ARTICLE IN PRESS
COMGEO:1526

Computational Geometry ••• (••••) •••–•••

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

The geodesic 2-center problem in a simple polygon ☆,☆☆

Eunjin Oh [a], Jean-Lou De Carufel [b], Hee-Kap Ahn [a,*]

[a] *Department of Computer Science and Engineering, Pohang University of Science and Technology, Republic of Korea*
[b] *University of Ottawa, Canada*

ARTICLE INFO

Article history:
Received 1 November 2016
Accepted 14 February 2018
Available online xxxx

Keywords:
Simple polygons
k-center problem
Geodesic distance

ABSTRACT

In the *geodesic* 2*-center problem in a simple polygon* with n vertices, we find a set S of two points in the polygon that minimizes the maximum geodesic distance from any point of the polygon to its closest point in S. In this paper, we present an $O(n^2 \log^2 n)$-time algorithm for this problem using $O(n)$ space.

1. Introduction

The *k-center problem* can be stated as follows: Given a set $\mathcal{P}$ of n points in a metric space, find a set S of k points in the metric space, which we call a k-center of $\mathcal{P}$, that minimizes

$$\max_{p \in \mathcal{P}} \min_{s \in S} d(s, p),$$

where $d(x, y)$ denotes the distance between two points x and y. Computing a k-center of points is a typical problem in clustering. Clustering is the task of partitioning a given set into subsets subject to various objective functions, which have applications in pattern-analysis, decision-making, and machine-learning situations including data mining, document retrieval, and pattern classification [11].

The Euclidean k-center problem, that is the k-center problem under the Euclidean metric, has been studied extensively. The 1-center of a set $\mathcal{P}$ of points in $\mathbb{R}^d$ coincides with the center of the minimum enclosing ball of $\mathcal{P}$, which can be computed in linear time for any fixed dimension [14]. For a set of n points in the plane, Chan showed that its 2-center can be computed in $O(n \log^2 n \log^2 \log n)$ time [5] while its k-center can be computed in $O(n^{O(\sqrt{k})})$ time for $k > 2$ [10]. Moreover, for the case that k is part of input, it is NP-hard to approximate the Euclidean k-center problem within a factor smaller than 1.822 even for the planar case [7]. Kim and Shin presented an $O(n \log^3 n \log \log n)$-time algorithm for computing a 2-center of a convex n-gon in Euclidean metric.

In this paper, we consider the *geodesic k-center problem in a simple polygon P* with n vertices. Namely, we want to find a set S of k points in P, which we call a geodesic k-center of P, that minimizes

☆ A preliminary version of this paper appears in the proceedings of the 26th International Symposium on Algorithms and Computation (ISAAC 2015).
☆☆ This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the SW Starlab support program (IITP–2017–0–00905) supervised by the IITP (Institute for Information & communications Technology Promotion) and NRF grant 2011–0030044 (SRC–GAIA) funded by the government of Korea.
* Corresponding author.
E-mail addresses: jin9082@postech.ac.kr (E. Oh), jdecaruf@uottawa.ca (J.-L. De Carufel), heekap@postech.ac.kr (H.-K. Ahn).

https://doi.org/10.1016/j.comgeo.2018.02.008
0925-7721/

Please cite this article in press as: E. Oh et al., The geodesic 2-center problem in a simple polygon, Comput. Geom. (2018), https://doi.org/10.1016/j.comgeo.2018.02.008

$$\max_{p\in P}\min_{s\in S} d(s,p),$$

where $d(x, y)$ is the length of the shortest path between x and y lying in P (also called the *geodesic distance*). Geometrically, this is equivalent to finding k smallest-radius geodesic disks with the same radius whose union contains P.

The geodesic 1-center problem has been studied extensively. Asano and Toussaint presented the first algorithm for computing the geodesic 1-center of a simple polygon with n vertices in $O(n^4 \log n)$ time [3]. In 1989, the running time was improved to $O(n \log n)$ time by Pollack et al. [17]. Their technique can be described as follows. It first triangulates the polygon and finds the triangle T that contains the center in $O(n \log n)$ time. Then it subdivides T further and finds a region containing the center such that the combinatorial structure of the geodesic paths from each vertex of P to all points in that region is the same. Finally, the problem is reduced to finding the lowest point of the upper envelope of a family of convex distance functions in the region, which can be done in linear time using a technique by Megiddo [15]. Recently, the running time for computing the geodesic 1-center was improved to linear by Ahn et al. [1], which is optimal. In their paper, instead of triangulating the polygon, they construct a set of $O(n)$ chords. Then they recursively subdivide the polygon into $O(1)$ cells by a constant number of chords and find the cell containing the center. Finally, they obtain a triangle containing the center. In this triangle, they find the lowest point of the upper envelope of a family of functions, which is the geodesic 1-center of the polygon, using an algorithm similar to the one of Megiddo [15].

Surprisingly, there has been no result for the geodesic k-center problem for $k > 1$, except the one by Vigan [18]. They gave an algorithm for computing a geodesic 2-center in a simple polygon with n vertices, which runs in $O(n^8 \log n)$ time. The algorithm follows the framework of Kim and Shin [12]. However, it is unclear whether the algorithm works correctly. Specifically, they claim that the decision version of the geodesic 2-center problem in a simple polygon can be solved using a technique similar to the one by Kim and Shin [12] without providing any proof or analysis on the time complexity. They apply parametric search using their decision algorithm, but they do not describe how their algorithm works in the claimed running time. It is again unclear whether the parallel algorithm by Kim and Shin extends to this problem.

The geodesic k-center problem can be extended in a *polygonal domain* of total complexity n, that is, a polygon with holes of total complexity n. In this setting, the problem becomes harder. Specifically, the best upper bound known on the total number of geodesic centers is $O(n^{10})$ in a polygonal domain of total complexity n [19] while there is a unique geodesic center in a simple polygon. This is because a shortest path in a polygonal domain is not unique. The best known algorithm for computing all geodesic centers in a polygonal domain of complexity n takes $O(n^{11} \log n)$ time [19].

1.1. Our results

In this paper, we present an $O(n^2 \log^2 n)$-time algorithm that solves the geodesic 2-center problem in a simple polygon with n vertices. We first observe that a simple polygon P can always be partitioned into two regions by a geodesic path $\pi(x, y)$ such that

- x and y are two points on the boundary of P, and
- the set consisting of the geodesic 1-centers of the two regions of P into which $\pi(x, y)$ divides P is a geodesic 2-center of P.

Then we consider $O(n)$ candidate pairs of edges of P such that one of them, namely (e, e'), satisfies $x \in e$ and $y \in e'$. We find these candidate pairs of edges in $O(n^2 \log n)$ time. Finally, we compute a 2-center restricted to each such pair of edges in $O(n \log^2 n)$ time.

2. Preliminary

A polygon P is said to be *simple* if it is bounded by a closed path, all vertices are distinct, and the edges intersect only at their endpoints. The vertices of a simple n-gon P are labeled $v_1, \ldots, v_n$ in clockwise order along the boundary of P. We set $v_{n+k} = v_k$ for all $k \geq 1$. An edge whose endpoints are v_i and v_{i+1} is denoted by e_i. For any two points x and y lying inside a simple polygon P, the *geodesic path* between x and y, denoted by $\pi(x, y)$, is the shortest path inside P between x and y. The length of $\pi(x, y)$ is called the *geodesic distance* between x and y, denoted by $d(x, y)$. The geodesic path between any two points in P is unique. The geodesic distance and the geodesic path between x and y can be computed in $O(\log n)$ and $O(\log n + k)$ time, respectively, after an $O(n)$-time preprocessing, where k is the number of vertices on the geodesic path [8]. The vertices of $\pi(x, y)$ excluding x and y are reflex vertices of P and they are called the *anchors* of $\pi(x, y)$. If $\pi(x, y)$ is a line segment, it has no anchor. In this paper, "distance" refers to geodesic distance unless specified otherwise.

Given a set X of points in P, we use ∂X to denote the boundary of X. A set $X \subseteq P$ is *geodesically convex* if $\pi(x, y) \subset X$ for any two points x and y in X. For any two distinct points u and w on ∂P, let $C[u, w]$ be the part of ∂P in clockwise order from u to w. For any point u on ∂P, let $C[u, u]$ be u itself. The subpolygon of P bounded by $C[u, w]$ and $\pi(u, w)$ is denoted by $P[u, w]$. It is not necessarily simple because a vertex of $\pi(u, w)$ might also appear on $C[u, w]$. However, we can apply the algorithms [1,2,8,9] for computing the geodesic center, the farthest-point geodesic Voronoi diagram, the shortest-path tree and the shortest path between two points to $P[u, w]$ without increasing their running times. To see this, observe that we can construct a larger polygon $\bar{P}[u, w]$ by slightly perturbing the vertices of $\pi(u, w)$ that also appear