Computational Geometry 57 (2016) 1–7

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Analysis of farthest point sampling for approximating geodesics in a graph

Pegah Kamousi [a], Sylvain Lazard [b], Anil Maheshwari [c], Stefanie Wuhrer [d,*]

[a] *Département d'Informatique, Université Libre de Bruxelles, Belgium*
[b] *Inria, Loria, Nancy, France*
[c] *School of Computer Science, Carleton University, Canada*
[d] *Inria Grenoble Rhône-Alpes, France*

ARTICLE INFO

Article history:
Received 31 October 2013
Received in revised form 12 February 2016
Accepted 4 May 2016
Available online 11 May 2016

Keywords:
Farthest Point Sampling
Approximate geodesics
Shortest paths
Planar graphs
Approximation algorithms

ABSTRACT

A standard way to approximate the distance between two vertices p and q in a graph is to compute a shortest path from p to q that goes through one of k sources, which are well-chosen vertices. Precomputing the distance between each of the k sources to all vertices yields an efficient computation of approximate distances between any two vertices. One standard method for choosing k sources is the so-called *Farthest Point Sampling* (FPS), which starts with a random vertex as the first source, and iteratively selects the farthest vertex from the already selected sources.

In this paper, we analyze the stretch factor $\mathcal{F}_{\text{FPS}}$ of approximate geodesics computed using FPS, which is the maximum, over all pairs of distinct vertices, of their approximated distance over their geodesic distance in the graph. We show that $\mathcal{F}_{\text{FPS}}$ can be bounded in terms of the minimal value $\mathcal{F}^*$ of the stretch factor obtained using an optimal placement of k sources as $\mathcal{F}_{\text{FPS}} \leqslant 2r_e^2\mathcal{F}^* + 2r_e^2 + 8r_e + 1$, where r_e is the length ratio of longest edge over the shortest edge in the graph. We further show that the factor r_e is not an artefact of the analysis by providing a class of graphs for which $\mathcal{F}_{\text{FPS}} \geqslant \frac{1}{2}r_e\mathcal{F}^*$.

1. Introduction

In the context of shape analysis, it is commonly required to compute and analyze geodesics between many pairs of vertices on a shape that is represented by a connected undirected graph G with n vertices and m edges. To compute shortest-path queries from a single-source on G, Dijkstra's algorithm [4] takes $O(n\log n + m)$ time. To compute all-pairs shortest-paths, we can run Dijkstra's algorithm starting from each of the n vertices and, while there are more efficient methods, this problem has a trivial $\Omega(n^2)$ lower bound.

To reduce this complexity, the problem of efficiently *approximating* the distance between any two vertices is often considered. A very recent method efficiently computes a $(1+\epsilon)$-approximation to a single such query in a planar graph in $O((\log\log n)^3/\epsilon^2 + \log\log\sqrt{\log\log((\log\log n)/\epsilon^2)}/\epsilon^2)$ time and $O(n((\log\log n)^2/\epsilon + (\log\log n)/\epsilon^2)))$ space [17].

In contrast to this work that builds a carefully chosen data structure, we are interested in a class of simple algorithms, commonly used in practice, that compute in a pre-processing phase a set $S = \{s_1, \ldots, s_k\}$ of k vertices, called *sources*, in G

* Corresponding author.
E-mail addresses: pegahk@gmail.com (P. Kamousi), sylvain.lazard@inria.fr (S. Lazard), anil@scs.carleton.ca (A. Maheshwari), stefanie.wuhrer@inria.fr (S. Wuhrer).

http://dx.doi.org/10.1016/j.comgeo.2016.05.005
0925-7721/

and runs Dijkstra's algorithm from each of them in $O(k(n\log n+m))$ total time. Then, the distance between any two vertices p and q is approximated as the minimum, over all k sources, of the distance from p to q through one of the sources. The quality of the worst approximation is characterized by the stretch factor, defined as the maximum, over all pairs of distinct vertices, of their approximated distance over their geodesic distance in the graph, that is

$$\mathcal{F} = \max_{(p,q)\in V,\ p\neq q} \min_{s_i\in S} \frac{d(p,s_i)+d(s_i,q)}{d(p,q)},$$

where V denotes the set of vertices of G and where the function $d(.,.)$ measures the shortest geodesic distance between two vertices. Throughout this paper, we use for simplicity the notation $\max_{p,q}$ for $\max_{(p,q)\in V,\ p\neq q}$.

A natural problem is thus to compute an optimal placement of k sources that yields a minimum stretch factor, denoted $\mathcal{F}^*$. We refer to this problem as the *k-center path-dilation problem*. This problem is *NP*-complete even for planar graphs because the existence of at most k sources so that the stretch factor is 1 is trivially equivalent to the existence of a vertex cover of size at most k, which is *NP*-complete even for planar graphs [7].[1] Furthermore, we show in [10] that the k-center path-dilation problem is also *NP*-complete in the case of planar triangle graphs (i.e., connected graphs whose faces have three edges and whose edges are incident to at most two faces), which are of particular interest for shape analysis.

For computing a set of at most k sources, Könemann et al. [12, Thm. 3] present a simple algorithm that yields a stretch factor $\mathcal{F}_K \leqslant 2\mathcal{F}^*+1+\epsilon$ in time $O(k(n\log n+m)\log(nr_e/\epsilon))$ for any $\epsilon>0$, where r_e is the lengths ratio of the longest over the shortest edges in G. For convenience of the reader, we detail in Section 2 their algorithm and proof because they missed the ϵ term and did not state the complexity.

In this work, we analyze the stretch factor of the even simpler and commonly used *Farthest Point Sampling* (FPS) heuristic for selecting a set of k sources [9,14]. FPS starts by selecting a random vertex and iteratively selects a vertex that has the largest geodesic distance to its closest already selected source, until k sources are picked. Running Dijkstra's algorithm from each of the sources directly yields a total running in $O(k(n\log n+m))$.

To the best of our knowledge, no theoretical results are known on the quality of the stretch factor, $\mathcal{F}_{\mathrm{FPS}}$, obtained by an FPS of k sources, compared to the minimal stretch factor $\mathcal{F}^*$. In this paper, we prove that for any connected undirected graph and any choice of k sources obtained by the FPS algorithm,

$$\mathcal{F}_{\mathrm{FPS}} \leqslant 2r_e^2(\mathcal{F}^*+1)+8r_e+1,$$

where r_e is the lengths ratio of the longest over the shortest edges in G (Theorem 1). We further show that the factor r_e is not an artefact of the analysis by providing a family of graphs for which $\mathcal{F}_{\mathrm{FPS}} \geqslant \frac{1}{2}r_e\mathcal{F}^*$ (Theorem 6). This shows that if the ratio r_e is large, $\mathcal{F}_{\mathrm{FPS}}$ can be much larger than the optimal stretch factor $\mathcal{F}^*$ but, on the other hand, $\mathcal{F}^*$ is likely to be large as well. Indeed, consider a graph with $k+1$ arbitrarily small edges such that all their adjacent edges are long enough: at least one of these edges is not incident to a source and for the endpoints of this edge, their approximated distance over their geodesic distance is arbitrarily large. Note that all our bounds also hold in the case where the edge lengths ratio r_e is defined only by pairs of edges that belong to one and the same shortest path in G.

The relevance of our bounds on $\mathcal{F}_{\mathrm{FPS}}$ is to give some theoretical insight on why FPS has been used successfully in heuristics for shape processing. However, it should be stressed that the edge lengths ratio r_e appears in the upper and lower bounds on $\mathcal{F}_{\mathrm{FPS}}$ but not on $\mathcal{F}_K$. Still, r_e appears in the running time for the latter and not for the former, but since it appears in logarithmic form, it is fair to expect that Könemann et al.'s algorithm would give better results in terms of the combination of stretch factors and running times than the widely used FPS algorithm. Nevertheless, we are not aware of any experimental study on the subject.

After discussing related work in Section 2, we prove our main results, Theorems 1 and 6 in Sections 3.1 and 3.2 respectively.

2. Related work

Computing geodesics on polyhedral surfaces is a well-studied problem for which we refer to the recent survey by Bose et al. [2]. While much work on surface processing compute geodesics that are allowed to pass through the interior of faces, much work also restrict geodesics to go through vertices and edges, as they are easy to compute. In this paper, we restrict geodesics to be shortest paths along edges of the underlying graph.

The FPS algorithm has been used for a variety of surface processing tasks. The algorithm was first introduced for graph clustering [9], and later independently developed for 2D images [6] and extended to 3D meshes [14]. This sampling strategy has been used to efficiently compute approximate geodesic distances [1,8], to recognize the class of an input shape [5,13], and to compute point-to-point correspondences between surfaces [3,16,15]. In practice, $\mathcal{F}_{\mathrm{FPS}}$ and r_e are typically reasonably small. For instance, for the Greek head model consisting of 6607 vertices used by Ruggeri and Saupe [15], it is $r_e=15.4$ and $\mathcal{F}_{\mathrm{FPS}}=18.8$ for $k=500$. In our implementation, we computed $\mathcal{F}_{\mathrm{FPS}}$ by averaging over five sets of randomly chosen sources computed using FPS.

[1] A *NP*-hardness proof for general graphs can also be found in [11, §2] but, as it comes as a corollary of another *NP*-hardness reduction, it is less trivial and it inherently uses non-planar graphs.