfplanes that are useless to the optimal solution and then proceed to reduce to the interval coverage problem using the remaining halfplanes. Our algorithm, in contrast, does not need such a pruning step. In addition, it seems difficult to extend their algorithm to the circle coverage or polygon coverage problem because it relies on the left-to-right order of all points. As such, our techniques appear to be more powerful. On the other hand, their approach can be used to solve a more general line-separable unit-disk coverage problem (the lower-only halfplane coverage is just a special case of the problem). Also, using their $O(n \log n)$ time new implementation for the lower-only halfplane coverage, their algorithm for the general halfplane coverage now runs in $O(n^3 \log n)$ time.

Outline. The rest of the paper is organized as follows. In Section 2, we present our algorithm for the lower-only halfplane coverage problem. By extending the techniques, we solve the star-shaped polygon coverage problem in Section 3. As mentioned above, with the $O(n \log n)$-time problem reduction in [1], this also solves the 2D instance-optimal ϵ-kernel problem in $O(n \log n)$ time. The general halfplane coverage problem is discussed in Section 4 and our approach uses an algorithm similar to the lower-only case algorithm as a subroutine. Due to the space limit, many proofs and details are omitted but can be found in the full paper.

2 Lower-only halfplane coverage

In this section, we present an $O(n \log n)$-time algorithm for the lower-only halfplane coverage problem. Let P be a set of points and H a set of lower halfplanes in $\mathbb{R}^2$. We wish to compute a smallest subset of halfplanes whose union covers P (i.e., covers all points of P). To simplify the notation, let $n = |P| = |H|$.

2.1 Preliminaries

We call a subset of H a *feasible solution* if the halfplanes of the subset together cover P; an *optimal solution* refers to a smallest feasible solution. We assume that the union of halfplanes of H covers P since otherwise there would be no feasible solution. Indeed, this can be easily determined $O(n \log n)$ time, e.g., by first computing the upper envelope of the bounding lines of all halfplanes of H and then check whether every point of P is below the upper envelope.

We make a general position assumption that no two points of P have the same x-coordinate. We also assume that no two halfplanes of H have their bounding lines parallel (since otherwise the one with lower bounding line is redundant and can be removed from H because it is completely contained in the other halfplane).

We sort all points of P from left to right and let $p_1, p_2, \ldots, p_n$ be the sorted list. For any two indices i, j with $1 \leq i \leq j \leq n$, let $P[i, j]$ denote the subsequence of points $\{p_i, p_{i+1}, \ldots, p_j\}$.

For any halfplane $h \in H$, let ℓ_h denote the bounding line of h.

Consider a halfplane $h \in H$. A subsequence $P[i, j]$ of P, with $1 \leq i \leq j \leq n$, is called a *maximal subsequence covered by* h if all points of $P[i, j]$ are in h but neither p_{i-1} nor p_{j+1} is in h (to make the definition rigorous, we could add two dummy points p_0 and p_{n+1} that are not covered by any halfplane of H). Let Γ_h denote the set of all maximal subsequences of P covered by h. It is not difficult to see that subsquences of Γ_h are pairwise disjoint.

2.2 Reducing to interval coverage

We reduce the problem to an instance of the interval coverage problem on a set P' of points and a set S of line segments (or intervals) on the x-axis. For each point $p_i \in P$, let p'_i be the (vertical) projection of p_i onto the x-axis (i.e., p'_i has the same x-coordinate as p_i). Define $P' = \{p'_i \mid 1 \leq i \leq n\}$. As such, P' has exactly n points. For any $1 \leq i \leq j \leq n$, let $P'[i, j] = \{p'_i, p'_{i+1}, \ldots, p'_j\}$.

We now define the set S. For each halfplane $h \in H$, we define a set S_h of segments as follows. For each subsequence $P[i, j] \in \Gamma_h$, we create a segment on the x-axis, denoted by $s[i, j]$, whose left and right endpoints are p'_i and p'_j, respectively; we add $s[i, j]$ to S_h. As such, $P' \cap s[i, j] = P'[i, j]$. We say that $s[i, j]$ is defined by h. Define $S = \bigcup_{h \in H} S_h$.

Consider the following interval coverage problem on P' and S: Find a smallest subset of segments of S whose union covers P'. This problem can be solved by a simple greedy algorithm in $O(|P'| + |S|)$ time after sorting the left endpoints of all segments of S along with all points of P'. Let $S^* \subseteq S$ be an optimal solution to the above interval coverage problem. Based on S^*, we create a subset $H^* \subseteq H$ as follows. For each segment $s \in S^*$, if s is defined by a halfplane $h \in H$, then we add h to H^* (it is possible that the same segment s is defined by multiple halfplanes, in which case we add an arbitrary such halfplane to H^*). In what follows, we show that H^* is an optimal solution to our original halfplane coverage problem for P and H. To this end, we first have the following lemma.

▶ **Lemma 1.**

1. *The union of all halfplanes of H^* covers P.*
2. *P' can be covered by k segments of S if and only if P can be covered by k halfplanes of H.*

The above lemma leads to the following corollary.

▶ **Corollary 2.**

1. *The size of a smallest subset of H for covering P is equal to the size of a smallest subset of S for covering P'.*
2. *No two segments of S^* are defined by the same halfplane.*
3. *H^* is a smallest subset of H for covering P.*

In light of Corollary 2, the above gives an algorithm that computes an optimal solution to the halfplane coverage problem for P and H. However, the algorithm is not efficient because the size of S could be $\Omega(n^2)$. Indeed, it is not difficult to see that $|\Gamma_h|$ (and thus $|S_h|$) could be $\Theta(n)$ for each halfplane $h \in H$. Hence, $|S|$ could be $\Omega(n^2)$ in the worst case. In the following, we reduce the time to $O(n \log n)$ by showing that a smallest subset of S for covering P' can be computed in $O(n \log n)$ time by using only a small subset of S.

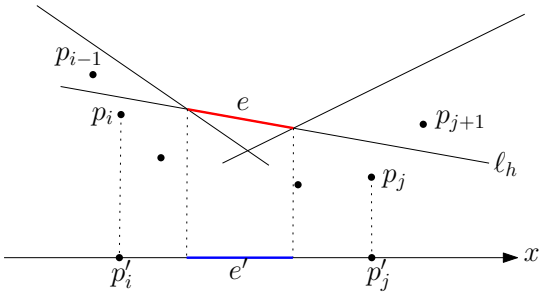


Figure 1 Illustrating the definition of $s(h) = s[i, j]$ for the case where ℓ_h contains an edge e of $\mathcal{U}$.

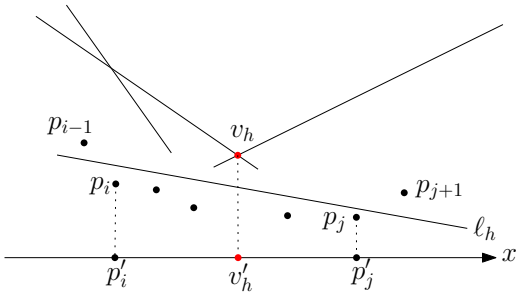


Figure 2 Illustrating the definition of $s(h) = s[i, j]$ when ℓ_h does not contain any edge of $\mathcal{U}$.

2.3 Improvement

The main idea is to show that it suffices to use at most one segment from S_h for each $h \in H$. More specifically, we show that for each halfplane $h \in H$, it is sufficient to define a segment, denoted by $s(h)$, for at most one subsequence of Γ_h, such that $\widehat{S} = \{s(h) \mid h \in H\}$, which is of size at most n and is a subset of S, must contain a smallest subset of S for covering P'. This implies that to compute a smallest subset of S for covering P', it suffices to compute a smallest subset of $\widehat{S}$ to cover P'. The latter problem can be solved faster as $|\widehat{S}| \leq n$.

Next, we first define the segment $s(h)$ for each $h \in H$. Then, we prove that $\widehat{S}$ contains a smallest subset of S for covering P'. Finally, we discuss how to compute $\widehat{S}$ efficiently.

Defining $s(h)$ and $\widehat{S}$. For each halfplane $h \in H$, we define a segment $s(h)$ as follows. Let $\mathcal{U}$ denote the upper envelope of the bounding lines of all halfplanes of H. Depending on whether the bounding line ℓ_h of h contains an edge on $\mathcal{U}$, there are two cases.

1. If ℓ_h contains an edge e on $\mathcal{U}$, then let e' be the vertical projection of e on the x-axis (see Fig. 1). Since e is on $\mathcal{U}$, no point of P is vertically above e. Hence, e' is contained in at most one segment $s[i, j]$ of S_h (note that such a segment may not exist, e.g., if e' is between p'_i and p'_{i+1} and neither point is in h). If such a segment $s[i, j]$ exists, then we define $s(h) = s[i, j]$; otherwise, $s(h)$ is not defined.
2. If ℓ_h does not contain any edge on $\mathcal{U}$, then let v_h be the unique vertex of $\mathcal{U}$ that has a tangent line parallel to ℓ_h (see Fig. 2). Let v'_h be the vertical projection of v_h onto the x-axis. Clearly, S_h has at most one segment $s[i, j]$ containing v'_h. If such a segment $s[i, j]$ exists, then we define $s(h) = s[i, j]$; otherwise, $s(h)$ is not defined.

Define $\widehat{S} = \{s(h) \mid h \in H\}$.

The following crucial lemma implies that a smallest subset of segments of $\widehat{S}$ whose union covers P' is an optimal solution to the interval coverage problem for P' and S.

▶ **Lemma 3.** *For any segment $s \in S \setminus \widehat{S}$, $\widehat{S}$ has a segment s' containing s.*

With Lemma 3, the lower-only halfplane coverage problem on P and H can be solved as follows. (1) Compute $\widehat{S}$. (2) Compute a smallest subset $\widehat{S}^*$ of segments of $\widehat{S}$ whose union covers P'. (3) Using $\widehat{S}^*$, obtain an optimal solution for P and H (in the same way as described in Section 2.2). Since $|\widehat{S}| \leq n$ and $|P'| = n$, the second and third steps can be implemented in $O(n \log n)$ time. Lemma 4 shows that the first step can be done in $O(n \log n)$ time too, by making use of ray-shooting queries in simple polygons [11, 12, 20]. We thus conclude with Theorem 5.

▶ **Lemma 4.** *Computing all segments of $\widehat{S}$ can be done in $O(n \log n)$ time.*

▶ **Theorem 5.** *Given in the plane a set of points and a set of lower halfplanes, one can compute a smallest subset of halfplanes whose union covers all points in $O(n \log n)$ time, where n is the total number of all points and halfplanes.*

Lower bound. Theorem 6 proves a lower bound even for certain special cases, by reductions from set equality and set inclusion problems [7].

▶ **Theorem 6.** *It requires $\Omega(n \log n)$ time to solve the lower halfplane coverage problem under the algebraic decision tree model, even if all points are given sorted by x-coordinates or if the bounding lines of all halfplanes are given sorted by their slopes.*

3 Star-shaped polygon coverage and 2D instance-optimal ϵ-kernels

In this section, we solve the star-shaped polygon coverage problem. Let $\mathcal{P}$ be a star-shaped polygon with respect to a center point o, i.e., the line segment $\overline{op} \subseteq \mathcal{P}$ for any point $p \in \mathcal{P}$. Let n be the number of vertices of $\mathcal{P}$. Let H be a set of n halfplanes that do not contain o. The goal is to compute a smallest subset of halfplanes whose union covers $\partial\mathcal{P}$, the boundary of $\mathcal{P}$. We present an $O(n \log n)$ time algorithm for the problem.

For any halfplane h, denote by $\overline{h}$ the complement halfplane of h. Define $\overline{H} = \{\overline{h} \mid h \in H\}$.

As in Section 2, we assume that no two halfplanes of H have their bounding lines parallel to each other. We also assume that H has a feasible solution (i.e., H has a subset of halfplanes whose union covers ∂P). Indeed, this can be determined in $O(n \log n)$ time as follows. We first compute the common intersection $\mathcal{U}$ of the halfplanes of $\overline{H}$, which must contain o. $\mathcal{U}$ can be computed in $O(n \log n)$ time. Notice that H has a feasible solution if and only if the interior of $\mathcal{U}$ does not intersect $\partial\mathcal{P}$ [1]. Since $\mathcal{P}$ is star-shaped with respect to o, whether the interior of $\mathcal{U}$ intersects $\partial\mathcal{P}$ can be determined in $O(n)$ by rotating a sweeping ray round o. As such, in total $O(n \log n)$ time, one can determine whether H has a feasible solution. In the following, we focus on finding an optimal solution.

Let C be a circle centered at o and containing $\mathcal{P}$. For any point p inside C with $p \neq o$, define its *projection point on C* as the intersection between C and the ray from o to p.

3.1 Reducing to circle coverage

Following a similar idea to Section 2, we reduce the problem to a *circle coverage* problem on C: Given a set of circular arcs on C, find a smallest subset of arcs that together cover C.

Consider a halfplane h. Recall that ℓ_h denote its bounding line. Let Γ_h denote the set of the maximal segments of $\mathcal{P} \cap \ell_h$ (intuitively Γ_h plays a similar role to the same notation in Section 2). For each segment $s \in \Gamma_h$, it "blocks" the visibility of a portion of $\partial\mathcal{P}$ from o,

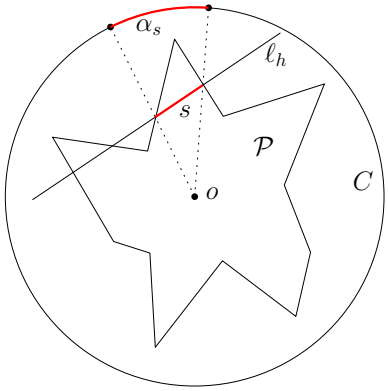


Figure 3 Illustrating the definition of an arc α_s.

and it also blocks the visibility of an arc of C from o, denoted by α_s (see Fig. 3). The two endpoints of α_s are exactly the projections of the two endpoints of s on C, respectively. Let S_h denote the set of arcs defined by the segments of Γ_h. Define $S = \bigcup_{h \in H} S_h$.

Now consider the circle coverage problem for S: Compute a smallest subset of arcs of S whose union covers C. To solve the problem, Lee and Lee [21] first gave an $O(|S| \log |S|)$ time algorithm, and Agarwal and Har-Peled [1] presented a much simpler solution with the same runtime. Suppose we have an optimal solution S^*. We create a subset H^* of H as follows. For each arc α of S^*, if h is the halfplance that defines α, then we include h in H^*. Lemma 7, analogous to Lemma 1 in Section 2, has been proved by Agarwal and Har-Peled [1].

▶ **Lemma 7** (Agarwal and Har-Peled [1]).
1. *The union of all halfplanes H^* covers $\partial\mathcal{P}$.*
2. *C can be covered by k arcs of S if and only if $\partial\mathcal{P}$ can be covered by k halfplanes of H.*

With Lemma 7, we can obtain results analogous to Corollary 2. In particular, H^* is a smallest subset of H for covering $\partial\mathcal{P}$. As such, the above gives an algorithm for computing a smallest subset of H to cover $\partial\mathcal{P}$. However, the algorithm is not efficient because the size of S could be $\Omega(n^2)$ (since $|\Gamma_h|$ and thus $|S_h|$ could be $\Theta(n)$ for each halfplane $h \in H$). In the following, we reduce the time to $O(n \log n)$ by showing that a smallest subset of S for covering $\partial\mathcal{P}$ can be computed in $O(n \log n)$ time by using only a small subset of S.

3.2 Improvement

As in Section 2, we will define a subset $\widehat{S} \subseteq S$ such that $\widehat{S}$ contains at most one arc $\alpha(h)$ defined by each halfplane $h \in H$ (and thus $|\widehat{S}| \leq n$) and $\widehat{S}$ contains a smallest subset of S for covering C. Further, we will show that $\widehat{S}$ can be computed in $O(n \log n)$ time. Consequently, applying the circle coverage algorithm [1, 21] can solve the problem in $O(n \log n)$ time.

Defining $\alpha(h)$ and $\widehat{S}$. For each halfplane $h \in H$, we define an arc $\alpha(h)$ on C as follows. As will be seen, $\alpha(h)$ is in S_h and thus is in S. Let $\mathcal{U}$ be the common intersection of all halfplanes of $\overline{H}$. As discussed before, $\mathcal{U}$ must be inside $\mathcal{P}$ since H has a feasible solution.

We represent a direction in $\mathbb{R}^2$ by a unit vector. Let $\mathbb{S}$ denote the set of all unit vectors (directions) in $\mathbb{R}^2$, i.e., $\mathbb{S} = \{v : \|v\| = 1\}$.

For each halfplane h, we define the *norm* of its bounding line ℓ_h as the direction perpendicular to ℓ_h and towards the interior of h. For each edge e of $\mathcal{U}$, it is contained in ℓ_h for some halfplane $h \in H$; the norm of e refers to the norm of ℓ_h.

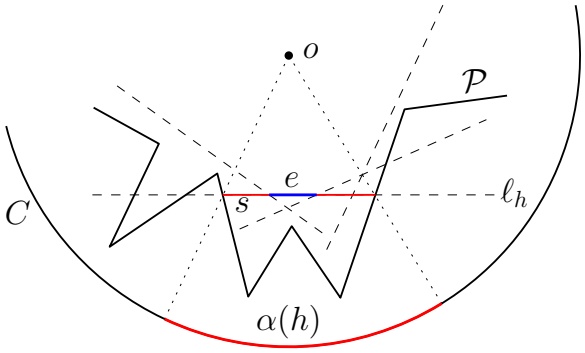


Figure 4 Illustrating the definition of $\alpha(h)$ when ℓ_h contains an edge e of $\mathcal{U}$. The dashed lines are halfplane bounding lines. The solid polyline is part of $\partial\mathcal{P}$. The red arc on the circle C is $\alpha(h)$.

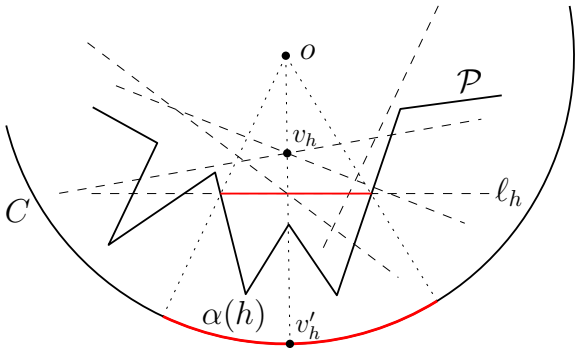


Figure 5 Illustrating $\alpha(h)$ (the red arc on circle C) when ℓ_h does not contain any edge of $\mathcal{U}$.

The norms of all edges of $\mathcal{U}$ partition $\mathbb{S}$ into $|\mathcal{U}|$ *basic intervals* such that the interior of each interval does not contain the norm of any edge of $\mathcal{U}$, where $|\mathcal{U}|$ is the number of edges of $\mathcal{U}$. Note that the endpoints of every basic interval are norms of two adjacent edges of $\mathcal{U}$.

To define $\alpha(h)$, depending on whether the bounding line ℓ_h of h contains an edge of $\mathcal{U}$, there are two cases.

1. If ℓ_h contains an edge e of $\mathcal{U}$, then since $\mathcal{U}$ is inside $\mathcal{P}$, e must be contained in a segment s of Γ_h. Let $\alpha(h)$ be the arc of C defined by s (see Fig. 4).
2. If ℓ_h does not contain any edge of $\mathcal{U}$, then let I be the basic interval of $\mathbb{S}$ that contains the norm of h. Let e_1 and e_2 be the two adjacent edges of $\mathcal{U}$ whose norms are endpoints of I. Define v_h to be the vertex of $\mathcal{U}$ incident to both e_1 and e_2 (Fig. 5). Let v'_h be the projection of v_h on C. Define $\alpha(h)$ to be the unique arc (if exists) of S_h containing v'_h.

Define $\widehat{S} = \{s(h) \mid h \in H\}$.

The following crucial lemma implies that a smallest subset of arcs of $\widehat{S}$ whose union covers C is also a smallest subset of arcs of S covering C.

▶ **Lemma 8.** *For any arc $\alpha \in S \setminus \widehat{S}$, $\widehat{S}$ must have an arc α' such that $\alpha \subseteq \alpha'$.*

With Lemma 8, a smallest subset of H for covering $\partial\mathcal{P}$ can be computed as follows. (1) Compute $\widehat{S}$. (2) Compute a smallest subset $\widehat{S}^*$ of $\widehat{S}$ for covering C. (3) Using $\widehat{S}^*$, obtain a smallest subset of H to cover $\partial\mathcal{P}$. Since $|\widehat{S}| \leq n$, the second and third steps can be done in $O(n \log n)$ time using the circle coverage algorithm in [1, 21]. Lemma 9 shows that the first step can be done in $O(n \log n)$ time too, using the ray-shooting queries in simple polygons [11, 12, 20]. As such, we conclude with Theorem 10.

▶ **Lemma 9.** *Computing all arcs of $\widehat{S}$ can be done in $O(n \log n)$ time.*

▶ **Theorem 10.** *The star-shaped polygon coverage problem is solvable in $O(n \log n)$ time, where n is the sum of the number of vertices of the polygon and the number of halfplanes.*

Computing instance-optimal ϵ-kernels in $\mathbb{R}^2$. As discussed in Section 1, computing an instance-optimal ϵ-kernel for a set of n points in the plane can be reduced in $O(n \log n)$ time to an instance of the star-shaped polygon coverage problem with n halfplanes and a star-shaped polygon of n vertices. Consequently, with our algorithm for the star-shaped polygon coverage problem, an instance-optimal ϵ-kernel can be computed in $O(n \log n)$ time.

Covering an x-monotone polyline. A special case of the star-shaped polygon coverage is as follows. Given in the plane an x-monotone polyline $\mathcal{P}$ of n vertices and a set H of n lower halfplances, we aim to compute a smallest subset of halfplanes so that their union covers $\mathcal{P}$. If we consider a point o with y-coordinate at $+\infty$, then the problem becomes a special case of the star-shaped polygon coverage problem and thus can be solved in $O(n \log n)$ time.

Lower bound. We finally have the lower bound in Theorem 11 by reduction from a problem in Theorem 6, which justifies the optimality of Theorem 10.

▶ **Theorem 11.** *Solving the star-shaped polygon coverage problem requires $\Omega(n \log n)$ time under the algebraic decision tree model.*

4 The general halfplane coverage

In this section, we consider the general halfplane coverage problem. Given in the plane a set P of n points and a set H of n halfplanes, the goal is to compute a smallest subset of halfplanes so that their union covers all points of P. Note that H may have both upper and lower halfplanes. We present an $O(n^{4/3} \log^{5/3} n \log \log^{O(1)})$ time algorithm for the problem.

For a halfplane h, denote by $\overline{h}$ the complement halfplane of h. Let $\overline{H} = \{\overline{h} \mid h \in H\}$.

We first determine whether the union of all halfplanes of H is the entire plane. This can be done by computing the common intersection $\mathcal{U}$ of all halfplanes of $\overline{H}$, which can be done in $O(n \log n)$ time. Notice that $\mathcal{U} = \emptyset$ if and only if the union of all halfplanes of H is the entire plane. Depending on whether $\mathcal{U} = \emptyset$, our algorithm will proceed in different ways. Below, we first discuss the case $\mathcal{U} \neq \emptyset$ in Section 4.1 and the other case is solved in Section 4.2.

4.1 The case $\mathcal{U} \neq \emptyset$

Assuming that $\mathcal{U} \neq \emptyset$, we solve this case in $O(n \log n)$ by an algorithm similar to those in the previous two sections. One may consider it a *cyclic version* of the lower-only halfplane coverage algorithm in Section 2 or a *point version* of the start-shaped polygon coverage algorithm in Section 3. Since $\mathcal{U} \neq \emptyset$, let o be any point inside $\mathcal{U}$. Then, no halfplanes of H cover o. Let C be a circle containing o and all points of P.

We reduce the problem to a *circular-point coverage problem*: Given on C a set P' of points and a set S of arcs, the goal is to compute a subset of arcs whose union covers all points. This problem is different from the circle coverage problem in Section 3 in that here we only need to cover points of P' instead of the entire circle C. To solve this new problem, we will show that the problem can be reduced to the circle coverage problem and consequently applying the circle coverage algorithm [1, 21] can solve the problem.

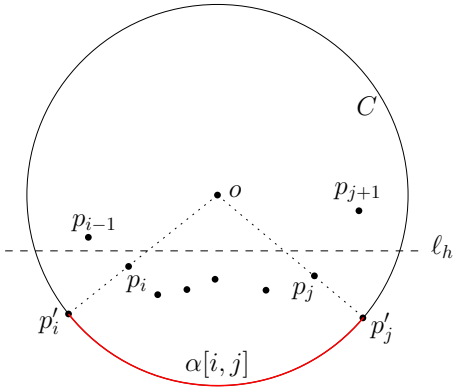


Figure 6 Illustrating the definition of arcs $\alpha[i,j]$.

4.1.1 Reducing to the circular-point coverage problem

We order the points of P counterclockwise around o and let $p_1, p_2, \ldots, p_n$ be the ordered list. For any point p in C with $p \neq o$, we define its projection point on C as the intersection between C and the ray from o to p. For each point $p_i \in P$, let p'_i be its projection on C. Hence, the points $p'_1, p'_2, \ldots, p'_n$ are ordered on C counterclockwise. Let P' be the set of all these projection points.

We consider P' as a cyclic sequence of points. Each point $p'_i \in P'$ refers to p'_j with $j = i \bmod n$ if $i > n$. For two indices i, j, we use $P'[i,j]$ to denote the subsequence of points $p'_i, p'_{i+1}, \ldots, p'_j$. We define the same notation for the points of P.

For each halfplane $h \in H$, we define a set S_h of arcs on C as follows. A subsequence $P[i,j]$ of P is called a *maximal subsequence covered by* h if all points of $P[i,j]$ are in h but neither p_{i-1} or p_{j+1} is in h. Let Γ_h denote the set of all maximal subsequences of P covered by h. For each subsequence $P[i,j]$ of Γ_h, we define an arc $\alpha[i,j]$ on C starting from p'_i counterclockwise until p'_j (see Fig. 6). Let S_h be the set of all arcs thus defined by the subsequences of Γ_h. Define $S = \bigcup_{h \in H} S_h$.

Now consider the circular-point coverage problem for S and P' on C: Compute a smallest subset of arcs of S whose union covers P'. We show in the full paper that the problem can be solved in $O((|S| + |P'|)\log(|S| + |P'|))$ time. Suppose we have an optimal solution S^*. We create a subset H^* of H as follows. For each arc α of S^*, if α is defined by a halfplane $h \in H$, then we add h to H^*. We have the following lemma, analogous to Lemmas 1 and 7.

▶ **Lemma 12.**

1. *The union of all halfplanes of H^* covers P.*
2. *P' can be covered by k arcs of S if and only if P can be covered by k halfplanes of H.*

With Lemma 12, we can obtain results analogous to Corollary 2. In particular, H^* is an optimal solution of H for covering P.

The above gives an algorithm for computing a smallest subset of H for covering P. However, the algorithm is not efficient because $|S|$ could be $\Omega(n^2)$ (since $|\Gamma_h|$ and thus $|S_h|$ could be $\Theta(n)$ for each halfplane $h \in H$). As for the lower-only halfplane coverage problem in Section 2, we can reduce the time to $O(n \log n)$ by showing that a smallest subset of S for covering P' can be computed in $O(n \log n)$ time by using only a small subset of S. More specifically, we can define a subset $\widehat{S} \subseteq S$ such that $\widehat{S}$ contains at most one arc $\alpha(h)$ defined by each halfplane $h \in H$ (and thus $|\widehat{S}| \leq n$) and $\widehat{S}$ contains a smallest subset of S for covering P'. Further, we can show that $\widehat{S}$ can be computed in $O(n \log n)$ time.

4.2 The case $\mathcal{U} = \emptyset$

We now consider the case $\mathcal{U} = \emptyset$. By Helly's theorem, there are three halfplanes in $\overline{H}$ whose common intersection is $\emptyset$. This means that there are three halfplanes in H whose union is the entire plane and thus covers all points of P. As such, the size τ^* of the smallest subset of H for covering P is at most three. Depending on whether τ^* is one, two, or three, there are three subcases.

If $\tau^* = 3$, then since H has three halfplanes whose union is $\mathbb{R}^2$, it suffices to find such three halfplanes. As discussed in [18] (see Lemma 4.1 [18]), this problem can be solved in $O(n)$ time using the linear-time linear programming algorithm [25].

If $\tau^* = 1$, then the problem becomes determining whether H has a halfplane containing all points of P, or alternatively, determining whether $\overline{H}$ has a halfplane that does not contain any point of P. For each halfplane $\overline{h} \in \overline{H}$, determining whether $\overline{h} \cap P = \emptyset$ can be easily done in $O(\log n)$ time by a halfplane range emptyness query, after constructing a convex hull of P. Therefore, the problem in this subcase can be solved in $O(n \log n)$ time.

In what follows, we discuss the subcase $\tau^* = 2$. Our goal is to find two halfplanes from H such that their union covers all points of P. In the following, we present an $O(n^{4/3} \log^{5/3} n \log^{O(1)} \log n)$-time algorithm for this problem. It turns out that the runtime for solving this case dominates the algorithm for the overall problem, which is surprising (and perhaps also interesting) because it means that this "special" case actually exhibits the difficulty of the general halfplane coverage problem. As discussed in Section 1, although we do not have a proof, we feel that $\Omega(n^{4/3})$ might be a lower bound for this problem, at least under a partition model [15].

4.2.1 Algorithm for the subcase $\tau^* = 2$

Our algorithm is based on a modification (and simplification) of Agarwal, Sharir, and Welzl's algorithm [4] (referred to as the ASW algorithm) for the decision version of the discrete two-center problem: Given a set Q of n points in $\mathbb{R}^2$ and a parameter r, determine whether there are two congruent disks centered at two points of Q with radius r that are together cover all points of Q. The ASW algorithm runs in $O(n^{4/3} \log^4 n)$ time, which was recently improved to $O(n^{4/3} \log^{7/3} n \log^{O(1)} \log n)$ by Wang [30]. Wang's algorithm follows the same idea as the ASW algorithm but uses a more efficient data structure developed in [30]. In the following, we modify the ASW algorithm to solve our problem.

Assume that H has two halfplanes h_1^* and h_2^* whose union covers P. If both of them are lower or upper halfplanes, then we can apply our lower-only halfplane coverage algorithm in Section 2 to find them in $O(n \log n)$ time. As such, we assume that h_1^* is a lower halfplane and h_2^* is an upper one. In the following, we describe an algorithm that can find a lower halfplane and an upper halfplane whose union covers P.

Let H_l denote the subset of all lower halfplanes of H and H_u the subset of all upper halfplanes. For each lower halfplene $h_l \in H_l$, we consider the *subproblem* of determining whether there is an upper halfplane $h_u \in H_u$ such that $P \subseteq h_l \cup h_u$, or equivalently, $P \cap \overline{h_l} \subseteq h_u$. Our eventual goal (referred to as the *original problem*) is to decide whether H_l has a halfplane h_l whose subproblem has an affirmative answer.

To solve the subproblems for all $h_l \in H$, we work in the dual setting. For any subset $H' \subseteq H$, let $\mathcal{D}(H')$ denote the set of dual points of the bounding lines of the halfplanes of H'. For any subset $P' \subseteq P$, let $\mathcal{D}(P')$ denote the set of dual lines of the points of P'. In the dual setting, the subproblem for h_l becomes determining whether $\mathcal{D}(H_u)$ has a dual point above all dual lines of $\mathcal{D}(P \cap \overline{h_l})$. If we consider each dual line of $\mathcal{D}(P)$ bounding an

upper halfplane, then it is equivalent to determining whether the common intersection of all upper halfplanes bounded by the dual lines of $\mathcal{D}(P \cap \overline{h_l})$ contains a point of $\mathcal{D}(H_u)$. Let $\mathcal{K}(h_l)$ denote the above common intersection. As such, the subproblem for h_l is to determine whether $\mathcal{K}(h_l) \cap \mathcal{D}(H_u) = \emptyset$.

With the above discussion, our original problem is to determine whether $\mathcal{K}(H_l) = \bigcup_{h_l \in H_l} \mathcal{K}(h_l)$ contains a point of $\mathcal{D}(H_u)$, i.e., whether $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) = \emptyset$. Note that $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) \neq \emptyset$ implies that there are a halfplane $h_l \in H_l$ and a dual point $h_u^* \in \mathcal{D}(H_u)$ such that $h_u^* \in \mathcal{K}(h_l)$. In the primal plane, this means that $P \subseteq h_l \cup h_u$, where h_u is the halfplane of H_u whose bounding line is dual to h_u^*. In the case where $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) \neq \emptyset$, our algorithm will return a halfplane $h_l \in H_l$ and a dual point $h_u^* \in \mathcal{D}(H_u)$ with $h_u^* \in \mathcal{K}(h_l)$.

Observation 13 has been proved in [4, Theorem 2.8].

▶ **Observation 13.** (ASW [4]) *For any two lower halfplanes $h_1, h_2 \in H_l$, the boundaries $\partial\mathcal{K}(h_1)$ and $\partial\mathcal{K}(h_2)$ cross each other at most twice.*

For a subset $H_l' \subset H_l$, define $\mathcal{K}(H_l') = \bigcup_{h_l \in H_l'} \mathcal{K}(h_l)$. The following observation holds particularly for our case (it does not hold for the disk case in [4]).

▶ **Observation 14.** *For any subset $H_l' \subseteq H_l$, the boundary $\partial\mathcal{K}(H_l')$ is x-monotone.*

Proof. For each halfplane $h_l \in H_l'$, $\partial\mathcal{K}(h_l)$ is the upper envelope of a set of lines and thus is x-monotone. Therefore, $\partial\mathcal{K}(H_l')$ is the lower envelope of the upper envelopes $\partial\mathcal{K}(h_l)$ for all $h_l \in H_l'$ and thus $\partial\mathcal{K}(H_l')$ must also be x-monotone. ◀

To determine whether $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) = \emptyset$, we run the ASW algorithm (see Section 4.3 [4]). The algorithm uses the divide-and-conquer approach. In the merge step, we are given implicit representations of $\mathcal{K}(H_l^1)$ and $\mathcal{K}(H_l^2)$ for two subsets H_l^1 and H_l^2 of H_l (i.e., $\mathcal{K}(H_l^1)$ and $\mathcal{K}(H_l^2)$ are implicitly maintained by a data structure so that certain operations needed by the algorithm can be efficiently supported by the data structure), and the problem is to obtain an implicit representation of $\mathcal{K}(H_l^1 \cup H_l^2)$. The merge step is done by sweeping a vertical line in the plane from left to right, which runs in $O(m^{4/3} \log^3 m)$ time, where $m = |H_l^1 \cup H_l^2|$. As such, the total time of the divide-and-conquer algorithm is $O(n^{4/3} \log^4 n)$. For our problem, we can improve the runtime of the merge step by a logarithmic factor based on Observation 14. Indeed, in the original ASW algorithm (which is for disks), their corresponding structure $\partial\mathcal{K}(H_l^i)$, $i = 1, 2$, may have $\Omega(m)$ edges intersecting the vertical sweepline and therefore the algorithm uses a balanced binary search tree to maintain all these intersections. In our problem, by Observation 14, $\partial\mathcal{K}(H_l^i)$, $i = 1, 2$, is x-monotone and thus intersects the vertical sweepline at most once. As such, we do not have to use a tree, which saves the runtime by a logarithmic factor. In this way, an implicit representation of $\mathcal{K}(H_l)$ can be computed in $O(n^{4/3} \log^3 n)$ time. Refer to Section 4.3 [4] for the algorithm details.

With the implicit representation of $\mathcal{K}(H_l)$, to determine whether $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) = \emptyset$, another sweeping procedure is done on both $\mathcal{K}(H_l)$ and the points of $\mathcal{D}(H_u)$. This takes $O(n^{4/3} \log^3 n)$ time for the disk case in [4]. Again, for our case, we do not need to use a tree in the sweeping procedure and thus the time of the procedure can be bounded by $O(n^{4/3} \log^2 n)$. Further, in the case where $\mathcal{K}(H_l) \cap \mathcal{D}(H_u) \neq \emptyset$, the algorithm will return a halfplane $h_l \in H_l$ and a point $h_u^* \in \mathcal{D}(H_u)$ with $h_u^* \in \mathcal{K}(h_l)$. As discussed above, this means $P \subseteq h_l \cup h_u$, where h_u is the halfplane of H_u whose bounding line is dual to h_u^*.

In summary, we can determine in $O(n^{4/3} \log^3 n)$ time whether H_l has a halfplane h_l and H_u has a halfplane h_u such that $P \subseteq h_l \cup h_u$. The runtime can be slightly improved using a recent result of Wang [30].

▶ **Theorem 15.** *Given in the plane a set of points and a set of halfplanes, one can compute a smallest subset of halfplanes whose union covers all points in* $O(n^{4/3}\log^{5/3} n \log^{O(1)}\log n)$ *time, where* n *is the total number of all points and halfplanes.*

References

1 Pankaj K. Agarwal and Sariel Har-Peled. Computing instance-optimal kernels in two dimensions. In *Proceedings of the 39th International Symposium on Computational Geometry (SoCG)*, pages 4:1–4:15, 2023. doi:10.4230/LIPIcs.SoCG.2023.4.

2 Pankaj K. Agarwal, Sariel Har-Peled, and Kasturi R. Varadarajan. Approximating extent measures of points. *Journal of the ACM*, 51:606–635, 2004. doi:10.1145/1008731.1008736.

3 Pankaj K. Agarwal and Jiangwei Pan. Near-linear algorithms for geometric hitting sets and set covers. *Discrete and Computational Geometry*, 63:460–482, 2020. doi:10.1007/s00454-019-00099-6.

4 Pankaj K. Agarwal, Micha Sharir, and Emo Welzl. The discrete 2-center problem. *Discrete and Computational Geometry*, 20:287–305, 1998. doi:10.1007/PL00009387.

5 Christoph Ambühl, Thomas Erlebach, Matúš Mihalák, and Marc Nunkesser. Constant-factor approximation for minimum-weight (connected) dominating sets in unit disk graphs. In *Proceedings of the 9th International Conference on Approximation Algorithms for Combinatorial Optimization Problems (APPROX), and the 10th International Conference on Randomization and Computation (RANDOM)*, pages 3–14, 2006. doi:10.1007/11830924_3.

6 Sayan Bandyapadhyay, William Lochet, Saket Saurabh, and Jie Xue. Minimum-membership geometric set cover, revisited. In *Proceedings of the 39th International Symposium on Computational Geometry (SoCG)*, pages 11:1–11:14, 2023. doi:10.4230/LIPIcs.SoCG.2023.11.

7 Michael Ben-Or. Lower bounds for algebraic computation trees (preliminary report). In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC)*, pages 80–86, 1983. doi:10.1145/800061.808735.

8 Therese Biedl, Ahmad Biniaz, and Anna Lubiw. Minimum ply covering of points with disks and squares. *Computational Geometry: Theory and Applications*, 94:101712, 2020. doi:10.1016/j.comgeo.2020.101712.

9 Paz Carmi, Matthew J. Katz, and Nissan Lev-Tov. Covering points by unit disks of fixed location. In *Proceedings of the International Symposium on Algorithms and Computation (ISAAC)*, pages 644–655, 2007. doi:10.1007/978-3-540-77120-3_56.

10 Timothy M. Chan and Elyot Grant. Exact algorithms and APX-hardness results for geometric packing and covering problems. *Computational Geometry: Theory and Applications*, 47:112–124, 2014. doi:10.1016/j.comgeo.2012.04.001.

11 Bernard Chazelle, Herbert Edelsbrunner, M. Grigni, L. Guibas, John Hershberger, Micha Sharir, and Jack Snoeyink. Ray shooting in polygons using geodesic triangulations. *Algorithmica*, 12:54–68, 1994. doi:10.1007/BF01377183.

12 Bernard Chazelle and Leonidas J. Guibas. Visibility and intersection problems in plane geometry. *Discrete and Computational Geometry*, 4:551–589, 1989. doi:10.1007/BF02187747.

13 Francisco Claude, Gautam K. Das, Reza Dorrigiv, Stephane Durocher, Robert Fraser, Alejandro López-Ortiz, Bradford G. Nickerson, and Alejandro Salinger. An improved line-separable algorithm for discrete unit disk cover. *Discrete Mathematics, Algorithms and Applications*, pages 77–88, 2010. doi:10.1142/S1793830910000486.

14 Stephane Durocher, J. Mark Keil, and Debajyoti Mondal. Minimum ply covering of points with unit disks. In *Proceedings of the 35th Canadian Conference on Computational Geometry (CCCG)*, pages 19–25, 2023.

15 Jeff Erickson. New lower bounds for Hopcroft's problem. *Discrete and Computational Geometry*, 16:389–418, 1996. doi:10.1007/BF02712875.

16 Tomás Feder and Daniel H. Greene. Optimal algorithms for approximate clustering. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 434–444, 1988. doi:10.1145/62212.62255.

17 Robert J. Fowler, Michael S. Paterson, and Steven L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Information Processing Letters*, 12:133–137, 1981. doi:10.1016/0020-0190(81)90111-3.

18 Sariel Har-Peled and Mira Lee. Weighted geometric set cover problems revisited. *Journal of Computational Geometry*, 3:65–85, 2012. doi:10.20382/jocg.v3i1a4.

19 Sariel Har-Peled and Benjamin Raichel. On the budgeted hausdorff distance problem. In *Proceedings of the 35th Canadian Conference on Computational Geometry (CCCG)*, pages 169–173, 2023.

20 John Hershberger and Subhash Suri. A pedestrian approach to ray shooting: Shoot a ray, take a walk. *Journal of Algorithms*, 18:403–431, 1995. doi:10.1006/jagm.1995.1017.

21 C.C. Lee and D.T. Lee. On a circle-cover minimization problem. *Information Processing Letters*, 18:109–115, 1984. doi:10.1016/0020-0190(84)90033-4.

22 Jian Li and Yifei Jin. A PTAS for the weighted unit disk cover problem. In *Proceedings of the 42nd International Colloquium on Automata, Languages and Programming (ICALP)*, pages 898–909, 2015. doi:10.1007/978-3-662-47672-7_73.

23 Gang Liu and Haitao Wang. On the line-separable unit-disk coverage and related problems, 2014. arXiv:2309.03162.

24 Gang Liu and Haitao Wang. On the line-separable unit-disk coverage and related problems. In *Proceedings of the 34th International Symposium on Algorithms and Computation (ISAAC)*, pages 51:1–51:14, 2023. doi:10.4230/LIPIcs.ISAAC.2023.51.

25 Nimrod Megiddo. Linear programming in linear time when the dimension is fixed. *Journal of the ACM*, 31:114–127, 1984. doi:10.1145/2422.322418.

26 Joseph S. B. Mitchell and Supantha Pandit. Minimum membership covering and hitting. *Computational Geometry: Theory and Applications*, 876:1–11, 2021. doi:10.1016/j.tcs.2021.05.002.

27 Nabil H. Mustafa, Rajiv Raman, and Saurabh Ray. Settling the APX-hardness status for geometric set cover. In *Proceedings of the 55th IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pages 541–550, 2014. doi:10.1109/FOCS.2014.64.

28 Nabil H. Mustafa and Saurabh Ray. PTAS for geometric hitting set problems via local search. In *Proceedings of the 25th Annual Symposium on Computational Geometry (SoCG)*, pages 17–22, 2009. doi:10.1145/1542362.1542367.

29 Logan Pedersen and Haitao Wang. Algorithms for the line-constrained disk coverage and related problems. *Computational Geometry: Theory and Applications*, 105-106:101883:1–18, 2022. doi:10.1016/j.comgeo.2022.101883.

30 Haitao Wang. Unit-disk range searching and applications. *Journal of Computational Geometry*, 14:343–394, 2023. doi:10.20382/jocg.v14i1a13.

31 Hai Yu, Pankaj K. Agarwal, Raghunath Poreddy, and Kasturi R. Varadarajan. Practical methods for shape fitting and kinetic data structures using coresets. *Journal of the ACM*, 52:378–402, 2008. doi:10.1007/s00453-007-9067-9.