International Journal of Foundations of Computer Science
© World Scientific Publishing Company

ONLINE ROUTING IN CONVEX SUBDIVISIONS*

PROSENJIT BOSE

School of Computer Science, Carleton University Ottawa, K1S 5B6, Canada

ANDREJ BRODNIK

IMFM, University of Ljubljana, Ljubljana, Slovenia and Department of Computer Science, Luleå Technical University, SE-971 87 Luleå, Sweden

SVANTE CARLSSON

University of Karlskona/Ronneby, 371 41 KARLSKRONA, Sweden

ERIK D. DEMAINE

Massachusetts Inst. Tech., Lab. for Computer Science Cambridge, MA, 02139, USA

RUDOLF FLEISCHER

Department of Computer Science The Hong Kong University of Science and Technology, Kowloon Hong Kong

ALEJANDRO LÓPEZ-ORTIZ

Department of Computer Science, University of Waterloo, Waterloo, N2L 3G1, Canada

PAT MORIN

School of Computer Science, Carleton University Ottawa, K1S 5B6, Canada

and

J. IAN MUNRO

Department of Computer Science, University of Waterloo, Waterloo, N2L 3G1, Canada

Received (received date)
Revised (revised date)
Communicated by Editor's name

*This research was funded by the Natural Sciences and Engineering Research Council of Canada.

ABSTRACT

We consider online routing algorithms for finding paths between the vertices of plane graphs. We show (1) there exists a routing algorithm for arbitrary triangulations that has no memory and uses no randomization, (2) no equivalent result is possible for convex subdivisions, (3) there is no competitive online routing algorithm under the Euclidean distance metric in arbitrary triangulations, and (4) there is no competitive online routing algorithm under the link distance metric even when the input graph is restricted to be a Delaunay, greedy, or minimum-weight triangulation.

Keywords: Online algorithms, routing, oblivious algorithms, computational geometry

1. Introduction

Path finding, or routing, is central to a number of fields including geographic information systems, urban planning, robotics, and communication networks. In many cases, knowledge about the environment in which routing takes place is not available beforehand, and the vehicle/robot/packet must learn this information through exploration. Algorithms for routing in these types of environments are referred to as *online*[1] routing algorithms.

In this paper we consider online routing in the following abstract setting[2]: The environment is a plane graph, G (i.e., the planar embedding of G) with n vertices. The source s and destination t are vertices of G, and a packet can only travel on edges of G. Initially, a packet only knows the coordinates of s, t, and $N(s)$, where $N(v)$ denotes the set of vertices adjacent to a node v. When a packet visits a node v, it learns the coordinates of $N(v)$.

Bose and Morin[2] classify routing algorithms based on their use of memory and/or randomization. A deterministic routing algorithm is *memoryless* or *oblivious* if, given a packet currently at vertex v and destined for node t, the algorithm decides where to forward the packet based only on the coordinates of v, t and $N(v)$. A randomized algorithm is oblivious if it decides where to move a packet based only on the coordinates of v, t, $N(v)$, and the output of a random oracle. An algorithm $\mathcal{A}$ is *defeated* by a graph G if there exists a pair of vertices $s, t \in G$ such that a packet stored at s will never reach t when being routed using $\mathcal{A}$. Otherwise, we say that $\mathcal{A}$ *works* for G.

Let $\mathcal{A}(G, s, t)$ denote the length of the walk taken by routing algorithm $\mathcal{A}$ when travelling from vertex s to vertex t of G, and let $SP(G, s, t)$ denote the length of the shortest path between s and t. We say that $\mathcal{A}$ is *c-competitive* for a class of graphs $\mathcal{G}$ if

$$\frac{\mathcal{A}(G, s, t)}{SP(G, s, t)} \leq c$$

for all graphs $G \in \mathcal{G}$ and all $s, t \in G$, $s \neq t$. We say that $\mathcal{A}$ is simply *competitive* if $\mathcal{A}$ is c-competitive for some constant c.

Recently, several papers have dealt with online routing and related problems in geometric settings. Kalyanasundaram and Pruhs[3] give a 16-competitive algorithm to *explore* any unknown plane graph, i.e., visit all of its nodes. This online exploration problem makes the same assumptions as those made here, but the goal of the problem is to visit all vertices of G, not just t. This difference leads to inherently different solutions.

Kranakis *et al.*[4] give a deterministic oblivious routing algorithm that works for any Delaunay triangulation, and give a deterministic non-oblivious algorithm that works for any connected plane graph.

Bose and Morin[2] also study online routing in geometric settings, particularly triangulations. They give a randomized oblivious routing algorithm that works for any triangulation, and ask whether there is a deterministic oblivious routing algorithm for all triangulations. They also give a competitive non-oblivious routing algorithm for Delaunay triangulations.

Cucka et al.[5] experimentally evaluate the performance of routing algorithms very similar to those described by Kranakis *et al.*[4] and Bose and Morin.[2] When considering the Euclidean distance travelled during point-to-point routing, their results show that the GREEDY routing algorithm[2] performs better than the COMPASS routing algorithm[2,4] on random graphs, but does not do as well on Delaunay triangulations of random point sets.[a] However, when one considers not the Euclidean distance, but the number of edges traversed (link distance), then the COMPASS routing algorithm is slightly more efficient for both random graphs and Delaunay triangulations.

In this paper we present a number of new fundamental theoretical results that help further the understanding of online routing in plane graphs.

1. We give a deterministic oblivious routing algorithm for all triangulations, solving the open problem posed by Bose and Morin.[2]

2. We prove that no deterministic oblivious routing algorithm works for all convex subdivisions, showing some limitations of deterministic oblivious routing algorithms.

3. We prove that the randomized oblivious routing algorithm RANDOM-COMPASS described by Bose and Morin[2] works for any convex subdivision.

4. We show that, under the Euclidean metric, no routing algorithm exists that is competitive for all triangulations, and under the link distance metric, no routing algorithm exists that is competitive for all Delaunay, greedy, or minimum-weight triangulations.

The remainder of the paper is organized as follows: In Section 2 we give our deterministic oblivious algorithm for routing in triangulations. Section 3 presents our results for routing in convex subdivisions. Section 4 describes our impossibility results for competitive algorithms. Finally, Section 5 summarizes and concludes with open problems.

[a]Cucka *et al.* call these algorithms P-DFS and D-DFS, respectively.

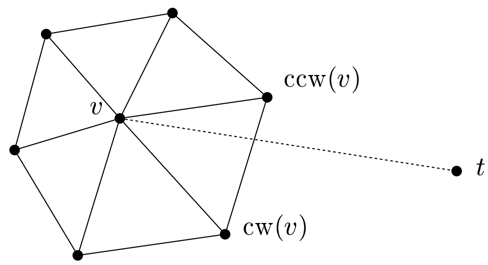


Figure 1: Definition of cw(v) and ccw(v).

2. Oblivious Routing in Triangulations

A *triangulation* T is a plane graph for which every face is a triangle, except the outer face, which is the complement of a convex polygon. In this section we describe a deterministic oblivious routing algorithm that works for all triangulations. The algorithm is a carefully designed combination of two existing algorithms.[2] The GREEDY algorithm always moves a packet to a neighbouring node that minimizes the distance to t. The COMPASS algorithm always moves a packet to the node that is most "inline" with t. Both these algorithms are defeated by certain types of triangulations, but the ways in which they are defeated are very different. By combining them, we obtain an algorithm that works for any triangulation.

We use the notation $\angle a, b, c$ to denote the angle formed by a b and c as measured in the counterclockwise direction. Let cw(v) be the vertex in $N(v)$ which minimizes the angle $\angle cw(v), v, t$ and let ccw(v) be the vertex in $N(v)$ which minimizes the angle $\angle t, v, \mathrm{ccw}(v)$. If v has a neighbour w on the line segment (v, t), then $\mathrm{cw}(v) = \mathrm{ccw}(v) = w$. In particular, the vertex t is contained in the wedge $\mathrm{cw}(v), v, \mathrm{ccw}(v)$. Refer to Fig. 1 for an illustration.

The GREEDY-COMPASS algorithm always moves to the vertex among $\{\mathrm{cw}(v), \mathrm{ccw}(v)\}$ that minimizes the distance to t. If the two distances are equal, or if $\mathrm{cw}(v) = \mathrm{ccw}(v)$, then GREEDY-COMPASS chooses one of $\{\mathrm{cw}(v), \mathrm{ccw}(v)\}$ arbitrarily.

Theorem 1 *Algorithm* GREEDY-COMPASS *works for any triangulation.*

Proof. Suppose, by way of contradiction that a triangulation T and a pair of vertices s and t exist such that GREEDY-COMPASS does not find a path from s to t.

In this case there must be a cycle of vertices $C = \langle v_0, \ldots, v_{k-1} \rangle$ of T such that GREEDY-COMPASS moves from v_i to v_{i+1} for all $0 \leq i \leq k$, i.e., GREEDY-COMPASS gets trapped cycling through the vertices of C (see also Lemma 1 of Bose and Morin[2]).[b] Furthermore, it follows from Lemma 2 of Bose and Morin[2] that the destination t is contained in the interior of C.

Claim 1 *All vertices of C must lie on the boundary of a disk D centered at t.*

Proof. (of claim) Suppose, by way of contradiction, that there is no such disk

[b] Here, and in the remainder of this proof, all subscripts are taken mod k.

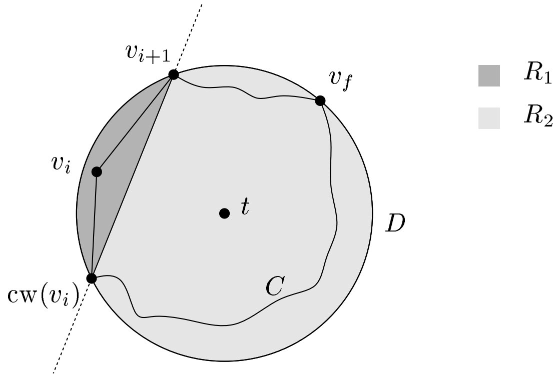


Figure 2: The proof of Theorem 1.

D. Then let D be the disk centered at t and having the furthest vertex of C from t on its boundary. Consider a vertex v_i in the interior of D such that v_{i+1} is on the boundary of D. (Refer to Fig. 2.) Assume, w.l.o.g., that $v_{i+1} = \mathrm{ccw}(v_i)$. Then it must be that $\mathrm{cw}(v_i)$ is not in the interior of D, otherwise GREEDY-COMPASS would not have moved to v_{i+1}. But then the edge $(\mathrm{cw}(v_i), \mathrm{ccw}(v_i))$ cuts D into two regions, R_1 containing v_i and R_2 containing t. Since C passes through both R_1 and R_2 and is contained in D then it must be that C enters region R_1 at $\mathrm{cw}(v_i)$ and leaves R_1 at $v_{i+1} = \mathrm{ccw}(v_i)$. However, this cannot happen because both $\mathrm{cw}(\mathrm{cw}(v_i))$ and $\mathrm{ccw}(\mathrm{cw}(v_i))$ are contained in the halfspace bounded by the supporting line of $(\mathrm{cw}(v_i), \mathrm{ccw}(v_i))$ and containing t, and are therefore not contained in R_1. □

Thus, we have established that all vertices of C are on the boundary of D. However, since C contains t in its interior and the triangulation T is connected, it must be that for some vertex v_j of C, $\mathrm{cw}(v_j)$ or $\mathrm{ccw}(v_j)$ is in the interior of D. Suppose that it is $\mathrm{cw}(v_j)$. But then we have a contradiction, since the GREEDY-COMPASS algorithm would have gone to $\mathrm{cw}(v_j)$ rather than v_{j+1}. □

3. Oblivious Routing in Convex Subdivisions

A *convex subdivision* is an embedded plane graph such that each face of the graph is a convex polygon, except the outer face which is the complement of a convex polygon. Triangulations are a special case of convex subdivisions in which each face is a triangle; thus it is natural to ask whether the GREEDY-COMPASS algorithm can be generalized to convex subdivisions. In this section, we show that there is no deterministic oblivious routing algorithm for convex subdivisions. However, there is a randomized oblivious routing algorithm that uses only one random bit per step.

3.1. Deterministic Algorithms

Theorem 2 *Every deterministic oblivious routing algorithm is defeated by some* *convex subdivision.*

Proof. We exhibit a finite collection of convex subdivisions such that any deterministic oblivious routing algorithm is defeated by at least one of them.

There are 17 vertices that are common to all of our subdivisions. The destination vertex t is located at the origin. The other 16 vertices $V = \{v_0, \dots, v_{15}\}$ are the vertices of a regular 16-gon centered at the origin and listed in counterclockwise order.[c] In all our subdivisions, the even-numbered vertices $v_0, v_2, \dots, v_{14}$ have degree 2. The degree of the other vertices varies. All of our subdivisions contain the edges of the regular 16-gon.

Assume, by way of contradiction, that there exists a routing algorithm $\mathcal{A}$ that works for any convex subdivision. Since the even-numbered vertices in our subdivisions always have the same two neighbours in all subdivisions, $\mathcal{A}$ always makes the same decision at a particular even-numbered vertex. Thus, it makes sense to ask what $\mathcal{A}$ does when it visits an even-numbered vertex, without knowing anything else about the particular subdivision that $\mathcal{A}$ is routing on.

For each vertex $v_i \in V$, we color v_i black or white depending on the action of $\mathcal{A}$ upon visiting v_i, specifically, black for moving counterclockwise and white for moving clockwise around the regular 16-gon. We claim that all even-numbered vertices in V must have the same color. If not, then there exists two vertices v_i and v_{i+2} such that v_i is black and v_{i+2} is white. Then, if we take $s = v_i$ in the convex subdivision shown in Fig. 3.a, the algorithm becomes trapped on one of the edges (v_i, v_{i+1}) or (v_{i+1}, v_{i+2}) and never reaches the destination t, contradicting the assumption that $\mathcal{A}$ works for any convex subdivision.

Therefore, assume w.l.o.g. that all even-numbered vertices of V are black, and consider the convex subdivision shown in Fig. 3.b. From this figure it is clear that, if we take $s = v_1$, $\mathcal{A}$ cannot visit x after v_1, since then it gets trapped among the vertices $\{v_{12}, v_{13}, v_{14}, v_{15}, v_0, v_1, x\}$ and never reaches t.

Note that we can rotate Fig. 3.b by integral multiples of $\pi/4$ while leaving the vertex labels in place and make similar arguments for v_3, v_5, v_7, v_9, v_{11}, v_{13} and v_{15}. However, this implies that $\mathcal{A}$ is defeated by the convex subdivision shown in Fig. 3.c since if it begins at any vertex of the regular 16-gon, it never enters the interior of the 16-gon. We conclude that no oblivious online routing algorithm works for all convex subdivisions. $\square$

We note that, although our proof uses subdivisions in which some of the faces are not strictly convex (i.e., have vertices with interior angle π), it is possible to modify the proof to use only strictly convex subdivisions, but doing so leads to more cluttered diagrams. These diagrams are shown in Fig. 4. We leave the details to the interested reader.

3.2. Randomized Algorithms

[c] In the remainder of this proof, all subscripts are implicitly taken mod 16.

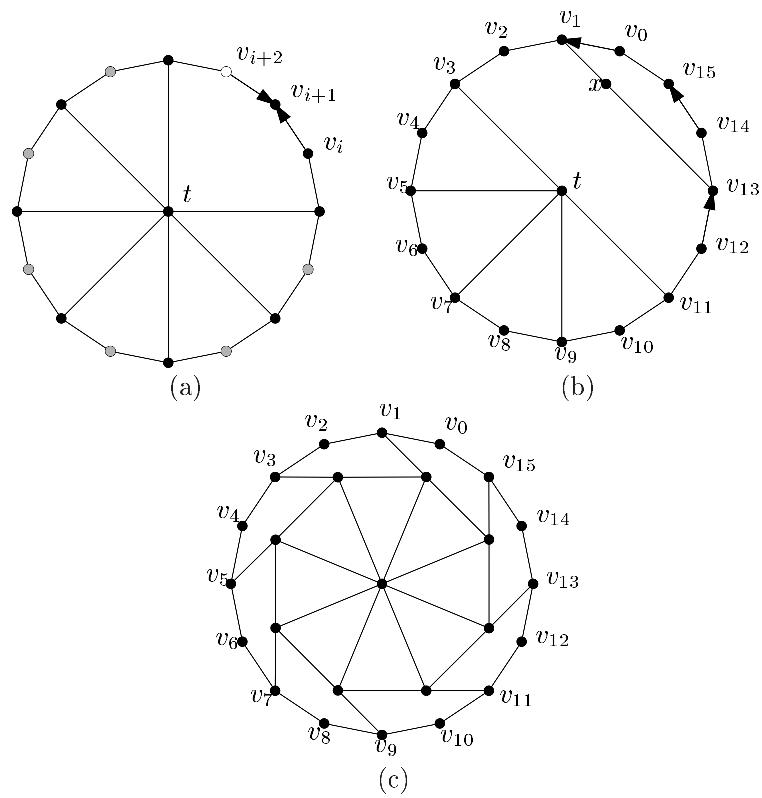


Figure 3: The proof of Theorem 2.

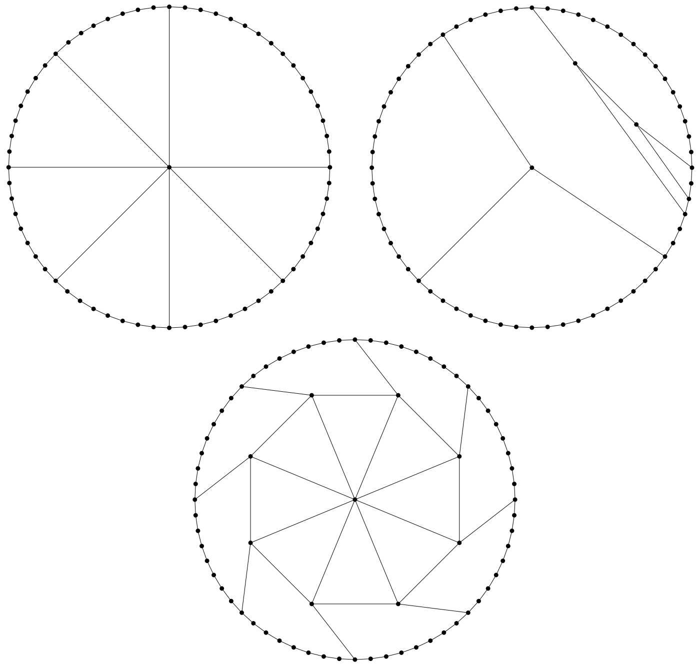


Figure 4: Strictly convex subdivisions that can be used in the proof of Theorem 2.

Bose and Morin[2] describe the RANDOM-COMPASS algorithm and show that it works for any triangulation. For a packet stored at node v, the RANDOM-COMPASS algorithm selects a vertex from $\{\mathrm{cw}(v), \mathrm{ccw}(v)\}$ uniformly at random and moves to it. In this section we show that RANDOM-COMPASS works for any convex subdivision.

Although it is well known that a random walk on any graph G will eventually visit all vertices of G, the RANDOM-COMPASS algorithm has two advantages over a random walk. The first advantage is that the RANDOM-COMPASS algorithm is more efficient in its use of randomization than a random walk. It requires only one random bit per step, whereas a random walk requires $\log k$ random bits for a vertex of degree k. The second advantage is that the RANDOM-COMPASS algorithm makes use of geometry to guide it, and the result is that RANDOM-COMPASS generally arrives at t much more quickly than a random walk. Nevertheless, it can be helpful to think of RANDOM-COMPASS as a random walk on a directed graph in which every node has out-degree 1 or 2 except for t which is a sink.

Before we can make statements about which graphs defeat RANDOM-COMPASS, we must define what it means for a graph to defeat a randomized algorithm. We say that a graph G defeats a (randomized) routing algorithm if there exists a pair of vertices s and t of G such that a packet originating at s with destination t has probability 0 of reaching t in any finite number of steps. Note that, for oblivious algorithms, proving that a graph does not defeat an algorithm implies that the algorithm will reach its destination with probability 1.

Theorem 3 *Algorithm* RANDOM-COMPASS *works for any convex subdivision.*

Proof. Assume, by way of contradiction, that there is a convex subdivision G with two vertices s and t such that the probability of reaching s from t using RANDOM-COMPASS is 0. Then there is a subgraph H of G containing s, but not containing t, such that for all vertices $v \in H$, $\mathrm{cw}(v) \in H$ and $\mathrm{ccw}(v) \in H$.

The vertex t is contained in some face f of H. We claim that this face must be convex. For the sake of contradiction, assume otherwise. Then there is a reflex vertex v on the boundary of f such that the line segment (t, v) does not intersect any edge of H. However, this cannot happen, since $\mathrm{ccw}(v)$ and $\mathrm{cw}(v)$ are in H, and hence v would not be reflex.

Since G is connected, it must be that for some vertex u on the boundary of f, $\mathrm{cw}(u)$ or $\mathrm{ccw}(u)$ is contained in the interior of f. But this vertex in the interior of f is also in H, contradicting the fact that f is a convex face of H. We conclude that there is no convex subdivision that defeats RANDOM-COMPASS. □

4. Competitive Routing Algorithms

If we are willing to accept more sophisticated routing algorithms that make use of memory, then it is sometimes possible to find competitive routing algorithms. Bose and Morin[2] give a competitive algorithm for Delaunay triangulations under the Euclidean distance metric. Two questions arise from this: (1) Can this result be generalized to arbitrary triangulations? and (2) Can this result be duplicated for the link distance metric? In this section we show that the answer to both these questions is negative.

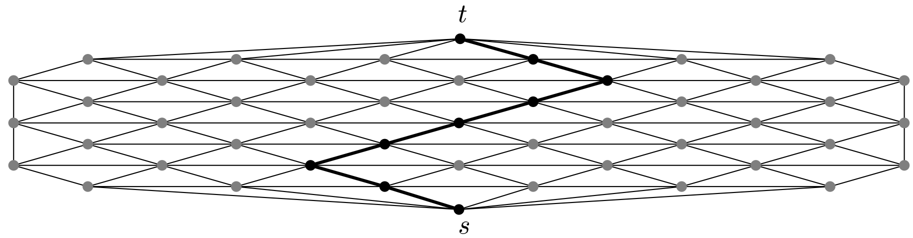
4.1. Euclidean Distance

In this section we show that, under the Euclidean metric, no deterministic routing algorithm is $o(\sqrt{n})$-competitive for all triangulations. Our proof is a modification of that used by Papadimitriou and Yannakakis[6] to show that no online algorithm for finding a destination point among n axis-oriented rectangular obstacles in the plane is $o(\sqrt{n})$-competitive.

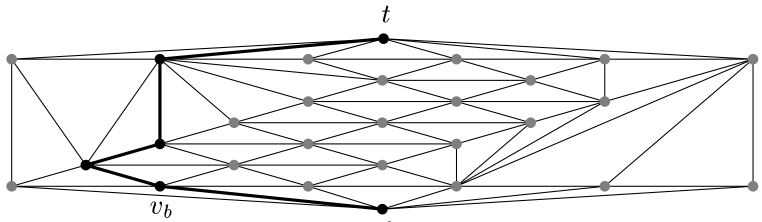
Theorem 4 *Under the Euclidean distance metric, no deterministic routing algorithm is $o(\sqrt{n})$ competitive for all triangulations.*

Proof. Consider an $n \times n$ hexagonal lattice with the following modifications. The lattice has had its x-coordinates scaled so that each edge is of length $\Theta(n)$. The lattice also has two additional vertices, s and t, centered horizontally, at one unit below the bottom row and one unit above the top row, respectively. Finally, all vertices of the lattice and s and t have been completed to a triangulation T. See Fig. 5.a for an illustration.

Let $\mathcal{A}$ be any deterministic routing algorithm and observe the actions of $\mathcal{A}$ as it routes from s to t. In particular, consider the first $n+1$ steps taken by $\mathcal{A}$ as it



(a)



(b)

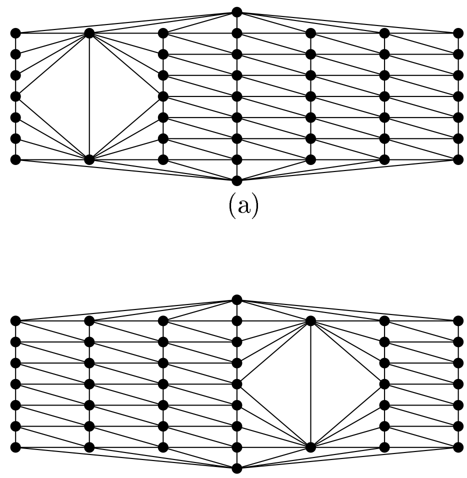
Figure 5: (a) The triangulation T with the path found by $\mathcal{A}$ indicated. (b) The resulting triangulation T' with the "almost-vertical" path shown in bold.

routes from s to t. Then $\mathcal{A}$ visits at most $n+1$ vertices of T, and these vertices induce a subgraph T_{vis} consisting of all vertices visited by $\mathcal{A}$ and all edges adjacent to these vertices.

For any vertex v of T not equal to s or t, define the *x-span* of v as the interval between the rightmost and leftmost x-coordinate of $N(v)$. The length of any x-span is $\Theta(n)$, and the width of the original triangulation T is $\Theta(n^2)$. This implies that there is some vertex v_b on the bottom row of T whose x-coordinate is at most $n\sqrt{n}$ from the x-coordinate of s and is contained in $O(\sqrt{n})$ x-spans of the vertices visited in the first $n+1$ steps of $\mathcal{A}$.

We now create the triangulation T' that contains all vertices and edges of T_{vis}. Additionally, T' contains the set of edges forming an "almost vertical" path from v_b to the top row of T'. This almost vertical path is a path that is vertical wherever possible, but uses minimal detours to avoid edges of T_{vis}. Since only $O(\sqrt{n})$ detours are required, the length of this path is $O(n\sqrt{n})$. Finally, we complete T' to a triangulation in some arbitrary way that does not increase the degrees of vertices on the first $n+1$ steps of $\mathcal{A}$. See Fig. 5.b for an example.

Now, since $\mathcal{A}$ is deterministic, the first $n+1$ steps taken by $\mathcal{A}$ on T' will be the same as the first $n+1$ steps taken by $\mathcal{A}$ on T, and will therefore travel a distance of $\Theta(n^2)$. However, there is a path in T' from s to t that first visits v_b (at a cost of $O(n\sqrt{n})$), then uses the "almost-vertical" path to the top row of T' (at a cost of $O(n\sqrt{n})$) and then travels directly to t (at a cost of $O(n\sqrt{n})$). Thus, the total cost of this path, and hence the shortest path, from s to t is $O(n\sqrt{n})$.



(a)

(b)

Figure 6: The point sets (a) $S_{49,2}$ and (b) $S_{49,5}$ along with their Delaunay triangulations.

We conclude that $\mathcal{A}$ is not $o(\sqrt{n})$-competitive for T'. Since the choice of $\mathcal{A}$ is arbitary, and T' contains $O(n)$ vertices, this implies that no deterministic routing algorithm is $o(\sqrt{n})$ competitive for all triangulations with n vertices. $\square$

4.2. Link Distance

The link distance metric simply measures the number of edges traversed by a routing algorithm. For many networking applications, this metric is more meaningful than Euclidean distance. In this section we show that competitive algorithms under the link distance metric are harder to come by than under the Euclidean distance metric. Throughout this section we assume that the reader is familiar with the definitions of Delaunay, greedy and minimum-weight triangulations (*cf.* Preparata and Shamos[7]).

We obtain this result by constructing a "bad" family of point sets as follows. Let C_i be the set of $\sqrt{n}$ points $\{(i\sqrt{n}, 1), (i\sqrt{n}, 2), \ldots, (i\sqrt{n}, \sqrt{n})\}$. We call C_i the *ith column.* Let $D_i = \{(i\sqrt{n}, 1), (i\sqrt{n}, \sqrt{n})\}$, and define a family of point sets $S = \bigcup_{j=1}^{\infty}\{S_{j^2}\}$ where $S_n = \{S_{n,1}, \ldots, S_{n,\sqrt{n}}\}$ and

$$S_{n,i} = \bigcup_{j=1}^{i-1} C_j \cup D_i \cup \bigcup_{j=i+1}^{\sqrt{n}} C_j \cup \{(\sqrt{n}/2, 0), (\sqrt{n}/2, \sqrt{n}+1)\} \tag{1}$$

Two members of the set S_{49} are shown in Fig. 6.

Theorem 5 *Under the link distance metric, no routing algorithm is $o(\sqrt{n})$-competitive for all Delaunay triangulations.*

Proof. We use the notation $DT(S_{n,i})$ to denote the Delaunay triangulation of $S_{n,i}$. Although the Delaunay triangulation of $S_{n,i}$ is not unique, we will assume $DT(S_{n,i})$ is triangulated as in Fig. 6. Note that, in $DT(S_{n,i})$, the shortest path between the topmost vertex s and bottom-most vertex t is of length 3, independent of n and i. Furthermore, any path from s to t whose length is less than $\sqrt{n}$ must visit vertices from one of the columns C_{i-1}, C_i, or C_{i+1}.

The rest of the proof is based on the following observation: If we choose an element i uniformly at random from $\{1, \ldots, \sqrt{n}\}$, then the probability that a routing algorithm $\mathcal{A}$ has visited a vertex of C_{i-1}, C_i, or C_{i+1} after k steps is at most $3k/\sqrt{n}$. Letting $k = \sqrt{n}/6$, we see that the probability that $\mathcal{A}$ visits a vertex of C_{i-1}, C_i, or C_{i+1} after $\sqrt{n}/6$ steps is at most $1/2$.

Letting d_i denote the (expected, in the case of randomized algorithms) number of steps when routing from s to t in $S_{n,i}$ using routing algorithm $\mathcal{A}$, we have

$$\frac{1}{\sqrt{n}} \cdot \sum_{i=1}^{\sqrt{n}} d_i \geq \sqrt{n}/12 \ . \tag{2}$$

Since, for any $S_{n,i}$, the shortest path from s to t is 3 there must be some i for which the competitive ratio of $\mathcal{A}$ for $S_{n,i}$ is at least $\sqrt{n}/36 \in \Omega(\sqrt{n})$. □

Theorem 6 *Under the link distance metric, no routing algorithm is $o(\sqrt{n})$-competitive for all greedy triangulations.*

Proof. This follows immediately from the observation that for any $S_{n,i}$, a Delaunay triangulation of $S_{n,i}$ is also a greedy triangulation of $S_{n,i}$. □

Theorem 7 *Under the link distance metric, no routing algorithm is $o(\sqrt{n})$-competitive for all minimum-weight triangulations.*

Proof. We claim that for members of S, any greedy triangulation is also a minimum-weight triangulation. To prove this, we use a result on minimum-weight triangulations due to Aichholzer *et al.*.[8] Let $K_{n,i}$ be the complete graph on $S_{n,i}$. Then an edge e of $K_{n,i}$ is said to be a *light edge* if every edge of $K_{n,i}$ that crosses e is not shorter than e. Aichholzer *et al.* prove that if the set of light edges contains the edges of a triangulation then that triangulation is a minimum-weight triangulation.

There are only 5 different types of edges in the greedy triangulation of $S_{n,i}$; (1) vertical edges within a column, (2) horizonal edges between adjacent columns, (3) diagonal edges between adjacent columns, (4) edges used to triangulate column i, and (5) edges used to join s and t to the rest of the graph. It is straightforward to verify that all of these types of edges are indeed light edges. □

5. Conclusions

We have presented a number of results concerning online routing in plane graphs. Table 1 summarizes what is currently known about online routing in plane graphs. An arrow in a reference indicates that the result is implied by the more general result pointed to by the arrow. An F indicates that the result is trivial and/or

Class of graphs	Deterministic oblivious		Randomized oblivious[d]		Euclidean competitive		Link competitive	
DT	Yes	[2,4]	Yes	[←]	Yes	[2]	No	[here]
GT/MWT	Yes	[↓]	Yes	[↓]	Yes	[9]	No	[here]
Triangulations	Yes	[here]	Yes	[2]	No	[here]	No	[↑]
Conv. Subdv.	No	[here]	Yes	[here]	No	[↑]	No	[↑]
Plane graphs	No	[F]	No	[F]	No	[F]	No	[F]

Table 1: A summary of known results for online routing in plane graphs.

folklore.

We have also implemented a simulation of the GREEDY-COMPASS algorithm as well as the algorithms described by Bose and Morin[2] and compared them under the Euclidean distance metric. These results will be presented in the full version of the paper. Here we only summarize our main observations.

For Delaunay triangulations of random point sets, we found that the performance of GREEDY-COMPASS is comparable to that of the COMPASS and GREEDY algorithms.[2,5,4] For triangulations obtained by performing Graham's scan[10] on random point sets, the GREEDY-COMPASS algorithm does significantly better than the COMPASS or GREEDY algorithms.

We also implemented a variant of GREEDY-COMPASS that we call GREEDY-COMPASS-2 that, when located at a vertex v, moves to the vertex $u \in \{\mathrm{cw}(v), \mathrm{ccw}(v)\}$ that minimizes $d(v,u) + d(u,t)$, where $d(a,b)$ denotes the Euclidean distance between a and b. Although there are triangulations that defeat GREEDY-COMPASS-2, it worked for all our test triangulations, and in fact seems to be twice as efficient as GREEDY-COMPASS in terms of the Euclidean distance travelled.

We note that currently, under the link distance metric, there are no competitive routing algorithms for any interesting class of geometric graphs (meshes do not count). The reason for this seems to be that the properties used in defining many geometric graphs make use of properties of Euclidean space, and link distance in these graphs is often independent of these properties. We consider it an open problem to find competitive algorithms, under the link distance metric, for an interesting and naturally occuring class of geometric graphs.

6. Acknowledgements

This work was initiated at Schloss Dagstuhl Seminar on Data Structures, held in Wadern, Germany, February–March 2000, and co-organized by Susanne Albers, Ian Munro, and Peter Widmayer. The authors would also like to thank Jorge Urrutia and Lars Jacobsen for helpful discussions.

[d] In this column, we consider only algorithms that use a constant number of random bits per step. Otherwise, it is well known that a random walk on any graph G will eventually visit all vertices of G.

1. A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.
2. P. Bose and P. Morin. Online routing in triangulations. In *Proceedings of the Tenth International Symposium on Algorithms and Computation (ISAAC'99)*, volume 1741 of *Springer LNCS*, pages 113–122, 1999.
3. B. Kalyanasundaram and K. R. Pruhs. Constructing competitive tours from local information. *Theoretical Computer Science*, 130:125–138, 1994.
4. E. Kranakis, H. Singh, and J. Urrutia. Compass routing on geometric networks. In *Proceedings of the 11th Canadian Conference on Computational Geometry (CCCG'99)*, 1999.
5. P. Cucka, N. S. Netanyahu, and A. Rosenfeld. Learning in navigation: Goal finding in graphs. *International Journal of Pattern Recognition and Artificial Intelligence*, 10(5):429–446, 1996.
6. C. H. Papadimitriou and M. Yannakakis. Shortest paths without a map. *Theoretical Computer Science*, 84:127–150, 1991.
7. F. P. Preparata and M. I. Shamos. *Computational Geometry.* Springer-Verlag, New York, 1985.
8. O. Aichholzer, F. Aurenhammer, S.-W. Cheng, N. Katoh, G. Rote, M. Taschwer, and Y.-F. Xu. Triangulations intersect nicely. *Discrete and Computational Geometry*, 16(4):339–359, 1996.
9. P. Bose and P. Morin. Competitive routing algorithms for greedy and minimum-weight triangulations. Manuscript, 2000.
10. R. L. Graham. An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, 1:132–133, 1972.