数理解析研究所講究録
1120 巻 1999 年 151-160

Spanning Trees Crossing Few Barriers

Tetsuo Asano
JAIST
Tatsunokuchi, Ishikawa, Japan

Mark de Berg
Utrecht University
Utrecht, the Netherlands

Otfried Cheong
HKUST
Kowloon, Hong Kong.

Leonidas J. Guibas
Stanford University
Stanford, CA 94305 USA

Jack Snoeyink
UBC
Vancouver, Canada

Hisao Tamaki
Meiji University
Tama-ku, Kawasaki, 214 Japan

Abstract We consider the problem of finding low-cost spanning trees for sets of n points in the plane, where the cost of a spanning tree is defined as the total number of intersections of tree edges with a given set of m barriers. We obtain the following results:

1. if the barriers are possibly intersecting line segments, then there is always a spanning tree of cost $O(\min(m^2, m\sqrt{n}))$;
2. if the barriers are disjoint line segments, then there is always a spanning tree of cost $O(m)$;
3. if the barriers are disjoint fat objects, discs for example, then there is always a spanning tree of cost $O(n+m)$.

All our bounds are worst-case optimal.

Key words and phrases spanning tree, barriers, minimum crossing

1 Introduction

Consider the problem of batched point location, where the goal is to efficiently locate n given points in a planar subdivision defined by m line segments. This problem arises in many applications, and in particular in the linear-time reconstruction of common geometric structures such as Voronoi and Delaunay diagrams, or convex hulls [6]. In these applications the desired diagram is constructed by adding the points in stages. In each stage, the group of points currently being added must be located among the regions of the diagram defined by all the previously inserted points. This batched point location can easily be solved by standard point location methods, or by line-sweep methods, at a logarithmic cost per point, but this would defeat the linear-time reconstruction goal. One way to avoid these logarithmic factors is to connect the points together by a structure, such as a spanning tree, that crosses the edges of the diagram only linearly many times. Then, once one of the points is located, the spanning structure can be traversed and the remaining points located as they are encountered.

The construction of such spanning trees motivated the current investigation, in which we generalize the subdivision edges to more general classes of geometric objects. Let $\mathcal{P}$ be a set of n points in the plane, which we call sites , and let $\mathcal{B}$ be a set of m geometric objects, which we call barriers. We assume that no site lies inside any of the barriers. An edge e, which is a straight line segment joining two sites, has a cost $c(e)$ that equals the number of barriers that e intersects. The cost of a spanning tree $\mathcal{T}$ for $\mathcal{P}$ is the sum of the costs of its edges:

$$c(\mathcal{T}) = \sum_{e \in \mathcal{T}} c(e).$$

(It would be more precise to speak of the cost with respect to $\mathcal{B}$, but since the barrier set will always be fixed and clear from the context, we omit this addition.) We are interested in cheap spanning trees, that is, spanning trees with small cost, for several types of barriers. We obtain the following results.

Section 2 deals with the case where the barriers are possibly intersecting line segments. Here we show that there are configurations where any spanning tree has cost $\Omega(\min(m^2, m\sqrt{n}))$. We also show how to construct a spanning tree with this cost.

Section 3 deals with various types of disjoint barriers. Here it turns out that much cheaper spanning trees can be constructed. For instance, we are able to obtain a bound of $O(n+m)$ when the barriers are fat objects—discs for example. This bound is tight in the worst case.

The major result in this paper is given in Section 4, where we prove that for any set of n sites and any set of m barriers that are (interior-)disjoint line segments, there is a spanning tree of cost $O(m)$, which is optimal in the worst case. Our proof shows in fact that a spanning tree is always possible in which no barrier segment is crossed more than four times. Such a linear cost spanning tree solves the batched point location problem we started with above (where the subdivision edges are the barriers).

All our proofs are constructive. Our construction in Section 3 indeed leads to an efficient $O((n+m)\log m)$ algorithm to produce a spanning tree of low cost. The existence proofs are more interesting, however, since a simple greedy algorithm will always construct a spanning tree of minimal cost (and for the linear-time reconstruction goal the computation of the tree happens during the preprocessing in any case).

The bounds mentioned above are significantly better then the naive $O(nm)$ bound. We close this introduction by noting that if we wish to construct a triangulation on the sites , not just a spanning tree, then the naive bound cannot be improved in the worst case. This can be seen by the example in in Fig. 1.

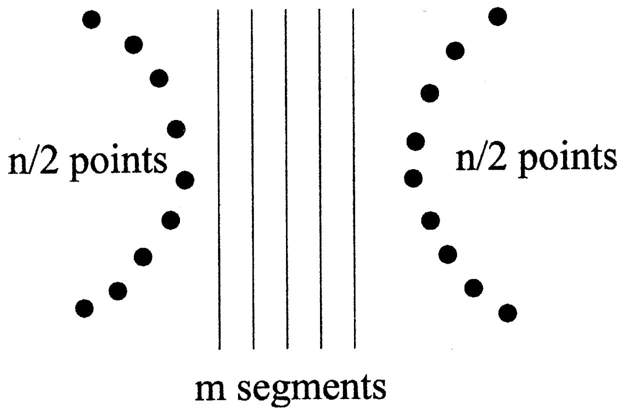

Figure 1: Any triangulation of the point set will have cost $\Omega(nm)$.

2 Intersecting segments

We start with the case where the barriers in $\mathcal{B}$ are possibly intersecting line segments.

Theorem 2.1 *(i) For any set $\mathcal{P}$ of n sites and any set $\mathcal{B}$ of m possibly intersecting segments in the plane, a spanning tree for $\mathcal{P}$ exists with a cost of* $O(\min(m^2, m\sqrt{n}))$.

(ii) For any n and m there is a set $\mathcal{P}$ of n sites and a set $\mathcal{B}$ of m segments in the plane, such that any spanning tree for $\mathcal{P}$ has a cost of $\Omega(\min(m^2, m\sqrt{n}))$.

Proof:

(i) First, extend the line segments in $\mathcal{B}$ to full lines. For each cell in the resulting arrangement, if the cell contains sites , choose a representative site and connect all sites in that cell to the representative. The edges used for this have zero cost, since the cells are convex and contain no barriers. Finally, compute a spanning tree on the set of representative sites with the property that any line intersects $O(\sqrt{n'})$ edges of the spanning tree [?], where n' is the number of representatives. The cost of the spanning tree is $O(m\sqrt{n'})$. This proves part (i) of the theorem, since $n' \leq \min(n, m^2)$.

(ii) First consider the case where $m \geq 2\sqrt{n} - 2$. We assume for simplicity that n is a square. We place the sites in a regular $\sqrt{n} \times \sqrt{n}$ grid. In between any two consecutive rows we place a bundle of $\lfloor m/(2\sqrt{n}-2) \rfloor$ horizontal barrier segments, and in between any two consecutive columns we place a bundle of $\lfloor m/(2\sqrt{n}-2) \rfloor$ vertical segments. The remaining segments are placed arbitrarily. Figure 2(a) shows the construction for the case $n = 25$ and $m = 16$. Any edge connecting two sites crosses at least one bundle. Hence, the cost of any spanning tree is at least $(n-1)\lfloor m/(2\sqrt{n}-2) \rfloor = \Omega(m\sqrt{n})$.

Now consider the case where $m < 2\sqrt{n} - 2$. We arrange the barrier segments as shown in Fig. 2(b) for the case $m = 8$: we have a group of $\lfloor m/2 \rfloor$ vertical segments and a group of $\lceil m/2 \rceil$ horizontal segments, such that any vertical segment intersects any horizontal segment. We place a site in each of the resulting "cells"; the remaining sites are placed in any cell. Any spanning tree for $\mathcal{P}$ will have cost $\Omega(m^2)$.

□

3 Disjoint uncluttered barriers

Let $\mathcal{P}$ be a set of n sites in the plane, $\mathcal{B}$ a set of m disjoint barriers. We give an algorithm that uses a binary space partition (BSP) for the set of barriers to construct a spanning tree for $\mathcal{P}$. We analyze the cost of the resulting spanning tree, assuming the BSP is orthogonal. Combining our results with known results on BSPs will then give us cheap spanning trees for so-called uncluttered scenes (defined below).

Given a BSP, our algorithm constructs a spanning tree for $\mathcal{P}$ recursively. Suppose we come to a node ν in the BSP with a set $\mathcal{P}_\nu$ of sites we wish to connect into a spanning tree; initially ν is the root of the BSP and $\mathcal{P}_\nu = \mathcal{P}$. There are three cases to consider.

(i) If $\mathcal{P}_\nu$ contains at most one site , then no spanning tree edges need to be added and we are done.

(ii) If $\mathcal{P}_\nu$ contains more than one site but ν is a leaf of the BSP, then we connect the sites into a spanning tree in an arbitrary manner.

(iii) The remaining case is where $\mathcal{P}_\nu$ contains more than one site and ν is an internal node of the BSP. Let ℓ_ν be the splitting line stored at ν. The line ℓ_ν partitions $\mathcal{P}_\nu$ into two subsets. (Points on the splitting line all go to the same subset, say the right one.) We recursively construct a spanning tree for each of these subsets by visiting the children of ν with the relevant subset. Finally, if both subsets are non-empty we connect the two spanning subtrees by adding an edge between the sites closest to ℓ_ν on either side of ℓ_ν.

We now analyze the cost of the spanning tree constructed in this manner for the special case of orthogonal BSPs. (An orthonal BSP for $\mathcal{B}$ is a BSP whose splitting lines are all horizontal or vertical.) We assume that the leaves of the BSP store at most c objects, for some constant c; thus the cells of the final subdivision are intersected by at most c objects. (Note that we cannot require $c = 0$ unless we restricted our attention to orthogonal barrier segments.)

The following result will imply the existance of spanning trees of linear cost for several classes of barriers, including orthogonal segments and convex fat objects.

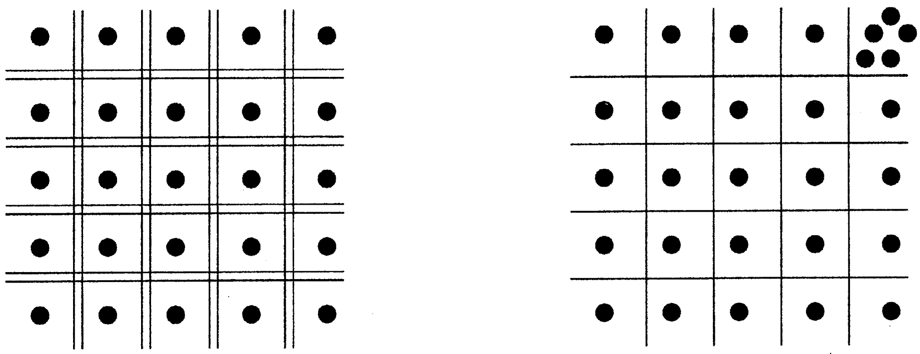

Figure 2: The lower bound constructions

Theorem 3.1 *Let $\mathcal{B}$ be a set of disjoint simply-connected barriers in the plane, and let $\mathcal{P}$ be a set of n sites in the plane. Suppose an orthogonal BSP for $\mathcal{B}$ exists that generates f fragments and whose leaf cells intersect at most c barriers. Then there is a spanning tree for $\mathcal{P}$ with cost at most $O(f+k+cn)$, where k is the total number of vertical and horizontal tangencies on barrier boundaries.*

Proof: The spanning-tree edges added in case (ii) intersect at most c barriers, so their total cost sums to at most $c(n-1)$.

Now consider an edge pq added in case (iii). Assume that the splitting line ℓ_ν is vertical. Let region(ν) denote the region corresponding to ν. Since the BSP uses only horizontal and vertical splitting lines, region(ν) is a rectangle, possibly unbounded to one or more sides. Define R_ν to be the intersection of region(ν) with the slab bounded by vertical lines through the sites p and q—see Fig. 3. Let b be a barrier intersected by pq. We will show how to charge this crossing to certain features of the barriers. These features are:

- The intersections between barrier boundaries and splitting lines. The number of these features is linear in the number of fragments f.
- Vertical and horizontal tangencies of barrier boundaries. There are k such features.

The charging of the intersection of pq with b

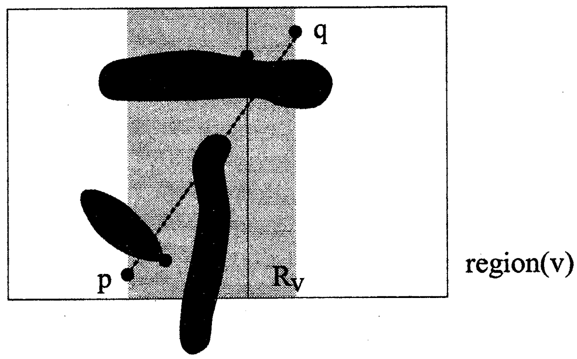

Figure 3: Illustration for the proof of Theorem 3.1.

is done as follows.

- If the boundary of b has a vertical tangent in the interior of R_ν, then we charge the intersection to this feature.
- Otherwise the boundary of b either intersects ℓ_ν in a point r lying in the interior of region(ν), or it intersects the boundary of region(ν) in a point r' that is also on the boundary of R_ν. Now we charge the intersection to r or r', respectively. Observe that both r and r' are features of b.

Fig. 3 shows for each of the three intersected barriers a feature to which the intersection can be charged. (Notice that there are actually more choices.) To bound the number of times a feature gets charged, we observe that the regions R_ν of nodes ν whose splitting line is vertical have disjoint interiors. It follows that a vertical tangency is charged at most once, and an intersection of a barrier boundary with a splitting line is charged at most twice (namely at most once for both fragments that have the point as a vertex). Similarly, a feature is charged at most twice from a node whose splitting line is horizontal. □ A

κ-cluttered scene in the plane is a set $\mathcal{B}$ of objects such that any square whose interior does not contain a bounding-box vertex of any of the the objects in $\mathcal{B}$ is intersected by at most κ objects in $\mathcal{B}$. A scene is called uncluttered if it is κ-cluttered for a (small) constant κ. It is known that any set of disjoint fat objects, discs for instance, is uncluttered—see the paper by de Berg et al. [2] for a overview of these models and the relations between them.

Theorem 3.2 *Let $\mathcal{B}$ be a set of m disjoint objects in the plane, each with a constant number of vertical and horizontal tangents, that forms a κ-cluttered scene, for a (small) constant κ. Let $\mathcal{P}$ be a set of n sites . Then there is a spanning tree for $\mathcal{P}$ with cost $O(m+n)$. This bound is tight in the worst case, even for unit discs. A spanning tree with this cost can be computed in time $O((m+n)\log m)$.*

Proof: De Berg [1] has shown that a κ-cluttered scene admits an orthogonal BSP that generates $O(m)$ fragments such that any leaf cell of the BSP is intersected by at most $O(\kappa)$ fragments. Then by Theorem 3.1 there is a spanning tree of cost $O(m+\kappa n)$.

To see that this bound is tight, take a disc as the only barrier and place the sites around the disc and so close to it that any edge connecting two sites crosses the disc. In this situation any spanning tree must have cost $\Omega(n)$. A row of m discs with two sites on either side is an example where any spanning tree must have cost $\Omega(m)$.

De Berg gives an algorithm that constructs the orthogonal BSP in time $O(m\log m)$, given only the corners of the bounding boxes of the barriers. The BSP induces a planar subdivision consisting of $O(m)$ boxes. We assign each site to the box containing it in time $O(n\log m)$ [1], and then construct the spanning tree from the leaves of the BSP upwards. Since we only need to maintain the leftmost, rightmost, topmost, and bottommost site in each node of the BSP, this can be done in time $O(n+m)$. □

Theorem 3.1 also implies that we can always find a spanning tree of cost $O(m)$ when the barriers are disjoint orthogonal segments, because Paterson and Yao [5] have shown that any set of orthogonal line segments in the plane admits an orthogonal BSP of size $O(m)$ whose leaf cells are empty. We can construct such a spanning tree in time $O((n+m)\log m)$: we need $O(m\log m)$ time to construct the BSP [3], plus $O((n+m)\log m)$ time to locate the sites in the BSP subdivision using an optimal point location structure [4], and $O(n+m)$ for the bottom-up construction of the spanning tree.

In the next section we will show that a linear-cost spanning tree exists for any set of disjoint barrier segments (even if they are not orthogonal), however, we do not know of an equally efficient way to construct the tree in the general case.

4 Disjoint segments

We now present the main result of our paper: Given any set $\mathcal{P}$ of n sites in the plane and any set $\mathcal{B}$ of m disjoint segments in the plane, there is a spanning tree for $\mathcal{P}$ whose cost is $O(m)$.

There are several ways in which we can obtain a spanning tree of cost $O(m\log(n+m))$. One possibility is to analyze a slightly adapted version of the BSP-based algorithm in terms of the depth of the underlying BSP, and use the fact that any set of m disjoint segments in the plane allows a BSP of size $O(m\log m)$ and depth $O(\log m)$ [5]. Another possibility is to use a divide-and-conquer approach based on cuttings. With neither of these two approaches we have been able to obtain a linear bound. The solution presented next therefore uses a different, incremental approach.

We assume that the segments and the sites are all *strictly* contained in a fixed bounding box, say an axis parallel unit square. We denote the upper-left and the upper-right corners of the bounding box by c_l and c_r, respectively. We will assume in the following that the sites , the endpoints of the segments of $\mathcal{B}$, and the two points c_l and c_r are in a general position collectively. This is not a serious restriction, but does make the description easier.

Let $\mathcal{T}$ be a spanning tree on $\mathcal{P} \cup \{c_l, c_r\}$ with straight edges and no self-intersections. (Note that the minimum cost spanning tree may require self-intersections. Our approach proves that self-intersections are not necessary to achieve the linear bound.) We call the path between c_l and c_r in $\mathcal{T}$ the *spine* of $\mathcal{T}$.

The spine of $\mathcal{T}$ partitions the bounding box into two parts: the part *above the spine* which is bordered by the spine and the upper edge of the bounding box, and the remaining part *below the spine*. Note that a point above the spine, in this definition, may see some edge of the spine above it since we are not assuming x-monotonicity of the spine. We say that the tree $\mathcal{T}$ is *spined* if

(1) all the sites are either on or above the spine, and
(2) both c_l and c_r are leaves of $\mathcal{T}$.

Lemma 4.1 *Let $\mathcal{P}$ be a set of sites and $\mathcal{B}$ a set of straight segments both strictly contained in the bounding box. Then there is a spined tree $\mathcal{T}$ of $\mathcal{P} \cup \{c_l, c_r\}$ such that each segment $s \in \mathcal{B}$ is stabbed by $\mathcal{T}$ at most $2 + u(s)$ times where $u(s)$ denotes the number of endpoints of s that are above the spine of $\mathcal{T}$ (and hence is at most two).*

Before proving the lemma, we show an example in Fig. 4. It consists of 5 segments and 9 sites , including the artificial sites c_l and c_r. The spine of the spined tree $\mathcal{T}$ is depicted by solid bold lines. **Proof:** The proof is by induction on the number of sites . We fix the segment set $\mathcal{B}$ throughout.

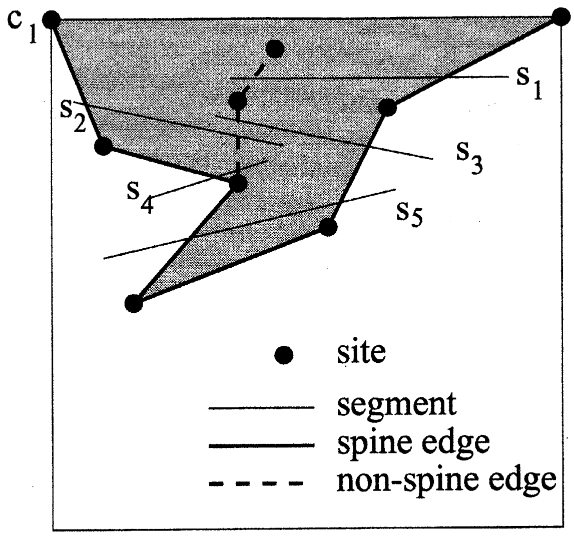

Figure 4: Spined tree

If $\mathcal{P}$ is empty, there are only two sites c_l and c_r. The edge between c_l and c_r does not stab any segment of $\mathcal{B}$, so the claim holds.

Assume now that $\mathcal{P}$ contains at least one site . Let p be the lowest site , that is, the site with the smallest y-coordinate, and let $\mathcal{P}' = \mathcal{P} \setminus p$. Let $\mathcal{T}'$ be the spined tree of $\mathcal{P}' \cup \{c_l, c_r\}$ provided by the induction hypothesis. For two sites q, r of $\mathcal{T}'$, we denote by $path(q, r)$ the path of $\mathcal{T}'$ between q and r. This notation will always be used where q and r are sites on the spine of $\mathcal{T}'$ so that $path(q, r)$ is a subpath of the spine. We will also write $lpath(q)$ for $path(c_l, q)$ and $rpath(q)$ for $path(q, c_r)$. We will abuse these notations allowing q or r to be an arbitrary point (not necessarily a site) on the spine of $\mathcal{T}'$ considered as a geometric curve.

We say that a point q in the bounding box is visible from p if the segment pq does not intersect the spine of $\mathcal{T}'$ except possibly at q. Let $Q = \{q_1, q_2, \ldots, q_t\}$ denote the set of sites of the spine of $\mathcal{T}'$ that are visible from p, listed in order from c_l to c_r. Note that $q_1 = c_l$ if c_l is visible from p and $q_t = c_r$ if c_r is.

We say that a segment $s \in \mathcal{B}$ *blocks* a site $q \in Q$ if s intersects both the segment pq and the spine of $\mathcal{T}'$. In this case, the maximal subsegment b of s lying below the spine of $\mathcal{T}'$ and being stabbed by pq is called a *blocker* of

q; we also say that s supports the blocker b and b blocks q. We call an endpoint of a blocker b that is on the spine of $\mathcal{T}'$ an anchor of b.

Suppose v is an anchor of a blocker of q. We call v a right anchor of the blocker if v is in $rpath(q)$; a left anchor if it is in $lpath(q)$. See Fig. 5 for pictorial illustration.

A blocker may have one anchor (left or right) or two anchors (both left and right). Note that, when a blocker has both left and right anchors, the left anchor may lie geometrically to the right of the right anchor, since the spine may not be x-monotone.

We say that two consecutive visible sites $q = q_i$, $r = q_{i+1}$ in Q and an edge e in $path(q, r)$ form a *good triple* (q, r, e) if the following three conditions hold (see Fig. 6).

(1) q and r do not have a common blocker.
(2) If any blocker of r has a left anchor then it is on e; if any blocker of q has a right anchor then it is also on e.
(3) If $q = c_l$ then e is incident to c_l; if $r = c_r$ then e is incident to c_r.

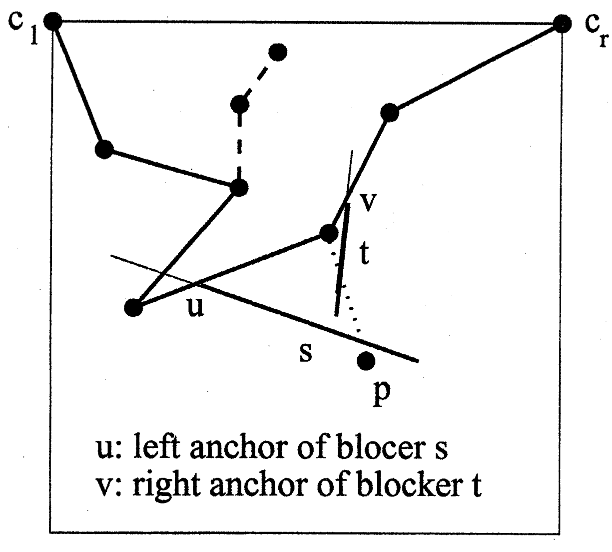

Figure 5: Blockers

Claim 4.2 *There is at least one good triple (q, r, e), $q, r \in Q, e \in path(q, r)$.*

We defer the proof of the claim and first show how we construct the spined tree of

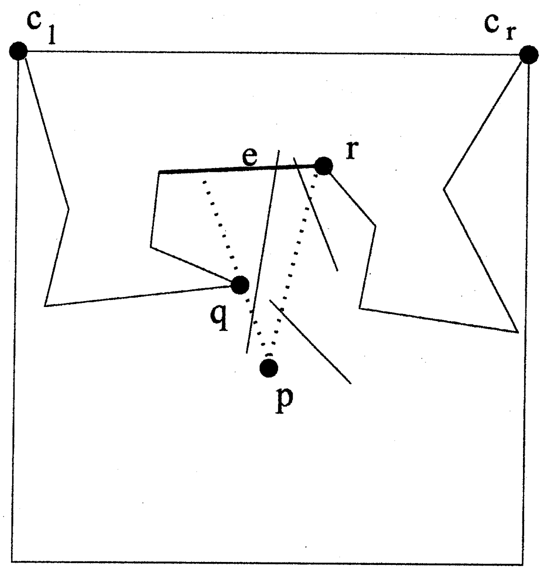

Figure 6: A good triple

$\mathcal{P} \cup \{c_l, c_r\}$ based on the claim. Let (p, q, e) be a good triple in Q. Our spined tree $\mathcal{T}$ is obtained from $\mathcal{T}'$ by adding two edges pq and pr and removing e. Since q and r are visible from p, we do not create any self-intersections, and since e is in $path(q, r)$, $\mathcal{T}$ remains a tree. The new spine goes through the edges pq and pr and it is clear that all sites are either on or above this spine. Condition (3) above guarantees that c_l and c_r remain leaves of the tree. Therefore, $\mathcal{T}$ is indeed a spined tree of $\mathcal{P} \cup \{c_r, c_r\}$.

New stabbings are created when a segment $s \in \mathcal{B}$ is stabbed by pq or pr. We consider three cases: (a) s is stabbed by both pq and pr, (b) s is stabbed by pq but not by pr, and (c) s is stabbed by pr but not by pq. Since case (c) is symmetric to case (b), we consider cases (a) and (b).

In case (a), s is not stabbed by any edge of $\mathcal{T}'$ because otherwise s would support a common blocker of q and r contradicting condition (1) of a good triple. Thus, the stabbing number of s is two without violating the induction hypothesis.

Next consider case (b): s is stabbed by pq but not by pr. Let C denote the closed curve formed by edges pq, pr and $path(q, r)$.

First suppose that s is not stabbed by $path(q,r)$. Then, one endpoint of s is in the interior of the cycle C. Since the interior of C is below the spine of $\mathcal{T}'$ and above the spine of $\mathcal{T}$, the number of endpoints of s above the spine is increased by one, accounting for the new stabbing and maintaining the induction hypothesis.

Next suppose that s is stabbed by $path(q,r)$. This means that s supports a blocker of q that has a right anchor. Condition (2) of a good triple implies that this right anchor lies in e, that is, e is the edge in $path(q,r)$ that stabs s. Since e is removed in forming $\mathcal{T}$, the induction hypothesis is maintained in this case as well. □

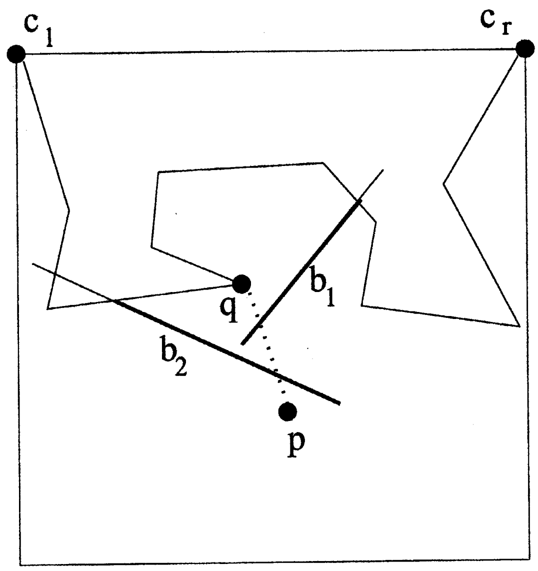

Figure 7: The partial order $b_1 \prec_q b_2$.

Before we prove the claim, we introduce some more notation. Let B denote the set of all blockers (the site $p \in \mathcal{P}$ is still fixed as the lowest site in $\mathcal{P}$).

For each blocker $b \in B$ and each site $q \in Q$ visible from p, we define a line segment $seg(q,b)$ as follows: If b blocks q, then $seg(q,b)$ denotes the line segment qq', where q' is the point at which pq stabs b. When b does not block q, we use the convention that $seg(q,b)$ is empty.

For any site $q \in Q$ visible from p, we can now introduce a partial order $\prec_q$ on B as follows: $b_1 \prec_q b_2$ if and only if b_1 and b_2 both block q and $seg(q,b_1)$ is properly contained in $seg(q,b_2)$ (see Fig. 7).

Finally, we define $\prec$ to be the transitive closure of the union of $\prec_q$ over all $q \in Q$. We claim that $\prec$ is anti-symmetric and is therefore indeed a partial order on B. To see this, consider a chain $b_1 \prec_{q_1} b_2 \prec_{q_2} \cdots$. A simple induction shows that $seg(q,b_j)$ contains $\bigcup_{1 \le i < j} seg(q,b_i)$ for every $q \in Q$ blocked by b_j. Therefore, $b_1 \prec b_j$ implies that $b_j \prec_q b_1$ does not hold for any q. Therefore, $\prec$ is anti-symmetric.

The following proposition will be used later.

Proposition 4.3 *Let b_1 and b_2 be two blockers with $b_1 \prec b_2$ and assume that b_1 blocks q. Then, the left anchor of b_2, if any, is in $lpath(q)$ and the right anchor of b_2, if any, is in $rpath(q)$.*

Proof: of Claim 4.2. If there is no blocker then the existence of a good triple is trivial. So assume that the set of blockers B is nonempty. Without loss of generality, we may assume that at least one of the blockers that is maximal with respect to $\prec$ has a right anchor (otherwise we argue symmetrically, swapping left and right). Among all the maximal blockers with a right anchor, choose the one whose right anchor is the closest to c_r in the spine of $\mathcal{T}'$ and call it b_0. Let v_0 denote the right anchor of b_0 and e the spine edge of $\mathcal{T}'$ that contains v_0. We define q (r, resp.) to be the first site visible from p when we traverse the spine of $\mathcal{T}'$ from v_0 towards c_l (c_r, resp.).

We study the three conditions of (q,r,e) being a good triple.

(1) q and r do not have a common blocker, since such a common blocker would contradict the maximality of b_0.

(2) Let q', r' be the points on $path(q, r)$ such that $path(q', r')$ is the maximal subpath of $path(q, r)$ that is visible from p. Note that p, q, q' are collinear with possibly $q = q'$ and p, r, r' are colinear with possibly $r = r'$.‘ Note therefore that if any blocker of q has a right anchor then it is on $rpath(q')$ and if any blocker of r has a left anchor then it is on $lpath(r')$.

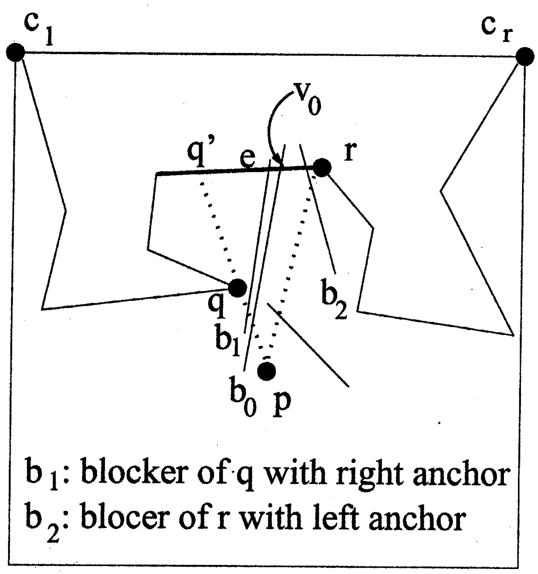

Figure 8: Case 1

Suppose r has a blocker b with left anchor v (see Fig. 8). Since b_0 is maximal, b cannot block q. Therefore, v must lie on $path(v_0, r)$. Combined with the above note, v must lie on $path(v_0, r')$. Since v_0 is on $rpath(q')$, it follows that $path(v_0, v)$ is entirely visible from p. This in turn implies that v_0 and v are on the same spine edge, namely e. This establishes the first half of condition (2). For the second half, that is the condition that the right anchor of any blocker of q lies in e, first note that the maximality of b_0, combined with the above note, implies that any such right anchor must lie on $path(q', v_0)$. Thus, if q' is on e, we are done. So suppose q' is not on e. This implies that e lies entirely within $path(r', r)$. Therefore we must have $r \neq r'$ in this case. Moreover, r does not have a blocker with a left anchor since any such anchor must lie on $path(v_0, r) \cap lpath(r')$, which is empty in this case. We conclude that condition (2) is satisfied as long as $r = r'$ or r has a blocker with a left anchor (see Fig. 9).

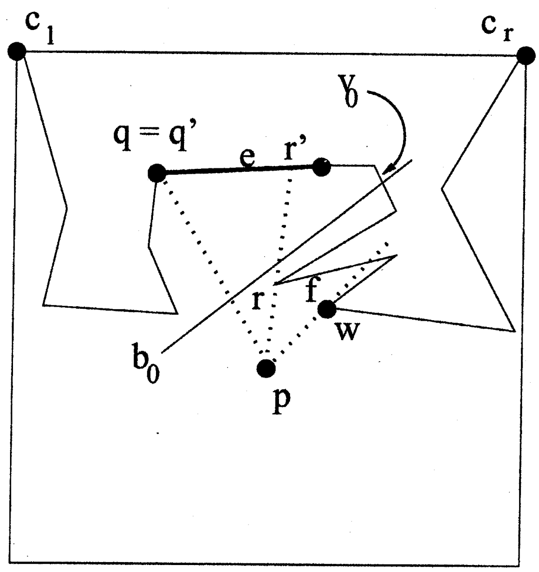

Figure 9: Case 2

(3) If $q = c_l$ then e is incident to c_l because it is impossible for the spine edge incident to c_l to be invisible from p in the neighborhood of c_l while c_l is visible. If $r = c_r$ then e is incident to c_r for an analogous reason.

To deal with the remaining case where condition (2) may not be satisfied by the triple (q, r, e), suppose that $r \neq r'$ and that r does not have a blocker with a left anchor. In this case, the last edge of $path(v_0, r)$ is invisible from p so that the first edge f of $rpath(r)$ is visible from p at least locally at r. Let w be the next visible site in the spine of $\mathcal{T}'$ after r. We claim that (r, w, f) is a good triple (see Fig. 9).

We first argue that r does not have a blocker at all. Suppose r did have a blocker b. Let b^* be the maximal blocker such that $b \prec b^*$. If b^* had a right anchor, then by Proposition 4.3 this anchor would have to be in $rpath(r)$, contradicting the choice of b_0. Therefore, b^* must have a left anchor u. By Proposition 4.3 again, u must lie on $lpath(r)$. However, u cannot lie on $lpath(v_0)$ because this would contradict the maximality of b_0. Therefore, u must lie on $path(v_0, r)$, but this implies that b^* is a blocker of r with a left anchor, contradicting our assumption. Therefore, r does not have a blocker at all. This trivially implies that r and w do not have a common blocker, condition (1) of (r, w, f) being a good triple. For condition (2) we need only consider blockers of w with left anchors. Since f is the only edge in $path(r, w)$ that has a part visible from p and f is incident to r, such left anchors must all lie on f. Finally, a reasoning analogous to the one for (q, r, e) shows that the triple (r, w, f) satisfies condition (3). □

Now we come to our main theorem.

Theorem 4.4 *Given a set $\mathcal{B}$ of non-intersecting line segments and a set $\mathcal{P}$ of sites in the plane, there is always a straight-edge spanning tree of $\mathcal{P}$ that stabs each line segment of $\mathcal{B}$ at most 4 times.*

Proof: We first compute a bounding box that properly contains all the objects of $\mathcal{B}$ and $\mathcal{P}$. Let c_l and c_r be the upper-left and upper-right corners of the bounding box, respectively. Then, applying the lemma to $\mathcal{B}$ and $\mathcal{P} \cup \{c_l, c_r\}$, we obtain a spined tree $\mathcal{T}$. Removing the artificial sites c_l and c_r from $\mathcal{T}$, it remains a tree since c_l and c_r are leaves of $\mathcal{T}$. It follows from the statement of the lemma that the resultant tree is a spanning tree over $\mathcal{P}$ that stabs each line segment of $\mathcal{B}$ at most 4 times. □

5 Conclusions

In this paper we have studied spanning trees among n points whose edges cross few among a given set of m barriers. When the barriers are disjoint, near-linear bounds for the cost of such a tree can be obtained by several simple arguments. Using more sophisticated techniques, we were able to show that a linear cost spanning tree is possible in many cases.

One of the techniques we used is the construction of certain BSPs on the barriers. This raises the question of whether our other methods, by 'reverse-engineering,' can help resolve some open questions about BSPs, including the long-standing one about the existence of a linear space BSP for disjoint line segments in the plane.

We note that the number of barriers crossed by linking two points is not a distance function and does not satisfy the triangle inequality. This means that the existence of other low-cost structures among the points, such as Hamiltonian tours and matchings, remains an interesting research problem.

References

[1] M. de Berg. Linear size binary space partitions for uncluttered scenes. Technical Report UU-CS-1998-12, Utrecht University, 1998

[2] M. de Berg, M. J. Katz, A. F. van der Stappen, and J. Vleugels. Realistic input models for geometric algorithms. In *Proc. 13th Annu. ACM Sympos. Comput. Geom.*, 294-303, 1997

[3] F. d'Amore and P. G. Franciosa. On the optimal binary plane partition for sets of isothetic rectangles. *Inform. Process. Lett.*, 44:255-259, 1992

[4] H. Edelsbrunner, L. J. Guibas, and J. Stolfi. Optimal point location in a monotone subdivision. *SIAM J. Comput.*, 15(2):317-340, 1986

[5] M. S. Paterson and F. F. Yao. Efficient binary space partitions for hidden-surface removal and solid modeling. *Discrete Comput. Geom.*, 5:485-503, 1990

[6] J. Snoeyink and M. van Kreveld. Linnear-time reconstruction of Delaunay triangulations with applications. In *Proc. Annu. European Sympos. Algorithms*, Lecture Notes in Computer Science 1284, 459-471, Springer-Verlag, 1997.