ELSEVIER

Information Processing Letters 69 (1999) 127–130

Information Processing Letters

Computing the optimal bridge between two convex polygons ☆

Leizhen Cai [a], Yinfeng Xu [b], Binhai Zhu [c,d,*]

[a] *Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong*
[b] *School of Management, Xi'an Jiaotong University, China*
[c] *Department of Computer Science, City University of Hong Kong, Kowloon, Hong Kong*
[d] *Department of Mathematics and Computer Science, Laurentian University, Ramsey Lake Road, Sudbury, Ontario, Canada P3E 2C6*

Received 1 August 1998; received in revised form 1 November 1998
Communicated by S.G. Akl

Abstract

We present an efficient algorithm for solving the following problem. Given two disjoint convex polygonal regions P, Q in the plane, add a line segment to connect them so as to minimize the maximum of the distances between points in one region and points in the other region. An $O(n^2 \log n)$ time algorithm is presented to find such a line segment (*optimal bridge*) (p, q), where n is the maximal cardinality of P, Q. We also present a very simple linear time constant factor approximate solution for this problem. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Computational geometry; Discrete optimization; Approximation

1. Introduction

In our daily life, given some separated regions, sometimes it is necessary to construct roads or bridges to connect them such that the length of a path between any two points in different regions is minimized (or is provably small). This problem can be thought of a variation of the Steiner tree problem [5] in two different ways: (1) we add a *Steiner* edge instead of a point; and (2) we want to minimize the maximum distance instead of the total distance. This note studies the simplest case for two separated convex polygonal regions.

It is easy to see that in order to determine the location of the line segment, we only need to consider the boundaries of the two regions. Therefore, we can formalize our problem as follows. Let P and Q be two disjoint convex polygons in the plane with boundaries ∂P and ∂Q, respectively. Denote the Euclidean distance between two points p and q by $d(p, q)$. We consider the problem of finding two points $p \in \partial P$ and $q \in \partial Q$ that minimize

$$\max_{p' \in \partial P} \{d(p', p)\} + d(p, q) + \max_{q' \in \partial Q} \{d(q, q')\}.$$

We refer to the line segment (p, q) as the *optimal bridge* of P and Q.

Let $V(P)$ and $V(Q)$ be the vertex sets of P and Q, respectively, and let n denote $\max\{|V(P)|, |V(Q)|\}$. In this paper we transform the above nonlinear optimization problem into a discrete optimization problem and then present an $O(n^2 \log n)$ algorithm to solve it. We also give a simple linear time algorithm that approximates the maximum distance within a factor of two.

☆ This research is supported by the UGC of Hong Kong, NSF of China, and City University of Hong Kong.

* Corresponding author.

0020-0190/99/$ – see front matter
PII: S0020-0190(99)00003-4

2. Geometric analysis and the algorithm

We first take a careful look at the optimal bridge (p, q). Clearly p and q must be *visible* to each other, i.e., the segment connecting p, q does not intersect the interior of P and Q. We might still have an infinite number of candidates. We first prove the following lemma.

Lemma 1. *p must be visible to q and must be falling in one of the following three cases:*

(1) *$p \in V(P)$, or*

(2) *there is $p^+ \in V(P)$ such that p is the intersection of the line segment (p^+, q) and the boundary of P, or*

(3) *p is the intersection point between ∂P and the bisector of two vertices $p_1, p_2 \in V(P)$.*

Proof. We remark that (1) is the degenerate case of (2) and (3). Without loss of generality, we assume q is already fixed. Assume that (p', q) is our optimal solution. As q is fixed we know that $d(p', q) + d(p', P)$ is minimized. Let $d(p', P) = d(p', p_j)$ and let p_i be some predecessor of p_j.

We consider the bisector of p_i, p_j and the line l containing the edge of P which contains p'. Assume their intersection is p^*. Elementary geometry shows that if q is within the sector centered at p^*, bounded by lines through (p_i, p^*), (p_j, p^*) and containing the bisector of p_i, p_j, then p' must be p^* (otherwise we can move p' to that position to minimize the maximum of $d(p', q) + d(p', p_i)$ and $d(p', q) + d(p', p_j)$, see Fig. 1). Note that if p' is not on ∂P then we might have a case degenerated to (1). Also, if q is out of the sector then one of p_i, p_j becomes p^+ (Fig. 2). □

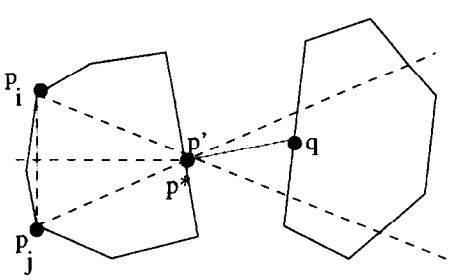

Fig. 1. Illustration for the proof of Lemma 1(1).

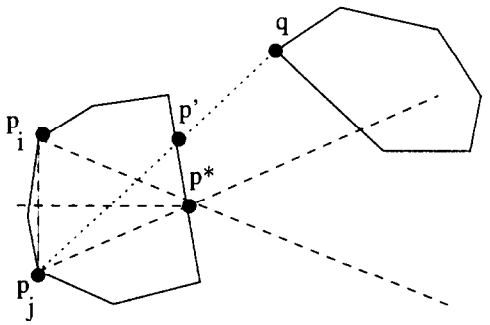

Fig. 2. Illustration for the proof of Lemma 1(2).

We would like to point out that the same conditions also hold for q. Lemma 1 enables us to obtain a polynomial time solution to compute the optimal bridge between P, Q. We now have the main result of this note.

Theorem 2. *The optimal bridge of P, Q can be computed in $O(n^2 \log n)$ time.*

Proof. With Lemma 1 we know that both p, q can be in one of the three situations. This will give us nine cases. Since all of them are very similar, we will only describe three cases in detail. We show the proof constructively in the following.

If both p, q fall into case (1), we enumerate all visible bridges between P, Q. This can be done easily in $O(n^2)$ time by finding the upper and lower tangents and then for each vertex of P find all the vertices of Q which are visible to it. Among them we compute the one, (p, q), such that

$$d(p, q) + \max_{p'' \in V(P)} \left[d(p, p'')\right] + \max_{q'' \in V(Q)} \left[d(q, q'')\right]$$

is minimized. We compute the farthest Voronoi Diagram for $V(P)$, $V(Q)$. Then we can use point location to find, for each vertex in $V(P)$ $(V(Q))$ its farthest neighbor in $V(Q)$ $(V(P))$ in $O(\log n)$ time. After this preprocessing, this step of computing the maximum distance between p'', q'' through p, q can be done in $O(1)$ time. Since we have $O(n^2)$ number of candidate bridges, all the computation in this step takes $O(n^2)$ time.

If p is in case (2) and q is in case (1), we first compute the tangents of P, Q. For all the vertices of P which are also on $CH(P \cup Q)$ we extend segments toward vertices in Q which are not on $CH(P \cup Q)$.

Among them we find the one which is the shortest. Assume this segment is (p^+, q). We compute the intersection of this segment with ∂P to obtain p. Again this process takes $O(n^2)$ time. (We can handle the situation when both p, q are in case (2) similarly.)

If both p, q are in case (3), then we have the most difficult situation. First of all we enumerate all the bisectors defined by two vertices in P (Q), then we compute the intersection of these bisectors with ∂P (∂Q). In fact we are only interested in those intersections which are visible to Q (P). We call these intersections *candidates* for P (Q). Certainly we have $O(n^2)$ number of candidates for p and q and they can be identified in $O(n^2 \log n)$ time (with some preprocessing on P, Q so that the ray shooting query takes $O(\log n)$ time). Clearly a brute force algorithm will force the algorithm to run in $O(n^4)$ time. We now show how to reduce the bound to $O(n^2 \log n)$.

First, for each candidate point in P (Q) we compute its farthest neighbor in P (Q). This can be done in $O(n^2 \log n)$ time. We first construct the farthest Voronoi Diagram [8] for the vertices of P (Q) and for each candidate point in P (Q) we perform a point location query in $O(\log n)$ time [4,6,7]. As the number of candidate points is $O(n^2)$, we have spent $O(n^2 \log n)$ time so far. Now for all candidate points in P we construct the *additively weighted* (nearest) Voronoi Diagram, $Vor(P)$ (under the L_2 metric) such that the initial positive weight for each candidate is the length of the distance from it to its farthest neighbor in P [2,3]. Finally we perform $O(n^2)$ number of point location queries in this Voronoi Diagram $Vor(P)$ for all the candidates in Q. We perform this symmetrically for Q. This way, we can find the weighted nearest neighbor among a pair of candidates in P, Q much faster: the additively weighted Voronoi Diagram can be constructed in $O(n^2 \log n^2) = O(n^2 \log n)$ time as it is transformable to the famous *power diagram* [1], point location queries also takes $O(\log n^2) = O(\log n)$ time (even though in the additively weighted Voronoi Diagram edges are parabola) [4,6,7]. Therefore, we can deal with this case in a total of $O(n^2 \log n)$ time.

Finally, among all the (p, q) candidates we simply find the one which minimizes the maximum distance between P, Q. □

3. A simple linear time constant factor approximate solution

In this section, we present a very simple factor-2 approximate solution for this problem. This algorithm involves no complicated subroutines like constructing additively weighted Voronoi Diagram and is very easy to implement. The algorithm is as follows.

Algorithm AppMinMax(P, Q).

(1) Compute the minimum distance (i.e., *separation*) between all the points of P and Q. Let p, q be the corresponding points on P, Q, respectively.
(2) Output (p, q) as the approximate solution for the optimal bridge.

The above algorithm certainly runs in $O(n)$ time as the computation of the separation between two convex polygons can easily be computed in $O(n)$ time with the hierarchical representations of P, Q [6]. The result is summarized as follows.

Theorem 3. AppMinMax(P, Q) *presents a factor-2 approximate solution to the optimal bridge between* P, Q.

Proof. Let $D(P)$, $D(Q)$ be the diameter of P, Q, respectively. Let (p_i, p^*), (p^*, q^*) and together with (q^*, q_j) be the optimal solution and the total optimal distance be $d_{\text{opt}}(P, Q)$. Without loss of generality, let

$$D(P) = d(p_1, p_2),$$

where $p_1, p_2 \in V(P)$. We claim that $d(p_i, p^*) \geqslant D(P)/2$, as $D(P)/2$ is the minmax distance for any point in the plane to reach p_1, p_2. Similarly we have $d(q^*, q_j) \geqslant D(Q)/2$. By the definition of the separation (p, q) we have $d(p^*, q^*) \geqslant d(p, q)$. Consequently

$$\begin{aligned} d_{\text{opt}}(P, Q) &= d(p_i, p^*) + d(p^*, q^*) + d(q^*, q_j) \\ &\geqslant D(P)/2 + d(p, q) + D(Q)/2. \end{aligned}$$

For any $p' \in V(P)$, $q' \in V(Q)$ we have $d(p', p) \leqslant D(P)$ and $d(q, q') \leqslant D(Q)$ (Fig. 3). Overall, we have

$$\begin{aligned} &d(p', p) + d(p, q) + d(q, q') \\ &\quad \leqslant D(P) + d(p, q) + D(Q) < 2d_{\text{opt}}(P, Q). \end{aligned}$$

□

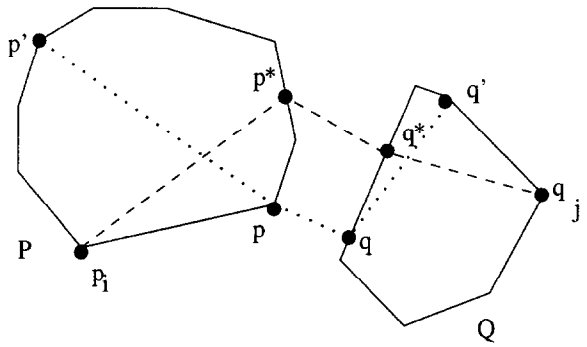

Fig. 3. Illustration for the proof of Theorem 3.

4. Remarks

In this note we present an $O(n^2 \log n)$ time solution to solve the optimal bridge problem. We also present a simple factor-2 approximate solution. It is interesting to know whether we can improve this upper bound as $\Omega(n)$ is the only known (trivial) lower bound. Another interesting question is whether we can have a better approximate solution.

References

[1] F. Aurenhammer, Power diagrams: Properties, algorithms and applications, SIAM J. Comput. 16 (1) (1987) 78–96.
[2] F. Aurenhammer, Voronoi diagrams: A survey of a fundamental geometric data structures, ACM Comput. Surveys 23 (3) (1991) 343–405.
[3] F. Aurenhammer, H. Imai, Geometric relations among Voronoi diagrams, Geom. Dedicata 27 (1988) 65–75.
[4] H. Edelsbrunner, L.J. Guibas, J. Stolfi, Optimal point location in a monotone subdivision, SIAM J. Comput. 15 (1986) 317–340.
[5] Z.A. Melzak, On the problem of Steiner, Canad. Math. Bull. 4 (1961) 143–148.
[6] F. Preparata, M. Shamos, Computational Geometry, Springer, Berlin, 1985.
[7] N. Sarnak, R.E. Tarjan, Planar point location using persistent search trees, Comm. ACM 29 (1986) 669–679.
[8] G. Toussaint, Computational Geometry, North-Holland, Amsterdam, 1985.