Optimal cover of points by disks in a simple polygon

Haim Kaplan *, Matthew J. Katz **, Gila Morgenstern***, and Micha Sharir†

Abstract. Let P be a simple polygon, and let Q be a set of points in P. We present an almost-linear time algorithm for computing a minimum cover of Q by disks that are contained in P. We generalize the algorithm above, so that it can compute a minimum cover of Q by homothets of a fixed compact convex set of constant description complexity $\mathcal{O}$ that are contained in P. This improves previous results of Katz and Morgenstern [19]. We also consider the disk-cover problem when Q is contained in a (not too wide) annulus, and present an $O(|Q| \log |Q|)$ algorithm for this case.

1 Introduction

Let P be a simple n-gon in the plane, and let Q be a set of m points in P. A *disk cover of Q with respect to P* is a set $\mathcal{D}$ of disks (of variable radii), such that the union of the disks of $\mathcal{D}$ covers (i.e., contains) Q and is contained in P. In other words, each disk $D \in \mathcal{D}$ is contained in P, and each point $q \in Q$, lies in at least one disk $D \in \mathcal{D}$. A *minimum disk cover of Q with respect to P* is a disk cover of Q with respect to P of minimum cardinality. The problem of computing a minimum disk cover of Q with respect to P was introduced and studied by Katz and Morgenstern [19]. They also considered the case where the covering objects are homothets (contained in P) of a fixed compact convex set $\mathcal{O}$ of constant description complexity. In both cases, exact polynomial-time solutions were presented. In this paper we present alternative and significantly faster solutions for both disks and homothets, and also consider a third new case, as mentioned in the abstract and detailed below. All our solutions run in close to linear time.

* School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel. E-mail: haimk@post.tau.ac.il. Work by Haim Kaplan was partially supported by the U.S.-Israeli Binational Science Foundation, project number 2006204, and by grant 975/06 from the Israel Science Fund.

** Department of Computer Science, Ben-Gurion University, Beer-Sheva 84105, Israel. E-mail: matya@cs.bgu.ac.il. Work by Matthew Katz was partially supported by the MAGNET program of the Israel Ministry of Industry, Trade & Labor (CORNET consortium), and by the Lynn and William Frankel Center for Computer Sciences.

*** Department of Computer Science, Ben-Gurion University, Beer-Sheva 84105, Israel. E-mail: gilamor@cs.bgu.ac.il. Work by Gila Morgenstern was partially supported by the Lynn and William Frankel Center for Computer Sciences.

† School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel, and Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, USA. E-mail: michas@post.tau.ac.il. Work by Micha Sharir was partially supported by NSF Grant CCF-08-30272, by grant 2006-194 from the U.S.-Israeli Binational Science Foundation, by grant 338/09 from the Israel Science Fund, Israeli Academy of Sciences, and by the Hermann Minkowski–MINERVA Center for Geometry at Tel Aviv University.

Background. Geometric covering problems have been studied extensively. These problems are instances induced by geometric settings of the well-known set cover problem. Most of these instances are known to be NP-hard. Let us briefly review several geometric covering problems that are related to the problems studied in this paper.

In the unit disk cover problem, the goal is to cover a given set of points with the smallest possible number of unit disks. A polynomial-time approximation scheme for this problem was given by Hochbaum and Maas [16]. In the discrete version of this problem, the covering unit disks must be selected from a given set of unit disks. Until recently only constant-factor approximation algorithms were known for the discrete version; see [2, 4, 5, 28]. This was recently improved by Mustafa and Ray [27], who presented a polynomial-time approximation scheme for the discrete version (as well as for several other problems), which is based on local search.

Hurtado et al. [18] studied the related problem of computing a minimum enclosing disk of a given set of m points, whose center must lie in a given convex n-gon; they presented an $O(m+n)$-time algorithm for this problem. The 2-center problem with obstacles was studied by Halperin et al. [15]. In this problem, the goal is to find two congruent disks of smallest radius whose union covers a given set of m points and whose centers lie outside a given set of disjoint simple polygons with a total of n edges. They presented a randomized $O(n\log^2(mn) + mn\log^2 m\log(mn))$ expected time algorithm for this problem. The analogous 1-center problem was studied by Halperin and Linhart [14], who presented an $O((m+n)\log(mn))$ time algorithm for this problem.

The solutions of Katz and Morgenstern [19] for the problems mentioned above are based on the "perfect graph approach". (The perfect graph approach was previously used in the solution of several art-gallery problems, under restricted models of visibility; see, e.g., [20, 21, 26, 25, 30].) In the perfect graph approach, one first defines a graph G corresponding to the input scene. Next, the following two theorems are proven: (i) There is a one-to-one correspondence between a minimum cover of the desired kind (e.g., disk cover) and a minimum clique cover of G, and (ii) G is perfect. Note that the second claim is crucial, since, in general, minimum clique cover is NP-complete, but is polynomial for chordal and perfect graphs [10, 12, 13]. The algorithms of Katz and Morgenstern consist of three stages. First, construct G, next, find a minimum clique cover of G, and finally, construct the cover of Q corresponding to the minimum clique cover of G. The bottleneck of their algorithms are the first and last stages, which, in the case of covering by disks, where implemented in $O(nm^2)$ time.

In this paper we take a different approach, avoiding the explicit construction of the graph G, and computing the cover itself by following the algorithm of Gavril [10] for finding a minimum clique cover in a chordal graph, and exploiting the special geometric structure of G. This leads to improved solutions, which, when carefully implemented, run in nearly linear time.

The rest of this paper is organized as follows: In Section 2, we describe an algorithm for computing a minimum cover of Q by disks contained in P in

$O((n + m(\log n + \log^6 m)))$ time and $O(n + m \log\log m)$ space. In Section 3, we extend this result to the case of $\mathcal{O}$-cover, that is, computing a minimum cover of Q by homothets of $\mathcal{O}$ contained in P, where $\mathcal{O}$ is as above. We show that if $\mathcal{O}$ is a convex polygon with k edges, then this can be done in $O(n + m \log n + m \log^{k-1} m \log\log m)$ time, using $O(n+m \log^{k-1} m)$ space, or otherwise in $O(n + m \log n + m^{1+\varepsilon})$ time, using $O(n + m^{1+\varepsilon})$ space, for any $\varepsilon > 0$. Finally, in Section 4 we consider the case where the point set Q is contained in a "not too wide" annulus R, and give an $O(m \log m)$-time algorithm for computing a minimum-disk cover of Q by disks contained in R.

2 Minimum disk cover in a simple polygon

Let P be a simple polygon with n edges and let Q be a set of m points inside P. We present a nearly linear algorithm for finding a minimum cover of Q by disks contained in P.

Consider the Voronoi diagram of the relatively open edges and reflex vertices of P, confined to within P. We refer to these relatively open edges and reflex vertices collectively as *boundary features* (or simply *features*) of P, and let P^* denote the set of these features. The *medial axis* M is the network of vertices, straight edges, and parabolic arcs of this Voronoi diagram which are strictly inside P and the non-reflex vertices of P. A vertex of M which is not a vertex of P is a point at equal and smallest distance from three features of P (including the case where two of these features are an edge e and a reflex endpoint of e). An edge of M is the locus of all points at equal (and smallest) distance from two features of P (this time excluding the case of an edge and one of its endpoints). It is well known (and easy to show) that the network M is in fact a tree; that is, it is a connected network without cycles. See Figure 1(a).

As we will argue shortly, when constructing a disk cover of Q inside P, it suffices to consider disks whose centers lie on M, and in fact consider only maximal such disks, whose boundary touches ∂P (necessarily in at least two points). The proof of Lemma 1 below can be found in the Appendix.

Lemma 1. *For each point $q \in Q$, the portion of the medial axis consisting of centers of maximal disks that contain q and are contained in P is connected.*

Consider the graph G whose vertices are the points of Q, and it contains an edge between two points $p, q \in Q$ if there exists a disk containing both p and q and contained in P. Clearly there is such a disk if and only if there is a disk containing p and q whose center is on M and whose radius is the distance from its center to ∂P. This follows by noting that every disk D contained in P is contained in a maximal disk of the above kind, which is obtained by inflating D about its center until it touches ∂P, and then by moving its center away from the contact with ∂P, maintaining that contact, until a second contact is made.

The proof of Lemma 1 shows that, for each $q \in Q$, the intersection of M with the Voronoi cell of q in the diagram of $P^* \cup \{q\}$ is a connected subset M_q of M. We refer to M_q as a *subtree* of M, but note that the leaves of M_q (points of M_q

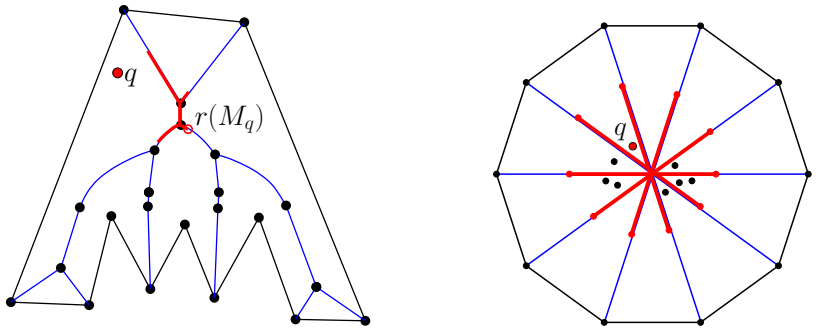

Fig. 1. (a) The medial axis of a simple polygon and the subtree M_q of a point $q \in Q$. (b) The complexity of each subtree M_q is $\Theta(n)$.

whose removal does not disconnect M_q) need not necessarily be vertices of M, but can lie in the relative interior of edges of M; see Figure 1(a). It follows by definition that, by identifying each point q of Q with its subtree M_q, G becomes the intersection graph of these subtrees. Therefore, by the characterization of Buneman and Gavril [3, 11], G is a *chordal* graph.[1]

We note that computing the subtrees M_q explicitly is too expensive, because their overall complexity can be $\Theta(mn)$ in the worst case, as depicted in Figure 1(b).

As shown by Katz and Morgenstern [19], a cover of Q by disks contained in P corresponds to a clique cover of G and vice versa. This also follows from the fact that pairwise intersecting subtrees of a tree T satisfy the 2-*Helly-property*: if every pair of subtrees in a given collection intersect, then they all have a common intersection [12]. Therefore, a clique of G can be covered by a single disk which is a maximal disk centered at a common point of all subtrees M_q corresponding to the points q of the clique. (The converse direction is trivial.) So our problem is to find a minimum clique cover of G. As is known [10], the chordality of G implies that this latter problem is solvable in polynomial time, but, exploiting the special geometric structure of the graph G, we are able to solve it particularly efficiently, by an algorithm with $O(n + m(\log n + \log^6 m))$ running time (improving upon the cubic algorithm of Katz and Morgenstern in [19] mentioned in the introduction).

We follow the algorithm of Gavril [10] for finding a minimum clique cover in a chordal graph. The algorithm is greedy and is based on the fact (see, e.g., [12]) that a chordal graph contains a *simplicial vertex* v, that is, a vertex whose neighbors induce a clique. The greedy clique cover algorithm takes as the first clique in the cover such a simplicial vertex v and its neighbors. It then deletes v and its neighbors and iterates this step on the subgraph induced by the remaining

[1] A graph is *chordal* if every cycle of at least four edges has a *shortcut*, i.e., an edge connecting two non-consecutive vertices of the cycle; see [12].

vertices (which is still chordal). It is not hard to see that the output clique cover is indeed minimum, because the simplicial vertices picked at each iteration form an independent set in the original G, and clearly the size of any clique cover is at least the size of this (or any) independent set.

We can implement this algorithm using the subtree representation of G, as follows. Root M at an arbitrary vertex of ∂P. Making M rooted induces a root for each subtree M_q, for $q \in Q$, which we denote by $r(M_q)$. Note that $r(M_q)$ is not necessarily a vertex of M but can lie in the relative interior of an edge. See Figure 1(a).

Pick M_q such that the (appropriately defined) subtree rooted at $r(M_q)$ contains no other root. The points of Q corresponding to all subtrees which contain $r(M_q)$ are the first clique in our cover (q is a simplicial vertex). The disk corresponding to this clique can be taken to be the maximal disk within P centered at $r(M_q)$. We then delete all subtrees containing $r(M_q)$ and iterate, until all of Q is exhausted.

We now present an efficient implementation of this algorithm. It consists of the following steps.

1. For each point $q \in Q$ compute an "anchor point" $A(q) \in M_q$.
2. Starting from each point $A(q)$, search M to find the corresponding root $r(M_q)$.
3. Maintain Q in a dynamic range searching data structure $\mathcal{D}$ that can efficiently report all points of Q contained in a query disk, and delete points from Q.
4. Search M in a bottom-up manner to find a simplicial vertex q of G (whose root $r(M_q)$ is lowest in M). Query $\mathcal{D}$ with the maximal disk centered at $r(M_q)$. The points in the query output form the first clique. Delete each such point q' from $\mathcal{D}$ and delete its corresponding root $r(M_{q'})$ from M.
5. Repeat the preceding step until Q becomes empty.

We now give the details of implementing each of these steps. We first compute M in $O(n)$ time [9]. We regard the edges and vertices of $M \cup \partial P$ as the edges and vertices of a planar map H. We further partition H into "pseudo-trapezoids" (referred to as "trapezoids", in short), by connecting each vertex of M to its nearest points on ∂P.

For each point $q \in Q$ we compute the trapezoid $T(q)$ in the decomposition of H containing q. To do so we preprocess the decomposition of H in $O(n)$ time and construct the point location data structure of Kirkpatrick [22], which supports logarithmic-time point-location queries. Then we locate the trapezoids $T(q)$ by m point location queries to this data structure, in $O(m \log n)$ time.

Let e be the feature of P^* on $\partial T(q)$ (that is, $T(q)$ is contained in the Voronoi cell of e). We compute the closest point q' to q on e and take $A(q)$ to be the intersection of the line $q'q$ with M which is closest to q (and which also lies on $\partial T(q)$). It is easy to see that a maximal disk centered at $A(q)$ inside P contains q, and therefore M_q indeed contains $A(q)$.

We compute the roots $r(M_q)$, for $q \in Q$, as follows. We fix a root for M, for example, the rightmost vertex of P. We then traverse M in depth-first order from

the root, and maintain its vertices on the stack (those vertices whose subtrees are still being traversed) in an array L, stored in their reverse order on the stack, i.e., in their order along the path of M from the root to the vertex currently being explored (L is in fact just a concrete structure for implementing the stack). When the search moves from a vertex v to a new vertex w, we insert w as the rightmost element of L, and when the search backtracks from a vertex v, we delete v from L.

When we traverse the edge (v, w) during the depth-first search, we find $r(M_q)$ for every point q such that $A(q)$ is on the edge (v, w), as follows. First observe that testing whether a point $z \in M$ belongs to M_q is easy to do in $O(1)$ time, provided we know the feature (edge or vertex) of M containing z: This is equivalent to testing whether $|zq|$ is at most the radius of the maximal disk centered at z, and this radius is readily available since we know the features of P^* nearest to z.

So let $q \in Q$ be a point whose anchor $A(q)$ lies on (v, w). If $v \notin M_q$ then $r(M_q)$ is on the edge (v, w). Furthermore, it is a point on (v, w) at equal distances to the two features of P defining the edge (v, w) and to q. We compute it by solving the appropriate system of algebraic equations (as is well known, there can be at most two solutions, for otherwise, since Voronoi regions are star-shaped, we would get an impossible planar embedding of $K_{3,3}$), and by taking the solution which lies on (v, w) closest to the root of M (i.e., to v).

If $v \in M_q$, we use binary search on the array L to find the farthest ancestor u of v which is still in M_q. The root $r(M_q)$ is on the edge from u to its parent, and we find it by solving a system of algebraic equations analogous to the one described above.

Having collected all the roots $r(M_q)$, we sort them along the edges of M containing them, and split the edges at these roots, making the roots additional vertices of M.

Finally we collect the roots which are centers of the maximal disks in our cover, using the greedy algorithm described above. For that we need a dynamic range reporting data structure $\mathcal{D}$ which, given a query disk D, can report all the points in D. The structure $\mathcal{D}$ initially contains all the points of Q, but, as we choose our centers, we will delete from $\mathcal{D}$ points that are already covered.

We traverse M again in depth-first order, stopping at each root $r(M_q)$. When we are about to backtrack from a root $r(M_q)$ which is still active (see below), we add the maximal disk D centered at $r(M_q)$ and contained in P to our cover. Using our range searching data structure $\mathcal{D}$, we report all the points of Q inside D, and delete them from $\mathcal{D}$. For each such point q we also mark the corresponding root $r(M_q)$ as inactive, so that we will ignore it as we backtrack through it later in the depth-first search. When the search terminates, we have obtained the desired minimum disk cover.

We now analyze the complexity of the algorithm. As above, let n denote the number of edges of P and put $m = |Q|$. Computing M and the Voronoi diagram H induced by $M \cup \partial P$, the triangulation of H into trapezoids, and the point location data structure for this triangulation, takes $O(n)$ time [9, 22]. For each point $q \in Q$, computing $A(q)$ takes $O(\log n)$ time, for a total of $O(m \log n)$ time.

The computation of the roots $r(M_q)$ takes $O(n + m \log n)$ time: maintaining the array L takes $O(n)$ time, and the binary searches for the roots take a total of $O(m \log n)$ time. Sorting the roots along the edges of M and subdividing M at these roots takes $O(n + m \log m)$ time. Finally, in the last step we again traverse M and perform a range reporting query for each disk (clique) in our cover. Using the recent data structure of Chan [6], we can find and delete all k points in a query disk in $O(k \log^6 m)$ amortized time. So the total time spent quering and updating the range searching structure $\mathcal{D}$ is $O(m \log^6 m)$ time. Summing up we obtain that the total running time of our algorithm is $O((n + m(\log n + \log^6 m))$. The space required by our algorithm is linear except for an additional $O(m \log \log m)$ needed for the data structure of [6].

In summary, we obtain:

Theorem 1. *Let P be a simple polygon with n edges and Q a set of m points contained in P. We can compute a minimum cover of Q by disks contained in P in $O((n + m(\log n + \log^6 m))$ time and $O(n + m \log \log m)$ space.*

3 Covering by homothetic copies of a convex set

Let P and Q be as in Section 2, and let $\mathcal{O}$ be some fixed compact convex set with nonempty interior. For a large part of the analysis, this is essentially all we assume about $\mathcal{O}$. For the algorithmic part, however, we need to assume that $\mathcal{O}$ has a sufficiently simple shape so as to facilitate certain operations on $\mathcal{O}$ as well as efficient construction of a dynamic range reporting data structure, similar to the structure $\mathcal{D}$ used above. For the time being, we only assume that P, Q and $\mathcal{O}$ are in general position, to avoid possible degeneracies in the constructs that extend those studied in the preceding section. Further assumptions will be elaborated below.

We fix some point inside $\mathcal{O}$ as its "center point", and assume that $\mathcal{O}$ is initially specified so that this point lies at the origin. Thinking of each point of $\mathcal{O}$ as a vector, $\lambda\mathcal{O}$ then denotes the convex set obtained from $\mathcal{O}$ by multiplying each vector in $\mathcal{O}$ by a fixed positive factor λ. The *convex distance function* induced by $\mathcal{O}$ is $d_{\mathcal{O}}(p, q) = \inf\{\lambda \mid q \in p + \lambda\mathcal{O}\}$. We refer to it as the *$\mathcal{O}$-distance*. Chew and Drysdale [7] were the first to study Voronoi diagrams under convex distance functions; see also Leven and Sharir [23]. Note that $d_{\mathcal{O}}$ is a metric if and only if $\mathcal{O}$ is centrally symmetric with respect to its center.

The medial axis $M_{\mathcal{O}}$ of P under the $\mathcal{O}$-distance is defined analogously to the medial axis of P under the Euclidean distance, as the locus of all points inside P whose $\mathcal{O}$-distance to the boundary is attained in at least two points. Assuming general position, $M_{\mathcal{O}}$ is a connected 1-dimensional network, consisting of vertices and edges. Each edge is the locus of all points which are at the same (nearest) $\mathcal{O}$-distance from two features of P^* (where P^* is defined as in the previous section). A vertex of $M_{\mathcal{O}}$ which is not a vertex of P is a point at the same (nearest) $\mathcal{O}$-distance from three features of P^*. The shape of the edges of $M_{\mathcal{O}}$ depends on the shape of $\mathcal{O}$. For example, if $\mathcal{O}$ is a convex polygon then each edge is a polygonal curve, whose breakpoints correspond to placements of

the center of $\mathcal{O}$ at which a vertex of $\mathcal{O}$ touches a vertex of P. See [7, 23] for more details. Finally, as in the previous section, it follows from basic properties of Voronoi diagrams that $M_{\mathcal{O}}$ is a tree.

As in the preceding section, for a point $q \in Q$, we denote by M_q the portion of $M_{\mathcal{O}}$ consisting of centers of maximal homothets of $\mathcal{O}$ that contain q (and are contained in P). Lemma 2 below is analogous to Lemma 1, asserting that for each $q \in Q$, M_q is a subtree of $M_{\mathcal{O}}$; its proof can be found in the Appendix.

Lemma 2. *M_q is connected for each point $q \in Q$.*

Consider the graph $G_{\mathcal{O}}$ defined similarly to G in the previous section. Its vertices are the points of Q, and it contains an edge between two points $p, q \in Q$ if there exists a homothet of $\mathcal{O}$ containing both p and q and contained in P. As in the case of disks, there is such a homothet if and only if there is a homothet containing p and q whose center is on $M_{\mathcal{O}}$ and whose boundary touches ∂P (at least twice). Indeed, take the given placement $\mathcal{O}_1$ of $\mathcal{O}$ and expand it about its center until it touches ∂P at some point v. Now move the center away from v along the line connecting it to v while expanding $\mathcal{O}$ so as to keep its boundary passing through v, until its center lies on $M_{\mathcal{O}}$. It is easily checked that the new placement of $\mathcal{O}$ contains $\mathcal{O}_1$ (and thus p and q) and is contained in P.

Again, if we identify each point q of Q with its subtree M_q, then $G_{\mathcal{O}}$ is the intersection graph of these subtrees and thus $G_{\mathcal{O}}$ is a chordal graph. As in Section 2, since pairwise intersecting subtrees of a tree T satisfy the 2-Helly property, covering Q by homothets of $\mathcal{O}$ is equivalent to a clique cover of $G_{\mathcal{O}}$, so our problem now is to find a minimum clique cover of $G_{\mathcal{O}}$.

We use the same high-level algorithm as in Section 2, but its implementation requires other, slightly more complex tools. Recall that so far the analysis did not require any further assumptions concerning the actual shape of $\mathcal{O}$. However, to facilitate an efficient implementation of the algorithm, we need to assume that this shape is sufficiently simple, so as to allow various operations (such as computing the $\mathcal{O}$-distance between two points, or between a point and a line segment, finding a point at the same $\mathcal{O}$-distance from two line segments, etc.) on $\mathcal{O}$ and on a constant number of other features (points and/or line segments) to be performed in constant time. The simplest way to enforce these properties is to assume that $\mathcal{O}$ has *constant description complexity* (see, e.g., [29]).

The algorithm proceeds through the following steps, analogous to those in the preceding algorithm. First, we compute the medial axis $M_{\mathcal{O}}$ in time $O(n)$, using the algorithm of Chin et al. [9]; the combinatorial complexity of $M_{\mathcal{O}}$ is $O(n)$. Next, we construct the planar map H (induced by $M_{\mathcal{O}} \cup \partial P$) and partition it into simply-shaped cells, by connecting each vertex of $M_{\mathcal{O}} \setminus \partial P$ (including breakpoints along edges which are equidistant from an edge of P and an endpoint of that edge) to its nearest point(s) (in the $\mathcal{O}$-distance) on ∂P. Each cell is bounded by two of these connecting segments, by a portion of a single edge of P (or just a vertex), and by an edge of $M_{\mathcal{O}}$ (or just a vertex). We then construct the point-location data structure of Kirkpatrick [22] on this triangulation of H. Computing this partition of H into trapezoids, and the point location data structure for this

triangulation, takes $O(n)$ time (in our assumed model of computation). We then compute, for each point $q \in Q$, the trapezoid $T(q)$ containing it, by making m queries to the point location data structure, in a total of $O(m \log n)$ time.

Let e be the feature of P^* on $\partial T(q)$ (that is, $T(q)$ is contained in the Voronoi cell of e). We compute the closest point q' to q on e under the $\mathcal{O}$-distance, and take $A(q)$ to be the intersection of the line $q'q$ with $M_{\mathcal{O}}$ which is closest to q (and which also lies on $\partial T(q)$). It is easy to see, arguing as above, that the maximal copy of $\mathcal{O}$ centered at $A(q)$ inside P touches ∂P at q' (and at another point) and contains q, and therefore M_q contains $A(q)$.

Computing the roots $r(M_q)$ is done exactly as in the previous section, in $O(n + m \log n)$ time. Sorting the roots along the edges of $M_{\mathcal{O}}$ and subdividing $M_{\mathcal{O}}$ at these roots also takes $O(n + m \log m)$ time, as before. Finally, in the last step we traverse $M_{\mathcal{O}}$ and perform a range reporting query for each homothet of $\mathcal{O}$ centered at one of the roots $r(M_q)$ (which represents a clique) in our cover, and delete the points of Q contained in this homothet from the data structure. For this we use the more general data structure of Agarwal et al. [1, Theorem 4.4]. Specialized to our setting (where we continue to assume that $\mathcal{O}$ has constant description complexity), it provides a dynamic data structure of size $O(m^{1+\varepsilon})$, which can be constructed in $O(m^{1+\varepsilon})$ time, for any $\varepsilon > 0$, so that, given a query homothetic placement $\mathcal{O}'$ of $\mathcal{O}$, it can report the points of Q inside $\mathcal{O}'$ in time $O(\log m + |Q \cap \mathcal{O}'|)$. A point can be deleted from Q (that is, from the data structure) in $O(m^{\varepsilon})$ amortized time, for any $\varepsilon > 0$. Hence the overall cost of this final part of the algorithm is $O(m^{1+\varepsilon})$, for any $\varepsilon > 0$. We thus obtain:

Theorem 2. *Let $\mathcal{O}$ be a fixed compact convex set of constant description complexity, let P be a simple polygon with n edges, and let Q be a set of m points contained in P. We can compute a minimum cover of Q by homothets of $\mathcal{O}$ contained in P, in $O(n + m \log n + m^{1+\varepsilon})$ time, using $O(n + m^{1+\varepsilon})$ storage, for any $\varepsilon > 0$.*

The algorithm can be made slightly more efficient in the special case where $\mathcal{O}$ is a convex polygon with k edges, for a constant k. In this case the dynamic range-reporting data structure can be implemented using a k-dimensional orthogonal range tree, where each dimension (coordinate) corresponds to the normal direction to one of the edges of $\mathcal{O}$. Using the data structure of [24] (See also [8]), this requires $O(m \log^{k-1} m)$ space and $O(m \log^{k-1} m \log \log m)$ time for all queries and subsequent deletions. The cost of the other steps of the algorithm remains the same as above. We thus obtain:

Theorem 3. *Let $\mathcal{O}$ be a fixed convex polygon with k edges, let P be a simple polygon with n edges, and let Q be a set of m points contained in P. We can compute a minimum cover of Q by homothets of $\mathcal{O}$ contained in P, in $O(n + m \log n + m \log^{k-1} m \log \log m))$ time, using $O(n + m \log^{k-1} m)$ space.*

4 Covering by arbitrary disks in a sufficiently narrow annulus

Assume that the points of Q lie in an annulus R rather than in a simple polygon. In this case, M, the medial axis of R, is not a tree but a circle. In particular, let c be the center of R and let r_1 and r_2 be inner and outer radii of R, respectively, then M is a circle of radius $r_M = \frac{r_1+r_2}{2}$, centered at c.

As in the previous sections, each point $q \in Q$ is associated with an arc M_q of the circle M, which is the portion of M consisting of centers of maximal disks that cover q (and are contained in R). See Figure 2(a), which illustrates a proof of the property that M_q is the intersection of M with the disk of radius $\frac{r_2-r_1}{2}$ centered at q. Again, we consider the intersection graph G of these arcs. However, unlike the previous cases, G is not chordal, since M is not a tree. Instead, G is a circular-arc graph (see, e.g., [12]). Moreover, we argue that if $r_2 < \frac{2+\sqrt{3}}{2-\sqrt{3}} r_1$ then G has the 2-Helly property, which means that the circular arcs in a clique of G have a point in common.

Indeed, since M_q is the intersection of M with the disk of radius $\frac{r_2-r_1}{2}$ centered at q, it is maximal when a_q, q, and b_q are collinear, where a_q and b_q are the endpoints of M_q. Let $\theta = \angle a_q c b_q$. If a_q, q, and b_q are indeed collinear then $\sin\frac{\theta}{2} = \frac{r_2-r_1}{r_2+r_1}$. So if $r_2 < \frac{2+\sqrt{3}}{2-\sqrt{3}} r_1$ then $\sin\frac{\theta}{2} < \frac{\sqrt{3}}{2}$, and therefore $\theta < 2\pi/3$. Consider a clique $\mathcal{C}$ in G. Since all arcs are of length smaller than $1/3$ the length of M, and they all intersect one specific arc of $\mathcal{C}$, then they cannot cover M completely. Consequently, $\mathcal{C}$ can be viewed as a set of pairwise intersecting intervals on a line and it follows that all the arcs of $\mathcal{C}$ must have a common intersection.

We conclude that, once again, a cover of Q by disks contained in R corresponds to a clique cover of G, and vice versa. We thus compute a minimum clique cover of G by applying the $O(m)$-time algorithm of Hsu and Tsai [17], after sorting the arcs M_q be their endpoints in $O(m \log m)$ time.

In summary, we obtain

Theorem 4. *Let R be an annulus such that $r_2 < \frac{2+\sqrt{3}}{2-\sqrt{3}} r_1$, where r_1, r_2 are the inner and outer radii of R, respectively, and let Q a set of m points contained in R. We can compute a minimum cover of Q by disks contained in R in $O(m \log m)$ time and $O(m)$ space.*

Theorem 4 also applies to the slightly more general case, where R is a disk with a circular hole, not necessarily concentric; see Figure 2(b). In this case, the medial axis is an ellipse with foci at the centers of R and of its hole. For each $q \in Q$, M_q is still a connected arc of M. Specifically, the endpoints of M_q are the points at equal distance to the two circles bounding R and its hole and to q, and there can be at most two such points. (Otherwise, arguing as in Section 2, we would get an impossible planar embedding of $K_{3,3}$.)

The graph G is a circular-arc graph, and it possesses the 2-Helly property provided that the hole is not too small. (The exact condition is that there do not exist three points $q_1, q_2, q_3 \in R$ whose arcs M_{q_1}, M_{q_2}, M_{q_3} cover M.) Hence,

if this condition holds then a minimum clique cover can be found as above, with the same asymptotic bounds on the running time and storage.

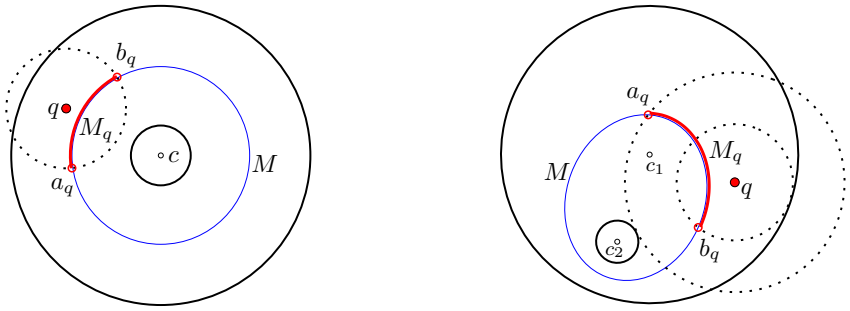

Fig. 2. (a) The arc M_q of a point $q \in Q$, obtained as the intersection of M with the (dotted) disk of radius $\frac{r_2-r_1}{2}$ around q. (b) R is a disk centered at c_1 with a hole centered at c_2; the medial axis, M is elliptic. The endpoints of M_q are in equal distance to the two circles bounding R and its hole and to q.

Finally, we do not know how critical is the assumption that R is not too wide, as in Theorem 4. Does the problem become hard if R is wider?

References

1. P. K. Agarwal, A. Efrat, and M. Sharir. Vertical decomposition of shallow levels in 3-dimensional arrangements and its applications. *SIAM J. Comput.*, 29: 912–953, 2000.
2. H. Brönnimann and M. T. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete & Computational Geometry*, 14(4):469–479, 1995.
3. P. Buneman. A characterization of rigid circuit graphs. *Discrete Math.,* 9:205-212, 1974.
4. G. Calinescu, I. I. Mandoiu, P-J Wan, and A. Zelikovsky. Selecting forwarding neighbors in wireless ad hoc networks. *MONET*, 9(2):101–111, 2004.
5. P. Carmi, M. J. Katz, and N. Lev-Tov. Covering points by unit disks of fixed location. In *ISAAC*, pages 644–655, 2007.
6. T. M. Chan. A dynamic data structure for 3-d convex hulls and 2-d nearest neighbor queries. In *SODA '06: Proc. 17th Annual ACM-SIAM Sympos. Discrete Algorithms*, pages 1196–1202, 2006.
7. L. P. Chew, and III R. L. Dyrsdale. Voronoi diagrams based on convex distance functions. In *SCG '85: Proc. First Annual Sympos. Comput. Geom.*, pages 235–244, 1985.
8. Y. Chiang and R. Tamassia. Dynamic algorithms in computational geometry. *Proceedings of the IEEE.* 80(9):1412–1434, 1992.

9. F. Y. L. Chin, J. Snoeyink, and C. A. Wang. Finding the medial axis of a simple polygon in linear time. *Discrete Comput. Geom.*, 21(3):405–420, 1999.
10. F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM J. Comput.*, 1(2):180-187, 1972.
11. F. Gavril. The intersection graphs of subtrees of a tree are exactly the chordal graphs. *J. Combinat. Theory Ser. B,* 16:47-56, 1974.
12. M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic Press, New York, 1980
13. M. Grötschel, L. Lovász, and A. Schrijver. Polynomial algorithms for perfect graphs. In C. Berge and V. Chvátal, editors, Perfect Graphs, volume 21, pages 325–356. North Holland, 1984.
14. D. Halperin and C. Linhart, The minimum enclosing disk with obstacles. Manuscript, 1999.
15. D. Halperin, M. Sharir, and K. Y. Goldberg. The 2-center problem with obstacles. *J. Algorithms*, 42(1):109–134, 2002.
16. D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and vlsi. *J. ACM*, 32(1):130–136, 1985.
17. W. L. Hsu and K. H. Tsai. Linear time algorithms on circular-arc graphs. *Inform. Process. Lett.*, 40(3):123–129, 1991.
18. F. Hurtado, V. Sacristán, and G. Toussaint. Some constrained minimax and maximin location problems. In *Studies in Locational Analysis*, pages 17–35, 2000.
19. M. J. Katz and G. Morgenstern. A scheme for computing minimum covers within simple regions. In F. Dehne, M. Gavrilova, J.R. Sack, and Cs. D. Tóth, editors, *Algorithms and Data Structures, 11th International Symposium (WADS)*, volume 5664 of *Lecture Notes in Computer Science*, pages 447-458. Springer, 2009.
20. M. J. Katz and G. Morgenstern, Guarding orthogonal art galleries with sliding cameras. In *25th European Workshop on Comput. Geom.*, pages 159–162, 2009.
21. M. J. Katz and G. S. Roisman, On guarding the vertices of rectilinear domains. *Comput. Geom.*, 39(3):219–228, 2008.
22. D. Kirkpatrick. Optimal search in planar subdivisions. *SIAM J. Comput.*, 12(1):28–34, 1983.
23. D. Leven and M. Sharir. Planning a purely translational motion for a convex object in two–dimensional space using generalized voronoi diagrams. *Discrete Comput. Geom.*, 2:9–31, 1987.
24. K. Mehlhorn and S. Näher. Dynamic Fractional Cascading. *Algorithmica*, 5(2):215–241, 1990.
25. R. Motwani, A. Raghunathan, and H. Saran. Covering orthogonal polygons with star polygons: The perfect graph approach, *J. Comput. Syst. Sci.*, 40(1): 19–48, 1990.
26. R. Motwani, A. Raghunathan, and H. Saran, Perfect graphs and orthogonally convex covers. *SIAM J. Discrete Math.*, 2(3):371–392, 1989.
27. N. H. Mustafa and S. Ray. PTAS for geometric hitting set problems via local search. In *Symposium on Computational Geometry*, pages 17–22, 2009.
28. S. Narayanappa and P. Vojtechovský. An improved approximation factor for the unit disk covering problem. In *CCCG*, 2006.
29. M. Sharir and P. K. Agarwal. *Davenport-Schinzel Sequences and their Geometric Applications.* Cambridge University Press, New York, 1995.
30. C. Worman and J. M. Keil. Polygon decomposition and the orthogonal art gallery problem. *Int. J. Comput. Geometry Appl.*, 17(2):105–138, 2007.

Appendix

Proof of Lemma 1

Consider the Voronoi cell $V(q)$ in the Voronoi diagram of $P^* \cup \{q\}$. The vertices of $V(q)$ are precisely the points at which the medial axis M crosses the boundary of this cell. Note also that $V(q)$ is a bounded cell, fully contained in the interior of P. See Figure 3.

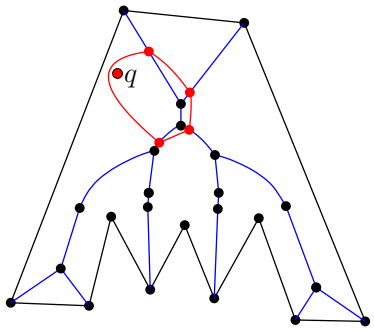

Fig. 3. The Voronoi cell $V(q)$, for a point $q \in Q$, in the Voronoi diagram of $P^* \cup \{q\}$.

Observe that the portion of M consisting of centers of maximal disks that contain q is precisely $M \cap V(q)$. We thus need to show that this portion of M is connected. That is, we have a sub-network of M which enters and leaves $V(q)$ through its vertices, and we want to rule out the possibility that it has two or more disconnected pieces.

There are two ways in which $M \cap V(q)$ can be disconnected: (i) It can have a component fully contained in the interior of $V(q)$. (ii) It can have two (or more) components, each entering and leaving $V(q)$ at its vertices.

Case (i) is impossible because M is connected.

Case (ii) is also impossible: Suppose to the contrary that $M \cap V(q)$ did have two (or more) such components. Then there must exist a feature $\xi \in P^*$, such that the Voronoi cell $V_0(\xi)$ of ξ in the medial-axis diagram (without q) intersects $V(q)$ in a region K such that $V(q) \setminus K$ is disconnected. For this, the intersection of K with $\partial V(q)$ has to be disconnected, and consist of at least two distinct Voronoi edges e_1, e_2 (of $V(q)$). See Figure 4.

This however is impossible, because P is a simple polygon. Indeed, take a point a_1 on e_1 and a point a_2 on e_2, and connect each a_i to q and to its nearest point on ξ by straight segments. Connecting these two nearest points to each other along ξ yields, together with the four other segments, a region E, which is either a pentagon (if ξ is an edge) or a quadrangle (if ξ is a vertex), and which

is fully contained in P. By assumption, though, it must contain a point v (an endpoint of e_1 or of e_2) equally nearest to q, ξ, and another feature $\xi' \in P^*$. Since ξ' lies *outside* E, the segment connecting v to its nearest point on ξ' must cross ∂E, which is impossible (because of the star-shapedness of Voronoi cells).

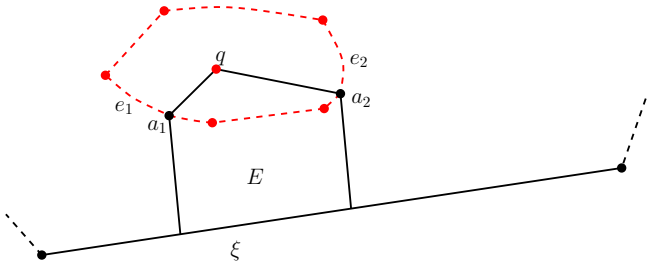

Fig. 4. There cannot be two disconnected edges e_1 and e_2 on the boundary between $V_0(\xi)$ and $V(q)$ where ξ is a feature of P^*.

This completes the proof.

Proof of Lemma 2

Proof of Lemma 2 is analogous to the proof of Lemma 1. The only difference is that in Case (ii) we connect a_1 and a_2 to their nearest points along ξ in the $\mathcal{O}$-distance, and use the star-shapedness of Voronoi cells, a property that holds for any convex distance function.