Computational Geometry 47 (2014) 651–667

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

Minimum-link paths revisited

Joseph S.B. Mitchell [a], Valentin Polishchuk [b,c,*], Mikko Sysikaski [b]

[a] Department of Applied Mathematics and Statistics, Stony Brook University, United States
[b] Helsinki Institute for Information Technology, Department of Computer Science, University of Helsinki, Finland
[c] Communications and Transport Systems, ITN, Linköping University, Norrköping, Sweden

A R T I C L E   I N F O

Article history:
Received 21 August 2011
Received in revised form 12 July 2012
Accepted 9 February 2013
Available online 30 December 2013
Communicated by M. Hoffmann and E. Welzl

Keywords:
Path planning
Link distance
Approximations
3SUM-hardness

A B S T R A C T

A path or a polygonal domain is C*-oriented* if the orientations of its edges belong to a set of C given orientations; this is a generalization of the notable rectilinear case ($C = 2$). We study exact and approximation algorithms for minimum-link C-oriented paths and paths with unrestricted orientations, both in C-oriented and in general domains.

Our two main algorithms are as follows:

A subquadratic-time algorithm with a non-trivial approximation guarantee for general (unrestricted-orientation) minimum-link paths in general domains.

An algorithm to find a minimum-link C-oriented path in a C-oriented domain. Our algorithm is simpler and more time-space efficient than the prior algorithm.

We also obtain several related results:

- 3SUM-hardness of determining the link distance with unrestricted orientations (even in a rectilinear domain).
- An optimal algorithm for finding a minimum-link rectilinear path in a rectilinear domain. The algorithm and its analysis are simpler than the existing ones.
- An extension of our methods to find a C-oriented minimum-link path in a general (not necessarily C-oriented) domain.
- A more efficient algorithm to compute a 2-approximate C-oriented minimum-link path.
- A notion of “robust” paths. We show how minimum-link C-oriented paths approximate the robust paths with unrestricted orientations to within an additive error of 1.

1. Introduction

Minimum-link problems arise in motion planning with turn costs, in line simplification, guarding applications, VLSI, wireless communication, and other areas. An instance of the problem is specified by an n-vertex polygonal domain P with h holes, and two points $s, t \in P$; the goal is to find an s–t path with the fewest edges (links). In the query version of the problem, the goal is to build a data structure (link distance map) to efficiently answer link distance queries with s fixed.

The algorithm of Mitchell, Rote and Woeginger [25] computes a minimum-link path in $O(n^2\alpha^2(n)\log n)$ time, where α is the inverse Ackermann function. It was believed that a faster algorithm is possible (e.g., in [5, p. 263] the result of [25] is called “suboptimal”). Nevertheless, the only previously known lower bound, also due to [25], was $\Omega(n \log n)$. The same bounds for the *rectilinear* case are given in [5,22]. Also, no approximation algorithm was previously known.

* Corresponding author.
E-mail addresses: joseph.mitchell@stonybrook.edu (J.S.B. Mitchell), valentin.polishchuk@liu.se (V. Polishchuk), mikko.sysikaski@helsinki.fi (M. Sysikaski).

0925-7721/$ – see front matter
http://dx.doi.org/10.1016/j.comgeo.2013.12.005

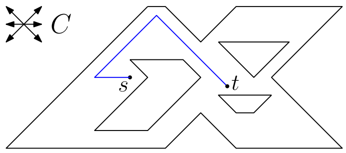

Fig. 1. A C-oriented domain and a minimum-link C-oriented path in it.

In this paper (Section 2) we give a subquadratic-time $O(\sqrt{h})$-approximation algorithm for the minimum-link path problem. We also observe (Theorem 2.1) that finding the exact solution is 3SUM-hard; this answers a question from the survey [26] and Problem 22 in The Open Problems Project [7].

Our 3SUM-hardness proof suggests that the problem's complexity stems from allowing the path edges to go in arbitrary directions. This—along with practical considerations—motivates the restricted, C-*oriented* setting [1,11–13,28,32,37] (Fig. 1) in which orientations of path edges come from a fixed set C of directions. (Abusing notation, we use C to denote also the cardinality of the set C.) Adegeest, Overmars and Snoyeink [1] presented two algorithms for finding minimum-link C-oriented paths in C-oriented domains—one running in $O(C^2 n \log n)$ time and space, the other in $O(C^2 n \log^2 n)$ time and $O(C^2 n)$ space.

In Section 3 we present an $O(C^2 n \log n)$-time $O(Cn)$-space algorithm, slightly improving on both algorithms from [1].[1] As a by-product, in Section 3.1, we reestablish the optimal time and space bounds claimed in [39] for computing a minimum-link *rectilinear* path amid rectilinear obstacles. Unlike the earlier papers on the rectilinear case, we use only elementary data structures, which simplifies the algorithm and its analysis. We also show how to find a C-oriented path in a general domain (Section 3.3.1), give an $O(Cn \log n)$-time $O(n)$-space 2-approximation algorithm for C-oriented paths (Section 3), and investigate in what sense C-oriented paths can approximate minimum-link paths with unrestricted orientation (Section 3.3.3).

All of our algorithms not only find minimum-link paths but also build, within the same time and space bounds, the corresponding link distance maps—exact or approximate. For instance, using our algorithms, one can construct approximate (additive or multiplicative) maps for general minimum-link paths in general domains in subquadratic time and linear space. This is in contrast with the exact link distance maps, which may have quartic complexity [35].

2. Paths with unrestricted orientations

The 3SUM-hardness of finding a minimum-link path can be seen easily, as we now observe. Start from an instance of the 3SUM-hard problem GeomBase considered in [8]: Given a set S of points lying on 3 parallel lines l_1, l_2, l_3, do there exist 3 points from S lying on a line $l \notin \{l_1, l_2, l_3\}$? Construct an instance of the minimum-link path problem as follows (Fig. 2, left): l_1, l_2, l_3 become obstacles, and each point $p \in S$ is a gap punched in the obstacle. The s–t link distance is 3 if and only if there exist 3 collinear gaps p_i, $i = 1, 2, 3$, such that $p_i \in l_i$.

We thus obtain:

Theorem 2.1. *Determining the link distance, for paths with unrestricted orientations, between two points of a polygonal domain with holes is 3SUM-hard. In particular, it is 3SUM-hard to decide if there exists a* 3*-link path between two points in a rectilinear domain.*

Remark 1. One can decide if the link distance between points s and t is 1 in time $O(n)$ (just test the segment st for intersection with each edge of the domain). One can test if the s–t link distance is $\leqslant 2$ in time $O(n \log n)$ (just compute the visibility polygons with respect to s and t, in time $O(n \log n)$, and test them for intersection, in time $O(n)$). One can test if the s–t link distance is $\leqslant 3$ in time $O(n^2)$ (assuming the visibility polygons with respect to s and t are disjoint, construct the visibility graph within the domain obtained by subtracting the two visibility polygons and check if there exists an (extended) visibility graph edge with endpoints on each of the two visibility polygons).

Several corollaries are immediate: finding a $4/3 - \varepsilon$ multiplicative approximation is 3SUM-hard; there is little hope to design an output-sensitive algorithm that would spend $o(n^2)$ time per link in the optimal path; computing an additive-1 approximation is 3SUM-hard; obstacles having few orientations of edges do not make the problem simpler, etc.

The proof can be strengthened to show that obtaining an $O(1)$ additive approximation is 3SUM-hard, and that obtaining a $(2 - \varepsilon)$ multiplicative approximation is equally hard. To see this, let $k = O(1)$ be an arbitrary integer. Take an instance of GeomBase of size n/k, and make k copies of it. Place the copies in a k-channel ("zig-zag") corridor with k channels, with one copy per channel (Fig. 2, right). There is a path utilizing a single link per channel (i.e., one link per copy of the instance) if and only if the GeomBase instance is feasible, otherwise 2 links per copy are needed. That is, if GeomBase is feasible, the s–t path will have $2 + k$ links, otherwise it will have $2 + 2k$ links. Distinguishing between the two cases is at least as hard as solving the GeomBase instance of size $\Theta(n/k) = \Theta(n)$.

Hence, we have:

[1] The algorithm was also presented at WADS 2011 [31].