NEW RESULTS FOR THE MINIMUM WEIGHT TRIANGULATION PROBLEM

Lenwood S. Heath Sriram V. Pemmaraju

Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061

December 7, 1992

Abstract

Given a finite set of points in a plane, a triangulation is a maximal set of non-intersecting line segments connecting the points. The weight of a triangulation is the sum of the Euclidean lengths of its line segments. Given a set of points in a plane, the minimum weight triangulation problem is to find a triangulation whose weight is minimum. No polynomial time algorithm is known to solve this problem, and it is also unknown whether the problem is NP-hard. The current best polynomial time approximation algorithm produces a triangulation that can be $O(\log n)$ times the weight of the optimal triangulation. We propose an algorithm that triangulates a set P, of n points in a plane in $O(n^3)$ time and that never does worse than the greedy triangulation. The algorithm produces an optimal triangulation if the points in P are the vertices of a convex polygon. The algorithm has the flavor of a heuristic proposed by Lingas and an analysis similar to his can be performed for our algorithm also, but experimental results indicate that our algorithm performs much better than the heuristic of Lingas. The results of experiments comparing the optimal triangulation with the performance of our algorithm, the heuristic of Lingas, and the greedy algorithm lead us to conjecture that the triangulations produced by our algorithm are within $O(1)$ of an optimal triangulation. We restate the minimum weight triangulation problem as a graph problem and show the NP-hardness of a closely related graph problem. Finally, we show that the constrained problem of computing the minimum weight triangulation, given a set of points in a plane and enough edges to form a triangulation, is NP-hard. These results are an advance towards a proof that the minimum weight triangulation problem is NP-hard.

Key words. Minimum weight triangulation, greedy triangulation, Delaunay triangulation, minimum spanning tree, local optimality, NP-hardness

Figure 1: A minimum weight triangulation and an arbitrary triangulation for a set of 10 points.

1 Introduction

Let $P = \{p_i : i = 1, 2, \ldots, n\}$ be a set of points in a plane, where each point p_i has the coordinates (x_i, y_i). To simplify our exposition, we assume that no three points in P are collinear. Let (p_i, p_j) where $i \neq j$ denote the line segment with endpoints p_i and p_j. Let $E(P)$ denote the set of line segments with endpoints in P, given by $E(P) = \{(p_i, p_j) : i \neq j\}$. We often think of the points in P as vertices and the line segments in $E(P)$ as edges of a graph and define various graph problems related to the minimum weight triangulation problem. Two line segments *cross* if they intersect at a point that is not a common endpoint. A *triangulation* $T(P)$ is a maximal set of mutually non-crossing line segments. Let $CH(P)$ denote the set of line segments bounding the convex hull of P, and let $|S|$ denote the cardinality of a set S. Then two properties of any triangulation $T(P)$ are the following.

1. $CH(P) \subseteq T(P) \subseteq E(P)$.

2. $|T(P)| = 3(n-1) - |CH(P)|$. As $3 \leq |C(P)| \leq n$, we have $2n - 3 \leq |T(P)| \leq 3n - 6$.

The *weight* of a line segment (p_i, p_j), denoted by $w(p_i, p_j)$, is the Euclidean distance between p_i and p_j and is given by $w(p_i, p_j) = ((x_i - x_j)^2 + (y_i - y_j)^2)^{1/2}$. The *weight* of a triangulation $T(P)$ is given by $W(T(P)) = \sum_{(p_i, p_j) \in T(P)} w(p_i, p_j)$. A *minimum weight triangulation* of a planar set of points P is a triangulation of P that has minimum weight among all triangulations. The minimum weight triangulation problem is

> Given $P = \{p_i : i = 1, 2, \ldots, n\}$, a set of n points in a plane, find a minimum weight triangulation of P.

We shall denote an arbitrary minimum weight triangulation of P by MWT(P) and its weight by W(MWT(P)).

To illustrate these concepts, Figure 1 shows a minimum weight triangulation and an arbitrary triangulation of a set of 10 points.

The minimum weight triangulation problem has applications in the numerical approximation of bivariate data. Yoeli [24] suggests an approach called the *polyhedron method* to calculate the value of a function f at any arbitrary point p, given the value of f at irregularly spaced points p_i, for $i = 1, 2, \ldots, n$. In this approach, the function surface is approximated by a triangulation of the points p_i, for $i = 1, 2, \ldots, n$. The point p lies within some face of the triangulation and $f(p)$ can be approximated by the linear interpolation of the three vertices of that face. A minimum weight triangulation has good numerical properties and provides a close approximation of the function surface.

The complexity of the minimum weight triangulation problem has been open since 1975 when it was mentioned by Shamos and Hoey [18]. Since then, several algorithms have been proposed to solve this problem [12, 16, 18]. None of these is known to produce even a constant approximation of a minimum weight triangulation. In the other direction, efforts to show the minimum weight triangulation problem NP-hard have also failed. In this paper we address this problem from both these directions.

In the first direction, we present an algorithm, called the greedy spanning tree triangulation algorithm (in brief, G-ST-T) that triangulates a set of n points in $O(n^3)$ time. G-ST-T has the flavor of a heuristic by Lingas [12], which we shall call the minimum spanning tree triangulation algorithm (in brief, MST-T). Both G-ST-T and MST-T follow a two step paradigm. In the first step, P is viewed as the vertices of a graph and $E(P)$ as its edges. Each algorithm choses a minimal subset $E^s(P)$ of pairwise non-crossing edges that span P. The choice of this set is crucial, and G-ST-T and MST-T differ in this choice. In the second step, an optimal triangulation containing $E^s(P)$ is obtained by using a dynamic programming algorithm that is an extension of an algorithm due to Gilbert [5] that optimally triangulates the interior of a simple polygon. If every point of P is in the convex hull, then both algorithms optimally triangulate P. MST-T never produces a triangulation that has weight greater than that of the Delaunay triangulation, while G-ST-T never produces a triangulation that has weight greater than that of the greedy triangulation. However, experiments indicate that G-ST-T rarely produces a non-optimal triangulation and produces an optimal triangulation far more frequently than the greedy algorithm. Even when G-ST-T fails to produce an optimal triangulation, its closeness to optimality is remarkable

and certainly much better than any current algorithm.

We also present a formulation of the minimum weight triangulation problem as a graph problem. This leads to a variety of graph theoretic problems whose solution may have a bearing on the status of the minimum weight triangulation problem. We prove the NP-hardness of one such problem. Our other NP-hardness result is motivated by the NP-completeness result of Lloyd [13]. We show that given P, and $E^s(P)$, a subset of $E(P)$ that contains at least one triangulation of P, the problem of finding a minimum weight triangulation of P that is a subset of $E^s(P)$ is NP-hard.

The paper is organized as follows. In Section 2, we survey previous work related to the minimum weight triangulation problem. In Section 3, we discuss a dynamic programming algorithm that optimally triangulates the interior of a convex polygon in $O(n^3)$ time. We present an extension of this algorithm that optimally triangulates more general planar regions called "cells". In Section 4, we review Lingas' algorithm (MST-T) and present our new algorithm (G-ST-T). Both algorithms run in $O(n^3)$ time. Though neither always produces an optimal triangulation, G-ST-T possesses properties that ensure that non-optimal triangulations are extremely rare. This is supported by experiments that compare the minimum weight triangulation with the performance of G-ST-T, MST-T, and the greedy triangulation. These we present in Section 5. In Section 6, we investigate issues of local optimality related to the known triangulation algorithms. In Section 7, we present a graph theoretic formulation of the minimum weight triangulation problem and prove two new NP-hardness results. The final section of the paper, Section 8, contains some open problems and conjectures that arise from this work.

2 Previous approaches

The complexity of the minimum weight triangulation is one of only four problems that remains open from Garey and Johnson's [4] original list of twelve open problems.

Attempts to show the minimum weight triangulation problem NP-hard have resulted in two related NP-hardness results. In the earliest result, Lloyd [13] shows that given a set P of points in a plane and a subset E^s of $E(P)$, the problem of determining whether $E^s(P)$ contains even one triangulation is NP-complete. In a later result, Lingas [10] shows that

the problem of determining the minimum weight geometric triangulation of multi-connected polygons is NP-complete.

Though there is no polynomial time algorithm that constructs even a constant approximation of the minimum weight triangulation, the greedy and the Delaunay triangulations have been studied closely as approximations of minimum weight triangulation. The greedy triangulation is constructed by a greedy algorithm that chooses the smallest edge that does not cross any previously chosen edge. Presently the most efficient known algorithm for the greedy triangulation uses $O(n^2)$ time and $O(n)$ space (see Levcopoulos and Lingas []). The Delaunay triangulation is a planar dual of the Voronoi diagram and can be constructed in $O(n \log n)$ time by a divide and conquer algorithm (see Shamos and Hoey [18]). Simple counterexamples to show that the greedy algorithm and the Delaunay algorithm do not produce an optimal triangulation are presented by Lloyd [13].

Let GT(P), DT(P), and MWT(P) denote a greedy triangulation, the Delaunay triangulation, and a minimum weight triangulation of P, respectively. A measure of how close a triangulation $T(P)$ is to a minimum weight triangulation is given by the ratio

$$R(T(P)) = \frac{W(T(P))}{W(\mathrm{MWT}(P))}.$$

Since neither the greedy triangulation nor the Delaunay triangulation is optimal, the worst-case ratios $R(\mathrm{GT}(P))$ and $R(\mathrm{DT}(P))$ give an indication of how well these triangulations approximate MWT(P). Levcopoulos [9] provides a lower bound for the greedy triangulation by showing that, for each n, there exists an n-point set P_1 with the property that

$$R(\mathrm{GT}(P_1)) = \Omega\left(n^{\frac{1}{2}}\right).$$

In a complementary result, Kirkpatrick [7] constructs, for each n, an n-point set P_2 with the property that

$$R(\mathrm{DT}(P_2)) = \theta(n).$$

Thus the greedy triangulation and the Delaunay triangulation can be arbitrarily bad approximations of a minimum weight triangulation.

Plaisted and Hong [16] present a triangulation algorithm and show that the weight of the triangulation that their algorithm produces is within $O(\log n)$ of the weight of the optimal

Figure 2: A 12 point cell.

triangulation. Their implementation of the heuristic has a time complexity of $O(n^5)$. Smith [19] improves on this by implementing the algorithm of Plaisted and Hong in $O(n^2 \log n)$ time. The upper bound on the performance of the heuristic of Plaisted and Hong is the best known for any algorithm. Lingas [12] presents a heuristic (MST-T) for the minimum weight triangulation problem that requires $O(n^3)$ time and seems to perform well in the average case.

Using certain separator theorems, Smith [19] gives the first subexponential time algorithm to compute a minimum weight triangulation.

3 An Improved Triangulation Algorithm

Before presenting our algorithm (G-ST-T), we make an observation that will be useful later.

Gilbert's dynamic programming algorithm [5] optimally triangulates the interior of a simple polygon. We extend this algorithm to triangulate a more general figure that we call a *cell*. A cell is any interior face of a straight line planar embedding of a graph. Any cell can be uniquely represented by a sequence of vertices $p_0, p_1, \ldots, p_{n-1}$ encountered if the boundary of the cell is traversed in, say, a counterclockwise order. Note that the vertices in the sequence are not, in general, distinct and that there may be edges that are traversed twice. A cell with the points labeled is shown in Figure 2. It can be represented completely by the sequence $1, 2, 3, 2, 4, 5, 6, 7, 8, 9, 10, 11, 10, 9, 12, 9, 8$. For the purposes of the algorithm every occurrence of the same point in the sequence is treated as being different. The dynamic programming algorithm that optimally triangulates the interior of a cell is similar to the one that optimally triangulates the interior of a convex polygon except that now the only line segments that can be considered are those that do not cross any line segments of the cell. This can be accomplished by assigning a weight of $+\infty$ to any line segment that crosses a line segment of the cell. Hence, we have the proposition

Proposition 3.1 *A cell of n points can be optimally triangulated in $O(n^3)$ time.*

The algorithm described above, that we shall call **Optimal cell triangulation (OCT)**,

will be employed as part of G-ST-T.

G-ST-T triangulates a set $P = \{p_1, p_2, \ldots, p_n\}$ of points in a plane in $O(n^3)$ time and has two distinct phases. In the first phase a spanning tree of the graph $(P, \mathrm{GT}(P))$ is chosen. How the spanning tree is chosen is explained later. The spanning tree along with the convex hull produce a single cell. In the second phase, the cell obtained in the first phase is optimally triangulated using the OCT algorithm. G-ST-T is similar in structure to the algorithm (MST-T) proposed by Lingas [12]. In fact, a similar analysis applies to G-ST-T also, and we include MST-T in our experimental analysis.

To introduce the two phase paradigm that is common to both MST-T and G-ST-T, we first present the minimum spanning tree triangulation algorithm.

Minimum spanning tree triangulation algorithm (MST-T)

Input : $P = \{p_1, p_2, \ldots, p_n\}$.

Output : A triangulation MST-T(P).

1. Compute the Delaunay triangulation DT(P).
2. Compute a minimum spanning tree $MST(P)$ of $(P, DT(P))$ using the following assignment of costs to line segments in $E(P)$

$$c((p_i, p_j)) = \begin{cases} \infty & \text{if } (p_i, p_j) \notin \mathrm{DT}(P) \\ 0 & \text{if } (p_i, p_j) \in CH(P) \\ w(p_i, p_j) & \text{otherwise} \end{cases}$$

3. Use OCT to optimally triangulate the single cell defined by $CH(P) \cup MST(P)$, producing MST-T(P).

The effect of the weights used in Step 2 is that all but one of the line segments in $CH(P)$ is in $MST(P)$. Steps 1 and 2 can each be performed in $O(n \log n)$ time while step 3 can be performed in $O(n^3)$ time. An example of a cell produced at the end of Step 2 in the MST-T algorithm is shown in Figure 3. The dark line segments constitute the minimum spanning tree chosen in step 2.

Let $I(P) = MST(P) - CH(P)$ be the interior line segments of the minimum spanning tree and let $jump(P) = \max\{w(p_i, p_j) : (p_i, p_j) \in I(P)\}$. Lingas [**?**] provides the following upper bound result for the performance of his algorithm as compared to the minimum weight triangulation.

Figure 3: A simple polygon created at the end of step 2 in the MST-T algorithm.

Figure 4: Counterexample to the MST-T algorithm.

Proposition 3.2 *For a set P of n points,*

$$\frac{W(MST\text{-}T(P))}{W(MWT(P))} = O\left(\frac{n \times jump(P)}{W(MWT(P))} + \log n\right).$$

Lingas also gives an example for which $n \times jump(P)/W(\mathrm{MWT}(P)) = \Omega(n)$ and yet the minimum weight triangulation is exactly the same as the triangulation produced by the MST-T algorithm. This shows that the bound is quite loose.

The following propositions follow immediately from the MST-T algorithm.

Proposition 3.3 *If P contains only vertices of a convex polygon, then* MST-T(P) *is optimal.*

Proposition 3.4 *For any set of points P,* MST-T(P) *is never worse than the Delaunay triangulation of P, i.e., $W(\mathrm{MST\text{-}T}(P)) \leq W(\mathrm{DT}(P))$.*

The MST-T algorithm produces an optimal triangulation if and only if all the line segments in the spanning tree that we chose are also in some particular minimum weight triangulation. That this is not always the case is shown in Figure 4. Here, three of the four convex hull edges are chosen to be in the minimum spanning tree. Whatever the choice may be, AC cannot be in the minimum spanning tree because it creates a cycle. Since, $w(DE) < w(AE) = w(CE) < w(BE)$, DE is the next edge chosen to be in the minimum spanning tree. The triangulation thus computed by MST-T algorithm is obviously non-optimal, since the optimal triangulation can be obtained by choosing edges AC, AE, CE and BE along with the convex hull edges.

The MST-T algorithm does not produce a non-optimal triangulation for any of the counterexamples in the literature [7, 9, 13, 14] that have been used to show either the greedy triangulation or the Delaunay triangulation to be non-optimal. This can be attributed to the fact that the algorithm picks the least number of edges needed to keep the set of input vertices connected and then proceeds to select the optimal set of the remaining edges. So

far we have not been able to say anything about the asymptotic behavior of the ratio R(MST-T(P)). We conjecture that R(MST-T(P)) = $O(1)$. This conjecture is motivated by the observation that the effects of a "bad" minimum spanning tree edge are local.

There are many ways of choosing a spanning tree in the first phase of the paradigm. The spanning tree chosen in the MST-T algorithm is one such choice. We now present an algorithm that differs from the MST-T algorithm in the choice of a spanning tree. Though the theoretical bound on the weight of the triangulation produced by our algorithm is loose, experimental results indicate that the choice of spanning tree that we prescribe produces triangulations that are almost always optimal. The time complexity of this algorithm remains $O(n^3)$. We call this algorithm the *greedy spanning tree triangulation algorithm.*

Greedy spanning tree triangulation algorithm. (G-ST-T).

Input : $P = \{p_1, p_2, \ldots, p_n\}$.

Output : A triangulation G-ST-T(P).

1. Compute a greedy triangulation GT(P) of P and the convex hull $CH(P)$.

2. Compute a minimum spanning tree $GMST(P)$ of $(P, \text{GT}(P))$ using the following assignment of costs to line segments

$$c((p_i, p_j)) = \begin{cases} \infty & \text{if } (p_i, p_j) \notin \text{GT}(P) \\ 0 & \text{if } (p_i, p_j) \in CH(P) \\ w((p_i, p_j)) & \text{otherwise} \end{cases}$$

3. Use OCT to optimally triangulate the single cell in $CH(P) \cup GMST(P)$, producing G-ST-T(P).

An analysis of the time complexity of each step follows.

1. As mentioned earlier, the most efficient known greedy triangulation algorithm uses $O(n^2)$ time and $O(n)$ space [].

2. Since the number of edges in any planar graph is $O(n)$ the greedy triangulation contains $O(n)$ line segments. This implies that the second step can be accomplished in $O(n \log n)$ time.

3. As in the MST-T algorithm, the third step is performed in $O(n^3)$ time.

Figure 5: Counterexample to the G-ST-T algorithm.

As before, the time complexity of the third step dominates and the time complexity of the algorithm is $O(n^3)$ time.

Like M-ST-T, G-ST-T produces optimal triangulations for all the counterexamples in the literature [7, 9, 13, 14]. The following two propositions follow quite directly from the G-ST-T algorithm.

Proposition 3.5 *If P contains only vertices of a convex polygon, then* G-ST-T(P) *is optimal.*

Proposition 3.6 *For any set of points P,* G-ST-T(P) *is never worse than the greedy triangulation of P, i.e., W*(G-ST-T(P)) $\leq W$(GT(P)).

Hence all upper bounds for the greedy triangulation also hold for triangulation produced by the G-ST-T algorithm. In particular all results about the asymptotic behavior of R(GT(P)) [8] are true for R(G-ST-T(P)).

The G-ST-T algorithm produces an optimal triangulation if and only if the line segments of the minimum spanning tree of the greedy triangulation are always in some particular minimum weight triangulation. That this is not always the case can be seen from the seven point counterexample in Figure 5. Here, GT(P) contains line segments BG, BD, GF, FE, ED, GE and BE (among others). The minimum spanning tree chosen from the line segments of GT(P) contains GF, FE, ED and BE (as well as two convex hull line segments). These edges are shown in dark. The triangulation that the G-ST-T algorithm computes is the same as the greedy triangulation. A slight improvement in the weight of the triangulation can be obtained by choosing line segments FD and BF instead of GE and BE. Examples for which both the MST-T and the G-ST-T algorithms produce non-optimal triangulations can be constructed by combining the examples in Figure 4 and Figure 5. G-ST-T produces an optimal triangulation for the example in Figure 4, while MST-T produces an optimal triangulation for the example in Figure 5, thus showing that G-ST-T and MST-T are not comparable as the G-ST-T and the greedy algorithm are.

Proposition 4.1 can be used to construct an upper bound on the size of the triangulation produced by the G-ST-T algorithm. For this we define $greedy_jump(P)$ as the largest line segment in $GMST(P) - CH(P)$; that is,

$$greedy_jump(P) = \max\{w(p_i, p_j) : (p_i, p_j) \in GMST(P) - CH(P)\}.$$

Then we have the following theorem.

Theorem 3.1 *For any set of points P of size n,*

$$R(\text{G-ST-T}(P)) = O\left(\frac{n \times greedy_jump(P)}{W(\text{MWT}(P))} + \log n\right).$$

It is a well-known fact that the heaviest edge in a minimum spanning tree is never heavier than the heaviest edge in an arbitrary spanning tree (see Aho, Hopcroft, and Ullman [1], Exercise 5.2). From this fact, it is clear that for any set of points P, $greedy_jump(P) \geq jump(P)$. If for all sets of points P, $greedy_jump(P) = O(jump(P))$, then the bound specified in proposition 4.1 also holds for the G-ST-T algorithm.

4 Experimental results.

The greedy, MST-T, and G-ST-T algorithms were tested with sets of sizes 15, 20 and 25 with 400 sets of each size. Testing larger sets was impractical due to the time required by the exhaustive search for finding a minimum weight triangulation. The points in each set were generated randomly with integer coordinates between 0 and $2^{16} - 1$. The performance of these algorithms was compared to the weight of the optimal triangulation, which was computed through a backtrack search of the space of triangulations. (The backtrack search processes the line segments in increasing order by length. Pruning the search tree by comparison to the smallest triangulation found so far yields a search capable of finding an optimal triangulation for sets of up to 25 points.) Three kinds of comparisons were made.

Table 1 lists, for each set size, the number of times each algorithm fails to produce an optimal triangulation for that set size. It should be noted that the G-ST-T algorithm failed to produce the optimal triangulation only twice in 1200 trials as compared to the greedy algorithm which failed 174 times. The MST-T algorithm also does quite well as compared

Set Size	Number of failures		
	Greedy	MST-T	G-ST-T
15	44	5	1
20	61	11	0
25	69	12	1
Total	174	28	2

Table 1

Set Size	Average triangulation weight			
	Optimal	Greedy	MST-T	G-ST-T
15	298697.76	309873.34	299236.84	298697.76
20	349238.62	373288.61	350176.36	349238.62
25	437233.88	479283.36	439846.97	437233.88

Table 2

to the greedy algorithm, failing 28 times in 1200 trials. These results are evidence that MST-T and G-ST-T rarely produce non-optimal triangulations.

Table 2 is a comparison of the average size of the triangulations produced by each algorithm for each set size. G-ST-T fails only once each for set size 15 and set size 25, and never for set size 20. Even when the algorithm fails, the weights of the triangulations produced by G-ST-T algorithm are very close to the minimum weight triangulation. These results are evidence that in the average case G-ST-T and MST-T produce triangulations that are within a small constant of approximation of the minimum weight triangulation.

Table 3 compares the worst case triangulation produced by each algorithm. For each triangulation $T(P)$ and for each set size, Table 3 shows the worst case ratio

$$\frac{W(T(P)) - W(\mathrm{MWT}(P))}{W(\mathrm{MWT}(P))} = R(T(P)) - 1.$$

In view of these experimental results, we speculate that both MST-T and G-ST-T algorithms may always produce triangulations that are within $O(1)$ of the minimum weight triangulation.

As both the MST-T and G-ST-T algorithms appear to be very good heuristics, we tested the two against each other for sets of points that are too large for calculation of their

Set Size	Worst case % over optimal		
	Greedy	MST-T	G-ST-T
15	4.4963	0.9701	0.0017
20	10.1832	1.0017	0.0000
25	17.3897	1.5323	0.0003

Table 3

Set Size	G-ST-T versus MST-T		
	G-ST-T > M-ST-T	G-ST-T = MST-T	G-ST-T < MST-T
30	0	157	43
35	0	169	31
40	0	169	31
45	1	151	48
50	0	153	47

Table 4

optimal triangulations. For each set size 30, 35, 40, 45, and 50, a total of 200 randomly selected point sets was triangulated by both MST-T and G-ST-T. The results are reported in Table 4, where the number of times that one algorithm produced a smaller triangulation than the other is indicated. To summarize, in 1000 trials, the G-ST-T algorithm produced a better triangulation than the MST-T algorithm 200 times, while the MST-T algorithm produced a better triangulation than the G-ST-T algorithm only once. This is further evidence of the superiority of G-ST-T as a heuristic to approximate a minimum weight triangulation.

5 Issues of Local Optimality

Throughout the paper we have been calling a minimum weight triangulation "optimal". We now elaborate on the notion of optimality. Given a set of points P and a triangulation $T(P)$ of P, we say that $T(P)$ is *locally optimal* if every convex quadrilateral in $T(P)$ that has no points in its interior is optimally triangulated. It follows from the definition that for every convex quadrilateral Q in a locally optimal triangulation $T(P)$, the shorter diagonal of Q belongs to $T(P)$. While a minimum weight triangulation is always locally optimal,

Figure 6: An example to show that the G-ST-T algorithm does not always produce locally optimal triangulations.

the converse does not always hold. The status of the four triangulations discussed in this paper with regard to local optimality is presented in the following theorem.

Theorem 5.1 *With local optimality as defined above, we have that*

1. *A greedy triangulation is always locally optimal.*

2. *A Delaunay triangulation is not always locally optimal.*

3. *A minimum spanning tree triangulation is not always locally optimal.*

4. *A greedy spanning tree triangulation is not always locally optimal.*

Proof: From the definition of a greedy triangulation, it follows immediately that, in any convex quadrilateral in the triangulation, the shorter diagonal is always present and, hence, the longer diagonal is always absent. Therefore, a greedy triangulation is always locally optimal.

An example showing that the Delaunay triangulation is not always locally optimal can be found in Figure **??**. In this example, $\square BCDA$ is a convex quadrilateral in the triangulation that is not optimally triangulated.

An example showing that the MST-T algorithm does not always produce a locally optimal triangulation can be found in Figure 4. In this example, $\square DAEC$ is a convex quadrilateral in the triangulation that is not optimally triangulated.

Since any greedy triangulation is locally optimal, it is surprising that G-ST-T is not always locally optimal. The example in Figure 6 demonstrates this. In this example the points A, B, C, D, E, and F are in the interior of a large convex polygon. In the upper figure the greedy triangulation of these points is shown with the minimum spanning tree indicated by the darkened line segments. In the lower figure the output of G-ST-T is shown. The two triangulations differ in that the greedy triangulation contains line segments DF and AF, while the triangulation produced by G-ST-T contains line segments AE and BE. AB is in the greedy triangulation and also in the minimum spanning tree of the greedy

triangulation. In the output of G-ST-T AB is the diagonal of quadrilateral $AEBC$ and diagonal AB is longer than diagonal EC. □

The really surprising result of Theorem 3 is that a triangulation produced by G-ST-T is not always locally optimal. The G-ST-T triangulation is derived from the locally optimal greedy triangulation, yet fails to inherit the local optimality. It is, of course, algorithmically easy to "repair" a G-ST-T triangulation to make it locally optimal by switching short diagonals for long ones in convex quadrilaterals. Such a repair may yield a triangulation that is not only locally optimal but is, in fact, globally optimal. In the few examples of non-optimal G-ST-T triangulations that we know about (see Figure 5 and Section 4), there is always some convex k-gon, where $4 \leq k \leq 6$, that is not optimally triangulated; replacing the triangulation of this k-gon with an optimal triangulation yields a globally optimal triangulation for all these examples. We speculate that it may be algorithmically inexpensive (i.e., polynomial time) to repair a G-ST-T triangulation to obtain a MWT, in an average-case sense if not in a worst-case sense.

6 NP-hardness results

In keeping with the notation of Garey and Johnson [4], the minimum weight triangulation problem can be formulated as a decision problem as follows.

MINIMUM WEIGHT TRIANGULATION (MWT)
INSTANCE. A set P of points in a plane, a positive rational number k.
QUESTION: Is there a triangulation of P whose weight is no more than k?

The computational complexity of this problem remains open. The minimum weight triangulation problem can be viewed in various ways. A particular view is to consider P as the set of vertices of a graph G and $E(P)$ as the set of edges of G. G is the complete graph on $|P|$ vertices. The lengths of the line segments in $E(P)$ can be thought of as weights associated with the edges of G. This graph $G = (P, E(P))$ made from the points and line segments has an obvious straight line embedding in the plane. Lloyd [13] proves the

Figure 7: A complete straight line graph and its crossing graph.

following problem related to MWT to be NP-complete. In keeping with Lloyd's notation we denote $E(P)$ by E.

TRIANGULATION EXISTENCE (TRI)

INSTANCE: A set P of points in a plane, a set of line segments $E' \subseteq E$.

QUESTION: Does E' contain a triangulation of P?

This problem is shown to be NP-complete by a reduction from SAT.

In this section, we prove two other problems related to MWT to be NP-complete. The first problem is gotten by first deriving a new graph from the straight line embedding of G. Let

$$C \quad = \quad \{(e_i, e_j) \in E(P) \times E(P) : e_i \text{ and } e_j \text{ cross}\}$$

be the set of crossings in the embedding. Then $G^c = (E(P), C)$ is the *crossing graph* of G.

Figure 7 illustrates a graph G and its crossing graph G^c. The minimum weight triangulation problem can be posed in terms of the crossing graph G^c as follows:

> **Minimum Weight Triangulation, Optimization Version:** Given a crossing graph G^c, where each vertex (line segment) is assigned a weight equal to its Euclidean length, find a maximal independent vertex set of minimum total weight.

By allowing the vertex weights to be arbitrary *rational* numbers and by restating the problem as a decision problem, we obtain the following:

RESTRICTED MINIMUM MAXIMAL INDEPENDENT SET (RMMI)

INSTANCE. A crossing graph $G^c = (E, C)$, positive rational weight $w(e)$ for each $e \in E$, and a positive rational number k.

QUESTION: Is there a maximal independent set I in G^c such that $\sum_{e \in I} w(e) \leq k$.

This close relative of MWT is easily shown to be NP-complete. The proof of the following theorem is essentially identical to a result of Anagnostou [2].

Theorem 6.1 *RMMI is NP-complete.*

Proof: Since the total weight of a set of vertices, each of which has a *rational* weight, is easily calculated in polynomial time, RMMI is clearly in NP. We show that RMMI is NP-hard by exhibiting a reduction of TRI to RMMI. Let P and E' constitute an instance of TRI. Let $G^c = (E, C)$ be the crossing graph associated with the set P. By definition, $E' \subset E$. Assign weights to the vertices of E as follows

$$w(e) = \begin{cases} 0 & \text{if } e \in E'; \\ 1 & \text{if } e \in E - E'. \end{cases}$$

Let the instance of RMMI consist of G^c, w, and $k = 0$. There is a triangulation of P contained in E' if and only if G^c contains a maximal independent set of size 0. Hence, TRI reduces to RMMI in polynomial time. We conclude that RMMI is NP-complete. □

The second NP-hardness result that we prove is for a geometric decision problem. Motivated by Lloyd's [13] NP-completeness result for TRI, we propose the following generalization of MWT.

GENERALIZED MINIMUM WEIGHT TRIANGULATION (GMWT)
INSTANCE: A set P of points in a plane, a set of line segments $E' \subseteq E$, such that E' contains a triangulation of P, a positive rational number k.
QUESTION: Is there a triangulation in E' whose weight is no more than k?

Theorem 6.2 *GMWT is NP-hard.*

Proof: We reduce TRI to GMWT. Let P and $E' \subseteq E(P)$ constitute an instance of TRI. In particular, from P and $E' \subseteq E(P)$ we construct a point set Q and a set of line segments $E'' \subseteq E(Q)$ and specify a positive integer k such that E' contains a triangulation of P if and only if E'' contains a triangulation of Q with weight no greater than k.

Let $P = \{(x_i, y_i) \mid 1 \leq i \leq n\}$. Without loss of generality, assume that $n \geq 3$, that all points in P are within a distance of $1/2$ from the origin, that no two points have the same x or y coordinate, and that points in P are indexed by increasing y-coordinate, (i.e., $i < j$ implies $y_i < y_j$). Let h_P be the number of points in the convex hull of P, $CH(P)$. No

line segment in E' has length greater than 1 and therefore any triangulation of P, $T(P)$ satisfies

$$W(T(P)) \leq 3(n-1) - h_P \leq 3n - 6.$$

The set of points Q is constructed by adding to P, points of the form

$$\begin{aligned} l_i &= \left(-1/2 - 9n, \frac{y_i + y_{i+1}}{2}\right) \\ r_i &= \left(+1/2 - 9n, \frac{y_i + y_{i+1}}{2}\right). \end{aligned}$$

Specifically, let $Q = P \cup L \cup R$ where

$$\begin{aligned} L &= \{l_i \mid 1 \leq i \leq n-1\} \\ R &= \{r_i \mid 1 \leq i \leq n-1\}. \end{aligned}$$

It follows that $|Q| = 3n - 2$. Note that the two points in P with maximum and minimum y coordinate, together with all the points in L and R, make up the convex hull of Q. Therefore

$$h_Q = |CH(Q)| = 2 + (n-1) + (n-1) = 2n.$$

Let

$$\begin{aligned} H &= \{(l_i, r_i) \mid 1 \leq i \leq n-1\} \\ S &= \{(l_i, l_{i+1}) \mid 1 \leq i \leq n-2\} \cup \{(r_i, r_{i+1}) \mid 1 \leq i \leq n-2\}. \end{aligned}$$

We complete the instance of GMWT by defining E'' as

$$E'' = E' \cup H \cup (L \times P) \cup (R \times P) \cup S.$$

Note that, for any edge $e \in (L \times P) \cup (R \times P)$, $9n < w(e) < 9n + 2$. Also note that $L \times P$ and $R \times P$ each contain 2 edges from $CH(Q)$ and S contains the remaining edges.

The sets Q and E'' satisfy the conditions for an instance of GMWT since

$$\begin{aligned} S \cup H \quad & \cup \quad \{(l_i, p_i) \mid 1 \leq i \leq n-1\} \\ & \cup \quad \{(l_i, p_{i+1}) \mid 1 \leq i \leq n-1\} \\ & \cup \quad \{(r_i, p_i) \mid 1 \leq i \leq n-1\} \\ & \cup \quad \{(r_i, p_{i+1}) \mid 1 \leq i \leq n-1\} \end{aligned}$$

forms a triangulation, as Figure **??** illustrates. The number of edges in any triangulation $T(Q)|$ of Q satisfies

$$|T(Q)| = 3(|Q| - 1) - h_Q = 7n - 9.$$

Finally, let

$$k = (3n - 6) + 2 + n'(9n - 2)$$

where

$$n' = |T(Q)| - |S| - (3(n - 1) - h_P) = 2(n - 1) + h_P.$$

We now show that E' contains a triangulation if and only if E'' contains a triangulation whose weight is no more than k. First suppose E' contains a triangulation, $T(P)$. Then

$$T(Q) = T(P) \cup S \cup D_L \cup D_R$$

triangulates Q. Here D_R (respectively, D_L) is any triangulation of the gap between the convex hull of P and R. One such triangulation $T(Q)$ is shown in Figure **??**. Clearly,

$$|D_L \cup D_R| = 2(n - 1) + h_P = n'.$$

Since, $W(T(P)) \leq 3n - 6$, $W(S) \leq 2$, and $W(D_L \cup D_R) \leq n'(9n + 2)$, we have that $W(T(Q)) \leq k$.

Suppose on the other hand that there is a triangulation $T(Q)$ in E'' whose weight is no greater than k. $T(Q)$ must contain all $2(n - 2)$ edges from S since these are part of the convex hull of Q. In addition, $T(Q)$ must contain at most n' edges from $H \cup (L \times P) \cup (R \times P)$. To see this, notice that the weight of m of these long edges, for $m > n'$, is greater than $m \cdot 9n$ which in turn is greater than k. It follows that $T(Q)$ contains at least $|T(Q)| - n' - 2(n-2) = 3(n - 1) - h_P$ edges from E', which must therefore triangulate P. □

GMWT is the closest problem to MWT that has been shown to be NP-hard. Additional insights may allow the above proof to be extended to MWT.

7 Conclusions and Conjectures

We have made progress in two directions towards determining the complexity of the minimum weight triangulation problem. In the first direction, we present two algorithms, one

of which is due to Lingas (MST-T) [12] which experimentally perform much better than previous algorithms. While MST-T never performs worse than the Delaunay triangulation, our algorithm (G-ST-T) never performs worse than the greedy triangulation. Experimental results show that our algorithm rarely produces non-optimal triangulations as compared to the greedy triangulation and the triangulation produced by Lingas' algorithm. Even when a non-optimal triangulation is produced by our algorithm, it is very close in weight to the minimum weight triangulation. Both algorithms also produce a minimum weight triangulation when P is convex. These results lead us to the following conjecture

> **Conjecture 1:** The MST-T and G-ST-T algorithms produce triangulations that are within $O(1)$ of the minimum weight triangulation in the worst case.

In the second direction, we present two NP-hardness results. These results show two generalizations of the minimum weight triangulation problem to be NP-hard. For our first NP-hardness result we formulate the minimum weight triangulation as a graph theoretic problem. This provides a new way of looking at the minimum weight triangulation problem. The second NP-hardness result is for a geometric decision problem that is a generalization of the minimum weight triangulation problem. Motivated by these results we have the following conjecture.

> **Conjecture 2:** The MWT problem is NP-hard.

In our graph theoretic formulation of the minimum weight triangulation problem we defined the class of Alternate characterizations of crossing-graphs are currently lacking but would help in designing Solutions to these problems may provide some insight into the minimum weight triangulation

Acknowledgments

We wish to thank the anonymous referee for helpful corrections and for pointing out the result of Anagnostou.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *Data Structures and Algorithms.* Addison Wesley Publishing Company, 1980.

[2] E. Anagnostou, Progress in minimum weight triangulation. Technical Report 232/90, Department of Computer Science, University of Toronto, 1990.

[3] M.R. Garey, D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, New York, 1979.

[4] P.D. Gilbert, New results on planar triangulations. M.S. thesis, Report No. UILUENG 78 2243, University of Illinois, 1979.

[5] D.G. Kirkpatrick, A note on Delaunay and optimal triangulations. *Information Processing Letters*, 10, 1980, 127–128.

[6] C. Levcopoulos and A. Lingas, On approximation behavior of the greedy triangulation for convex polygons. *Algorithmica,* 2, 1987, 175–193.

[7] C. Levcopoulos, An $O(\sqrt{n})$ lower bound for the non-optimality of the greedy triangulation. *Information Processing Letters*, 25, 1987, 247–251.

[8] A. Lingas, The greedy and Delaunay triangulations are not bad in the average case and minimum weight triangulation of multi-connected polygons is NP-complete. *Foundations of Computation Theory*, Lecture Notes in Computer Science 158, Springer-Verlag, Heidelberg, 1983, 270–284.

A. Lingas, A new heuristic for the minimum weight triangulation. *SIAM Journal of Algebraic and Discrete Methods*, 8, 1987, 646–658.

[9] E.L. Lloyd, On triangulations of a set of points in the plane. *Proceedings of the Eighteenth IEEE Symposium on Foundations of Computer Science*, 18, 1977, 228–240.

[10] G.K. Manacher and A.L. Zobrist, Neither the greedy nor the Delaunay triangulation of a planar point set approximates the optimal triangulation. *Information Processing Letters*, 9, 1979, 31–34.

[11] D.A. Plaisted and J. Hong, A heuristic triangulation algorithm. *Journal of Algorithms*, 8, 1987, 405–437.

[12] M.I. Shamos and D. Hoey, Closest point problems. *Proceedings of the Sixteenth IEEE Symposium on Foundations of Computer Science*, 1975, 151–162.

[13] W.D. Smith, *Studies in Computational Geometry Motivated by Mesh Generation.* Ph.D. dissertation, Princeton University, 1989.

[14] P. Yoeli, Compilation of data for computer-assisted relief cartography. In *Display and Analysis of Spatial Data,* J.C. Davis and M. J. McCullagh, editors, John Wiley & Sons, New York, 1975.