ELSEVIER

Computational Geometry 23 (2002) 31–42

Computational Geometry
Theory and Applications

www.elsevier.com/locate/comgeo

Packing two disks in a polygon

Sergei Bespamyatnikh

Department of Computer Science, University of British Columbia, Vancouver, BC, V6T 1Z4 Canada

Received 30 November 2000; accepted 30 July 2001

Communicated by R. Klein

Abstract

We present two algorithms for packing two largest disks in a polygon. The first algorithm locates two disks in a simple polygon in time $O(n \log^2 n)$ improving the best previous deterministic result (Bespamyatnikh, 1999) by a factor of $\log n$. The second algorithm finds two disks in a convex polygon such that the disks are separated by a diagonal of the polygon. It runs in time $O(n \log^2 n)$ improving previous result (Kim et al., 2000) by a linear factor.

Keywords: Computational geometry; Optimization; Parametric searching

1. Introduction

In this paper we consider the following problem.

Two disk problem. Find two largest non-overlapping disks of equal radius in a simple polygon P with n vertices.

This problem is an example of the obnoxious facility location [4,10,13–15]. Obnoxious location models are models in which customers no longer consider the facility desirable and try to have it as close as possible to their own location, but instead avoid the facility and stay away from it. Typical applications are optimal locations of nuclear reactors, garbage dumps, or water purification plants. The centers of the disks can be viewed as the locations of two obnoxious facilities.

The two disk problem has another nice motivation. Biedl et al. [6] consider a gift wrapping problem which asks whether a given polyhedron can be wrapped up, or hidden, using a given piece of paper.

E-mail address: besp@cs.ubc.ca, besp@cs.duke.edu (S. Bespamyatnikh).
URL address: http://www.cs.ubc.ca/~besp (S. Bespamyatnikh).

0925-7721/01/$ – see front matter
PII: S0925-7721(01)00044-X

They study a problem of wrapping the disk by a piece of paper that has a polygonal form. The two disk problem is to the disk wrapping using only one straight fold.

Kim et al. [17] consider the following problem.

Folding line problem. Given a convex polygon P with n vertices, find an optimal *folding line* which is a diagonal of P separating two largest non-overlapping disks of equal radius in P.

Related to the two disk problem is the largest-empty-circle problem: find the largest disk that fits in a given polygon. The standard solution uses the fact that the center of the largest disk is a vertex of the medial axis [22]. The medial axis can be computed in linear time [12].

Biedl et al. [6] obtained a polynomial-time algorithm for finding the largest disk that can be hidden by one simple fold of an input polygon, i.e., an algorithm for the two disk problem. Their algorithm takes $\Omega(n^2)$ time because it looks through all pairs of edges in the medial axis. Bespamyatnikh [5] gave an $O(n \log^3 n)$ algorithm for the two disk problem. Bose et al. [7] obtained a linear-time algorithm for packing two disks in a convex polygon. Very recently, Bose et al. [8] obtained a simple randomized $O(n \log n)$ algorithm for finding two disks in polygons with holes.

We improve the previous algorithm [5] for the two disk problem by a factor of $\log n$. The algorithm is based on parametric searching of Megiddo [18]. We show that the decision problem that asks whether there exist two non-overlapping disks of fixed radius, can be solved in linear time improving the best previous deterministic result [5].

Kim et al. [17] gave an $O(n^2 \log^2 n)$ algorithm for computing an optimal vertex folding line. Their algorithm is based on parametric searching. We show that the folding line problem can be solved by searching in a binary tree avoiding the use of parametric searching. The running time of our algorithm is $O(n \log^2 n)$, substantially improving upon the previous bound.

2. Preliminaries

The *medial axis* M [1,12] of a simple polygon P is the locus of all centers of disks that are contained in P and touch the boundary of P in two or more points. It is related to the *Voronoi diagram* [3]. The Voronoi diagram of a set of sites is the partition of the plane into connected regions having the same set of closest sites. This partition consists of *Voronoi cells, edges* and *vertices*.

If we select the vertices and edges (open line segments) of the polygon P to be the sites, then Voronoi vertices and edges clipped by P form the medial axis M. An edge of the medial axis can be either a line segment or a piece of parabola, see Fig. 1. The points of a line edge are equidistant from two edges of P. The points of a parabolic edge are equidistant from an edge of P and a non-convex vertex of P.

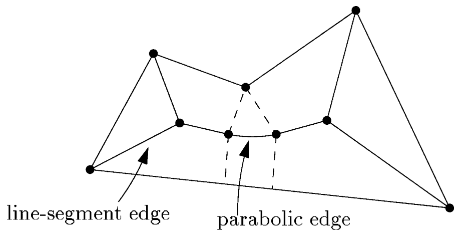

Fig. 1. Medial axis.

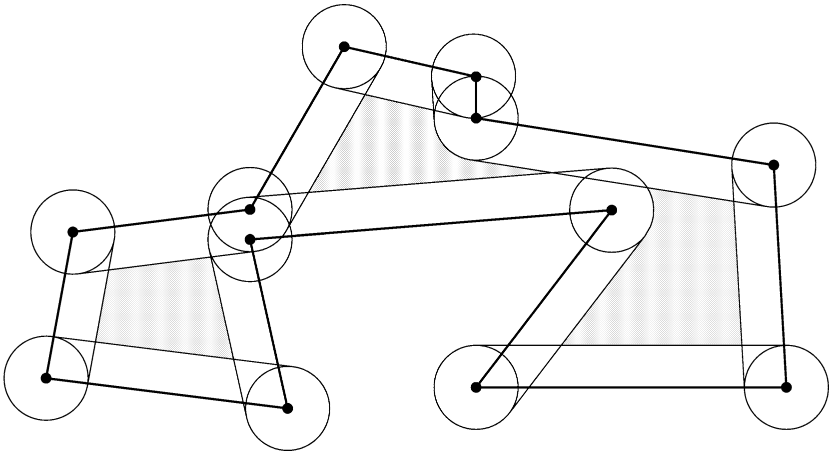

Fig. 2. Contracted region is shaded.

The medial axis can be computed in linear time by an algorithm of Chin et al. [12]. Their algorithm uses the technique of Aggarwal et al. [1] for computing a medial axis of a convex polygon. Note that the medial axis of a convex polygon P is a tree whose edges are line segments and leaves are the vertices of P.

The centers of two largest disks in a simple polygon belong to its medial axis [6]. Let δP denote the boundary of a polygon P. For a point $p \in P$, $D(p)$ denotes the distance from p to δP. For any $r > 0$, the *contracted region* $P(r)$ is the locus of points of the polygon P at distance at least r from δP.

Using the medial axis one can compute the distances $D(p)$ for all vertices p of M in linear time. The medial axis has a tree form. Each edge of M contains the points closest to the same sites (vertices and open edges of P) of the Voronoi diagram. To compute the distances $D(p)$ we find the associated sites for each edge of M (it can be done simultaneously with computing the medial axis). The sites can be found by contracting the tree of the medial axis M. The basic procedure is the deletion of leaves that are adjacent to same node.

The contracted region $P(r)$ can be disconnected, see Fig. 2. Each connected component of $P(r)$ is bounded by line segments and circle arcs. The vertices of $P(r)$ can be computed in linear time by inspecting edges of M. Each line-segment edge of M contains at most one vertex of $P(r)$. Note that a parabolic edge may contain two vertices of $P(r)$ [6].

3. Two disks in simple polygon

We apply the parametric searching technique [18]. The radius r of two disks is a parameter of the decision problem that asks whether there exists a pair of non-intersecting disks of the radius r in the polygon P. We obtain two algorithms, sequential and parallel, for the decision problem. Let T_s be the running time of the sequential decision algorithm. Let T_p and P' be the running time and the number of processors of the parallel algorithm, respectively. The parametric searching scheme allows us to solve the optimization problem, i.e., the two disk problem, in $O(P'T_p + T_pT_s \log P')$ time.

We show that the decision problem can be solved in $O(n)$ time using one processor and in $O(\log n)$ time using $O(n)$ processors. Hence two largest disks in a simple polygon can be found in $O(n \log^2 n)$ time.

3.1. Sequential algorithm

The idea of the algorithm is to construct the convex hull of the contracted polygon $P(r)$. $P(r)$ could be disconnected and we use connectivity induced by the medial axis M. Unfortunately the edges of $P(r)$ and segments connecting components of $P(r)$ that are induced by M, may intersect properly (not at endpoints). In spite of this we found a simple way to solve the decision problem in linear time. The algorithm contains four steps.

Algorithm `TwoDisks`(P, r).
Input: A simple polygon P with n vertices, a real number r.
Output: Answer whether there exist two disks of radius r in P.

1. Compute the vertices of the contracted region $P(r)$.
2. Connect each pair of consecutive (in topological order) vertices of $P(r)$ by a line segment. If the distance between consecutive vertices is at least $2r$, then the algorithm stops and replies "yes" (there are two non-intersecting disks of the radius r in the polygon P).
3. The segments form a simple closed chain C. Find the convex hull of the chain C.
4. Compute diameter of $CH(C)$. If the diameter is at least $2r$ then the answer is "yes". Otherwise the answer is "no".

Theorem 1. *The decision algorithm* `TwoDisks` *is correct and runs in linear time using linear space.*

Proof. The first step can be done in linear time by examining each edge of the medial axis. Each line-segment edge contains at most one vertex of $P(r)$. Note that parabolic edge can produce two vertices of $P(r)$.

In Step 2 the medial axis tree is traversed in topological order so that every edge is traversed twice, see Fig. 3 (note that $p_2 = p_7$ and $p_3 = p_6$). We connect every two consecutive vertices of $P(r)$ found along the traversal. If a parabolic edge produces two vertices we list them in the order according to the walk. If the algorithm does not stop in Step 2, then it builds the chain C of the segments (to make chain closed

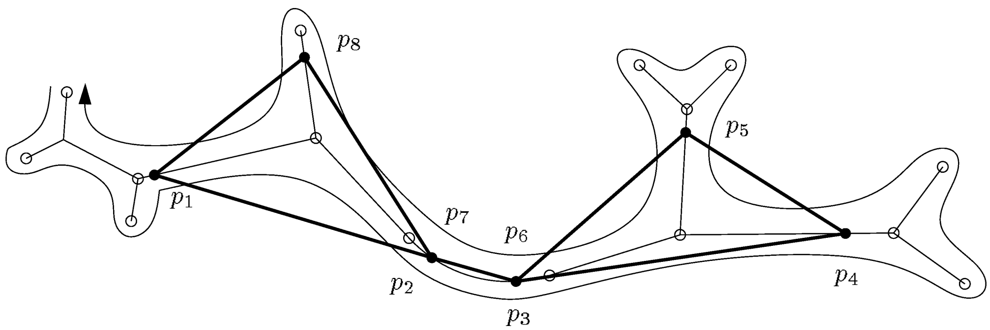

Fig. 3. Step 2 of `TwoDisks`. A walk of M produces closed chain $p_1 p_2 \ldots p_8 p_1$.

we connect the last found vertex and the first one, see Fig. 3) and its convex hull in Step 3. The segments of the chain do not intersect properly by Lemma 2. We use the linear algorithm [20] to construct the convex hull of the chain C. The convex hull of C and the convex hull of the vertices of $P(r)$ coincide by Lemma 3.

To compute the diameter of the set of vertices of $P(r)$ we apply calipers [21] to the convex hull $CH(C)$. The reply in Step 4 is correct by Lemma 3. □

Lemma 2. *If two edges of the chain C have lengths at most $2r$ then they do not intersect properly.*

Proof. Let $e_1 = (p_1, q_1)$ and $e_2 = (p_2, q_2)$ be two edges of the chain C such that their lengths are at most $2r$. We can assume that the segment e_2 is horizontal and the point p_1 is above the line p_2q_2, see Fig. 4. For any points p and q of the medial axis, let $\pi(p, q)$ denote the path of the medial axis connecting vertices p and q. The disks of radius r centered at points p_i, q_i lie in the polygon P because their centers are vertices of $P(r)$. Hence the segments $[p_i, q_i]$, $i = 1, 2$, lie inside the polygon P. Suppose that the segments $[p_1, q_1]$ and $[p_2, q_2]$ intersect properly.

Let R_i, $i = 1, 2$, denote the region bounded by the path $\pi(p_i, q_i)$ and the segment $[p_i, q_i]$ (if $\pi(p_i, q_i)$ intersects $[p_i, q_i]$, then the region R_i is defined as a set of points that cannot be moved away to infinity without crossing the path $\pi(p_i, q_i)$ or the segment $[p_i, q_i]$). The regions R_1 and R_2 lie inside the polygon P because their boundaries lie inside P.

The paths $\pi(p_1, q_1)$ and $\pi(p_2, q_2)$ do not intersect because, for each i, the points p_i and q_i are consecutive in topological order defined by the medial axis. Hence an endpoint of one of the edges e_i, $i = 1, 2$, is surrounded by another edge and the corresponding path. In other words, this endpoint, say p_1, belongs to the region R_2, see Fig. 4.

Consider the subtree of the medial axis that is defined as a set of points p of the medial axis such that the path $\pi(p, p_1)$ lies in the region R_2. Let p be a point of this subtree with largest y-coordinate. The point p does not belong to the path $\pi(p_2, q_2)$ due the topological order of choosing the segments $[p_i, q_i]$. The point p cannot be a vertex of the polygon P because it belongs to the region R_2 lying in the interior of P. The point p is the center of a disk that is contained in P and touches the boundary of P in two or more points. At least one of these points, say q, is above the horizontal line passing through p. The point q lies outside the region R_2. Hence the segment $[p, q]$ intersects the path $\pi(p_2, q_2)$.

Let p' be a point of intersection of the segment $[p, q]$ and the path $\pi(p_2, q_2)$. The point q is the closest point of the boundary of P to p'. The circle with center at p' and radius $|p' - q|$ touches the boundary of P in only one point q. Hence p' cannot belong to the medial axis. Contradiction. □

Lemma 3. *The diameter of $P(r)$ coincides with the diameter of the set of vertices of $P(r)$.*

Proof. The endpoints of a diameter of $P(r)$ belong to the boundary of the contracted region $P(r)$. The boundary of $P(r)$ consists of segments and arcs. If $P(r)$ is bounded by only segments, then $P(r)$ is a union of simple polygons and the furthest points are vertices of $P(r)$. In general case, if one of the points defining the diameter of $P(r)$ belongs to an arc, then it is an endpoint of the arc because the arc is concave, see Fig. 2. □

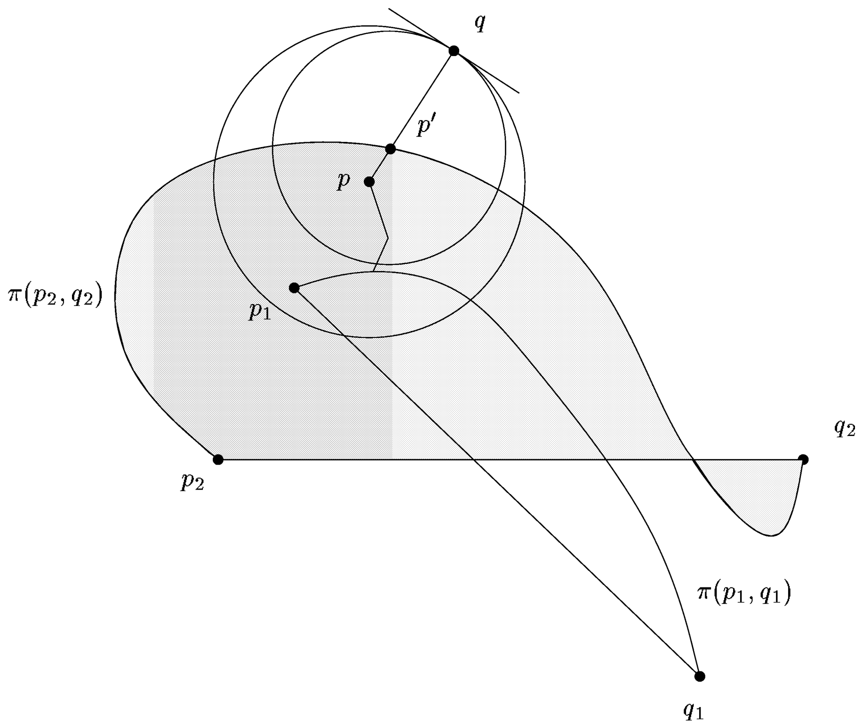

Fig. 4. Region R_2 is shaded.

3.2. Parallel algorithm

The parallel algorithm uses the medial axis of P and it can be difficult to compute the medial axis in $\mathrm{O}(\log n)$ time using $\mathrm{O}(n)$ processors. Instead, we precompute the medial axis in linear time [12] by sequential algorithm and assume that it is accessible in the parallel algorithm. In other words, we assume that the input of the decision algorithm includes the medial axis of P.

First, the parallel decision algorithm constructs the vertices of the contracted region $P(r)$ in O(1) time using $\mathrm{O}(n)$ processors. We assign one processor to each edge of the medial axes. An edge e of the medial axes can be either a line segment or a parabolic arc. The processor corresponding to the edge e computes one or two points of e at distance r from the boundary of the polygon. The convex hull of these points can be computed in $\mathrm{O}(\log n)$ time using an algorithm of Miller and Stout [19].

In order to compute the diameter of the convex polygon $CH(P(r))$ we can find *all furthest neighbors*, i.e., for each vertex p of the polygon, find its furthest neighbor among the vertices of the polygon. Clearly, the diameter is the largest distance from a vertex to its furthest neighbor. There are several papers describing parallel algorithms for computing all-furthest-neighbors in a convex polygon [2,9]. We apply a parallel algorithm of Atallah and Kosaraju [2] for computing all-furthest-neighbors in $\mathrm{O}(\log n)$ time using $\mathrm{O}(n)$ processors.

We conclude the following theorem.

Theorem 4. *Two largest disjoint disks in a simple polygon P can be found in $\mathrm{O}(n\log^2 n)$ time.*

4. Vertex folding line

Kim et al. [17] solved the folding line problem by adapting the parametric search technique. We show that one can avoid the parametric searching since the polygon has a simple shape (it is convex). We prove that the number of candidates for the folding line is at most linear although there are $\Omega(n^2)$ possible folding lines.

Lemma 5. *Let P be a convex polygon with n vertices. The number of optimal folding lines in P is at most n.*

Proof. Let $p_1, p_2, \ldots, p_n,\ n \geqslant 3$, be the vertices of P in clockwise order. We show that the number of optimal folding lines passing through a vertex p of P is at most 2. Without loss of generality $p = p_1$. Consider a directed line pp_i, $i = 3, 4, \ldots, n-1$. The directed line pp_i splits P into two polygons, the left polygon P_i and the right polygon Q_i, see Fig. 5. Let D_1 be the largest disk in the polygon P_i. Let D_2 be the largest disk in the polygon Q_i. Let $r_j,\ j = 1, 2$, be the radius of the disk D_j.

Roughly speaking, the idea is as follows. The left polygons grow when i increases. Hence the largest disk inscribed in the left polygon increases simultaneously with i. Similarly the largest disk in the right polygon strictly decreases when i increases. The minimum of two radii (left and right disks) defines concave sequence and the largest value is achieved at some unique folded line pp_i or two consecutive lines $pp_i,\ pp_{i+1}$. We give the detailed proof.

First we prove the property of the disks D_i that at least one of them touches the line pp_i. Indeed, the sum of the angles $\angle p_2pp_n$ and $\angle p_{i+1}p_ip_{i-1}$ (measured in clockwise order, see Fig. 5) is less than 2π. This implies that the sum of the angles $\angle p_2pp_i$ and $\angle pp_ip_{i-1}$ or the sum of the angles $\angle p_ipp_n$ and $\angle p_{i+1}p_ip$ is less than π (both sums can be less than π). Without loss of generality we can assume that $\angle p_2pp_i + \angle pp_ip_{i-1} < \pi$. If the disk D_1 is not tangent to the line pp_i then it can be translated in the direction of the bisector of lines p_2p and $p_{i-1}p_i$, see Fig. 6(a). Note that the size of translated disk can be increased. Contradiction.

We consider two cases comparing the radii $r_j,\ j = 1, 2$.

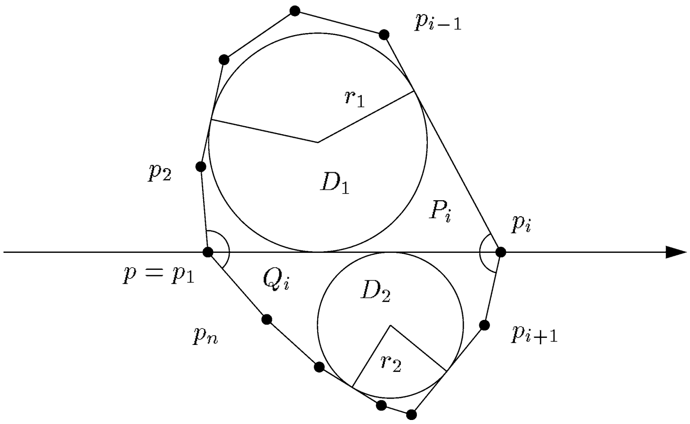

Fig. 5. Disks D_1 and D_2.

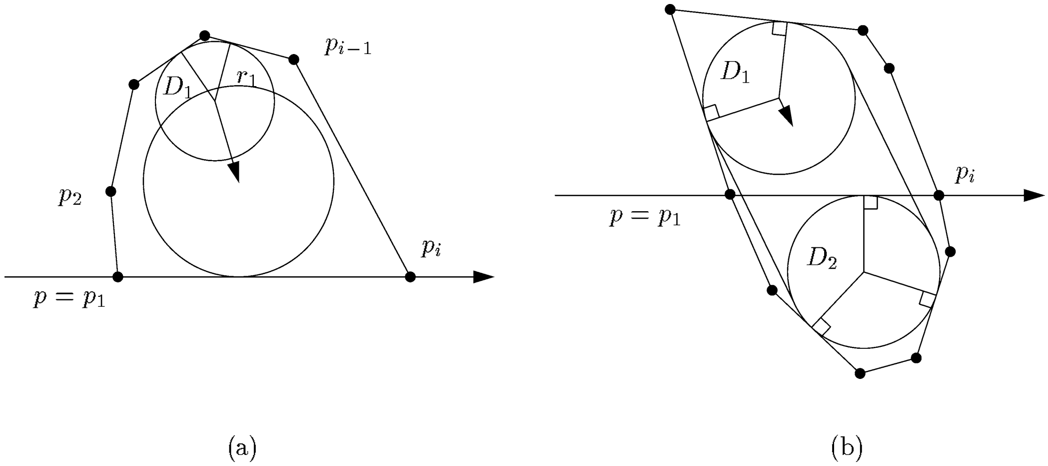

Fig. 6. Pushing disk D_1.

Case 1. $r_1 = r_2$. In this case, both disks D_i, $i = 1, 2$, touch the line pp_i. Suppose, on the contrary, that the disk D_1 does not touch the line pp_i. Consider the convex hull of the union of two disks. Its boundary contains two common tangent segments of the disks, see Fig. 6(b). This region is contained in P since P is convex polygon. The disk D_1 can be slightly translated in direction of the disk D_2 and radius of D_1 can be increased keeping it inside P and above pp_i. Contradiction.

In the case of the equal radii the line pp_i can be a unique optimal folding line passing through p (any other line pp_j, $j \neq i$, decreases the size of either D_1 or D_2).

Case 2. $r_1 \neq r_2$. We assume that $r_1 < r_2$. Consider the convex hull of two disks D_1 and D_2. Moving the disk D_1 towards D_2 and increasing linearly its radius keep the disk D_1 inside P. The only barrier preventing the motion is the diagonal pp_i. Thus, the disk D_1 touches the line pp_i. It implies that the largest disk in the polygon P_j, $j = 3, \ldots, i-1$, has smaller radius than r_1. Therefore there is at most one optimal folding line pp_i whose corresponding radii are in relation $r_1 < r_2$. Similarly there is at most one optimal folding line with $r_1 > r_2$.

The total number of optimal folding lines is at most n since every diagonal has two endpoints (the definition of the optimal folding lines is symmetric). Lemma follows. □

Theorem 6. *The folding line problem can be solved in* $O(n \log^2 n)$ *time using* $O(n)$ *space.*

Proof. The proof of Lemma 5 implies that, for every vertex $p \in P$, there are at most two folding lines maximizing $\min(r_1, r_2)$ over all folding lines pp_i. We call these lines *candidate* folding lines. The algorithm finds all the candidate folding lines. We show that, after preprocessing the points into a data structure, the candidate folding lines for a vertex of P can be found in $O(\log^2 n)$ time. Lemma 5 provides the binary search for finding a candidate folding lines. In fact the algorithm can answer a *line query* which asks, for a line l, sizes of the largest inscribed disks in the polygons obtained by cutting P along l. It suffices to make a query algorithm with $O(\log n)$ running time.

First, we construct the medial axis M in linear time [1]. The edges of the medial axis for a convex polygon are line segments. By a symbolic perturbation, we may assume that each internal vertex a of the tree M has degree 3. The point a is equidistant from three edges of P. Each edge b of M is the locus of points equidistant from two edges of P, say $e_1(b)$ and $e_2(b)$. The medial axis can be viewed as a binary tree if we chose any vertex of M as the root. Unfortunately, the binary tree can be unbalanced which makes a problem for efficient search.

We apply the balancing technique using *centroid decomposition*. A binary tree T can be split into two parts, each of size at least $\lfloor(|T|+1)/3\rfloor$, by removing a *centroid edge*. Guibas et al. [16] give an algorithm that computes a centroid decomposition of a binary tree T in linear time. The decomposition also can be obtained by a polygon cutting theorem of Chazelle [11].

We describe how to answer a line query using the centroid decomposition of M. Let l be a query line passing through vertices $p_1, p_j \in P$. Let P' be one of the polygons cut off by l. We want to find the largest disk inscribed in P'. Note that the center of any largest disk in P' (the largest disk could be not unique) belongs to the medial axis M since it touches at least three edges of P' and at least two of them differ from $p_1 p_j$ (and, therefore, define an edge of M). We find an edge of M, say e_D, containing the center of the largest disk in $O(\log n)$ time. Let e_M be the centroid edge of the tree M. The removal of e_M splits M into two subtrees M_1 and M_2. It is suffices to determine whether e_D belongs to $M_1 \cup \{e_M\}$ or $M_2 \cup \{e_M\}$ in O(1) time.

We observe the following property of the medial axis: the locus of the centers of the largest disks is either a vertex of M or an edge equidistant from two parallel edges of P. Let $L(M)$ denote the locus of the centers of the largest disks in P. Another useful property is that, for any path $\Pi \subset M$ from a vertex of P to a point of $L(M)$, the distances $D(p)$ for points $p \in \Pi$ define a monotone function.

Let D_1 and D_2 be the disks corresponding to the endpoints of e_M. Let r_i, $i = 1, 2$, be the radius of the disk D_i and let O_i be the center of D_i. Without loss of generality we assume that $r_1 \leqslant r_2$, see Fig. 7(a). We show how to locate e_D by considering the following three cases.

Case 1. The line l does not intersect the disk D_1.

Note that the line l does not intersect any disk of radius $D(p)$ centered at point p in a sufficiently small neighborhood of O_1. The radius $D(p)$ increases when it center moves from O_1 toward O_2. (Note that the radius does not change if $r_1 = r_2$. In this case, r_1 is the radius of the largest disk in P' since P' contains two parallel sides.) Therefore some part of the edge e_M containing O_1 is an edge of the medial axis of P'. By the monotonicity property of the medial axis, the subtree M_1 does not contain e_D and can be discarded.

Case 2. The line l intersect the disk D_1 but not the disk D_2.

By the monotonicity property of the medial axis M, the points of $M_1 \cup \{e_M\}$ are within distance r_2 from the boundary of P (the distance to $\delta P'$ can only decrease). Therefore $M_1 \cup \{e_M\}$ can be discarded.

Case 3. The line l intersects both the disk D_1 and the disk D_2.

There are two subcases depending on the relation between l and $e_i(e_M)$. Suppose that the edges $e_1(e_M)$ and $e_2(e_M)$ lie on the same side of the line l. Let A_1 and A_2 be the points of the edge $e_1(e_M)$ such that $O_i A_i$ is perpendicular to $e_1(e_M)$, see Fig. 7(b). Without loss of generality we can assume that $p_1 A_2 A_1 B_1 B_2 p_j$ is the clockwise order of points on the boundary of P'. Note that the interior of the quadrilateral $A_1 O_1 O_2 A_2$ (and $O_1 B_1 B_2 O_2$) does not intersect the medial axis M. The edge e_D is contained either in the polygon bounded by the closed path $A_1 \ldots B_1 O_1 A_1$ in clockwise order (the path between A_1 and B_1 lies on the boundary of P) or in the polygon bounded by $A_2 O_2 B_2 \ldots A_2$. The point O_1 is closer to the edge $e_1(e_M)$ than to the line l. There is a unique point a_1 of $A_1 O_1$ equidistant from $e_1(e_M)$ and l. The point a_1 can be computed in O(1) time. Note that the projection of O_1 onto l belongs to the edge $p_1 p_j$ since the disk of radius $|a_1 A_1|$ centered at a_1 is contained in the disk D_1 and, thus, it

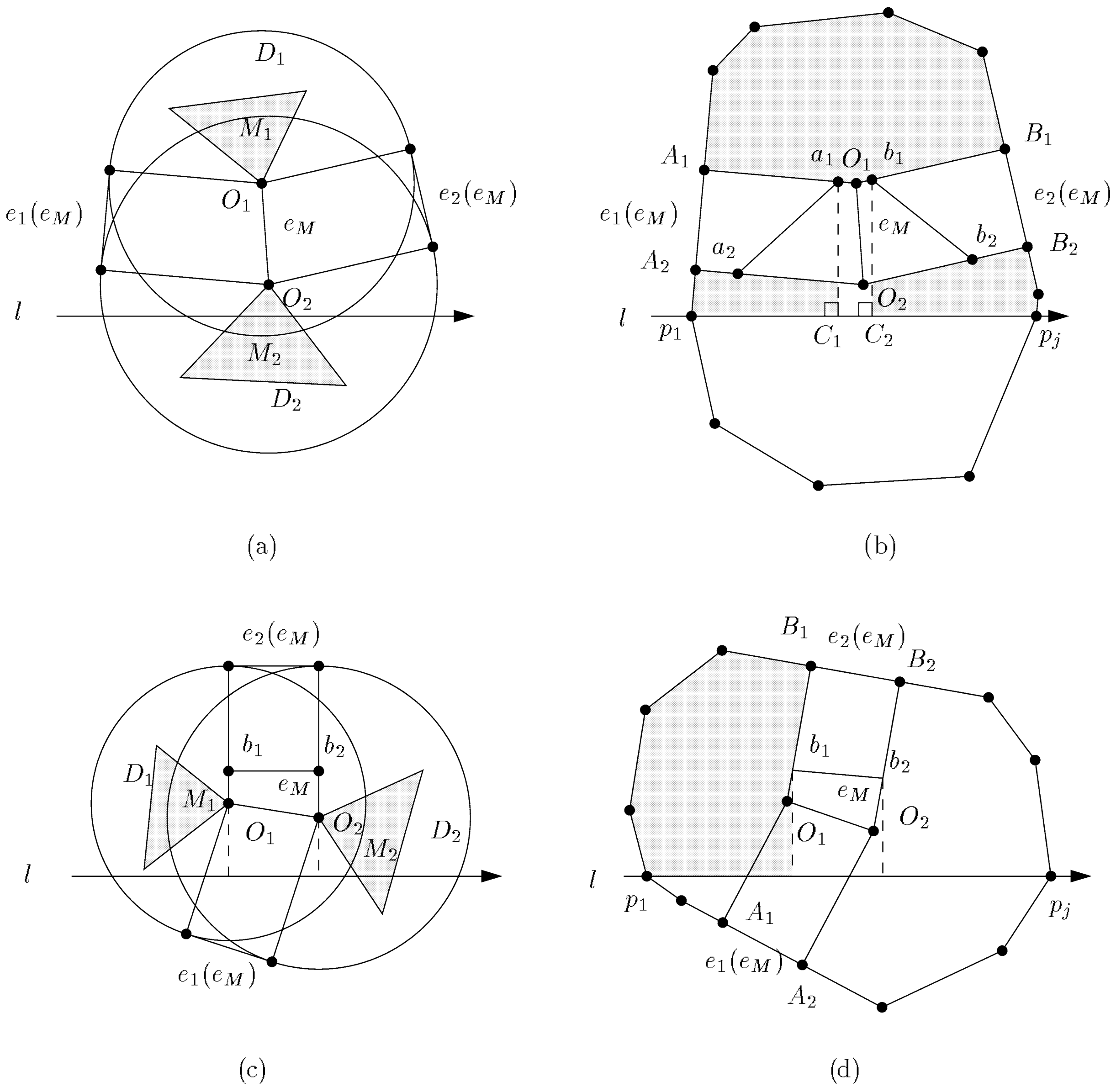

Fig. 7. Case 3 of line query.

cannot be tangent to a l at point outside of the edge $p_1 p_j$. Similarly, there is a unique point a_2 of $A_2 O_2$ equidistant from $e_2(e_M)$ and l.

The segment $a_1 a_2$ is contained in the medial axis of P'. Similarly, there is a unique segment $b_1 b_2$, $b_1 \in B_1$, $b_2 \in B_2$, which is a part of the medial axis of P'. The intersection of the 6-gon $A_1 O_1 B_1 B_2 O_2 A_2$ and the medial axis of P' is $[a_1 a_2] \cup [b_1 b_2]$. The medial axis of P' outside of this 6-gon consists of three connected subtrees located in the shaded areas in Fig. 7(b). We compute the distances from $a_i, b_i, i = 1, 2$

to l. We can decide which part contains the center of the largest disk inscribed in P'. We use one of the subtrees M_1 or M_1 to proceed the binary search.

Suppose that the line l separates the segments $e_1(e_M)$ and $e_2(e_M)$. Without loss of generality we can assume that the segment B_1B_2 is on the boundary of the polygon P', see Fig. 7(c). We define the points b_1 and b_2 as above. If the line l is parallel to the line B_1B_2, then the disk of radius $|b_1B_1|$ centered at b_1 is the largest disk in P'. We assume that the ray emanating from B_1 toward B_2 intersects l, see Fig. 7(d). Then $|b_1B_1| > |b_2B_2|$ and the largest disk in P' cannot be tangent to an edge of the chain $B_2 \ldots p_j$ in clockwise order by the monotonicity property of the medial axis of P'. Therefore we proceed the search in M_1. Similarly, if the ray B_2B_1 intersects l then we proceed the search in M_2.

Finally, only one edge of the medial axis M survives. The solution can be found as follows. We use the same notation as above. If the disk D_2 does not intersect the segment p_1p_j, then it is the largest disk in P. Otherwise, the disk D_2 intersects p_1p_j but the disk D_1 does not. There is a unique point of O_1O_2 that defines the center of the largest disk tangent to $e_1(e_M)$, $e_2(e_M)$ and p_1p_j. It can be computed in O(1) time. □

5. Conclusion

We presented two algorithms for packing two disks in a polygon. The first algorithm packs disks in a simple polygon and the second one packs disks in a convex polygon with the extra constraint that the disks are separated by a diagonal. Both algorithms run in $O(n \log^2 n)$ time. It would be interesting to improve these algorithms.

References

[1] A. Aggarwal, L.J. Guibas, J. Saxe, P.W. Shor, A linear-time algorithm for computing the Voronoi diagram of a convex polygon, Discrete Comput. Geom. 4 (6) (1989) 591–604.
[2] M.J. Atallah, S.R. Kosaraju, An efficient parallel algorithm for the row minima of a totally monotone matrix, J. Algorithms 13 (1992) 394–413.
[3] F. Aurenhammer, Voronoi diagrams: A survey of a fundamental geometric data structure, ACM Comput. Surv. 23 (3) (1991) 345–405.
[4] B. Ben-Moshe, M.J. Katz, M. Segal, Obnoxious facility location: complete service with minimal harm, Internat. J. Comput. Geom. Appl. 10 (6) (2000) 581–592.
[5] S. Bespamyatnikh, Efficient algorithm for finding two largest empty circles, in: Proc. 15th European Workshop on Comput. Geometry, 1999, pp. 37–38.
[6] T. Biedl, E. Demaine, M. Demaine, A. Lubiw, G. Toussaint, Hiding disks in folded polygons, in: Proc. 10th Canadian Conference on Comput. Geom., 1998, pp. 36–37.
[7] P. Bose, J. Czyzowich, E. Kranakis, A. Maheshwari, Algorithms for packing two circles in a convex polygon, in: Japan Conference on Discrete and Comput. Geom. (JCDCG98), Lecture Notes in Computer Sciences, Vol. 1763, Springer, 2000, pp. 93–103.
[8] P. Bose, P. Morin, A. Vigneron, Packing two disks into a polygonal environment, Report HKUST-TCSC-2001-02, Hong Kong University of Science and Technology, Department of Computer Science, 2001.
[9] P.G. Bradford, R. Fleischer, M. Smid, More efficient parallel totally monotone matrix searching, J. Algorithms 23 (1997) 386–400.
[10] P. Cappanera, Discrete facility location and routing of obnoxious activities, Report, Dip. di Matematica, Univ. di Milano, 1999 (Ph.D. Thesis).

[11] B. Chazelle, A theorem on polygon cutting with applications, in: Proc. 23rd Ann. IEEE Sympos. Found. Comput. Sci., 1982, pp. 339–349.
[12] F. Chin, J. Snoeyink, C.-A. Wang, Finding the medial axis of a simple polygon in linear time, Discrete Comput. Geom. 21 (3) (1999) 405–420.
[13] Z. Drezner, Location, in: P.M. Pardalos, M.G.C. Resende (Eds.), Handbook of Applied Optimization, Oxford University Press, 2001.
[14] Z. Drezner, E. Erkut, On the continuous p-dispersion problem, J. Oper. Res. Soc. 46 (1995) 516–520.
[15] Z. Drezner, H. Hamacher, Facility Location: Applications and Methods, 1st Edition, Springer, Berlin, 2001.
[16] L. Guibas, J. Hershberger, D. Leven, M. Sharir, R.E. Tarjan, Linear time algorithms for visibility and shortest path problems inside triangulated simple polygons, Algorithmica 2 (1987) 209–233.
[17] S.K. Kim, C.-S. Shin, T.-C. Yang, Placing two circles in a convex polygon, Inform. Process. Lett. 73 (1–2) (2000) 33–39.
[18] N. Megiddo, Applying parallel computation algorithms in the design of serial algorithms, J. ACM 30 (4) (1983) 852–865.
[19] R. Miller, Q.F. Stout, Efficient parallel convex hull algorithms, IEEE Trans. Comput. C-37 (12) (1988) 1605–1618.
[20] F.P. Preparata, M.I. Shamos, Computational Geometry: An Introduction, 3rd Edition, Springer, 1990.
[21] M.I. Shamos, Computational Geometry, Ph.D. Thesis, Department of Computer Science, Yale University, New Haven, CT, 1978.
[22] M.I. Shamos, D. Hoey, Closest-point problems, in: Proc. 16th Ann. IEEE Sympos. Found. Comput. Sci., 1975, pp. 151–162.