Self-approaching Graphs

Soroush Alamdari[1], Timothy M. Chan[1], Elyot Grant[2],
Anna Lubiw[1], and Vinayak Pathak[1]

[1] Cheriton School of Computer Science, University of Waterloo, Waterloo, Canada
{s26hosse,tmchan,alubiw,vpathak}@uwaterloo.ca
[2] Massachusetts Institute of Technology, Cambridge, USA
elyot@mit.edu

Abstract. In this paper we introduce *self-approaching* graph drawings. A straight-line drawing of a graph is *self-approaching* if, for any origin vertex s and any destination vertex t, there is an st-path in the graph such that, for any point q on the path, as a point p moves continuously along the path from the origin to q, the Euclidean distance from p to q is always decreasing. This is a more stringent condition than a greedy drawing (where only the distance between vertices on the path and the destination vertex must decrease), and guarantees that the drawing is a 5.33-spanner.

We study three topics: (1) recognizing self-approaching drawings; (2) constructing self-approaching drawings of a given graph; (3) constructing a self-approaching Steiner network connecting a given set of points.

We show that: (1) there are efficient algorithms to test if a polygonal path is self-approaching in $\mathbb{R}^2$ and $\mathbb{R}^3$, but it is NP-hard to test if a given graph drawing in $\mathbb{R}^3$ has a self-approaching uv-path; (2) we can characterize the trees that have self-approaching drawings; (3) for any given set of terminal points in the plane, we can find a linear sized network that has a self-approaching path between any ordered pair of terminals.

Keywords: self-approaching, increasing-chord, graph drawing.

1 Introduction

A straight-line graph drawing (or "geometric graph") in the plane has points for vertices, and straight line segments for edges, where the weight of an edge is its Euclidean length. The drawing need not be planar. Rao *et al.* [27] introduced the idea of greedy drawings. A *greedy drawing* of a graph is a straight-line drawing in which, for each origin vertex s and destination vertex t, there is a neighbor of s that is closer to t than s is, i.e., there is a *greedy* st-path $P = (s = p_1, p_2, \ldots, p_k = t)$ such that the Euclidean distances $D(p_i, t)$ decrease as i increases. This idea has attracted great interest in recent years (e.g. [3,7,17,21,23,26]) mainly because a greedy drawing of a graph permits greedy local routing.

It is a very natural and desirable property that a path should always get closer to its destination, but there is more than one way to define this. Although every vertex along a greedy path gets closer to the destination, the same is not true of intermediate points along edges. See Figure 1.

W. Didimo and M. Patrignani (Eds.): GD 2012, LNCS 7704, pp. 260–271, 2013.

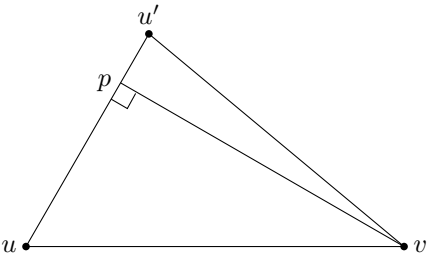

Fig. 1. As we move from u towards u', distance to v first decreases (until p), then increases. However, $D(u', v) < D(u, v)$.

Another disadvantage of greedy paths is that the length of a greedy path is not bounded in terms of the Euclidean distance between the endpoints. This is another natural and desirable property for a path to have, and is captured by the *dilation* (or "stretch factor") of a graph drawing—the maximum, over vertices s and t, of the ratio of their distance in the graph to their Euclidean distance. The dilation factor of greedy graph drawings can be unbounded.

Icking *et al.* [22] introduced a stronger notion of "getting closer" to a destination, that addresses both shortcomings of greedy paths. A curve from s to t is *self-approaching* if for any three points a, b, c appearing in that order along the curve, we have $D(a, c) \geq D(b, c)$. Icking *et al.* proved that a self-approaching curve has *detour* at most 5.3332, where the *detour* or *geometric dilation* of a curve is the supremum over points p and q on the curve, of the ratio of their distance along the curve to their Euclidean distance $D(p, q)$. This is stronger than dilation in that we consider all pairs of points, not just all pairs of vertices.

In this paper we introduce the notion of a *self-approaching* graph drawing—a straight-line drawing that contains, for every pair of vertices s and t, a self-approaching st-path and a self-approaching ts-path (which need not be the same). We also explore the related notion of an *increasing-chord* graph drawing, which has the stronger property that every pair of vertices is joined by a path that is self-approaching in both directions. Rote [28] proved that increasing-chord paths have geometric dilation at most 2.094.

Our first result is a linear time algorithm to recognize a self-approaching polygonal path in the plane. This extends to $\mathbb{R}^3$, with some slow-down. We do not know the complexity of recognizing self-approaching graph drawings in the plane or higher dimensions. One approach would be to find, for every pair of vertices u and v, a self-approaching path from u to v in the graph drawing. This problem is open in $\mathbb{R}^2$ but we show that it is NP-hard in $\mathbb{R}^3$.

Next, we consider the question of constructing a self-approaching drawing for a given graph. We give a linear time algorithm to recognize the trees that have self-approaching drawings. Finally, we consider the problem of connecting a given set of terminal points in the plane by a network that has a self-approaching path between every pair of terminals. We show that this can be done with a linear sized network.

Due to space constraints, many details are omitted. They can be found in the long version of the paper.

2 Background

A *spanner* is a graph of bounded dilation. Spanners have been very well-studied—see for example the book by Narasimhan and Smid [25] and the survey by Eppstein [15]. A main goal is to efficiently construct a spanner on a given set of points, with the objective of minimizing dilation while keeping the number or total length of edges small. For recent results, see, e.g., [4,16]. If Steiner vertices are allowed, their number should also be minimized, and different versions of the problem arise if we include the Steiner points in measuring the dilation, see [14].

The *detour* of a graph drawing is defined to be the supremum, over all points p, q of the drawing (whether at vertices, or interior to edges) of the ratio of their distance in the graph to their Euclidean distance. Note that if two edges cross in the drawing, then the detour is infinite. By contrast, a self-approaching drawing may have crossing edges, for example, any complete geometric graph is self-approaching. Constructing a network to minimize detour has also been considered [13,12], though not as extensively as spanners.

Relevant background on greedy drawings is as follows. Answering a conjecture of Papadimitriou and Ratajczak [26], Leighton and Moitra [23] showed that any 3-connected planar graph has a greedy drawing, and Goodrich and Strash [17] reduced the number of bits needed for the coordinates in the embedding. Moitra [24] used combinatorial conditions to classify the trees that have greedy embeddings. Connecting the ideas of greedy drawings and spanners, Bose *et al.* [7] showed that every triangulation has an embedding in which local routing produces a path of bounded dilation.

Self-approaching drawings are related to *monotone drawings* in which, for every pair of vertices s and t, there is an st-path that is monotone in some direction. This concept was introduced by Angelini, et al., [1] who gave algorithms to construct monotone planar drawings of trees and planar biconnected graphs. A follow-up paper [2] considers the case where a planar embedding is specified. Self-approaching drawings are not necessarily monotone, and monotone drawings are not necessarily self-approaching. The one relationship is that any increasing-chord drawing is a monotone drawing.

Although a monotone path need not be self-approaching, there is a stronger condition that does imply self-approaching, namely that the path is monotone in both the x- and y-directions. Thus, a network with an xy-monotone path between every pair of terminals is a self-approaching network. A *Manhattan network* has horizontal and vertical edges and includes an L_1 shortest path between every pair of terminals. So a Manhattan network is self-approaching. There is considerable work on finding Manhattan networks of minimum total length (so-called "minimum Manhattan networks"). There are efficient algorithms with approximation factor 2, and the problem has been shown to be NP-hard [11]. More relevant to us is the result of Gudmundsson et al. [19] that every point set admits a Manhattan network of $O(n \log n)$ vertices and edges, and there are point sets for which any Manhattan network has size at least $\Omega(n \log n)$. This contrasts with our result that every point set admits a self-approaching network of linear size.

For results on computing the dilation or detour of a path or graph, see the survey by Gudmundsson and Knauer [18] and the paper by Wulff-Nilsen [29].

The Delaunay triangulation has several good properties: it has dilation factor below 2 [30], and is a greedy drawing [8], although greedy paths in a Delaunay triangulation do not necessarily have bounded dilation. We show that the Delaunay triangulation is not necessarily self-approaching.

3 Preliminaries

We let $D(u, v)$ denote the Euclidean distance between points u and v in $\mathbb{R}^d$. Formally, a *curve* is a continuous function $f\colon [0,1] \to \mathbb{R}^d$, and an *st-curve* is a curve f with $f(0) = s$ and $f(1) = t$. The *reverse curve* is $f(1-t), t \in [0,1]$. For convenience, we will identify a curve with its image, and ignore the particular parameterization. When we speak of points a and b *in order along the curve*, or with b *later than* a on the curve, we mean that $a = f(t_1)$ and $b = f(t_2)$ for some $0 \le t_1 \le t_2 \le 1$. A curve is *self-approaching* if for any three points a, b, c in order along the curve, we have $D(a, c) \ge D(b, c)$ (see Figure 2(a)). Note that this definition is sensitive to the direction of the curve—it may happen that a curve is self-approaching but its reverse is not.

A curve has *increasing chords* if for any four points a, b, c, d in order along the curve we have $D(a, d) \ge D(b, c)$ (see Figure 2(b) for an example). Note that if a curve has increasing chords then the reverse curve also has increasing chords, and the curve and its reverse are both self-approaching. The converse also holds: if a curve and its reverse are both self-approaching then the curve has increasing chords, as we then have $D(a, d) \ge D(a, c) \ge D(b, c)$ for any points a, b, c, d in order along the curve.

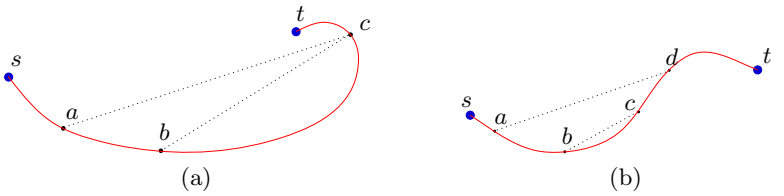

Fig. 2. (a) A self-approaching *st*-curve and (b) an increasing-chord curve in $\mathbb{R}^2$

The following characterization of self-approaching curves is straightforward:

Lemma 1. *([22]) A piecewise-smooth curve is self-approaching iff for each point a on the curve, the line perpendicular to the curve at a does not intersect the curve at a later point.*

Corollary 1. *A piecewise-smooth curve has increasing chords iff each line perpendicular to the curve intersects the curve at no other point.*

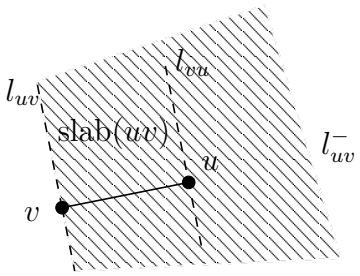

Fig. 3. For distinct points u and v, let $\overline{uv}$ be the line passing through u and v. Let l_{uv} denote the line that passes through v and is perpendicular to $\overline{uv}$, noting that l_{uv} and l_{vu} are distinct parallel lines. Let l^-_{uv} denote the open half-plane containing u with boundary l_{uv}, and let l^+_{uv} denote $\mathbb{R}^2 \setminus l^-_{uv}$, the closed half-plane that is the complement of l^-_{uv}. Let slab(uv) be the open strip bounded by l_{uv} and l_{vu}—in other words, the intersection of l^-_{uv} and l^-_{vu}.

When dealing with straight-line drawings of graphs, we apply Lemma 1 to piecewise-linear curves. Using the notation defined in Figure 3, we can restate the lemma as follows:

Corollary 2. *Let $P = (v_1, v_2, \ldots, v_n)$ be a directed path embedded in $\mathbb{R}^2$ via straight line segments. Then, P is self-approaching iff for all $1 < i < j \leq n$, the point v_j lies in $l^+_{v_{i-1}v_i}$. Equivalently, P is self-approaching iff for all $1 < i \leq n$, the convex hull of $\{v_i, v_{i+1}, \ldots, v_n\}$ lies in $l^+_{v_{i-1}v_i}$.*

Analogous characterizations are also possible in higher dimensions, with the half-planes $l^+_{v_{i-1}v_i}$ replaced by half-spaces bounded by hyperplanes orthogonal to $\overline{v_{i-1}v_i}$.

4 Testing Whether Paths Are Self-approaching

Corollary 2 implicitly suggests an algorithm to determine whether a directed path embedded in a Euclidean space is self-approaching. In this section, we provide improved algorithms for this task in two and three dimensions, as well as a lower bound. We assume a real RAM model in which all simple geometric operations can be performed in $O(1)$ time, and we assume that a straight-line drawing of a path $P = (v_1, v_2, \ldots, v_n)$ is represented explicitly as a list of n points (requiring $O(n)$ space).

Theorem 1. *Given a straight-line drawing of a path $P = (v_1, v_2, \ldots, v_n)$ in the plane, it is possible to test whether P is self-approaching in linear time.*

We prove this theorem by giving an $O(n)$ time algorithm to check that for all $1 < i \leq n$, the convex hull of v_i through v_n lies in $l^+_{v_{i-1}v_i}$. We can do this by iteratively processing and storing the convex hull.

In three dimensions, we can obtain a similar result with slightly worse running time using an existing convex hull data structure that supports point insertion and half-space range emptiness queries.

Theorem 2. *Given a straight-line drawing of a path $P = (v_1, v_2, \ldots, v_n)$ in $\mathbb{R}^3$, it is possible to test whether P is self-approaching in $O(n \log^2 n / \log \log n)$ time.*

Next, we show that Theorem 2 is tight up to a factor of $\log n / \log \log n$ by proving a lower bound of $\Omega(n \log n)$ on the running time of any algorithm for determining whether a directed path embedded in $\mathbb{R}^3$ is self-approaching. We do this by reducing from the *set intersection problem*, for which a solution requires $\Omega(n \log n)$ time on an input of size n in the algebraic computation tree model [5]. We can show the following:

Theorem 3. *Given a straight-line drawing of a path $P = (v_1, v_2, \ldots, v_n)$ in $\mathbb{R}^3$, at least $\Omega(n \log n)$ time is required in the algebraic computation tree model to test whether P is self-approaching.*

To prove this, we build an embedding of a path in $\mathbb{R}^3$ using 'cannons' and 'targets', where a slab perpendicular to a 'cannon' collides with a 'target' if and only if the corresponding elements of the sets A and B are identical.

The same construction also yields the following:

Corollary 3. *Given a straight-line drawing of a path $P = (v_1, v_2, \ldots, v_n)$ in $\mathbb{R}^3$, at least $\Omega(n \log n)$ time is required in the algebraic computation tree model to test whether P has increasing chords.*

5 Finding Self-approaching Paths in Graphs

We do not know how to test in polynomial time if a given graph drawing is self-approaching. This contrasts with the situation for greedy drawings where it suffices to find, for every pair of vertices s and t, a "first edge" (s, a) with $D(a, t) < D(s, t)$. In this section we explore the problem of finding a self-approaching path between two vertices s and t in a graph drawing. If we could do this in polynomial time, then we could test if a drawing is self-approaching in polynomial time. We are unable to settle the complexity in two dimensions, but, by employing the cannons and targets introduced in Section 4, we can show that the problem is hard in three or more dimensions:

Theorem 4. *Given a straight-line drawing of a graph G in $\mathbb{R}^3$, and a pair of vertices s and t from G, it is NP-hard to determine if a self-approaching st-path exists. It is also NP-hard to determine if an increasing-chord st-path exists.*

To prove this theorem, we reduce from 3SAT. Our proof uses similar 'cannons' and 'targets' to those used in the proof of Theorem 3, but this time, the cannons correspond to variable assignments and the targets correspond to literals in clauses.

6 Recognizing Graphs Having Self-approaching Drawings

In this section we characterize trees that have self-approaching drawings and give a linear time recognition algorithm. This is similar to Moitra's characterization of trees that admit greedy drawings [24].

Lemma 2. *In a self-approaching drawing of a tree T, for each edge (u, v), there is no edge or vertex of $T \setminus uv$ that intersects* slab(uv).

Proof. Since there is a unique path connecting vertices s and t in any tree T, a drawing of T is self-approaching if and only if it has increasing chords. The result then follows from Corollary 1. □

As it turns out, we can quickly determine whether a tree admits a self-approaching drawing:

Theorem 5. *Given a tree T, we can decide in linear time whether or not T admits a self-approaching drawing.*

Proof. To prove this theorem, we completely characterize trees that admit self-approaching drawings. We require two definitions of special graphs.

A *windmill* having *sweep length* k is a tree constructed by subdividing the edges of $K_{1,3}$ $k-1$ times iteratively and then attaching a leaf to each subdivision vertex. The term *sweep* shall denote one of the three new disjoint paths of length k that replace edges of $K_{1,3}$ during the subdivision process. A windmill is depicted in Figure 4(a).

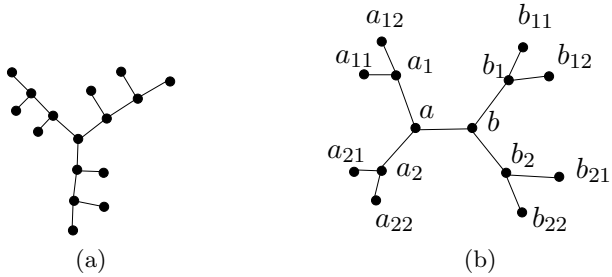

Fig. 4. (a) A windmill with sweeps of length 2 and (b) an embedding of the crab

The *crab graph* is the tree with vertices $\{a, b, a_1, a_2, b_1, b_2, a_{11}, a_{12}, a_{21}, a_{22}, b_{11}, b_{12}, b_{21}, b_{22}\}$ and edges $\{(a, b), (a, a_1), (a, a_2), (b, b_1), (b, b_2), (a_1, a_{11}), (a_1, a_{12}), (a_2, a_{21}), (a_2, a_{22}), (b_1, b_{11}), (b_1, b_{12}), (b_2, b_{21}), (b_2, b_{22})\}$ as depicted in Figure 4(b). A graph G is *crab-free* if it has no subgraph that is isomorphic to some subdivision of a crab.

We prove Theorem 5 in two steps. Write Δ_T for the maximum degree of a vertex in T.

1. First we show that a tree T with $\Delta_T \geq 4$ admits a self-approaching drawing if and only if T is a subdivision of $K_{1,4}$.
2. Then we show that a tree T with $\Delta_T \leq 3$ admits a self-approaching drawing if and only if it is a subdivision of a windmill, which happens if and only if T is crab-free.

To establish the first result, the following can be proved:

Lemma 3. *In an increasing-chord drawing of a path, the sum of the sizes of the angles in any consecutive chain of k left turns (or right turns) is at least $\pi(k-1)$ if $k > 1$ and at least $\pi/2$ if $k = 1$.*

Corollary 4. *If T admits a self-approaching drawing, then $\Delta_T \leq 4$. Also, if $\Delta_T = 4$, then there is only one vertex of degree 4 in T, and the four angles at the vertex of degree 4 have all size $\pi/2$, and the rest of the angles have size π.*

This concludes the first step of the proof. For the second step, we prove the following three structural lemmas, which establish the equivalence of a tree being a subdivision of a windmill, being crab-free, and admitting a self-approaching drawing.

Lemma 4. *Let T be a tree with $\Delta_T \leq 3$ that is crab-free. Then T is a subdivision of a windmill.*

Lemma 5. *Let T be a tree that is a subdivision of a windmill. Then T admits a self-approaching drawing.*

Lemma 6. *Let T be a tree that contains a subdivision of the crab. Then T does not admit a self-approaching drawing.*

Combining these results, we obtain the second step of the proof of the theorem. This completes the characterization of all trees that admit self-approaching drawings. To complete the proof of Theorem 5, it suffices to observe that it is possible, in linear time, to check whether a tree T is a subdivision of $K_{1,4}$ or of a windmill. □

7 Constructing Self-approaching Steiner Networks

We now turn our attention to the following problem: Given a set P of points in the plane, draw a graph N with straight edges and $P \subseteq V(N)$ such that for each ordered pair of points $p, q \in P$ there is a self-approaching path from p to q in the drawing of N. We call the points in $V(N) \backslash P$ *Steiner points* and the graph N a *self-approaching Steiner network*. An increasing-chord Steiner network is defined similarly.

We show that small increasing-chord Steiner networks (and hence small self-approaching Steiner networks) can always be constructed for any given set of points in the plane.

Theorem 6. *Given a set P of n points in the plane, there exists an increasing-chord Steiner network having $O(n)$ vertices and edges.*

Proof. Given points p and q, let θ_{pq} denote the angle between the line pq and the x-axis (we take the smaller of the two angles formed, so that $\theta_{pq} \in [0, \pi/2]$). A path is *$xy$-monotone* if every vertical line intersects the path at most once and every horizontal line intersects the path at most once. Clearly, an xy-monotone

path is self-approaching. We will construct a linear-size Steiner network G with the following property:

> For every pair of points $p, q \in P$ with $\theta_{pq} \in [\pi/8, 3\pi/8]$, there is an xy-monotone path from p to q in G.

To handle the remaining pairs of points, we can rotate the coordinate axes by $\pi/4$ and apply the same construction to obtain another Steiner network G'. We can then return the union of G and G'.

To construct G, we first build a *quadtree* [20], defined as follows: The root stores an initial square enclosing P. At each node, we divide its square into four congruent subsquares and create a child for each subsquare that is not empty of points of P. The tree has n leaves.

To ensure that the tree has $O(n)$ internal nodes, we compress each maximal path of degree-1 nodes by keeping only the first and last node in the path. The result is a *compressed quadtree*, denoted T.

For each square B in the compressed quadtree T, we add the four corner vertices and edges of B to G. (Note that we allow overlapping edges in our construction; it is not difficult to avoid overlaps by subdividing the edges appropriately.) For each leaf square B in T containing a single point $p \in P$, we add a 2-link xy-monotone path in G from p to each corner of B. For each degree-1 square B in T having a single child square B', we add a 2-link xy-monotone path in G from each corner of B' to the corresponding corner of B. By induction, it then follows that for every point $p \in P$ inside a square B in T, there is an xy-monotone path in G from p to each corner of B. The number of vertices and edges in G thus far is $O(n)$.

Given a parameter $\varepsilon > 0$, a *well-separated pair decomposition* of P is a collection of pairs of sets $\{A_1, B_1\}, \ldots, \{A_s, B_s\}$, such that[1]

1. for every pair of points $p, q \in P$, there is a unique index i with $(p, q) \in A_i \times B_i$ or $(p, q) \in B_i \times A_i$;
2. A_i and B_i are *well-separated* in the sense that both the diameter of A_i and the diameter of B_i is at most $\varepsilon d(A_i, B_i)$, where $d(A_i, B_i)$ is the minimum distance between A_i and B_i.

It is known that a well-separated pair decomposition consisting of $s = O(n/\varepsilon^2)$ pairs always exists [9]. Furthermore, such a decomposition can be constructed by a simple quadtree-based algorithm (for example, see [20] or [10]), where the sets A_i and B_i are in fact squares appearing in the compressed quadtree T.

To finish the construction of G, we consider each pair $\{A_i, B_i\}$ in the decomposition such that A_i and B_i are separated by both a vertical line and a horizontal line. Without loss of generality, suppose that A_i is to the left of and below B_i. We add a 2-link xy-monotone path in G from the upper right corner of A_i to the lower left corner of B_i. The overall number of vertices and edges in G is $O(n/\varepsilon^2)$.

[1] In the original definition [9], A_i and B_i are subsets of P, but for our purposes, we will take A_i and B_i to be regions in the plane (namely, squares).

To show that G satisfies the stated property, let $p, q \in P$ with $\theta_{pq} \in [\pi/8, 3\pi/8]$. Suppose that $(p, q) \in A_i \times B_i$. If A_i and B_i are intersected by a common horizontal line, then θ_{pq} must be upper-bounded by $O(\varepsilon)$ because A_i and B_i are well-separated; this is a contradiction if we make the constant ε sufficiently small. Thus, A_i and B_i must be separated by a horizontal line, and similarly by a vertical line via a symmetric argument. Without loss of generality, suppose that A_i is to the left of and below B_i. By concatenating xy-monotone paths in G, we can get from p to the upper right corner of A_i, then to the lower left corner of B_i, and finally to q. □

In the above construction, the edges we add for each well-separated pair $\{A_i, B_i\}$ may cross other edges, although it is possible to modify the construction to ensure that the network G is planar (and similarly G'). However, we do not know how to avoid crossings in the final network obtained by unioning G and G', while keeping the number of edges linear. Our construction can be carried out in $O(n \log n)$ time, since that is the cost for building the compressed quad tree and the well-separated pair decomposition. The theorem generalizes to any constant dimension.

We note that our construction bears some similarity to the construction used independently by Borradaile and Eppstein [6] to create small low-weight plane Steiner spanners in which the paths stay within a bounded range of angles.

Whether planar self-approaching Steiner networks of linear size can be constructed or not is an interesting question. Delaunay triangulations seemed to be a potential candidate, however, Figure 5 shows a configuration of 6 points in the plane whose Delaunay triangulation is not a self-approaching drawing.

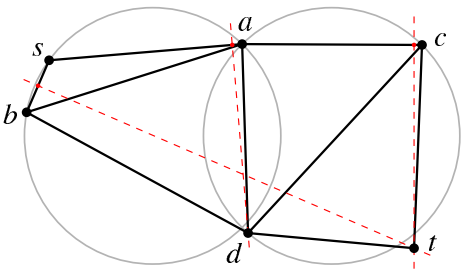

Fig. 5. The Delaunay triangulation of these six points does not have a self-approaching path from s to t. Forbidden edge-vertex pairs are indicated with dashed lines. From s we must take edge sa, because t lies in the forbidden region for edge sb. Then we cannot go to d since it is in the forbidden region of sa, nor can we use edge ac since t is in its forbidden region.

8 Conclusions

We have introduced the notion of self-approaching and increasing-chord graph drawings, with rich connections to greedy drawings, spanners, dilation and detour, and minimum Manhattan networks.

Our results are preliminary. We leave open the following questions:

– Can we test, in polynomial time, if a straight-line graph drawing in the plane is self-approaching [or increasing-chord]? Or is the problem NP-complete?

- Given a graph G, can we efficiently produce a self-approaching drawing of G if one exists?
- What classes of graphs have self-approaching [or increasing-chord] drawings? Does, for example, every 3-connected planar graph have a self-approaching drawing? Even more interesting, which graphs have a self-approaching drawing such that local routing finds a self-approaching path? For example, if 3-connected graphs had such drawings, this would have the significant implication that every 3-connected planar graph has an embedding where local routing gives paths of bounded detour (hence bounded dilation). Bose *et al.* [7] recently proved the weaker result that every triangulation has an embedding where local routing gives paths of bounded dilation.

Acknowledgements. Anna Lubiw thanks Marcus Brazil, Victor Chepoi, and Martin Zachariasen for workshop discussions that inspired this line of enquiry. This work was done as part of an Algorithms Problem Session at the University of Waterloo, and we thank the other participants for helpful discussions. We thank Prosenjit Bose and Pat Morin for help finding the example in Figure 5.

References

1. Angelini, P., Colasante, E., Battista, G.D., Frati, F., Patrignani, M.: Monotone drawings of graphs. J. Graph Algorithms Appl. 16(1), 5–35 (2012)
2. Angelini, P., Didimo, W., Kobourov, S., Mchedlidze, T., Roselli, V., Symvonis, A., Wismath, S.: Monotone Drawings of Graphs with Fixed Embedding. In: van Kreveld, M., Speckmann, B. (eds.) GD 2011. LNCS, vol. 7034, pp. 379–390. Springer, Heidelberg (2012)
3. Angelini, P., Frati, F., Grilli, L.: An algorithm to construct greedy drawings of triangulations. J. Graph Algorithms Appl. 14(1), 19–51 (2010)
4. Aronov, B., de Berg, M., Cheong, O., Gudmundsson, J., Haverkort, H., Smid, M., Vigneron, A.: Sparse geometric graphs with small dilation. Computational Geometry 40(3), 207–219 (2008)
5. Ben-Or, M.: Lower bounds for algebraic computation trees. In: Proc. 15th ACM Symposium on Theory of Computing, New York, pp. 80–86 (1983)
6. Borradaile, G., Eppstein, D.: Near-linear-time deterministic plane Steiner spanners and TSP approximation for well-spaced point sets. In: Proceedings of the 24th Annual Canadian Conference on Computational Geometry, CCCG, Charlottetown, PEI, Canada (2012)
7. Bose, P., Fagerberg, R., van Renssen, A., Verdonschot, S.: Competitive routing in the half-θ_6-graph. In: Proc. 23rd ACM-SIAM Symposium on Discrete Algorithms, pp. 1319–1328 (2012)
8. Bose, P., Morin, P.: Online routing in triangulations. SIAM J. Comput. 33(4), 937–951 (2004)
9. Callahan, P.B., Kosaraju, S.R.: A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. J. ACM 42, 67–90 (1995)
10. Chan, T.M.: Well-separated pair decomposition in linear time? Inform. Process. Lett. 107, 138–141 (2008)

11. Chin, F.Y.L., Guo, Z., Sun, H.: Minimum Manhattan network is NP-complete. Discrete & Computational Geometry 45(4), 701–722 (2011)
12. Dumitrescu, A., Tóth, C.D.: Light orthogonal networks with constant geometric dilation. Journal of Discrete Algorithms 7(1), 112–129 (2009)
13. Ebbers-Baumann, A., Grune, A., Klein, R.: The geometric dilation of finite point sets. Algorithmica 44, 137–149 (2006)
14. Ebbers-Baumann, A., Grüne, A., Klein, R., Karpinski, M., Knauer, C., Lingas, A.: Embedding point sets into plane graphs of small dilation. Int. J. Comput. Geometry Appl. 17(3), 201–230 (2007)
15. Eppstein, D.: Spanning trees and spanners. In: Sack, J., Urrutia, J. (eds.) Handbook of Computational Geometry, pp. 425–461. North-Holland (2000)
16. Giannopoulos, P., Klein, R., Knauer, C., Kutz, M., Marx, D.: Computing geometric minimum-dilation graphs is NP-hard. Int. J. Comput. Geometry Appl. 20(2), 147–173 (2010)
17. Goodrich, M.T., Strash, D.: Succinct Greedy Geometric Routing in the Euclidean Plane. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 781–791. Springer, Heidelberg (2009)
18. Gudmundsson, J., Knauer, C.: Dilation and detour in geometric networks. In: Gonzalez, T. (ed.) Handbook on Approximation Algorithms and Metaheuristics. Chapman & Hall/CRC Press (2007)
19. Gudmundsson, J., Klein, O., Knauer, C., Smid, M.: Small Manhattan networks and algorithmic applications for the earth mover's distance. In: Proc. 23rd European Workshop on Computational Geometry, pp. 174–177 (2007)
20. Har-Peled, S.: Geometric Approximation Algorithms. AMS (2011)
21. He, X., Zhang, H.: On succinct convex greedy drawing of 3-connected plane graphs. In: Proc. 22nd ACM-SIAM Symposium on Discrete Algorithms, pp. 1477–1486 (2011)
22. Icking, C., Klein, R., Langetepe, E.: Self-approaching curves. Math. Proc. Camb. Phil. Soc. 125, 441–453 (1995)
23. Leighton, T., Moitra, A.: Some results on greedy embeddings in metric spaces. Discrete and Computational Geometry 44, 686–705 (2010)
24. Moitra, A.: A solution to the Papadimitriou-Ratajczak conjecture. Master's thesis, Massachusetts Institute of Technology (2009)
25. Narasimhan, G., Smid, M.: Geometric Spanner Networks. Cambridge University Press (2007)
26. Papadimitriou, C.H., Ratajczak, D.: On a conjecture related to geometric routing. Theor. Comput. Sci. 344, 3–14 (2005)
27. Rao, A., Ratnasamy, S., Papadimitriou, C., Shenker, S., Stoica, I.: Geographic routing without location information. In: Proc. 9th International Conference on Mobile Computing and Networking, pp. 96–108 (2003)
28. Rote, G.: Curves with increasing chords. Mathematical Proceedings of the Cambridge Philosophical Society 115, 1–12 (1994)
29. Wulff-Nilsen, C.: Computing the maximum detour of a plane geometric graph in subquadratic time. Journal of Computational Geometry 1(1), 101–122 (2010)
30. Xia, G.: Improved upper bound on the stretch factor of Delaunay triangulations. In: Proc. 27th ACM Symposium on Computational Geometry, pp. 264–273 (2011)