Growing a Tree from its Branches*

Prosenjit Bose and Godfried Toussaint

School of Computer Science
McGill University
3480 University Street
Montréal, Québec, Canada
H3A 2A7

Abstract

Given a set L of n disjoint line segments in the plane, we show that it is always possible to form a spanning tree of the endpoints of the segments, such that each line segment is an edge of the tree and the tree has no crossing edges. Such a tree is known as an *encompassing* tree and can be constructed in $O(n \log n)$ time when no three endpoints in L are collinear. In the presence of collinear endpoints, we show firstly that an encompassing tree with no crossing edges exists and can be computed in $O(n^2)$ time, and secondly that the maximum degree of a node in the minimum weight spanning tree formed by these line segments is seven, and there exists a set of line segments achieving this bound. Finally, we show that the complexity of finding the minimum weight spanning tree is optimal $\Theta(n \log n)$ when we assume that the endpoints of the line segments are in general position.

1 Introduction

Given a set of disjoint line segments, determining whether the set admits certain combinatorial structures has received considerable attention. One of the best-studied such structures has been the simple circuit or polygon through a set of line segments. The question of deciding whether a set of disjoint line segments admits a simple circuit is conjectured to be NP-complete since Rappaport [11] has shown that deciding whether a set of line segments (which are allowed to intersect at their endpoints) admits a simple circuit is an NP-complete problem. For certain special cases, however, polynomial-time algorithms have been obtained. Avis and Rappaport [1] gave an $O(n^4)$ time and $O(n^2)$ space algorithm to decide whether a set of disjoint line segments admits a simple monotone circuit. Rappaport, Imai, and Toussaint [12] have shown that the decision problem is $O(n \log n)$ when every line segment in the set has at least one endpoint on their convex hull (such a configuration is known as a *convexly independent* set of line segments). Although not every convexly independent set of line segments admits a simple circuit, Mirzaian [9] has shown that such a set always admits a simple polygon such that the line segments are

*Research supported in part by an NSERC scholarship, and by grants NSERC-A9293 and FCAR-EQ1678

either part of the boundary of the polygon or form internal diagonals. Mirzaian's result does not hold for arbitrary sets of disjoint line segments, as was shown by Urabe and Watanabe [16], but it is conjectured that the result is true if the line segments are afforded the additional freedom of forming external diagonals of the polygon. O'Rourke and Rippel [6] proved the hamiltonicity of the visibility graph of certain restricted classes of line segments.

Surprisingly, the problem of determining whether a set of line segments admits a spanning tree on the endpoints such that each line segment is an edge of the tree and the tree has no crossing edges has not been studied at all. This notion leads to several intriguing questions. Does a set of line segments always admit such a tree? Are there efficient algorithms for finding such a tree if one exists? Is there a bound on the maximum degree of such a tree? We answer all these questions in the affirmative.

2 Notation and Definitions

Let us first introduce some terminology. Most of the geometric and graph theoretic terminology used is standard and for details, we refer the reader to O'Rourke [7], Bondy and Murty [2], and Preparata and Shamos [8]. We begin by reviewing some of the main geometric and graph theoretic terminology used in this paper.

Let E^2 denote the Euclidean space of dimension two. A domain D in E^2 is *convex* if for every two points q_1 and q_2 in D, the segment $\overline{q_1 q_2}$ is entirely contained in D. The *Convex Hull* of a set of points S in E^2 is the boundary of the minimum area convex domain in E^2 containing S.

Two line segments in the plane are *disjoint* if they do not intersect. A set of line segments is disjoint if every pair of line segments in the set is disjoint. Given a set L of disjoint line segments in the plane, we say that two points in the plane are *visible* with respect to L, if the open line segment determined by the two points does not intersect any line segment in L.

Given a set L of disjoint line segments, we define the *visibility* graph of L, denoted as $V(L)$, as follows. The endpoints of the line segments in L correspond exactly to the vertices of $V(L)$. If two endpoints are visible with respect L, then their corresponding vertices are adjacent in $V(L)$. The *augmented visibility* graph of the set L, denoted $AV(L)$, is the same as $V(L)$, with the addition that every line segment in L has a corresponding edge in $AV(L)$. Thus, $AV(L)$ contains two types of edges: *line segment edges*, and *visibility edges* (refer to Figure 1).

3 Existence of Spanning Trees

We begin by showing that every set of disjoint line segments admits a spanning tree, such that every line segment is a tree edge and the tree has no crossing edges. By this, we mean that the augmented visibility graph of the set of line segments contains a spanning tree that has no crossing edges and includes every edge corresponding to a line segment in L. A spanning tree that has this property is said to *encompass* the set L. We assume in this section that no three endpoints of line segments in the set L are collinear.

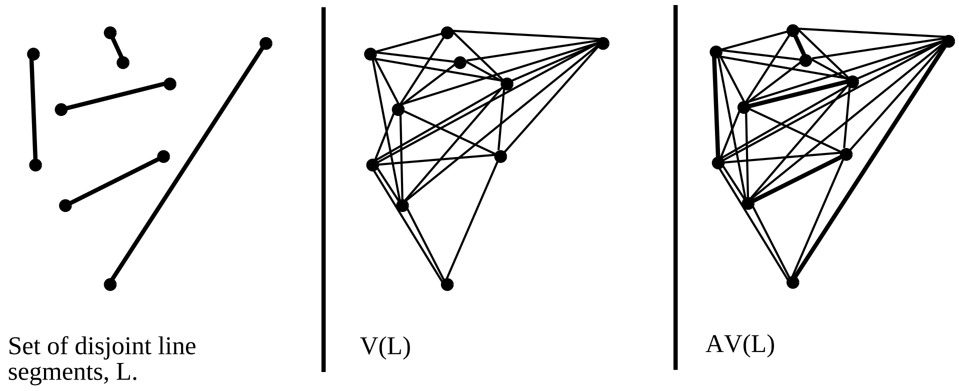


Figure 1: A set of line segments and corresponding visibility graphs.

First, we show that the augmented visibility graph of a set of disjoint line segments L is connected. Since a triangulation of L is a subgraph of $AV(L)$ and a triangulation is connected, this result follows from [3]. However, we include an alternate direct proof here for completeness.

Lemma 3.1 *The augmented visibility graph of a set of disjoint line segments is connected.*

Proof: By induction on the number of line segments (refer to Figure 2).

Basis: $n = 1$. When there is only one line segment, the graph is trivially connected.

Inductive Hypothesis: Given a set of $n \leq k$, $k \geq 1$, disjoint line segments, the augmented visibility graph is connected.

Inductive Step: $n = k + 1$. Pick a line segment s which has at least one endpoint e_1 on the convex hull. By removing the segment s from the set L, we have a smaller set L'. By the inductive hypothesis, $AV(L')$ is connected. By inserting s into the set L', the graph might become disconnected because some visibility edges may no longer exist. Consider an arbitrary visibility edge, $\overline{ab}$, in $AV(L')$ that is not in $AV(L)$. We will show that there exists a path from a to b in $AV(L)$. Therefore, $AV(L)$ must be connected, since there exists a path from the two endpoints of any visibility edge deleted from $AV(L')$ by the insertion of s, and s itself is connected by at least one endpoint on the convex hull.

The line segment $\overline{ab}$ must intersect s at some point x. Consider the convex hull of the endpoints inside the triangle(a, x, e_1), excluding x. Either the convex hull is a single edge from a to e_1 when the triangle is empty, or there is a path from a to e_1 when the triangle contains at least one endpoint. Similarly, either there is a single edge from b to e_1, or there is a path from b to e_1. ■

Lemma 3.2 *[2] Every connected graph contains a spanning tree.*

What remains to be shown is that this connected graph contains the appropriate spanning tree. Let us modify $AV(L)$, by giving weights to the edges in the graph. Each edge representing

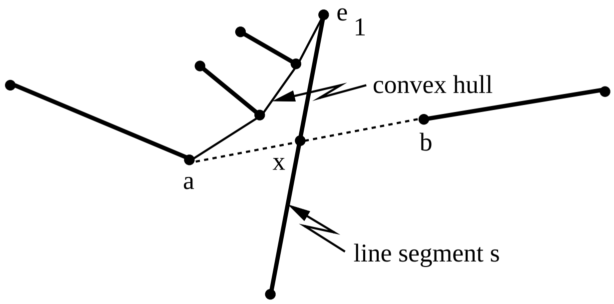


Figure 2: Illustration for proof of Lemma 3.1.

a line segment is assigned weight 0 and every other edge is assigned weight 1. In $AV(L)$ the set of edges representing the line segments do not form any cycles since the line segments are disjoint by definition. Therefore any minimum weight spanning tree of the modified $AV(L)$ will include all the line segment edges and the existence of the desired spanning tree is guaranteed. We have therefore established the following result.

Lemma 3.3 *Every set of disjoint line segments admits an encompassing spanning tree.*

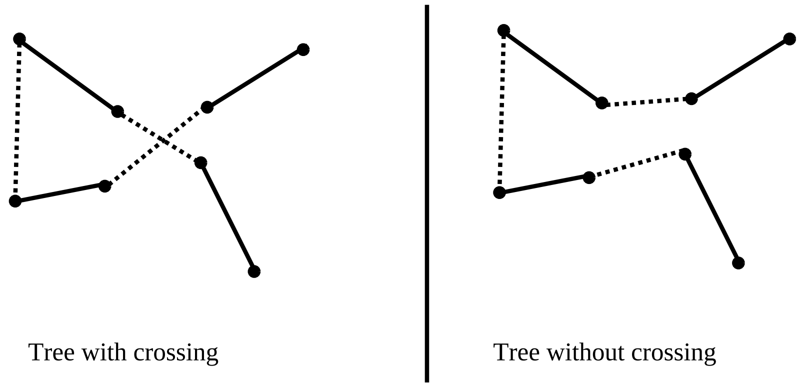


Figure 3: Spanning tree may contain intersections.

The spanning tree that is found in the augmented visibility graph, can be embedded directly on the set of line segments from which the graph arises. The question then becomes: can the spanning tree always be embedded without intersection? For example, the tree illustrated in Figure 3 has an intersection, but an alternate tree can be found.

Lemma 3.4 *[3] The augmented visibility graph of a set of disjoint line segments has a triangulated planar subgraph containing all the line segments as edges.*

Proof: A set of points can always be triangulated [8]. Therefore, we can view triangulating a set of line segments as triangulating the endpoints and then subsequently adding the line segments and re-adjusting the triangulation. We now prove that this can always be done by induction on the number of given line segments that must be present in a triangulation.

Basis: $n = 0$. Given a set of points P with no prescribed line segments, the set can be triangulated.

Inductive Hypothesis: Given a set of points P and a set L of $n \leq k$, $k \geq 0$, disjoint line segments whose endpoints come from the set P, the set admits a triangulation that includes those segments.

Inductive Step: $n = k + 1$. We are given a set of points P and a set of disjoint line segments L of size n, defined on P. Pick a line segment s. By removing the segment s from the set L, we obtain a smaller set L'. By the inductive hypothesis, L' admits a triangulation $T(L')$. See Figure 4 for an illustration of the following argument. We show that L can be triangulated by inserting s into $T(L')$ and re-triangulating.

The endpoints of s are already in $T(L')$, since we triangulated the set of points P. The segment s can only intersect visibility edges since the set L is a set of disjoint line segments. By removing all the intersected edges, we are left with a planar subdivision of the interior of the convex hull of P consisting of triangles and at most two regions R_1 and R_2 on either side of s. The regions R_1 and R_2 are simple and weakly-visible from s and can therefore be triangulated by the algorithm presented in [15]. A polygon P is weakly-visible from an edge e if $\forall x \in P, \exists y \in e$ such that the line segment $\overline{xy}$ is in P.

■

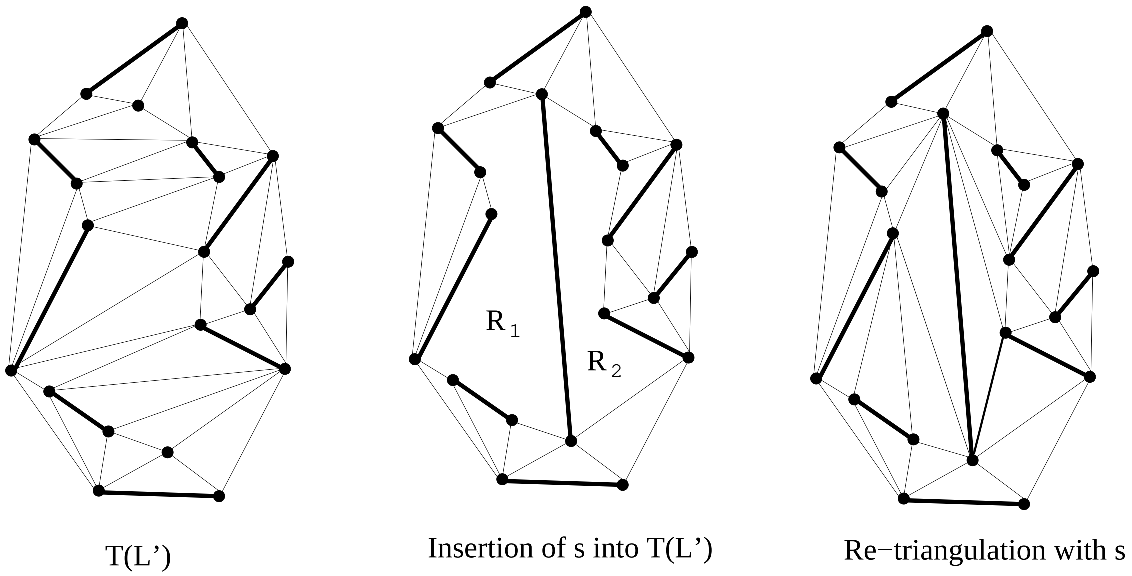


Figure 4: Illustration for proof of Lemma 3.4.

A triangulated graph is a planar graph where every face is a triangle except possibly the outer face. In [4], it is shown how to triangulate a set of disjoint line segments in $O(n \log n)$ time. By assigning a weight 0 to each line segment edge, and a weight 1 to every other edge of the triangulation, we see that any minimum weight spanning tree will be an intersection-free encompassing spanning tree. Since there are a linear number of edges in a planar graph, the minimum weight spanning tree can be constructed in $o(n \log n)$ time using standard algorithms ([8], [17]). Therefore we have the following theorem.

Theorem 3.1 *Every set of disjoint line segments admits an encompassing spanning tree with no intersecting edges, and such a tree can be constructed from the set of line segments in $O(n \log n)$ time.*

4 Handling Collinearities

Until now we have assumed that no three endpoints in a set of line segments lie on a line. In this section, we show that an encompassing spanning tree with no intersecting edges exists even in the presence of collinear endpoints. We will assume, in this section, that a disjoint set of line segments has collinear endpoints but that the convex hull of the set is not a line segment. When the convex hull is a line segment, the solution is trivial.

Lemma 4.1 *A set of points P with collinear points whose convex hull is not a line segment admits a triangulation.*

Proof: By induction on the number of points.

Basis: $n = 3$. Three points whose convex hull is not a line segment admit a triangulation.

Inductive Hypothesis: A set of $n \leq k$, $k \geq 3$ points P with collinearities and whose convex hull is not a line segment admit a triangulation.

Inductive Step: $n = k + 1$. Remove a point $p \in P$ such that the convex hull of the smaller set P' is not a line segment. Such a point must exist since P has at least 4 points. By the inductive hypothesis, P' admits a triangulation $T(P')$. Now, we insert p into $T(P')$. If p is outside $T(P')$, the set can be triangulated by inserting edges from p to all visible vertices of the convex hull of P. If p is in a triangle of $T(P')$, the set can be triangulated by connecting p to the three vertices of the triangle. Finally, if p is on an edge e of a triangle of $T(P')$, the set can be triangulated by connecting p to the apex of each of the two possible triangles that have e as a base. ■

Lemma 4.2 *The augmented visibility graph of a set of disjoint line segments with collinear endpoints has a triangulated planar subgraph containing all the line segments as edges.*

Proof: A set of points with collinearities can be triangulated by Lemma 4.1. Similar to the proof of Lemma 3.4, we show that line segments can be added to a triangulation of the points and the triangulation re-adjusted.

Basis: $n = 0$. Given a set of points P with no prescribed line segments, the set can be triangulated by Lemma 4.1.

Inductive Hypothesis: Given a set of points P and a set L of $n \leq k$, $k \geq 0$, disjoint line segments whose endpoints come from the set P, the set admits a triangulation that includes those segments.

Inductive Step: $n = k + 1$. We are given a set of points P and a set of disjoint line segments L of size n, defined on P. Pick a line segment s. By removing the segment s from the set L, we have a smaller set L'. By the inductive hypothesis, L' admits a triangulation $T(L')$. We show that L can be triangulated by inserting s into $T(L')$ and re-triangulating.

The endpoints of s are already in $T(L')$, since we triangulated the set of points P. The segment s can only intersect visibility edges since the set L is a set of disjoint line segments. By removing all the intersected edges, we are left with a planar subdivision of the interior of the convex hull of P consisting of triangles and at most two regions R_1 and R_2 on either side of s. The regions R_1 and R_2 are simple and weakly-visible from s. The regions can be triangulated by the algorithm presented below which is a modification of the algorithm TR-POL in [15]. ∎

Given three points a, b, c in the plane, (a, b, c) is a left turn if c lies to the left of ray(a, b), a right turn if c lies to the right of ray(a, b), a zero turn if c lies on ray(a, b), past b, and a 180 turn if c lies on ray(a, b), before b. We use the following notation in the algorithm. q_t, q_{t-1} represent the stack top and just under top, respectively. x is the current polygon vertex under consideration. $x = x + 1$ increments current vertex to next vertex in counter-clockwise order. The input to the algorithm is a simple polygon with collinearities that is weakly-visible from an edge uv. Let the counter-clockwise order of the vertices of the polygon be $v, p_0, p_1, \ldots, p_{n-2}, u$.

Algorithm TR-COL

Let T be the list of edges of the triangulation. Initialize $T = P$.
Let S be a stack. Initially, S is empty.
Push v on stack
Push p_0 on stack
Initialize $x = p_1$.
While $x \neq v$ do {
    If (q_{t-1}, q_t, x) is a right turn or zero turn {
        push x
        $x = x + 1$
    } Else if (q_{t-1}, q_t, x) is a left turn {
            If $(q_{t-1}, x, x + 1)$ is a 180 turn {
                Add diagonal $(q_t, x + 1)$
                $x = x + 1$
            }
            Else {
                Add diagonal (q_{t-1}, x)
                Pop stack
            }

}
If $q_t = v$ {
push x
$x = x + 1.$
}
}

The proof of correctness of the algorithm TR-COL is almost identical to that of TR-POL found in [15] and is quite lengthy. Therefore, we only discuss the modifications needed for the introduction of the zero and the 180 turns. The reader is encouraged to refer to [15] for the rest of the proof which will not be duplicated here.

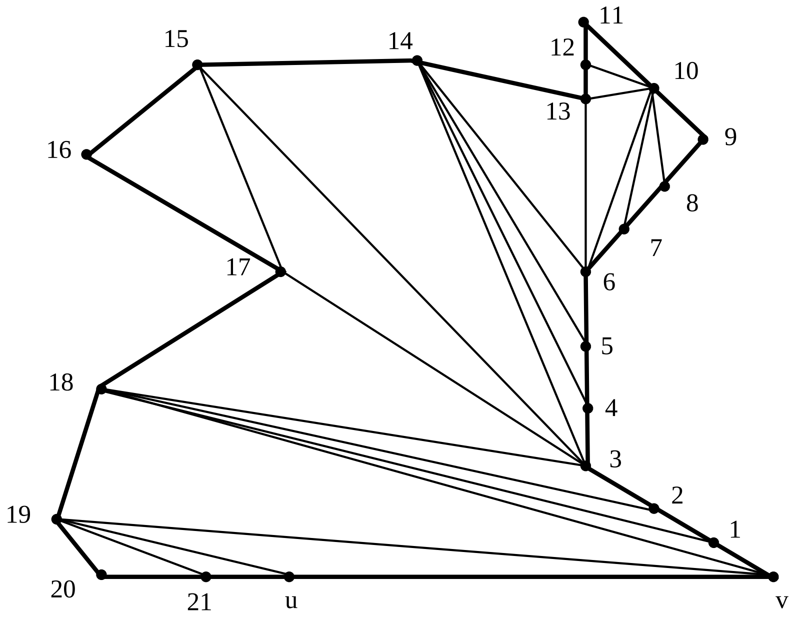


Figure 5: Polygon weakly visible from uv triangulated with TR-COL.

The zero turns do not introduce concavities in the stack, and can be treated as degenerate right turns in TR-POL. Therefore, let us consider the 180 turns. Such turns are degenerate left turns. Consider a left turn in TR-POL, such as (q_{t-1}, q_t, x) (or (6,10,11) in Figure 5). With no collinearities, we may add diagonal (q_{t-1}, x) and pop the stack. With collinearities allowed, such a diagonal may lie on top of other edges not yet scanned (such as edge (11,12) and edge (12,13) in the figure). However, due to the fact that P is weakly-visible from edge uv, the triangle(q_{t-1}, q_t, x) must be empty (triangle(6,10,11) in the figure). Therefore, any vertex, say $x + 1$ (or 12 and 13 in the figure) giving a 180 turn from q_{t-1}, is visible from q_t. The algorithm therefore upon encountering a 180 turn does not add diagonal (q_{t-1}, x), but diagonal $(q_t, x + 1)$ instead.

A set of line segments in the presence of collinearities can be triangulated in $O(n^2)$ time where n is the number of line segments. The algorithm follows from the proof of Lemma 4.2

and the previous discussion. Therefore, we have the following theorem:

Theorem 4.1 *Every set of disjoint line segments with collinearities admits an encompassing spanning tree with no intersecting edges, and such a tree can be constructed from the set of line segments in $O(n^2)$ time.*

5 Bounded Degree Spanning Trees

In this section, we investigate properties of the minimum weight spanning tree of a set of disjoint line segments. Let us first define the assignment of weights. In the augmented visibility graph of a set of disjoint line segments, a weight of zero is given to every edge corresponding to a line segment, and the euclidean distance between the endpoints represents the weight of every visibility edge. Therefore the minimum weight spanning tree of a set of disjoint line segments is an encompassing spanning tree. For this section, we will assume that the endpoints of the line segments may be collinear.

Since we are considering the whole augmented visibility graph when determining the minimum weight spanning tree as opposed to an intersection-free subgraph (triangulation) as was done in the previous section, it is no longer clear whether the minimum weight spanning tree is intersection-free.

Lemma 5.1 *The minimum weight spanning tree of a set of disjoint line segments is intersection-free.*

Proof: (refer to Figure 6) Suppose that the minimum weight spanning tree, T, contained an intersecting pair. The intersecting pair must consist of two visibility edges since the set of line segments is disjoint. Let $\overline{ab}$ and $\overline{cd}$ be the pair intersecting at a point x.

First, we show that there exists a vertex disjoint path between one of the pairs (a, d), (b, d), (b, c), or (a, c) which uses neither of the two edges $\overline{ab}$ or $\overline{cd}$ in T. We will refer to such paths as *valid* vertex disjoint paths. Since T is a tree, there must be a path from b to c. Suppose the path uses neither of the edges $\overline{ab}$ or $\overline{cd}$, then it is a valid vertex disjoint path from b to c and we are done.

Suppose then that the path from b to c uses both the edges $\overline{ab}$ and $\overline{cd}$. This path would start from the edge $\overline{ba}$; then, it would follow a path from a to d; and finally use the edge $\overline{dc}$. Thus, if the path from b to c uses both edges, there is a valid vertex disjoint path from a to d.

Suppose now that the path from b to c contains the edge $\overline{ab}$ but not $\overline{cd}$. This path would go from b to a and then follow a path from a to c implying that the path from a to c is a valid vertex disjoint path.

Similarly, suppose that the path from b to c uses the edge $\overline{cd}$ but not $\overline{ab}$, then there is a valid vertex disjoint path from b to d.

Thus, one of the pairs of vertices will have a vertex disjoint path in the tree T, which uses neither of the two edges $\overline{ab}$ or $\overline{cd}$. Assume, without loss of generality, that this pair is (a, d).

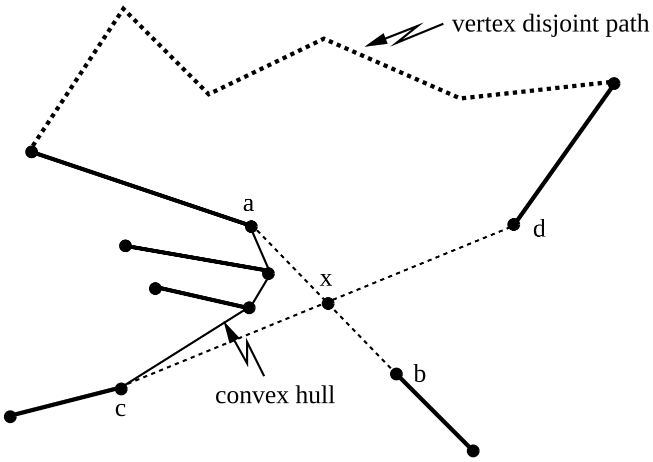


Figure 6: Illustration for proof of Lemma 5.1.

The removal of the edges $\overline{ab}$ and $\overline{cd}$ from the tree T decomposes the tree into three trees. One tree T_1 contains the vertices a and d, since there is a valid vertex disjoint path between them. One tree T_2 contains the vertex b and one tree T_3 contains the vertex c.

Let S denote the set of endpoints (of line segments) inside the triangle(a, c, x), including a and c. Consider the convex hull of S. Aside from a and c, let the other convex hull vertices be denoted by $c_1, c_2, \ldots, c_k$, where k is two less than the total number of convex hull vertices. If $k = 0$, then the convex hull simply consists of the edge $\overline{ac}$. In this case, let Π_s denote the edge. Otherwise, the convex hull consists of the edge $\overline{ac}$ and a path $\Pi_s = c, \overline{cc_1}, c_1, \ldots, \overline{c_k a}$. Observe that the size of the perimeter of triangle (a, c, x) must be greater than the size of the perimeter of the convex hull since both are convex sets enclosing the set S and the convex hull of S is the smallest convex set enclosing the set S. Therefore, since $\overline{ac}$ is common to both convex sets, $\|\Pi_s\| < \|\overline{ax}\| + \|\overline{xc}\|$.

Similarly, for the set R of endpoints (of line segments) inside triangle (d, b, x), including b and d, we have $\|\Pi_r\| < \|\overline{dx}\| + \|\overline{xb}\|$.

Since Π_s connects a with c and Π_r connects b with d, we have that $G' = T - ab - cd + \Pi_s + \Pi_r$ is connected. Thus, G' has a minimum weight spanning tree, say T'. But $weight(T') \leq weight(G') < weight(T)$. Contradiction. ∎

The question now is whether we can bound the maximum degree of a node in the minimum weight spanning tree of a set of disjoint line segments. It is not unreasonable to believe that the maximum degree can be bounded since we know the following.

Lemma 5.2 *[10] Every finite set of points S in the plane admits a minimum weight spanning tree whose maximum node degree is five.*

In a finite set of points, every point can see every other point, which implies a quadratic number of edges in the visibility graph of the set of points, but in a disjoint set of line segments, there are some points which do not see each other. In fact, it is shown in [13] that $5n - 4$ is a tight lower bound on the number of edges in the augmented visibility graph of a set of n disjoint line segments. Although there may be only a linear number of edges in the augmented visibility graph, we establish the following lemma.

Lemma 5.3 *The minimum weight spanning tree of a set of disjoint line segments has maximum degree seven.*

Proof: Suppose that the minimum weight spanning tree T had a vertex v with $d(v) \geq 8$; then vertex v must be adjacent to one line segment edge and at least seven visibility edges. Thus, there must be two adjacent visibility edges such that the angle between them is less than $\frac{\Pi}{3}$. Let $\overline{va}$ and $\overline{vb}$ be two such edges and refer to Figure 7. Without loss of generality, assume that $\|\overline{vb}\| \geq \|\overline{va}\|$. Let α be the angle between the two edges.

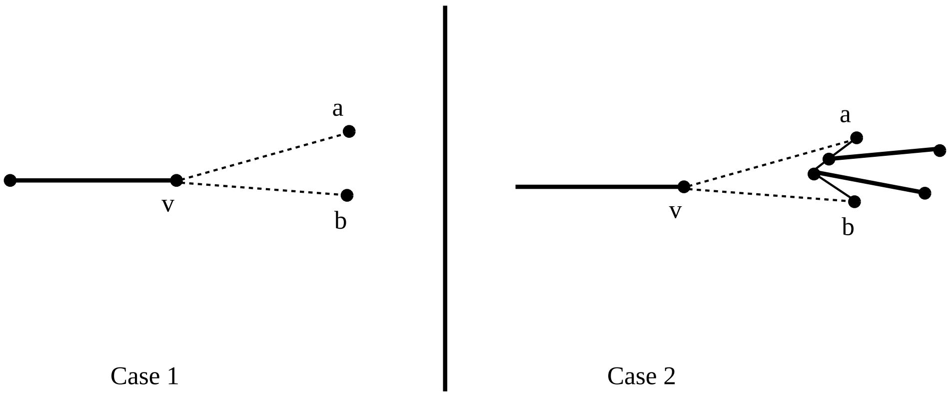


Figure 7: Illustration for proof of Lemma 5.3.

Case 1: Suppose a sees b. Since $\alpha < \frac{\Pi}{3}$, the edge $\overline{vb} > \overline{ab}$. Thus, the tree $T' = T - \overline{vb} + \overline{ab}$ has lower total weight, contradicting the fact that T has minimum weight.

Case 2: Suppose a does not see b. There must be at least one line segment blocking visibility. That line segment must therefore intersect $\overline{ab}$. Furthermore, since $\overline{va}$ and $\overline{vb}$ are visibility edges, one endpoint of the line segment must lie in the triangle(a, v, b). Consider the convex hull of all the endpoints inside the triangle(a, b, v) including a and b. Aside from a and b, let the other convex hull vertices be denoted by $c_1, c_2, \ldots, c_k$, where k is two less than the total number of convex hull vertices. Since there is at least one endpoint in the triangle, $k \geq 1$. The convex hull consists of the edge $\overline{ab}$ and a path $\Pi = \overline{bc_1}, c_1, \overline{c_1c_2}, c_2, \ldots, \overline{c_ka}$.

Removing the edge $\overline{vb}$ from T, disconnects the tree into two components. One of the components contains b and the other contains a, since there is an edge between v and a. Therefore, one of the convex hull edges must connect the two components, since Π connects a to b. This edge, let us denote it by e_c, is smaller than $\overline{vb}$ since it is wholly contained in triangle(a, b, v), and $\overline{vb}$ is the triangle's longest side due to the fact that $\alpha < \frac{\pi}{3}$. Thus, the tree $T' = T - \overline{vb} + e_c$ has lower total weight, contradicting the fact that T is minimum weight.

■

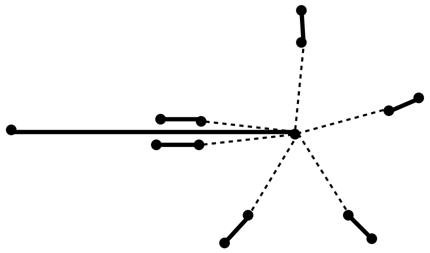


Figure 8: Minimum encompassing tree with degree 7.

The bound on the maximum degree in the minimum weight spanning tree is tight because the set of line segments in Figure 5 exhibits a minimum encompassing spanning tree with maximum degree seven. The angle between every pair of visibility edges adjacent to the vertex of degree seven is greater than $\frac{\pi}{3}$ except for the pair whose visibility is blocked by the line segment. We have therefore proved the following theorem.

Theorem 5.1 *The minimum weight spanning tree of a set of line segments contains no intersecting edges and has maximum degree seven. There exists a set of line segments achieving the maximum degree.*

6 Algorithms

In this section, we present an $O(n \log n)$ time algorithm to find the minimum weight spanning tree of a set of disjoint line segments. The algorithm is based on the constrained delaunay triangulation of the set of line segments and therefore we assume that the endpoints are in general position (no three are collinear and no four are co-circular).

Let the euclidean distance between two points p and q be denoted by $d(p,q)$. The $lune(p,q)$ of two points p and q is the interior of the of the intersection of two circles with radii equal to $d(p,q)$ centered at p and q, respectively. Notice that for any point x lying inside $lune(p,q)$, we have that both $d(x,p) < d(p,q)$ and $d(x,q) < d(p,q)$.

We now define an object known as the *constrained relative neighborhood graph* [5]. The relative neighborhood graph constrained to a set of line segments L, denoted as $CRNG(L)$, is a graph defined in the following way. A node of $CRNG(L)$ corresponds to an endpoint of a line segment in L. An edge exists between two nodes if and only if their corresponding endpoints e_1 and e_2 have one of the following properties:

1. The two endpoints are the endpoints of a line segment in L.

2. No endpoint lying in $lune(e_1, e_2)$ is visible from both e_1 and e_2.

Lemma 6.1 *The minimum weight spanning tree of a set of disjoint line segments, L, is a subgraph of the constrained relative neighborhood graph on that set.*

Proof: Suppose an edge of the minimum weight spanning tree, T, is not an edge of the constrained relative neighborhood graph. This edge must be a visibility edge, since by definition all line segment edges are contained in the $CRNG(L)$.

Let e be such an edge, with endpoints a and b. Since e is not an edge in the $CRNG(L)$, the $lune(a,b)$ must contain an endpoint x which is visible to both a and b. The removal of e from the tree, disconnects the tree into two trees T_1 and T_2, with $a \in T_1$ and $b \in T_2$. Since x was in T, it must be in one of the two components. Without loss of generality, let $x \in T_1$. Thus, the tree $T' = T - e + \overline{xb}$ has lower weight than T contradicting the fact that T has minimum weight. ■

Su and Chang [14] presented an algorithm to compute the constrained relative neighborhood graph of a set of disjoint line segments in $O(n \log n)$ time by pruning the constrained delaunay triangulation of the set. Since there are a linear number of edges in such a graph, we can then apply any $o(n \log n)$ time minimum weight spanning tree algorithm ([8], [17]) to the resulting constrained relative neighborhood graph. Therefore we have the following theorem.

Theorem 6.1 *The minimum weight spanning tree of a set of disjoint line segments can be computed in $O(n \log n)$ time.*

In the following section, we show that this algorithm is optimal.

7 Lower Bound

In this section, we show that the problem of finding the minimum spanning tree of a set of disjoint line segments requires time $\Omega(n \log n)$.

Theorem 7.1 *The problem of sorting n real numbers is $O(n)$ transformable to the problem of finding a minimum weight spanning tree of a set of disjoint line segments; thus, finding the minimum spanning tree of a set of disjoint line segments requires $\Omega(n \log n)$ time.*

Proof: Given a set S of n positive real numbers, $x_1, \ldots, x_n$, we show how any minimum weight spanning tree algorithm can be used to sort them with only linear overhead. For convenience, let $s_1, \ldots, s_n$ represent the sorted order of the real numbers.

Let x_{max} represent the largest real number in S and x_{min} the smallest. Let $d = (x_{max}^2 - x_{min}^2) + (x_{max} - x_{min})$. Let $m = x_{max}^2$. All these values can be easily computed in linear time.

For each number x_i, we construct a corresponding line segment, l_i, i.e., we associate the number i with the line segment (refer to Figure 9). The line segment l_i is constructed in the following way. One endpoint has coordinates (x_i, x_i^2). This endpoint is known as the *lower endpoint*. The other endpoint has coordinates $(x_i, d + 4m - 2x_i^2)$, and is known as the *upper endpoint*.

All of the lower endpoints lie on the parabola $y = x^2$ and all of the upper endpoints lie on the parabola $y = -2x^2 + d + 4m$. Since the parabola $y = x^2$ opens upwards and the parabola $y = -2x^2 + d + 4m$ opens downwards, the endpoints of l_{s_i} can only see the endpoints of $l_{s_{i-1}}$ and $l_{s_{i+1}}$. Observe that the set of visibility edges between all lower endpoints together with the given line segments forms an encompassing spanning tree. We will now show that such a tree is the only minimum weight spanning tree.

Suppose the minimum weight spanning tree, T, uses an edge, e, between two upper endpoints. Let l_i and l_j be the two line segments joined by e, with $x_i < x_j$. Therefore, $\|e\| = \sqrt{(x_j - x_i)^2 + (2x_i{}^2 - 2x_j^2)^2}$. Let e' be the edge between the two lower endpoints of l_i and l_j. We see that $\|e'\| = \sqrt{(x_j - x_i)^2 + (x_j^2 - x_i^2)^2}$. But, $\|e'\| < \|e\|$. Therefore, the tree $T' = T - e + e'$ has lower total weight, contradicting the fact that T has minimum weight. Therefore, the minimum weight spanning tree does not use edges between upper endpoints. Similarly, we see that the minimum weight spanning tree does not use edges between upper and lower endpoints, since the distance between an upper endpoint and a lower endpoint is at least $d + 2m$. Therefore, the minimum weight spanning tree consists only of edges between lower endpoints together with the given line segments.

Since the minimum weight spanning tree consists of visibility edges between all lower endpoints together with the given line segments, a simple depth-first traversal of the tree starting from the leftmost vertex of degree one enables us to uncover the sorted order of the input in linear time from the output delivered by any algorithm. ■

8 Conclusion

In this paper, we have shown that a set of disjoint line segments always admits an encompassing spanning tree with no crossing edges. This tree can be computed in $O(n \log n)$ time. We have shown that in the presence of collinearities, a set of disjoint line segments always admits an encompassing spanning tree with no crossing edges that can be computed in $O(n^2)$ time. We have also shown that the minimum weight spanning tree of the set of line segments contains no crossing edges and has maximum node degree seven. Furthermore, there exists a set of line segments that achieves this bound. The encompassing minimum weight spanning tree is a subgraph of the constrained relative neighborhood graph and can be computed in time $O(n \log n)$ when the endpoints of the line segments are in general position. We have shown a lower bound of $\Omega(n \log n)$ for the problem establishing that our algorithm is optimal. There are a number of open problems still to be considered.

1. Is it possible to remove the general position assumption for the minimum weight spanning tree and still compute it more efficiently than brute force? In computing the minimum weight spanning tree, we use the constrained delaunay triangulation whose computation is difficult when points are not in general position.

2. Is it NP-Hard or NP-Complete to compute a simple polygon or a simple hamiltonian path through a set of disjoint line segments? Rappaport [11] has shown that it is NP-Complete

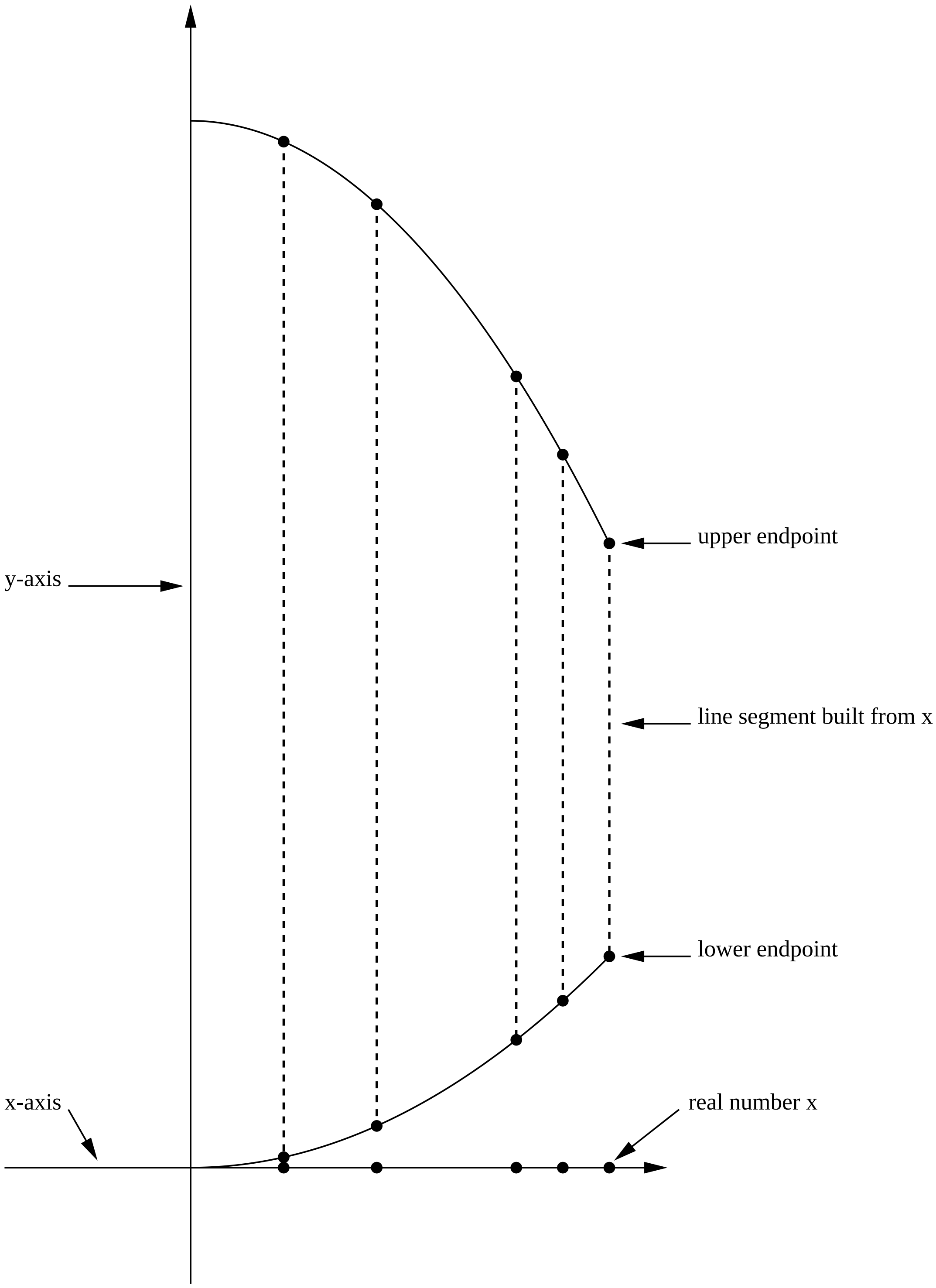
when the line segments can intersect at endpoints.

3. Is it possible to compute a simple polygon through a set of disjoint line segments, where the line segments are either part of the boundary, internal diagonals or external diagonals? Urabe and Watanabe [16] have shown that if the line segments are limited to the boundary and internal diagonals then it is not always possible.

4. Is the augmented visibility graph of a set of disjoint line segments hamiltonian?

Acknowledgements We thank David Bremner, Anna Lubiw, Henk Meijer, Joe O'Rourke, Eduardo Rivera-Campo and Jorge Urrutia for fruitful discussions on the topic. We thank Luc Devroye for his expertise in PostScript and for making Figure 9. We also thank an anonymous referee for helpful comments. Finally, we thank David Thompson, Marie Lee, and Al Van Houtte for providing a stimulating research environment.

Figure 9: Illustration for proof of Theorem 7.1.



References

[1] D. AVIS AND D. RAPPAPORT. Computing monotone simple circuits in the plane. *Computational Morphology*. Elsevier Science, G.T. Toussaint (editor), North Holland, 1988.

[2] J.A. BONDY AND U.S.R. MURTY. *Graph theory with applications*. Elsevier Science, New York, New York, 1976.

[3] H. ELGINDY AND G. TOUSSAINT. Efficient algorithms for inserting and deleting edges from triangulations. *Proc. International Conference on Foundations of Data Organization*, Kyoto, Japan, 1985.

[4] H. ELGINDY AND G. TOUSSAINT. On triangulations of line segments. *Proc. International Conference on Modelling and Simulation*, Athens, Greece, pp. 83–109, 1984.

[5] J. JAROMCZYK AND G. TOUSSAINT. Relative neighborhood graphs and their relatives. *Proceedings of the IEEE,* **80, 9**, pp. 1502-1517, 1992.

[6] J. O'ROURKE AND J. RIPPEL. Two segment classes with hamiltonian visibility graphs. Technical Report 15, Smith College, 1992.

[7] J. O'ROURKE. *Art gallery theorems and algorithms*. Oxford University Press, New York, New York, 1987.

[8] F. PREPARATA AND M. SHAMOS. *Computational geometry: an introduction*, Springer-Verlag, New York, New York, 1985.

[9] A. MIRZAIAN. Hamiltonian triangulations and circumscribing polygons of disjoint line segments. *Computational Geometry: Theory and Applications*,**2, 1**, pp. 15-30, 1992.

[10] C. MONMA AND S. SURI. Transitions in geometric minimum spanning trees. *Proc. ACM Symp. on Comp. Geom.*, pp. 239-249, 1991.

[11] D. RAPPAPORT. Computing simple circuits from a set of line segments is NP-complete. *the 3rd ACM Symposium on Computational Geometry*, Waterloo, Ontario, pp. 52-60, 1987.

[12] D. RAPPAPORT, H. IMAI, AND G.T. TOUSSAINT. Computing simple circuits from a set of line segments. *Discrete and Computational Geometry*, **5, 3**, pp. 289-304, 1990.

[13] X. SHEN AND H. EDELSBRUNNER. A tight lower bound on the size of visibility graphs, *Information Processing Letters*, **26**, pp. 61-64, 1987.

[14] T. SU AND R. CHANG. Computing the constrained relative neighborhood graphs and constrained Gabriel graphs in Euclidean plane. *Pattern Recognition*, **24, 3**, pp. 221-230, 1991.

[15] G. TOUSSAINT AND D. AVIS. On a convex hull algorithm for polygons and its application to triangulation problems, *Pattern Recognition*, **15**, 23–29, 1982.

[16] M. URABE AND M. WATANABE. On a counterexample to a conjecture of Mirzaian. *Computational Geometry: Theory and Applications*, **2, 1**, pp. 51-53, 1992.

[17] A. YAO. An $O(E \log \log V)$ algorithm for finding minimum spanning trees, *Information Processing Letters*, **4**, pp. 21-23, 1975.