Discrete Comput Geom (2008) 40: 377–394
DOI 10.1007/s00454-007-9025-y

Tight Bounds for Connecting Sites Across Barriers

David Krumme · Eynat Rafalin ·
Diane L. Souvaine · Csaba D. Tóth

Received: 19 February 2007 / Revised: 17 June 2007 /
Published online: 5 February 2008

Abstract Given m points (*sites*) and n obstacles (*barriers*) in the plane, we address the problem of finding a straight line minimum cost spanning tree on the sites, where the cost is proportional to the number of intersections (crossings) between tree edges and barriers. If the barriers are infinite lines, it is known that there is a spanning tree such that every barrier is crossed by $O(\sqrt{m})$ tree edges, and this bound is asymptotically optimal. Asano et al. showed that if the barriers are pairwise disjoint line segments, then there is a spanning tree such that every barrier crosses at most 4 tree edges and so the total cost is at most $4n$. Lower bound constructions are known with 3 crossings per barrier and $2n$ total cost.

We obtain tight bounds on the minimum cost of spanning trees in the special case where the barriers are interior disjoint line segments that form a convex subdivision of the plane and there is a point in every cell of the subdivision. In particular, we show that there is a spanning tree such that every barrier crosses at most 2 tree edges, and there is a spanning tree of total cost $5n/3$. Both bounds are the best possible.

Work by Eynat Rafalin and Diane Souvaine was supported by the National Science Foundation under Grant #CCF-0431027.

E. Rafalin's research conducted while at Tufts University.

D. Krumme · D.L. Souvaine (✉)
Department of Computer Science, Tufts University, Medford, MA 02155, USA
e-mail: dls@cs.tufts.edu

D. Krumme
e-mail: dwk@cs.tufts.edu

E. Rafalin
Google Inc., Mountain View, CA 94043, USA
e-mail: eynat@google.com

C.D. Tóth
Department of Mathematics, University of Calgary, Calgary, Canada
e-mail: cdtoth@ucalgary.ca

Keywords Crossing number · Spanning trees

1 Introduction

Chazelle and Welzl [2, 14] proved that for n points in d-space, there is a spanning tree such that every hyperplane intersects at most $O(n^{1-1/d})$ edges of the tree. This bound is tight apart from the constant factor: every spanning tree on n points of a d-dimensional integer lattice section crosses an axis-aligned hyperplane at least $\Omega(n^{1-1/d})$ times. This result and its extensions by Matoušek [10] were used in efficient range searching data structures in finite VC-dimensional range spaces.

In the plane, in particular, there is a spanning tree for n points and a set of barrier lines such that every line crosses at most $O(\sqrt{n})$ edges [9, 15]. Interestingly, if the barriers are disjoint line segments rather than infinite lines, then one can construct a spanning tree that crosses every barrier at most $O(1)$. The study of spanning trees across disjoint barriers was motivated by the *multi-point location problem*, where a set of points lies within an underlying geometric data structure. This question often arises in geometric modeling systems (e.g., in robotics, vision, radio wave propagation prediction, CAD/CAM, and others). Given a subdivision of the plane (e.g. a triangulation) with $O(n)$ edges and vertices and a set S of m distinct points, we wish to find the face containing s_i, for every $i = 1, \dots, m$. Through iterated use of a worst-case optimal planar point location algorithms (e.g. [3, 7], or [11]), one can locate all points in $O(m \log n)$ time. A potential strategy for locating all m points in $O(m+n)$ time is to find a walk w of $O(m+n)$ combinatorial complexity that visits all m points. Snoeyink and van Kreveld [13] demonstrated empirically that walking to the next point to be processed with the method of Guibas and Stolfi [4] is better than traversing all faces of T in a pre-specified order. A pre-order traversal of a spanning tree T of S provides such a walk in time 2 cost(T), where cost(T) is the number of crossings between tree edges and the boundaries of the cells of the subdivision.

Snoeyink [12] posed a specific version of this problem: Given a set S of m distinct points (sites) and a set L of n segments (barriers) in the plane where the relative interiors of the barriers are pairwise disjoint and no site lies on any barrier, does a spanning tree T of S always exist that, when embedded with straight line edges, has the property that no barrier of L crosses more than a constant number of edges of T?

Asano et al. [1] gave an upper bound of 4 on the number of crossings per barrier, which implies an upper bound of $4n$ on the total cost. Hoffmann and Tóth improved this bound to 3 in the special case that all segments are axis-parallel [6], and they constructed an example (which can be realized with axis-parallel segments) where every spanning tree crosses some barrier at least three times [5]. For the restricted problem where barriers form a convex planar subdivision and each cell contains a site, Krumme et al. [8] showed that a spanning tree with at most 3 crossings per barrier always exists.

Contribution We obtain tight bounds on the minimum cost of a spanning tree in the special case that the barriers are interior disjoint line segments that form a convex subdivision of the plane and there is a site in every cell. In this case, we may as well assume that every cell contains exactly one site by specifying a single site in each cell (since we can connect the designated site to all other sites of the same cell without crossing any barriers). We prove that there exists a straight line spanning tree that crosses every barrier at most twice and one with a total cost of at most $\frac{5}{3}n$. On the other hand, there are examples where any spanning tree crosses some barrier at least twice and others where any spanning tree has total cost at least $\frac{5}{3}n - O(\sqrt{n})$.

Organization Sects. 2 and 3 present the proofs for our upper and lower bounds for the number of crossings per barrier and the total number of crossings required for connecting all sites. Section 4 discusses some directions for future work.

2 The Maximal Number of Crossings Per Barrier

Theorem 1 *Given a set $L = \{l_1, \ldots, l_n\}$ of n pairwise non-crossing line segments* (barriers) *in the plane that subdivide the plane into $n+1$ convex cells, and given a set S of $n+1$ points* (sites), *one in each convex cell, then there exists a straight line spanning tree T on the sites such that the edges of T cross every barrier at most twice.*

Figure 1 depicts an example where every spanning tree on the sites crosses some barrier at least twice, verifying that Theorem 1 is tight. The n barriers are n half-lines with their initial portion lying along the sides of a regular n-gon C. Site s_0 lies at the center of C. Sites $s_1, s_2, \ldots, s_n$ are centrally symmetric around s_0 such that any segment $s_0 s_i$, $i = 1, 2, \ldots, n$, crosses two consecutive barriers along C. Every spanning tree T must contain an edge $e_1 = s_0 s_i$ for some $i = 1, 2, \ldots, n$ that crosses both barriers that bound the cell containing s_{i+1}. Any edge $e_2 \in T$ incident to s_{i+1} crosses one of these two barriers a second time.

We prove Theorem 1 using an iterative algorithm that computes a spanning graph on the sites. For ease, we assume that there is a small opening at the left endpoint of each barrier, that can let an edge connecting the two sites in the two adjacent cells through it without introducing new crossings.

At each iteration, we augment the graph by adding a single vertex (site) and an edge. For each new site, we add the bent edge that connects it to the site in an adjacent cell through a small openings in the set of barriers, this keeps the graph connected

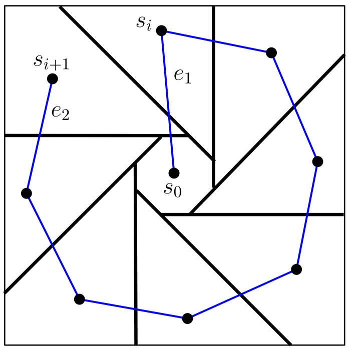

Fig. 1 Lower bound construction. At least one barrier has to cross two edges of a spanning tree

and the new bent edge does not cross any barrier. Intuitively, the algorithm then transforms the intermediate graph to a straight-line graph by contracting the bent edge as if it were a rubber band, without introducing intersections between edges. The rubber band contracts to a straight edge unless it hits "obstacles" formed by sites or by regions swept by previous rubber bends. The beauty of the algorithm is the ingenious use of a weakly simple polygon to bound the area already swept by the algorithm and to verify that a bent edge can be pulled across a barrier at most once from each end, guaranteeing at most two crossings per barrier.

In step i, $1 \le i \le n$, we produce a connected straight line graph T_i with vertex set $V(T_i) \subseteq S$, $|V(T_i)| \ge i + 1$, that crosses every barrier at most twice. A significant portion of the proof of correctness involves identifying a set of invariants and proving that they are maintained throughout the progress of the algorithm. Section 2.1 presents some basic assumptions and describes invariants maintained throughout the algorithm, Sect. 2.2 describes the progress of the algorithm and Sect. 2.5 proves that it terminates and achieves the cost of 2 crossings per barrier.

2.1 Basic Assumptions and Invariants

We assume that the barriers and the sites lie in a bounding box B and no two barriers share a common endpoint. We choose a coordinate system such that B is *not* axis-parallel, no barrier is vertical, and the left endpoints of the barriers have distinct x-coordinates. Let b_0 denote the leftmost corner of the bounding box. Observe that the leftmost corner of every cell of the subdivision is either b_0 or a left endpoint of a barrier.

Ordering The enumeration of all sites s_i, $i = 1, 2, \dots, n$, and consequently all cells defines a method for creating a unique set of bent edges that connects all sites without crossing each other or any barriers. Assume that $L = \{\ell_1, \dots, \ell_n\}$ represents the barriers in increasing order of the x-coordinates of their left endpoints $q_1, q_2, \dots, q_n$. Let C_0 be the cell adjacent to b_0, and, for every $i = 1, 2 \dots, n$, let C_i denote the cell whose leftmost point is q_i. Denote by s_i the site contained in C_i for $i = 0, 1, \dots, n$, see Fig. 3a. Let $\hat{C}_i$ denote the cell adjacent to C_i such that ℓ_i and q_i lie on the line separating C_i and $\hat{C}_i$; and let $\hat{s}_i$ denote the site in $\hat{C}_i$. For $i = 1, 2, \dots, n$, the *V-shape* of s_i is the 2-edge path $w_i = (s_i, q_i, \hat{s}_i)$, see Fig. 3b. It can also be considered a single edge from s_i to $\hat{s}_i$ with a bend at q_i. Let W denote the set of all straight line segments $s_i q_j$ occurring in V-shapes. Observe that the set $L \cup W$ contains pairwise non-crossing line segments.

Our algorithm proceeds in at most n steps. In step i, $i = 1, 2, \dots, n$, it computes a connected straight line graph T_i and a weakly simple polygon P_i. For clarity, we adopt the terminology that graphs have *vertices* and *edges*; polygons have *corners* and *sides*; and barriers have *endpoints*.

Weakly Simple Polygons Our algorithm will maintain a weakly simple polygon whose corners are either sites or apices of V-shapes. Intuitively, this polygon covers the region that was previously swept by the algorithm. It is used for guiding the process of transforming bent edges to straight-line edges but preventing intersections between new edges and any previously drawn edges.

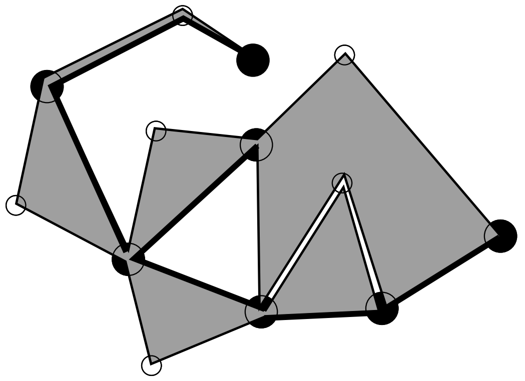

Fig. 2 A weakly simple polygon (perturbed by ϵ into a *simple* polygon) that can be constructed by our algorithm. Polygon sides *in bold* represent tree edges of the intermediate graph T_i'. All other sides are part of some V-shape. *Circles* represent polygon corners. *Full circles* represent sites. *Empty circles* represent apices of V-shapes

A closed polygonal chain $P = (p_0, p_1, \dots, p_{k-1}, p_k = p_0)$ where any point p in the plane could occur several times represents a *weakly simple polygon* if the sides $p_i p_{i+1}$ are pairwise non-crossing segments and if each point p_i can be moved by a distance at most an arbitrarily small $\epsilon > 0$ to a position $p_i{}'$ so that the closed polygonal chain $P' = (p_0{}', p_1{}', \dots, p_{k-1}{}', p_k{}' = p_0{}')$ represents the boundary of a simple polygon in counterclockwise order, with the interior lying to the left. A corner $p_i \in P$ is *convex* (*reflex*) if the clockwise angle $\angle(p_{i-1}, p_i, p_{i+1})$ measures less (more) than 180°. Figure 2 depicts a weakly simple polygon.

We denote the interior of the polygon by $\text{int}(P)$. The weakly simple polygon P and its interior $\text{int}(P)$ jointly cover a *closed polygonal region*, which we denote by $\overline{P}$. The boundary of this region is denoted by $\partial\overline{P}$. The polygonal region $\overline{P}$ may have holes and its boundary may be disconnected. The edges of P may be traveled several times during a traversal along the boundary of the polygon.

Initialization T_0 has vertex set $\{s_0\}$ and no edges. P_0 is a degenerate polygon with a single corner s_0.

Invariants The vertex set $V(T_i)$ and the region $\overline{P}_i$ are monotone increasing in every step. That is, $V(T_i) \subset V(T_{i+1})$ and $\overline{P}_i \subset \overline{P}_{i+1}$. We maintain the following invariants.

1. *The graph T_i.* T_i is a connected straight line graph.
   (a) Its vertex set is $V(T_i) \subseteq S$, with $|V(T_i)| \geq i + 1$.
   (b) Every edge of T_i is a side of the polygon P_i.
2. *The polygon P_i.* P_i is a weakly simple polygon with the following properties.
   (a) The *convex corners* of $\overline{P}_i$ are in the set $V(T_i) \cup \{q_1, \dots, q_n\}$.
   (b) If a left endpoint q_i of a barrier is a *convex corner* of $\overline{P}_i$, then the V-shape associated with q_i lies on the boundary of the region $\overline{P}_i$, that is, $w_i \subset \partial\overline{P}_i$.
   (c) Every *side* of P_i is either part of the graph T_i or part of a V-shape.
   (d) If a *side* of P_i intersects a *line segment* $\ell \in L \cup W$ at point r, then the portion of ℓ between r and one of ℓ's endpoints is completely contained in the interior of P_i.
   (e) The *interior* of P_i contains *no* sites.

By maintaining the invariants, we guarantee that at every step of the algorithm at most two edges of the graph can cross a barrier.

Proposition 1 *If T_i and P_i satisfy invariants* 1b *and* 2d, *then at most two edges of T_i can cross a barrier.*

Proof By invariant 1b, every edge of T_i is a side of the polygon P_i. By invariant 2d, if an edge of P_i intersects a barrier $\ell \in L$ at point r, then the portion of ℓ between r and one of the endpoints of ℓ is completely contained in the interior of P_i. Since every barrier $\ell \in L$ has only two endpoint, the boundary of P_i can cross ℓ at most twice. □

2.2 Progress of the Algorithm

We start with a graph T_{i-1} and a weakly simple polygon P_{i-1} satisfying the above invariants. To generate T_i and P_i, we consider two cases. In both cases, we compute an intermediate graph T_i' on a vertex set $V(T_i')$, $|V(T_i')| > |V(T_{i-1})|$, which has one or two edges with a bend, and an intermediate polygon P_i'. Section 2.3 describes how the intermediate graph T_i' is transformed into a straight line graph T_i by straightening the bent edges and updating P_i'.

Case 1 *For every site $s_j \notin V(T_{i-1})$, the segment s_jq_j is disjoint from $\overline{P}_{i-1}$.* Consider the smallest index $1 \leq j \leq n$ such that $s_j \notin V(T_{i-1})$. Note that the leftmost point of $\hat{C}_j$ is to the left of q_j, and so $\hat{s}_j \in V(T_{i-1})$. We augment the graph T_{i-1} by the V-shape w_j. We put $T_i' = T_{i-1} + w_j$, where w_j is an edge *with one bend*. We append the V-shape w_j to the weakly simple polygon with two different orientations by letting $P_i' = P_{i-1} + (\hat{s}_j, q_j, s_j, q_j, \hat{s}_j)$. (See Fig. 3b, f, h.)

Case 2 *There is a site $s_j \notin V(T_{i-1})$ such that the segment s_jq_j is not disjoint from $\overline{P}_{i-1}$.* By invariant 2e, the $\mathrm{int}(P_{i-1})$ contains no site and so $s_j \notin \overline{P}_{i-1}$, hence s_jq_j must intersect the boundary $\partial\overline{P}_{i-1}$. Let r be the first intersection point of s_jq_j and $\partial\overline{P}_{i-1}$. By invariant 2c and since all V-shapes are noncrossing, r lies on an edge s_as_b of T_{i-1}. We replace edge s_as_b by two edges (each *with one bend*), $e_{j_1} = (s_j, r, s_a)$ and $e_{j_2} = (s_j, r, s_b)$ to get $T_i' = T_{i-1} - s_as_b + e_{j_1} + e_{j_2}$. We append the segment rs_j to the weakly simple polygon with two different orientations by letting $P_i' = P_{i-1} + (r, s_j, r)$. (See Fig. 3d.)

In Case 1, it follows from invariant 2d that the segments q_js_j and $q_j\hat{s}_j$ are disjoint from $\mathrm{int}(P_{i-1})$. In Case 2, the segment rs_j is disjoint from $\mathrm{int}(P_{i-1})$ by definition. It follows that P_i' is a weakly simple polygon, and it is easy to check that T_i' and P_i' satisfy all conditions of invariant 2. The graph T_i' is *not* a straight line graph; it is a connected graph where each edge is a *polyline* and satisfies invariants 1a and 1b. Note, however, that every bend in the graph T_i' maps to a reflex corner of the weakly simple polygon $\overline{P_i'}$.

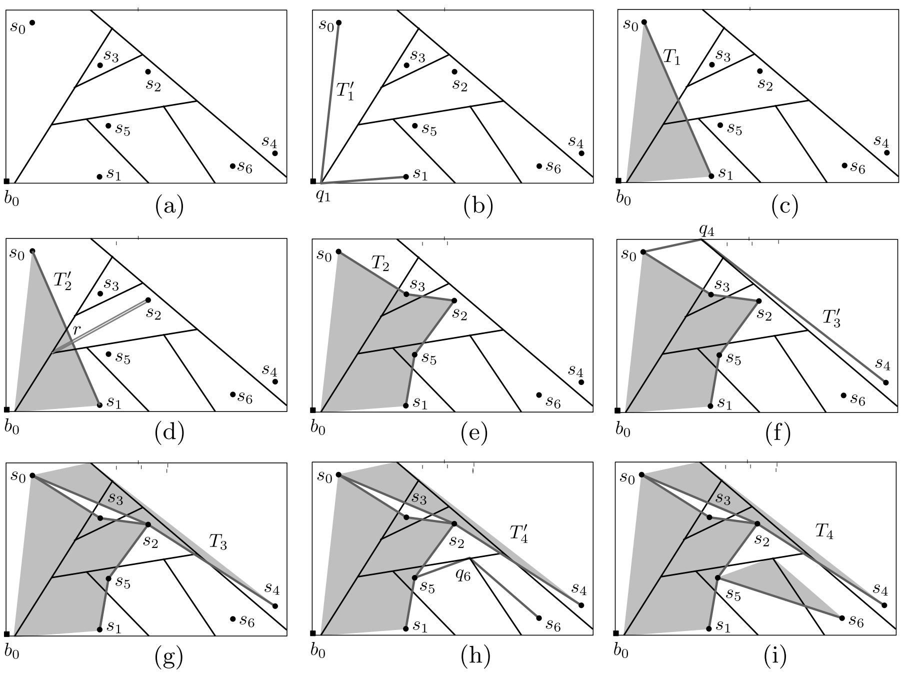

Fig. 3 Six segments, seven sites, and the steps of our algorithm. In each instance the *shaded region* is bounded by a weakly simple polygon. Parts **b**, **c** reflect a Case 1 update, parts **d**, **e** reflect a Case 2 update and parts **f**, **g** and **h**, **i** reflect a Case 1 update

2.3 Removing a Bend

We are given a connected graph $G = T_i'$ with bends satisfying invariants 1a and 1b and a weakly simple polygon $P = P_i'$ satisfying invariant 2. G and P also satisfy the following invariant.

3. Each edge of G has at most two bends and each bend maps to a *reflex corner* of $\overline{P}$.

Intuitively, we place a rubber band along an edge e with bends. Ideally the rubber band contracts to a straight line edge, and we add the area swept by the rubber band to the polygon P (for example, Fig. 3b–c). Since we want to generate a weakly simple polygon whose interior is disjoint from the sites, the rubber band may wrap around obstacles represented by S and convex corners of P.

Remove the bends recursively. Consider an edge with at most two bends $e = (s_a, r_1, r_2, s_b)$. By invariant 1b, the polyline e lies along consecutive sides of P and by invariant 3 these sides form a reflex chain along P_i. Let $\pi = \pi(s_a, r_1, r_2, s_b)$ denote the shortest path between s_a and s_b that is homotopic to the path (s_a, r_1, r_2, s_b) in the presence of the obstacles $\overline{P} \cup S$. (See Fig. 4.)

Consider the polyline $\pi = (s_a = t_0, t_1, t_2, \dots, t_{k-1}, t_k = s_b)$, $k \geq 1$, passing through a sequence of sites and convex corners of P (Fig. 4). For every segment $t_{j-1}t_j \subset \pi$, $j = 1, 2, \dots, k$, we design a new edge e_j (possibly with bends) and a path f_j. If $t_{j-1}, t_j \in S$, then $e_j = f_j = t_{j-1}t_j$. If $t_{j-1} \in S$ and $t_j \in \{q_1, q_2, \dots, q_i\}$

is an apex of a V-shape, then by invariant 2b, t_j is incident to the sides $t_j s_h$ and $t_j s_{h'}$ along polygon P, where $s_h, s_{h'} \in V(G)$. We may assume w.l.o.g. that s_h is on the same side of the line $t_j s_{h'}$ as t_{j-1}. Let $e_j = (t_{j-1}, t_j, s_h)$ with one bend, and let $f_j = (t_{j-1}, t_j, s_h, t_j)$ that traverses the side $t_j s_h$ twice. The case that $t_{j-1} \in \{q_1, q_2, \ldots, q_i\}$ and $t_j \in S$ is analogous. Finally, if both t_{j-1} and t_j are apices of V-shapes, then we design an edge $e_j = (s_g, t_{j-1}, t_j, s_h)$ with two bends such that s_g and s_h are sites adjacent to t_{j-1} and t_j, respectively (Fig. 4). We also design a path $f_j = (t_{j-1}, s_g, t_{j-1}, t_j, s_h, t_j)$ that traverses the sides $t_{j-1}s_g$ and $t_j s_h$ twice, once in each direction.

We update the graph by setting $G := G - e + \sum_{j=1}^{k} e_j$. If Δ denotes the region enclosed by the closed curve $(s_a, r_1, r_2, s_b) \cup \pi(s_a, r_1, r_2, s_b)$, then we let $P = P - e + \sum_{j=1}^{k} f_j$, and so $\overline{P} = \overline{P} \cup \Delta \cup \bigcup_{j=1}^{k} e_j$. Note that s_a or s_b may occur as an internal vertex in π. Our argument goes through without any change if this happens, but needs to allow that G has loops.

Call this subroutine recursively while G has an edge with bends. The number of edges may increase in each step. Let

$$f(G,P) = \#(\text{non-straight edges of } G) + 2 \cdot \#(\text{convex verticesof } \overline{P}) + 2|S \setminus V(G)|$$

Observe that $f(G,P)$ decreases in each step. Since the number of convex vertices of $\overline{P}$ and $|S \setminus V(G)|$ are bounded, the recursion terminates with a straight line graph $T_i = G$ and a polygon $P_i = P$.

2.4 Invariant Maintenance

Invariant 2 We have noted that we maintain invariant 2 when passing from P_{i-1} to P'_i. Here we show that the subroutine in Sect. 2.3 also maintains invariant 2. Since the path $\pi(s_a, r_1, r_2, , s_j)$ is homotopic to (s_a, r_1, r_2, s_j), the open region Δ contains no additional sites or corners (invariant 2e). Because $\pi(s_a, , r_1, r_2, s_b)$ is a shortest path homotopic to a reflex chain along $\partial\overline{P}$, π is also a reflex chain, and so we do not add any new convex corners to polygon P (invariant 2a). The two endpoints s_a and s_b of π are sites, and so the neighbors of a convex corner q of $\overline{P}$ remain the same (invariant 2b). Note, however, that if π passes through a convex corner r of $\overline{P}$, then r will not be a convex corner anymore and invariant 2b no longer poses restriction on r (see e.g. Fig. 4). Every new side along the boundary $\partial\overline{P}$ is covered by new edges of G (invariant 2c).

The edges of V-shapes are segments in W, so the portions of the path $\sum_{j=1}^{k} f_j$ along the V-shapes cannot cross barriers. Since $\pi(s_a, r_1, r_2, s_b)$ is a homotopic shortest path to a reflex chain (s_a, r_1, r_2, s_b) along $\overline{P}$, for any barrier partially contained in

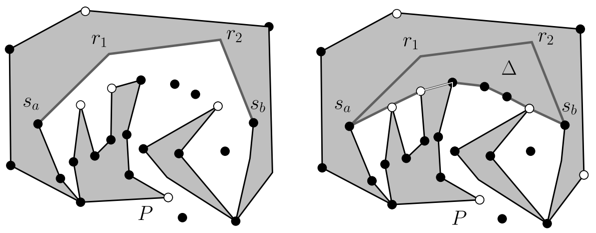

Fig. 4 The shortest path $\pi(s_a, r_1, r_2, s_b)$ and region Δ which is added to P. *Empty circles* represent apices of V-shapes on $\partial\overline{P}$

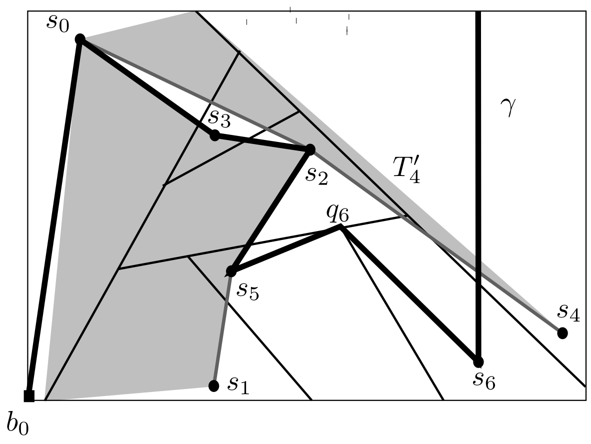

Fig. 5 The path γ for $h=6$ of the instance of the algorithm depicted in Fig. 3

this region, the portion between the intersection point and one of its endpoints must be fully contained in the region (invariant 2d).

Invariants 1 and 3 We have noted that we maintain the connectivity of the graph when we pass from T_{i-1} to T_i, and the subroutine of Sect. 2.3 transforms T'_i into a straight line graph T_i. It remains to show that the subroutine maintains the connectivity of the graph G. In each step of the subroutine, we replace an edge $e=(s_a,r_1,r_2,s_b)$ by a sequence of edges $M(E)=(e_1,e_2,\dots,e_k)$, where two consecutive edges are either adjacent or incident to two sites of a V-shape $(s_{i'},q_{i'},\hat{s}_{i'})$ along $\partial\overline{P}$. Invariants 1 and 3 guarantee that there is a path $A(i')\subset G$ between the sites $s_{i'}$ and $\hat{s}_{i'}$. Hence, it is enough to show that the path $A(i')$ is not disconnected when we remove edge e, that is, we need to show that $A(i')$ does not pass through the edge e.

Note a global view of the algorithm. The main algorithm applied Case 1 in step $i=1$ because polygon P_0 lies in the interior of cell C_0. For a step i, let $h=h(i)$, $1\le h\le i$, denote the last step before (and including) step i when the algorithm applied Case 1.

Proposition 2 *The edges of T_{h-1} are never removed from our graph.*

Proof Graph T_{h-1} has no edge that intersects a cell containing a site of $S\setminus V(T_{h-1})$, otherwise Case 2 would have been applied in step h. So Case 2 (which replaces an edge by two bent edges) is never applied to the edges of T_{h-1} in subsequent steps. □

Return to step i, and consider the subroutine that replaces an edge of T'_i recursively by new edges. In one step of this subroutine, a sequence $M(e)$ of edges replaces a single edge e. Our key observation is that if two consecutive edges $e_{j-1},e_j\in M(e)$ are incident to two sites of a V-shape $(s_{i'},q_{i'},\hat{s}_{i'})$ added to the polygon before step h, then the sites $s_{i'}$ and $\hat{s}_{i'}$ are connected via edges of T_{h-1}, that are present in graph G. We need focus only on the single V-shape $(s_h,q_h,\hat{s}_h)$, inserted into P in step h.

We show that in steps $h,h+1,\dots,i$, no shortest path π hits the apex of the V-shape h, proving that Invariants 1 and 3 are maintained. First assume that the leftmost vertex of the V-shape $(s_h, q_h, \hat{s}_h)$ is the apex q_h. By our ordering scheme, the apices of V-shapes added previously to the polygon lie to the left of q_h, and so all internal vertices of $\pi(s_h, q_h, \hat{s}_h)$ are sites. In every subsequent step i', $h < i' \le i$, we recursively extend the edges of $\pi(s_h, q_h, \hat{s}_h)$ to the right along segment $s_{i'}q_{i'}$, and so by our ordering scheme, no apex of any V-shape can be an internal vertex of any path π.

Next let us assume that the leftmost vertex of the V-shape $(s_h, q_h, \hat{s}_h)$ is $\hat{s}_h$, and so its rightmost vertex is s_h. Assume w.l.o.g. that the line segment $\hat{s}_h s_h$ lies *below* the V-shape $(s_h, q_h, \hat{s}_h)$ (if it lies above the V-shape, we can argue analogously with the vertical mirror image). We define a path γ that partitions the bounding box B into two regions and passes through the V-shape $(s_h, q_h, \hat{s}_h)$: let γ start with the line segment $b_0 s_0$, it follows some path in the graph T_{h-1} from s_0 to $\hat{s}_h$; then it follows the V-shape from $\hat{s}_h$ to s_h, and terminates with a vertical line segment $s_h b_1$ connecting s_h to the top side of B (refer to Fig. 4). It suffices to show that no edge created in steps $h, h+1, \ldots, i$ lies in the region above γ. Indeed, no newly created edge crosses any of $b_0 s_0$, T_{h-1}, and $(s_h, q_h, \hat{s}_h)$. It remains to show that the new edges cannot extend from the right side of the vertical segment $s_h b_1$ to its left side. In step h, the new edges may extend along V-shapes inserted into P prior to step h whose apices lie below γ. Since the apices of all previous V-shapes lie to the left of q_h, these V-shapes cannot cross $s_h b_1$. In every subsequent step i', $h < i' \le i$, we recursively extend the edges created in step h to the right along a segment $s_{i'}q_{i'}$, and so these edges cannot extend to the left of $s_h b_1$, either. We conclude that no shortest path can hit the apex q_h of the V-shape $(s_h, q_h, \hat{s}_h)$.

2.5 Termination

The final graph T_i is a connected straight line graph over all sites. Since all invariants are maintained throughout the algorithm, we can apply Proposition 1 and conclude that at most two edges of T_i cross each barrier, proving Theorem 1.

3 Total Number of Crossings

In this section, we show that for every input (L, S) of n line segments forming a convex subdivision and $n+1$ sites, one in each convex cell, one can construct a straight line spanning tree T of total cost at most $\frac{5}{3}n$. We also present a family of inputs for which any spanning tree has a total cost of at least $\frac{5}{3}n - O(\sqrt{n})$. Our upper bound is based on a two phase algorithm: In the first phase, we greedily add an edge between sites s_i and s_j if $s_i s_j$ crosses at most one barrier and this barrier does not cross any previously added edge. The resulting graph G crosses every barrier exactly once, but is not always connected. We analyze the structure of the connected components of G and show that in the second phase of our algorithm one can augment the graph G to a spanning graph T such that the new edges increase the total cost by at most $\frac{2}{3}n$.

3.1 A Greedy Algorithm that Almost Solves the Problem

We assume that the barriers and the sites lie in a bounding box B, and the endpoints of the barriers are all distinct. The first phase in our algorithm for constructing a spanning tree T is the following simple greedy procedure: Initialize G to be a graph with vertex set S and no edges. For any two sites $s_i, s_j \in S$ lying in two adjacent cells, if the line segment $s_i s_j$ crosses at most one barrier $\ell \in L$ and no edge of G crosses ℓ, then let $G = G + s_i s_j$. It is clear that the output graph G crosses every barrier at most once. We can implement this greedy algorithm in $O(n)$ time if we pre-compute the list of $O(n)$ pairs of adjacent cells and maintain during the algorithm the list of barriers already crossed by an edge of G.

If G is a spanning graph over all sites S, then our algorithm is complete and we have a spanning tree $T \subseteq G$ of total cost at most n. Assume, that G has $k \geq 2$ connected components denoted by $S_1, S_2, \ldots, S_k$. The second phase of our algorithm, in Sect. 3.3, will add edges between the components of G. But first we show, in Sect. 3.2, that the components of G have a very special structure.

3.2 Nested Structure of Components

For every $i = 1, 2, \ldots, k$, let M_i denote the union of the cells corresponding to the vertices of the component S_i of G. Every M_i, $i = 1, 2, \ldots, k$, is a polygonal *region* in the bounding box B containing all sites and all intersection points of barriers. Since edges of G connect sites in adjacent cells, every region M_i is connected and the regions jointly form a subdivision of the bounding box B.

Lemma 1 *For every barrier $\ell \in L$ there is an edge $e \in G$ that crosses ℓ.*

Proof Select a coordinate system where ℓ is horizontal, and let q_{left} and q_{right} denote its left and right endpoints, respectively. Let us denote the cells above ℓ by $A_1, A_2, \ldots, A_\alpha$ along ℓ such that A_1 is incident to q_{left}, and let a_i denote the site lying in A_i for $i = 1, 2, \ldots, \alpha$. Similarly, the cells below ℓ are denoted by $B_1, B_2, \ldots, B_\beta$ along ℓ such that B_1 is incident to q_{left}, and let b_i denote the site lying in B_i for $i = 1, 2, \ldots, \beta$ (see Fig. 6).

It is enough to show that there are two indices $i \in \{1, 2, \ldots, \alpha\}$ and $j \in \{1, 2, \ldots, \beta\}$ such that the segment $a_i b_j$ crosses no other barrier but ℓ. It follows that the greedy algorithm, when processing the first such adjacent pair (a_i, b_j), augment G with the edge $a_i b_j$ unless a previous edge of G already crosses ℓ.

Consider two adjacent cells A_i and B_j lying on opposite sides of ℓ. Their common boundary $t_{ij} = A_i \cap B_j$ is an interval along ℓ, which has non-zero length due to our assumption that no two barriers share an endpoint. We say that the pair (i, j) is *left-leaning* (*right-leaning*), if the segment $a_i b_j$ crosses the line through ℓ on the left

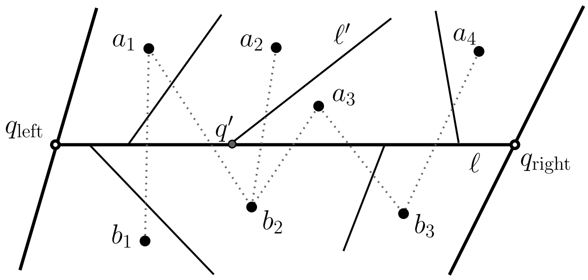

Fig. 6 Construction in the proof of Lemma 1. Pairs $(1, 1)$, $(1, 2)$ and $(2, 2)$ are right-leaning. The pairs $(3, 3)$ and $(4, 3)$ are left-leaning and $(3, 2)$ is neither left- nor right-leaning

(right) of the interval t_{ij}, see an illustration in Fig. 6. It suffices to show that there is pair (i, j) which is neither left- nor right-leaning, and so $a_i b_j$ crosses no other barrier but ℓ. Since q_{left} and q_{right} must each lie either in the relative interior of another segment or on the bounding box, the pair $(1, 1)$ cannot be left-leaning and (α, β) cannot be right-leaning. Assume that the pair $(1, 1)$ is right-leaning, and let (i, j) be the first pair along ℓ that is *not* right-leaning. This means that $a_i b_j$ intersects l to the left of the right endpoint of t_{ij}. We may assume w.l.o.g. the previous pair is $(i-1, j)$, and since it is right-leaning, $a_{i-1}b_j$ crosses ℓ to the right of the left endpoint of t_{ij}. The pair (i, j) is not right-leaning, and we show that it cannot be left-leaning, either: Let q' denote the left endpoint of t_{ij}, and let ℓ' be the barrier along the boundary of A_{i-1} and A_i incident to l at q'. Since $(i-1, j)$ is right-leaning, ℓ' must intersect the line segment $a_{i-1}b_j$. This implies that a_i and b_j are on the same side of the line through ℓ' and so $a_i b_j$ intersects ℓ to the right of q'. □

Since M_i is not necessarily simply connected its boundary ∂M_i may not be connected. A *frame* γ is a connected component of a boundary ∂M_i for $i = 1, 2, \ldots, k$. A frame is a closed curve along portions of barriers and portions of ∂B.

Lemma 2 *For every $i = 1, 2, \ldots, k$, the boundary ∂M_i cannot contain an entire barrier.*

Proof Suppose, to the contrary, that there is a frame γ along the boundary of a region M_i that contains a barrier $\ell \in L$. Since ℓ is on the boundary of M_i, every cell along one side of ℓ belongs to region M_i and every cell along the opposite side of ℓ is outside M_i. By Lemma 1, there are two cells on two opposite sides of ℓ whose sites are connected by an edge of G (note that ℓ cannot cross any edge of G), a contradiction. We conclude that γ cannot contain an entire barrier. □

We define the *arc* as a maximal (nonempty) connected component of the intersection of a frame γ with either a barrier or ∂B. By Lemma 2, an arc cannot be an entire barrier. For an arc t, $t \not\subseteq \partial B$, we denote by $\ell(t) \in L$ the barrier containing t (note that the same barrier may appear several times along a frame γ). For every arc t, we define an *orientation*: $\sigma(t) \in \{$clockwise, counter-clockwise, neutral$\}$. Let $\sigma(t) =$ neutral if $t \subseteq \partial B$ or t does not contain either endpoint of $\ell(t)$; $\sigma(t) =$ clockwise if the *first* endpoint of t is an endpoint of the barrier $\ell(t)$ when traversing γ in clockwise order; $\sigma(t) =$ counter-clockwise if the *second* endpoint of t is an endpoint of the barrier $\ell(t)$ (by Lemma 1, t cannot contain both endpoints of $\ell(t)$), see Fig. 7.

Lemma 3 *All arcs along a frame have the same orientation.*

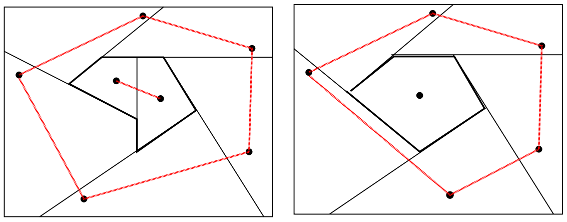
Fig. 7 Two examples of frames (*in bold*) created by the algorithm. *In both cases*, all of the arcs are oriented clockwise

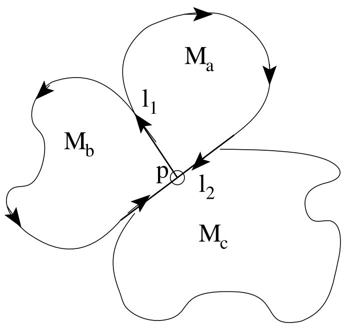

Fig. 8 Proof of Lemma 4

Proof Consider the cyclic sequence of the arcs $(t_1,\dots,t_\tau)$ along a frame γ. If $\gamma\neq\partial B$, then this sequence consists of more than one arc. The common point of every two consecutive arcs is an endpoint of a barrier, which contains one of the arcs: this implies that clockwise and counterclockwise arcs cannot be consecutive, and every neutral arc not on ∂B is preceded by a counter-clockwise and followed by a clockwise arc. Since a clockwise arc is always followed by another clockwise arc, there cannot be two consecutive arcs with different orientations. □

It follows that every frame is either ∂B or disjoint from ∂B: if a frame γ contains parts of the boundary ∂B and portions of barriers, then it contains a segment endpoint, and so it contains arcs of both neutral and non-neutral orientation, which is impossible by Lemma 3. Only ∂B consists of arcs of neutral orientations; all other frames consist of arcs of all clockwise or counter-clockwise orientation.

Lemma 4 *Any point of a frame lies in the boundary of at most two regions of the subdivision* $\{M_1,M_2,\dots,M_k\}$.

Proof Suppose, to the contrary, that point p lies on the boundary of three regions M_a, M_b, and M_c. Since each region is the union of cells and the endpoints of the barriers are distinct, p must be the intersection of two barriers: that is, p is the endpoint of a barrier $\ell_1\in L$ and lies in the relative interior of another barrier $\ell_2\in L$ (Fig. 8). Assume that M_a and M_b lie on opposite sides of ℓ_1; and ℓ_1 and M_c are on opposite sides of ℓ_2. The orientation of the arcs along M_a and M_b are different (clockwise and counterclockwise) since p is an endpoint of ℓ_1 and it determines the orientation of the arcs along ℓ_2. This incurs two different orientations on the arcs along ℓ_2 containing p: A contradiction. □

The above lemma reveals an important feature of the graph created by the greedy algorithm. It proves that the regions are nested one in the other and that they admit a specific partial ordering: Every region is a polygon with holes, where the holes are filled by other regions and every frame is the outer boundary of exactly one region.

3.3 Total Number of Crossings

To build T, we connect the components of G by edges that we call *bridge*s. Consider a frame γ separating regions M_i and M_j. For every point $f\in\gamma$, we define a potential

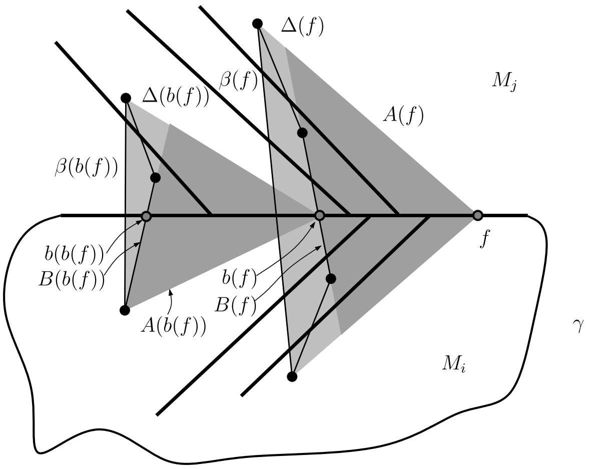

Fig. 9 Construction of $\Delta(f)$, $\beta(f)$, $A(f)$, $B(f)$ (the *two right triangles* and their associated primary edges). $\Delta(b(f))$, $A(b(f))$ and $b(b(f))$ are the *two left triangles* and the depicted point

bridge $B(f)$ as follows. Let $\Delta(f)$ be the closed triangle whose vertices are f and the two sites in the two faces adjacent to f. Let $\beta(f)$ be the edge of this triangle opposite f. We say that $\beta(f)$ is the *main* edge of $\Delta(f)$ and the other two edges are the *secondary* edges. If the interior of $\Delta(f)$ contains no sites, then let $B(f)=\beta(f)$. Otherwise, consider the convex hull of the sites in $\Delta(f)$, including the endpoints of $\beta(f)$. Traversing the boundary of this convex hull in the interior of $\Delta(f)$ from one endpoint of $\beta(f)$ to the other, one encounters sites from both S_i and S_j; choose as $B(f)$ any segment on the convex hull which connects a site in S_i to a site in S_j.

We define $A(f)$ to be the triangle whose vertices are f and the intersections of the line determined by $B(f)$ with the edges of $\Delta(f)$. We say that the edge of this triangle which is an extension of $B(f)$ is its *main* edge and the other two are its *secondary* edges (Fig. 9).

Note that the use of the convex hull ensures that the triangle $A(f)$ is contained within the triangle $\Delta(f)$ and that the secondary edges of $A(f)$ are subsegments of the secondary edges of $\Delta(f)$. The secondary edges impose strong constraints exploited in the proofs below.

Lemma 5 (a) *No barrier intersects a secondary edge except at the endpoint of a secondary edge at a frame.* (b) *Secondary edges do not intersect except at their endpoints.*

Proof (a) By construction, the interior of each secondary edge lies entirely within one (convex) face, due to the convexity of the faces. Consequently, (b) an intersection of a secondary edge with the interior of another means two different sites (endpoints of those edges) within a single face. □

Lemma 6 *No site lies in the interior of $A(f)$.*

Proof This follows from $B(f)$ lying on the boundary of the convex hull of the sites in $A(f)$. □

For each point f on the frame, choose a point $b(f)$ in the intersection of the frame with $B(f)$. Since the endpoints of $B(f)$ are in different regions, there must be at least one such point. If there are several such points, choose $b(f)$ arbitrarily. We apply this process iteratively. For example, $b^3(f)$ denotes $b(b(b(f)))$.

Lemma 7 *If $b(f) \neq f$, then triangles $A(f)$ and $A(b(f))$ are interior disjoint.*

Proof The line through segment $B(f)$ defines two halfplanes. We show that $A(f)$ and $A(b(f))$ lie on opposite sides the line through $B(f)$. Since $A(f)$ lies entirely in the halfplane containing f, it is enough to prove that both endpoints of $\beta(b(f))$ lie in the opposite halfplane. Suppose, to the contrary, that an endpoint s of $\beta(b(f))$ lies in the same halfplane as f. By Lemma 6, s lies in the exterior of triangle $A(f)$. Hence, the secondary edge $b(f)s$ crosses one of the secondary edges of $\Delta(f)$, contradicting Lemma 5. □

Lemma 8 *If $b^3(f) \neq b^2(f) \neq b(f)$, then $A(b^2(f)) \cap A(f) = \emptyset$.*

Proof By Lemma 7, $b^2(f)$ lies on the opposite side of the line determined by $B(f)$ from f. If a secondary edge of $A(b^2(f))$ crosses a secondary edge of $A(f)$, then Lemma 5 is violated. If a secondary edge of $A(b^2(f))$ crosses $B(f)$ and terminates within $A(f)$, then either Lemma 6 is violated or the associated secondary edge of $\alpha(b^2(f))$ crosses a secondary edge of $A(f)$, violating Lemma 5. If $B(b^2(f))$ is the only edge of $A(b^2(f))$ which intersects $A(f)$, then a secondary edge of $A(f)$ crosses $B(b^2(f))$ and once again either Lemma 6 is violated or the associated secondary edge of $\Delta(f)$ crosses a secondary edge of $A(b^2(f))$, violating Lemma 5. There are no other ways for the triangles to intersect. □

Lemma 9 *If a barrier intersects the boundary of $A(f)$ in two points, then one of them is f.*

Proof If not, then one of the points is in a secondary edge of $A(f)$, violating Lemma 5. □

Lemma 10 *Suppose there is no point f such that $b(f) = f$. Then there is a sequence of points on the frame $(f_0, f_1, \ldots, f_d = f_0)$, $d \geq 3$, such that $f_i = b(f_{i-1})$ for all $1 \leq i \leq d$.*

Proof Since there are only finitely many sites, the existence of such a sequence for some d is assured. By hypothesis, $d \geq 2$. But $b^2(f) = f$ would mean the entire segment $fb(f)$ would lie within both $A(f)$ and $A(b(f))$, contradicting Lemma 7. □

Lemma 11 *A total of at most $\frac{5}{3}n$ crossings suffice to construct a spanning tree of S.*

Proof Construct G. Suppose there are k regions $M_1, \ldots, M_k$. Every M_i, $i = 1, 2, \ldots, k$, has a unique frame γ_i as an outer boundary. Graph G has $n + 1$ vertices, k components, and by Lemma 1 it has n edges. We can remove $k - 1$ edges from G

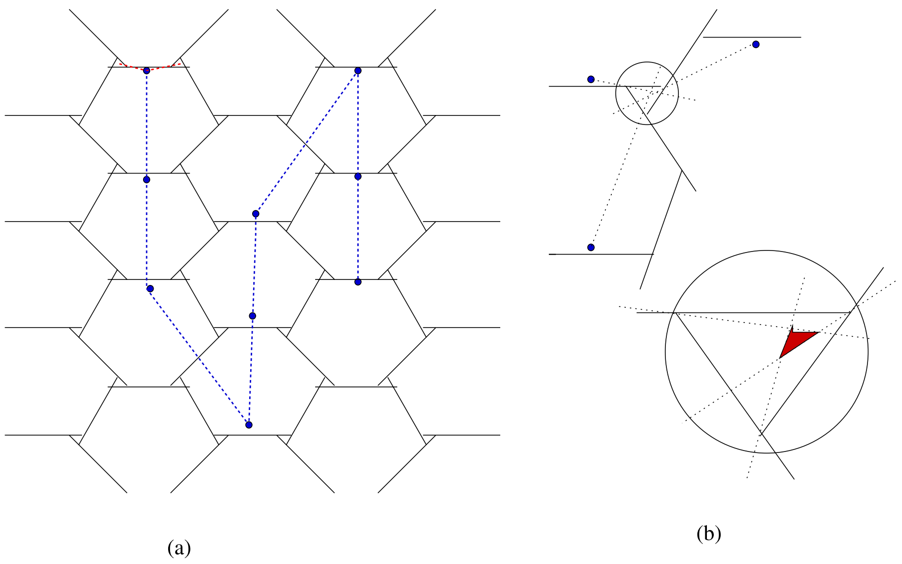

Fig. 10 Construction of the lower bound using a *honey-comb*: **a** The construction of arrangement. Only *blue* sites within hexagonal faces and *blue-blue* edges connecting two blue sites are drawn. **b** Enlargement of one section in the honey-comb. An edge incident to a *red* site in the *shaded region* of a triangular face will always cross at least two barriers. This proves the lower bound of two crossings per barrier and total of $\frac{5}{3}n$ crosses

and obtain a graph G^- with $n-k+1$ edges and the same connected components as G.

For each frame, add a bridge determined as follows. If the frame contains a point f for which $b(f)=f$, use $B(f)$. Otherwise, consider $B(f_0)$, $B(f_1)$, and $B(f_2)$ from Lemma 10. Choose the bridge that intersects the fewest barriers, say it is $B(f_i)$. Note that by Lemmas 7 and 8, the triangles $A(f_0)$, $A(f_1)$, and $A(f_2)$ have pairwise disjoint interiors.

We now label all barrier endpoints that lie on this frame to facilitate counting. If a barrier is crossed by a bridge $B(f_j)$ and its endpoint lies in the interior of triangle $A(f_j)$, then we label the endpoint *crossed*. We label all remaining endpoints of barriers *uncrossed*. By our choice of bridges, and since the interiors of the triangles $A(f_i)$ for $i=1,2,3$ are disjoint, at most $\frac{1}{3}$ of the endpoints are labeled as crossed.

For every bridge, each barrier endpoint labeled *crossed* corresponds to a crossing between the barrier and the bridge, with one possible exception. The exception is the one barrier which, under Lemma 9, has no endpoint in the interior of triangle $A(f_i)$ but may be crossed as well. Since there are k bridges, there are at most k exceptional barriers. There are at most $2n$ endpoints of barriers and therefore at most $2n/3+k$ crossings of barriers by bridges. The tree G^- creates $n-k$ crossings. The total cost is thus at most $5n/3$. □

3.4 Lower Bound for the Total Cost

Lemma 12 *There are examples where every straight line spanning tree on S has a total cost of at least $\frac{5}{3}n$.*

Proof Construct a honey-comb with n barriers as depicted in Fig. 10. Suppose T is a spanning tree of connectors. We distinguish two types of sites: *blue* sites within the hexagonal faces and *red* sites within the triangular faces. The perimeter's effect vanishes (there are $O(\sqrt{n})$ cells along the perimeter) as the example grows in size. Ignoring the perimeter effect, the number of *red* sites is $\frac{2}{3}n$ and the number of *blue* sites is $\frac{1}{3}n$. We classify the edges between the sites as *blue-blue*, *red-blue*, and *red-red*. Each blue-blue edge crosses some barrier, and each red-blue or red-red edge crosses two or more barriers. Consider the graph (a forest) R formed from just the red sites and the red-red edges. Because T is a spanning tree, each component of R is connected to some blue site with a distinct red-blue edge. Thus the combined number of red-blue and red-red edges cannot be less than the number of red sites. Thus T has $n-1$ edges, all of which cross some barrier and at least $\frac{2}{3}n$ of which cross at least two barriers. This yields a total of at least $\frac{5}{3}n-1$ crossings. □

4 Summary and Future Work

We defined a restricted version of a problem posed by Snoeyink and proved a worst-case optimal upper bound of 2 on the number of crossings per edge and an asymptotically tight upper bound of $\frac{5}{3}n$ on the total cost. The problem presented by Snoeyink remains open, with a lower bound of 3 crossings per edge and $2n-2$ total cost and an upper bound of 4 and $4n$, respectively.

The lower bound of 3 crossings per edge indicates that the techniques presented in Sect. 2 cannot be extended to the original, more general case, since our algorithm cannot build any graph with more than 2 crossings per barrier. In addition, the techniques described in Sect. 3 for the analysis of the total number of crossings does not seem to apply for the general problem, either. The greedy algorithm in Sect. 3 generates a graph with a nested structure in the restricted setting, but this property no longer holds in the general case.

Two natural extensions of our results appear to be plausible. Given a set of n pairwise disjoint barriers (segments) and a set of sites (points) such that the sites jointly "see" the entire plane (that is, for every point p in the plane there is a site s such that the open segment sp is disjoint from the barriers), is it true that there is a spanning tree on the sites that crosses every barrier at most twice and has a total cost of at most $5n/3$? We are also studying a connection between the infinite line barriers (see [2]) and pairwise disjoint segment barriers: What is the minimum total number of crossings between n line segments with $O(n^{\alpha})$ intersecting pairs (where $0 \le \alpha \le 2$) and the edges of a spanning tree over all straight line spanning trees on m points?

References

1. Asano, T., de Berg, M., Cheong, O., Guibas, L.J., Snoeyink, J., Tamaki, H.: Spanning trees crossing few barriers. Discrete Comput. Geom. **30**(4), 591–606 (2003)

2. Chazelle, B., Welzl, E.: Quasi-optimal range searching in spaces of finite VC-dimension. Discrete Comput. Geom. **4**(5), 467–489 (1989)
3. Edelsbrunner, H., Guibas, L., Stolfi, J.: Optimal point location in a monotone subdivision. SIAM J. Comput. **15**(2), 317–340 (1983)
4. Guibas, L.J., Stolfi, J.: Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. ACM Trans. Graph. **4**(2), 74–123 (1985)
5. Hoffmann, M., Tóth, Cs.D.: Connecting points in the presence of obstacles in the plane. In: Proc. 14th Canad. Conf. on Comput. Geom., pp. 63–67, 2002
6. Hoffmann, M., Tóth, Cs.D.: Spanning trees across axis-parallel segments. In: Proc. 18th Canadian Conf. on Comput. Geom., pp. 101–104, 2006
7. Kirkpatrick, D.G.: Optimal search in planar subdivisions. SIAM J. Comput. **12**(1), 28–35 (1983)
8. Krumme, D., Perkins, G., Rafalin, E., Souvaine, D.L.: Upper and lower bounds for connecting sites across barriers. TUFTS-CS Technical Report 2003-6, Tufts University, Medford, MA (2003)
9. Matoušek, J.: Spanning trees with low crossing number. RAIRO Inform. Théor. Appl. **25**(2), 103–123 (1991)
10. Matoušek, J.: Efficient partition trees. Discrete Comput. Geom. **8**, 315–334 (1992)
11. Sarnak, N., Tarjan, R.: Planar point location using persistent search trees. Commun. ACM **29**(7), 669–679 (1986)
12. Snoeyink, J.: Open problem presented at the 9th Canadian Conference on Computational Geometry, 1997
13. Snoeyink, J., van Kreveld, M.: Linear-time reconstruction of Delaunay triangulations with applications. In: Proc. 5th European Sympos. on Algorithms. Lecture Notes in Comp. Sci., vol. 1284, pp. 459–471. Springer, Berlin (1997)
14. Welzl, E.: Partition trees for triangle counting and other range searching problems. In: Proc. 4th Sympos. on Comput. Geom., pp. 23–33. ACM Press, New York (1988)
15. Welzl, E.: On spanning trees with low crossing numbers. In: Data Structures And Efficient Algorithms. Lecture Notes in Comp. Sci., vol. 594, pp. 233–249. Springer, Berlin (1992)