Approximating shortest paths in arrangements of lines

Prosenjit Bose William Evans David Kirkpatrick
Michael McAllister Jack Snoeyink
Department of Computer Science
University of British Columbia

1 Introduction

Arrangements, such as the decomposition of the plane into regions by a set of n lines, are central to much of the research on algorithms in computational geometry. They express a large number of geometric features that have a highly regular structure, such as the $\binom{n}{2}$ intersections between pairs of lines. Powerful techniques, such as ϵ-nets, random sampling, lower envelopes, and implicit representation, have been developed that use the regularity to look at less than the full structure of the arrangements in solving problems on arrangements [3, 4]. These techniques are most applicable to problems of a combinatorial nature; they have less to say about metric properties.

In this paper, we consider shortest paths on an arrangement: Given a set L of n lines in the plane, and two points s and t that lie on lines of L, find a shortest path from s to t that is restricted to remain on the lines of L. We use the Euclidean distance metric.

One approach to solve this problem is to compute the entire arrangement $\mathcal{A}$ and then compute a shortest path by Dijkstra's algorithm [2]. This ignores the regularity in the arrangement, however, and leads to an algorithm with $O(n^2 \log n)$ complexity. The more interesting question is how to compute a shortest path in subquadratic time; this has been posed, for example, by Marc van Kreveld at the Fourth Dagstuhl Seminar on Computational Geometry [1].

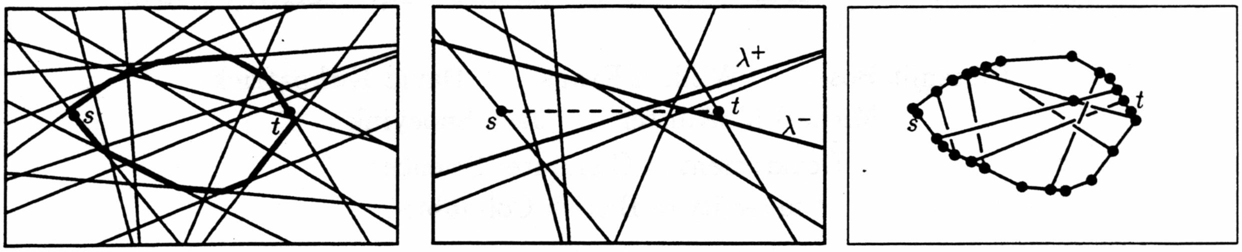
Computing the shortest path without computing the arrangement seems difficult. In particular, we have not succeeded in answering van Kreveld's question. Neither have we succeeded in proving an $\omega(n \log n)$ lower bound for a non-trivial model. However, we show how to compute, in $O(n \log n)$ time, a path P from s to t such that the length of P is at most twice that of the actual shortest path. Our algorithm is simple: run Dijkstra's in a subgraph of the arrangement. The proof of the approximation bound makes extensive use of the structure of shortest paths on arrangements and sub-arrangements formed by lines of restricted slope.

2 Algorithm

We begin with a few definitions that help us specify the subgraph of the arrangement and then present the algorithm itself.

Let $\mathcal{A}$ be an arrangement of n lines and let s and t be points on lines of $\mathcal{A}$. Throughout this paper, we use Roman letters to denote points and Greek letters to denote lines.

Definition 1 *The lines of $\mathcal{A}$ that intersect the closed segment $\overline{st}$ are called the* cross lines *of $\mathcal{A}$ relative to s and t (figure 1b).*



(a) (b) (c)

Figure 1: (a) An arrangement of lines with cell $Cell(st)$ in bold, (b) cross lines with critical lines λ^+ and λ^-, and (c) the graph G.

Definition 2 *The* extreme lines λ^- and λ^+ *of $\mathcal{A}$ for s and t are cross lines with the largest negative slope and smallest positive slope respectively. If there is no cross line of negative slope then λ^- is a cross line with maximum positive slope. Similarly, if there is no cross line of positive slope then λ^+ is a cross line with smallest negative slope.*

Definition 3 *The* cell of s and t in $\mathcal{A}$, $Cell(st)$, *is the boundary of the convex face that contains $\overline{st}$ in the following arrangement: the arrangement $\mathcal{A}$ without the cross lines and including all lines that contain s or t. A* cell edge *is an edge of the arrangement $\mathcal{A}$ that lies on the boundary of $Cell(st)$. A* cell vertex *is an endpoint of a cell edge; it is an* upper cell vertex *if it is above the line $\overline{st}$ and is a* lower cell vertex *if it is below the line $\overline{st}$; vertices s and t are both upper and lower cell vertices. A* cell line *is a line of $\mathcal{A}$ that contains a cell edge.*

We now describe our approximation algorithm. Let G be the graph whose vertices are the cell vertices of $Cell(st)$ and the intersection point of λ^+ and λ^-. Two vertices are joined by an edge if they lie on the same input line and no cell vertex lies between them. The length of the edge is the distance between its endpoints in the arrangement (figure 1c). The algorithm outputs the shortest st-path in G. This path well-approximates the shortest st-path in the arrangement (see Section 3); i.e. its length is at most twice that of the shortest st-path in the arrangement.

The algorithm runs in $O(n \log n)$ time and uses linear space. It determines the cross lines in $O(n)$ time. It determines $Cell(st)$ in $O(n \log n)$ time by intersecting the halfplanes, defined by input lines, that contain both s and t. It determines the cell vertices and the graph G by intersecting the cross lines with $Cell(st)$, also in $O(n \log n)$ time. Finally, since the graph G has $O(n)$ vertices and edges, the algorithm finds the shortest st-path in G, using Dijkstra's algorithm, in $O(n \log n)$ time.

3 Approximation Guarantee

We require two bounds to prove that the path output by the algorithm well-approximates the shortest st-path in the arrangement: a lower bound on the length of the shortest st-path in the arrangement and an upper bound on the length of the path that we generate. These bounds appear in Section 3.1. We then demonstrate how the bounds apply to the graph of the arrangement in Section 3.2.

3.1 Path length bounds

We begin with a some simple properties of shortest paths and some definitions. The lower bound and upper bound on path lengths appear in Lemmas 3 and 4.

Lemma 1 *Let P be a shortest path from s to t and let α be any line. If P intersects α at two points u and v then all points of P between u and v lie on α.*

> **Proof:** Let w be a point on P that lies between u and v. If w does not lie on α then $|\overline{uw}| + |\overline{wv}| < |\overline{uv}|$ by the triangle inequality and we can find a shorter path than P from s to t by following α, which contradicts the optimality of P. ■

Corollary 2 *The shortest path from s to t in an arrangement $\mathcal{A}$ is contained in $Cell(st)$.*

The analysis of the approximation factor for our short path focuses on the extreme lines λ^- and λ^+. Intuitively, these lines provide the fastest traversal across the face of $Cell(st)$.

Since $Cell(st)$ is convex, it has two supporting tangents parallel to λ^- and two supporting tangents parallel to λ^+(figure 2). These tangents divide $Cell(st)$ into four contiguous chains of cell vertices.

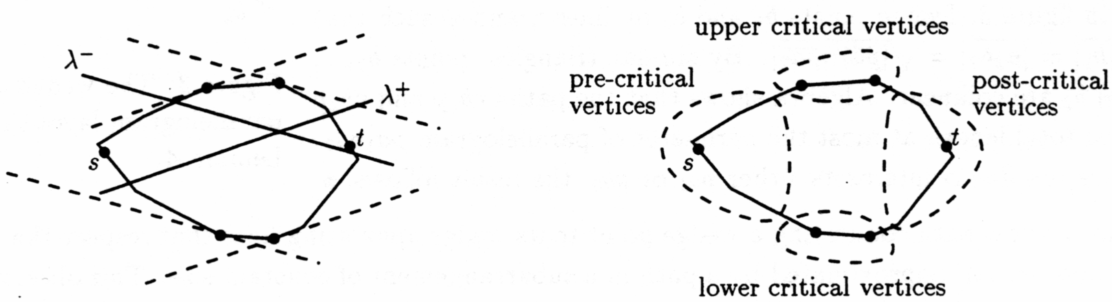


Figure 2: Division of $Cell(st)$ into pre-critical vertices, critical vertices, and post-critical vertices.

Definition 4 *The cell vertices of $Cell(st)$ in the two chains that do not contain s or t, including the vertices with the tangents parallel to λ^- and λ^+, are* critical vertices. *The cell vertices in the same chain as the point s are* pre-critical vertices. *The cell vertices in the same chain as t are* post-critical vertices.

Definition 5 *An edge of $Cell(st)$ is a* critical edge *if its two end vertices are critical vertices.*

We will consider shortest paths that use a restricted set of lines from an arrangement. The restrictions are made by looking at *wedges*:

Definition 6 *Let γ and ψ be distinct non-parallel lines, which divide the plane into four quadrants. Let r be a point not on γ or ψ. The* wedge *$W(\gamma, \psi, r)$ is the quadrant of the plane that contains the point r. The intersection point $\gamma \cap \psi$ is the* apex *of the wedge. A line α* respects a wedge *if the line parallel to α through the apex of the wedge does not intersect the interior of the wedge.*

The following pair of lemmas provide upper and lower bounds on the length of a shortest path that respects a wedge.

Lemma 3 *Let $W = W(\gamma, \psi, r)$ be a wedge with apex p. Let P be the shortest path from r to p in an arrangement of lines that respect the wedge W. Then the length of P is at least half the perimeter of the parallelogram R with edges parallel to γ and ψ and diagonal $\overline{pr}$.*

Proof Sketch: Proceed by induction on the number of edges in the path. In the induction step, a path either contains some vertex inside R, where we break the path and apply the induction hypothesis to each half, or the path is completely outside R, where the result follows immediately. ∎

Lemma 4 *Let $W = W(\gamma, \psi, r)$ be a wedge with apex p. If α is any line through r, then the length of the shortest path from r to p in the arrangement formed by lines α, γ, and ψ is at most the perimeter of the parallelogram with edges parallel to γ and ψ and diagonal $\overline{rp}$.*

Proof Sketch: Begin with the wedge W and parallelogram pa_1ra_2 as in figure 3. Define points b_1 and b_2 on lines γ and ψ such that $|\overline{a_1b_1}| = |\overline{a_2b_2}| = \sqrt{|\overline{pa_1}||\overline{pa_2}|}$. By similar triangles, points b_1, r, and b_2 are collinear. Algebra shows that the paths rb_1p and rb_2p have total length at most the perimeter of parallelogram pa_1ra_2. Since the line α intersects either pb_1 or pb_2, the result follows. ∎

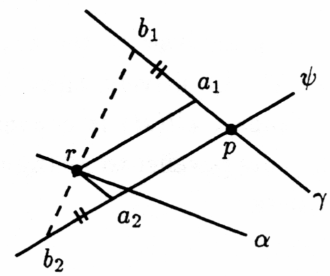


Figure 3: The wedge and parallelogram layout for Lemma 4.

The shortest path connecting a wedge point to its wedge apex using lines that respect the wedge can always be well-approximated by a path in a subarrangement of constant size. This observation, summarized in the corollary below, forms the building block for the proof of our approximation algorithm.

Corollary 5 *Using the notation of Lemmas 3 and 4, in any arrangement of lines containing α, γ, and ψ, if the shortest path from r to p uses only edges that respect W then this path is well-approximated by the shortest path from r to p in the arrangement consisting only of α, γ, and ψ.*

3.2 Approximation in the Arrangement Graph

To prove that Dijkstra's algorithm extracts a well-approximating st-path, we show that every shortest st-path in the arrangement either contains no critical vertices or at least one critical vertex that satisfies some special conditions (Lemma 7). In the former case, a simple path in the arrangement graph well-approximates the shortest path (Lemma 6). In the latter case, a more complex approximate path is found (Lemma 8).

Lemma 6 *If P is a shortest path from s to t which does not contain a critical vertex then the shortest path from s to t on the arrangement of λ^+, λ^-, and $Cell(st)$ well-approximates P.*

Proof: We prove the approximation factor for the the arrangement of λ^+, λ^-, and the lines that contain s and t. The path that uses $Cell(st)$ can only be shorter.

Let c be the point $\lambda^+ \cap \lambda^-$, let $W_s = W(\lambda^+, \lambda^-, s)$ and let $W_t = W(\lambda^+, \lambda^-, t)$. Note that since λ^+ and λ^- both separate s and t, the wedges W_s and W_t are opposite quadrants and hence, any line that respects one respects the other.

It follows from Lemma 3 that a shortest path P from s to t that visits no critical vertex (and hence all its edges respect both W_s and W_t) can be assumed to lie entirely within $W_s \cup W_t$ (and hence P must pass through c). Thus by Lemma 4, P can be well-approximated by the concatenation of the paths that well-approximate the subpath of P from s to c and the subpath of P from c to t.

Since the arrangement graph G contains the point $\lambda^+ \cap \lambda^-$ as a vertex and it is connected to $Cell(st)$, the well-approximating path exists in G for Dijkstra's algorithm to find. ■

Lemma 7 *Let P be a shortest path from s to t. Either P contains no critical vertices or P contains at least one critical vertex between the last pre-critical vertex a and the first post-critical vertex b in P.*

Proof: Assume that there is a shortest path P that contradicts the theorem; this implies that the subpath of P from s to a contains a critical vertex, the subpath from b to t contains a critical vertex, or they both contain critical vertices. We may assume without loss of generality that the subpath from b to t contains a critical vertex (otherwise switch the roles of s and t). Let z be the first critical vertex in P after b. We may also assume that b is a lower cell vertex. Thus, z is on the upper cell (otherwise P would backtrack on the lower cell).

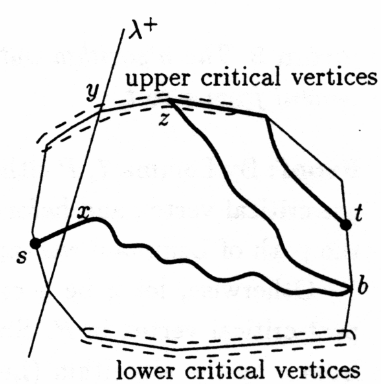


Figure 4: Illustration of the analysis of the structural properties of a shortest path.

Since λ^+ is a cross line, it separates s and t. Post-critical vertex b lies on the same side of λ^+ as t. Once the path P crosses λ^+ from s to b, it does not re-cross λ^+. Thus, the vertex z that follows b in P is on the same side of λ^+ as b and t.

Since λ^+ separates s from z and λ^+ does not intersect the upper pre-critical region, λ^+ intersects the upper critical region at a vertex y that precedes z in the upper cell ordering. Let x be the vertex where the path P and the line λ^+ first intersect.

If the path P does not visit a lower critical vertex between x and z then we can shorten P by following λ^+ from x to y and and then following cell edges from y to z (see figure 4).

Therefore, the path P must visit a lower critical vertex u between x and z. The path P cannot visit an upper pre-critical vertex after u because they are all on the other side of λ^+ from u and t. It cannot visit a lower pre-critical vertex after u (otherwise P would backtrack on the lower cell). ■

Lemma 8 shows that the critical vertex of Lemma 7 is well-approximated. Theorem 9 then provides the final guarantee on the length of the approximate shortest path.

Lemma 8 *Let P be any shortest path from s to t. Each cell vertex in P up to (but not including) the first post-critical vertex is well-approximated.*

Proof Sketch: The proof is by induction on a $\prec$-order of the cell vertices. The ordering is somewhat involved and has been omitted for space from this abstract.

Let z be a cell vertex that occurs before the first post-critical vertex in P. If z is adjacent to cell vertex y by a cell edge or to cell vertex x by a cross edge in P then, by induction, x and y are well-approximated. Thus z is well-approximated.

Otherwise, consider the edges in P that precede z. We claim that these edges respect the wedge $W_z = W(\overleftarrow{yz}, \overleftarrow{xz}, s)$. If they don not then there is a last edge $\overline{ab}$ in P that does not respect W_z.

We show that the vertex b lies on the cross edge $\overline{xz}$ of W_z. Since $\overleftrightarrow{ab}$ must separate s from z and t, and z is not a post-critical vertex, the line $\overleftrightarrow{ab}$ respects the wedge W_z. ■

Theorem 9 *The algorithm outputs a path that well-approximates any shortest path P on the arrangement from s to t.*

Proof: By Lemma 7, P either contains no critical vertex or visits a critical vertex after the last pre-critical vertex and before the first post-critical vertex. If P contains no critical vertex then the path of Lemma 6 well-approximates P.

Otherwise, let z be a critical vertex after the last pre-critical vertex and before the first post-critical vertex in P. Since z precedes the first post-critical vertex, z is well-approximated from s by the algorithm (Lemma 8). Since z follows the last pre-critical vertex, z is also well-approximated from t (by a symmetric argument). Thus the path generated by the algorithm is at most twice the length of P. ■

Acknowledgments

We thank Günter Rote and anonymous referees for their comments and improvements on an earlier version of our results.

References

[1] H. Alt, B. Chazelle, and R. Seidel (editors). Computational geometry. Dagstuhl-Report; 109, 1995.

[2] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[3] H. Edelsbrunner. *Algorithms in Combinatorial Geometry*, volume 10 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, Heidelberg, West Germany, 1987.

[4] K. Mulmuley. *Computational Geometry: An Introduction Through Randomized Algorithms.* Prentice Hall, Englewood Cliffs, NJ, 1993.