On Hard Instances of the Minimum-Weight Triangulation Problem

Sándor P. Fekete[1], Andreas Haas[1], Yannic Lieder[1], Eike Niehs[1], Michael Perk[1], Victoria Sack[1], and Christian Scheffer[1]

1 **Department of Computer Science, TU Braunschweig, Germany**
{s.fekete, a.haas, y.lieder, e.niehs, m.perk, v.sack, c.scheffer}@tu-bs.de

Abstract

We present a study on the practical nature of the NP-hard problem of finding a Minimum Weight Triangulation (MWT) of a planar point set: Can we deliberately construct *practically* difficult instances? This requires identifying point sets for which all of a number of previously developed exact and heuristic methods *simultaneously* encounter a combination of pitfalls. We show that for instances of medium size, this seems unlikely, implying that one of several alternative methods may offer a path to an optimal solution. This complements recent work on the practical performance of these heuristic methods for specific classes of large benchmark instances, indicating that MWT problems may indeed be practically easier to solve than implied by its NP-hard complexity.

1 Introduction

The complexity of finding a minimum-weight triangulation (MWT) of a planar point set was a famous open problem for 27 years [8], until Mulzer and Rote [16] gave an NP-hardness proof, based in intricately constructed gadgets of considerable size.

While this shows that finding an MWT is difficult in a well-defined, theoretical sense, it does not necessarily imply that the problem is also intractable for instances of practically relevant size. In recent work, Haas [13] was able to extend, refine and streamline a number of previous ideas to compute provably optimal solutions for point sets of up to $30,000,000$ uniformly distributed points and real-world benchmark instances with up to $744,710$ points. This suggests that the MWT may indeed be much simpler than indicated by its theoretical complexity, at least for standard classes of instances.

We present a complimentary study on the practical nature of the theoretical hardness: Can we deliberately construct practically difficult instances of the MWT problem? This requires identifying point sets for which the previously developed methods *simultaneously* encounter a number pitfalls. We show that for instances of medium size, this seems unlikely, implying that one of several alternative methods may always provide a path to an optimal solution.

1.1 Related Work

There are efficient algorithms for computing optimal MWT solutions for special classes of instances. Independently, Gilbert [9] and Klincsek [15] showed that for simple polygons, the MWT problem can be solved in $O(n^3)$ time with dynamic programming. This can be generalized to polygons with k inner points. Hoffmann and Okamoto [14] showed how to obtain the MWT of such a point set in $O(6^k n^5 \log n)$ time. Grantson et al. [11] improved the algorithm to $O(n^4 4^k k)$ and showed another $O(n^3 k! k)$-time decomposition strategy [12].

For general instances, there are polynomial-time heuristics for including or excluding edges with certain properties from an MWT. Das and Joseph [4] showed that every edge in an MWT has the *diamond property*: For a point set S, an edge e cannot be in its minimum weight triangulation $\mathrm{MWT}(S)$ if both of the two isosceles triangles with base e and base

36th European Workshop on Computational Geometry, Würzburg, Germany, March 16–18, 2020.
This is an extended abstract of a presentation given at EuroCG'20. It has been made public for the benefit of the community and should be considered a preprint rather than a formally reviewed paper. Thus, this work is expected to appear eventually in more final form at a conference with formal proceedings and/or in a journal.

angle $\pi/8$ contain other points of S. Drysdale et al. [7] improved the angle to $\pi/4.6$. This property can exclude large portions of the edge set and works exceedingly well on uniformly distributed point sets, for which only an expected number of $O(n)$ edges remain. Dickerson et al. [5, 6] proposed the *LMT-skeleton heuristic*, which is based on a simple local-minimality criterion fulfilled by every edge in MWT(S). The LMT-skeleton algorithm often yields a connected graph, such that the remaining polygonal faces can be triangulated with dynamic programming to obtain the minimum weight triangulation.

The combination of the diamond property and the LMT-skeleton made it possible to compute the MWT for large, well-behaved point sets. Beirouti and Snoeyink [1] showed an efficient implementation of these two heuristics and reported that their implementation could compute the exact MWT of 40,000 uniformly distributed points in less than 5 minutes and even up to 80,000 points with the improved diamond property.

In more recent work, Haas [13] refined a number of these ideas. Based on a variety of improvements and additional data structures, he could compute provably optimal solutions for instances with up to 30,000,000 uniformly distributed points in less than 4 minutes on commodity hardware; the limiting factor turned out to be memory, not runtime. He achieved the same performance for normally distributed point sets, as well real-world benchmark instances from the TSPLIB [18] and the VLSI library of size up to 744,710 points. This shows that a wide range of huge MWT instances can be solved to provable optimality with the right combination of theoretical insight and algorithm engineering.

1.2 Our Results

We conduct a study of the practical difficulty of arbitrary MWT instances. In addition to the proven methods based on diamond property and LMT-skeleton, we present an integer program that strengthens Haas' toolbox by providing a practically useful alternative for determining optimal triangulation edges in unresolved faces. We also show that with this extended set of methods, any considered instance with up to 300 points can be solved to provable optimality within short time, even point sets deliberately constructed to be difficult.

2 Tools

Solving MWT instances to provably optimality relies on a number of different tools, which we briefly sketch in the following. The cited *Diamond Property* filters out a set of only $O(n)$ edges that *may* be in an MWT. The mentioned *LMT Skeleton* consists of a (possibly large) set of edges that *must* be contained in an MWT, but may still leave a number of untriangulated faces; see Figure 1. These faces can be triangulated with different versions of Dynamic Programming (Section 2.1) or with Integer Programming (Section 2.2).

2.1 Dynamic Programming (DP)

Empty faces of the LMT-skeleton can be triangulated using a dynamic programming approach for simple polygons, in time $\mathcal{O}(n^3)$ for an empty face with $n \in \mathbb{N}$ boundary vertices [10, 15]. For faces containing inner points, one of the following dynamic programming approaches can be used: A non-empty face with $n \in \mathbb{N}$ boundary vertices and $k \in \mathbb{N}$ inner points can be triangulated in $O(n^3 k! k)$ [12] or a non-empty face with $k \in \mathbb{N}$ connected components (resulting from the LMT-skeleton) can be triangulated in $O(n^{k+2})$ [19], respectively.

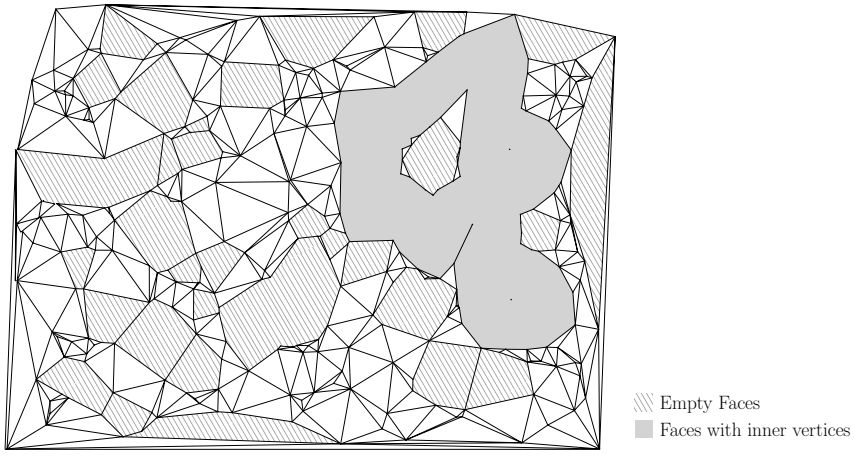

Figure 1 The LMT-skeleton (a subset of MWT(S)) of a point set S may contain untriangulated faces. These can be empty (hatched) or contain points and edges of the LMT-skeleton (filled).

2.2 Integer Programming

Another approach to compute the MWT of the remaining faces of the LMT-skeleton makes use of the following integer program (IP); see Yousefi and Young [20] as well as Dantzig, Hoffman and Hu [3] for related work. The objective function minimizes the sum of the perimeters $||\triangle||$ of all triangles, used in the triangulation. The variables $x_\triangle \in \{0,1\}$ indicate whether $\triangle$ is used in the triangulation. (Note that this description is slightly simplified because of limited space; a practically complete description provides additional adjustments for fixed edges along face boundaries.)

$$\min_{x_\triangle} \quad \sum_{\triangle} ||\triangle|| \cdot x_\triangle \tag{1}$$

$$\text{s.t.} \quad \sum_{\triangle \in \delta(e)} x_\triangle = 1 \qquad \forall e \in \text{boundary component} \tag{2}$$

$$\sum_{\triangle \in \delta^+(e)} x_\triangle = 1 \qquad \forall e \in \text{antennas} \tag{3}$$

$$\sum_{\triangle \in \delta^-(e)} x_\triangle = 1 \qquad \forall e \in \text{antennas} \tag{4}$$

$$\sum_{\triangle \in \delta^+(e)} x_\triangle - \sum_{\triangle \in \delta^-(e)} x_\triangle = 0 \qquad \forall e \in \text{inner edges} \tag{5}$$

$$x_\triangle \in \{0,1\} \tag{6}$$

Given a non-triangulated face, we distinguish three kinds of edges. *Boundary edges* lie on the outer face boundary or inner hole boundaries. Boundary edges are part of exactly one triangle in the face (Equation (2)). *Antenna edges* have the same face on both of its sides (see filled face in Figure 1), so they are part of two triangles in the face (Equation (3) and (4)). The third kind are *inner edges*, i.e., all remaining edges inside a face that are not fixed by the LMT-skeleton and not excluded by the diamond property. For these edges, the difference of the number of triangles on their left side equals the number of triangles on their right side (Equation (5)). The first three constraints imply either zero or one triangle on each side.

Equation 2 and Equation 5 are sufficient to solve the IP for a given polygon, with the boundary edges containing both the outer boundary and hole boundaries. Equation 3 and Equation 4 are auxiliary constraints, fixing edges of the LMT skeleton that have the same face on both sides.

3 Hard Instances

While previous studies on large classes of *specific* MWT instances showed that even huge instances can be solved optimally, this does not imply that there are *no* practically hard ones. For any NP-hard problem, natural candidates for such instances are the ones constructed in an NP-hardness reduction. However, the intricate constructions in the seminal proof by Mulzer and Rote [16] produce instances of tremendous size: While the clause gadgets have dimensions of $250,000 \times 250,000$, the connector gadgets (representing variable-clause incidences in a planar embedding of a 3SAT instance) require 14 points per subsegment of a length less than 28. As a consequence, representing even a Planar 1-in-3SAT instance with a handful of clauses (and thus, three handfuls of connector gadgets) easily requires millions of points. Given that "... modern SAT solvers can often handle problems with millions of constraints [i.e., clauses] and hundreds of thousands of variables" [17], it is clear that insufficient memory becomes a limiting factor long before the algorithmic difficulty of 3SAT.

This motivates the complimentary question to the results of [13]: Can we deliberately construct practically difficult instances of moderate size? It follows from the availability of the tools described in the previous section that such an instance must satisfy the following three properties.

1. It contains at least one complex face; otherwise it can be solved in $O(n^3)$.
2. The complex face must contain a relatively large number of connected components; otherwise, it can be solved in polynomial time with Dynamic Programming.
3. The Linear Programming relaxation of the IP for a face must yield a fractional optimal solution; otherwise, the IP is easy to solve.

We have employed a number of systematic methods to generate such instances. Figure 2 illustrates the workflow of an evolutionary strategy and a local perturbation algorithm. An example of how this leads to more complex instances can be seen in Figure 3.

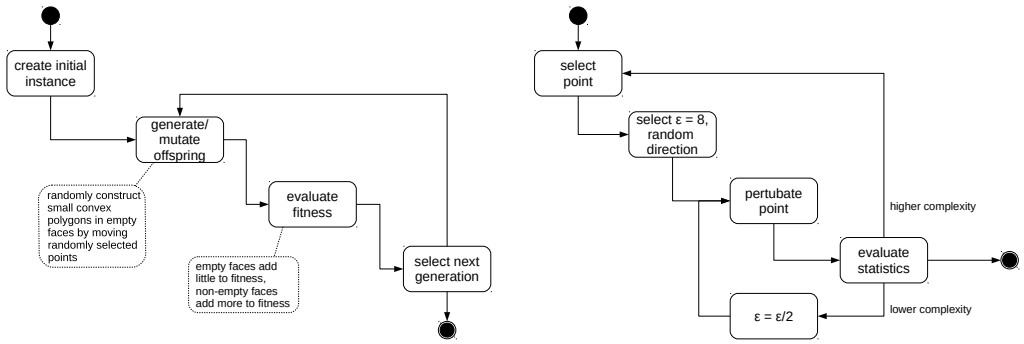

Figure 2 Modifying a point set to produce more complex LMT faces: **(Left)** Evolutionary strategy. **(Right)** Local Perturbation.

There are known classes of instances with a complex face of the LMT-skeleton that contain many connected components; see Figure 4. On the other hand, there are also known classes of

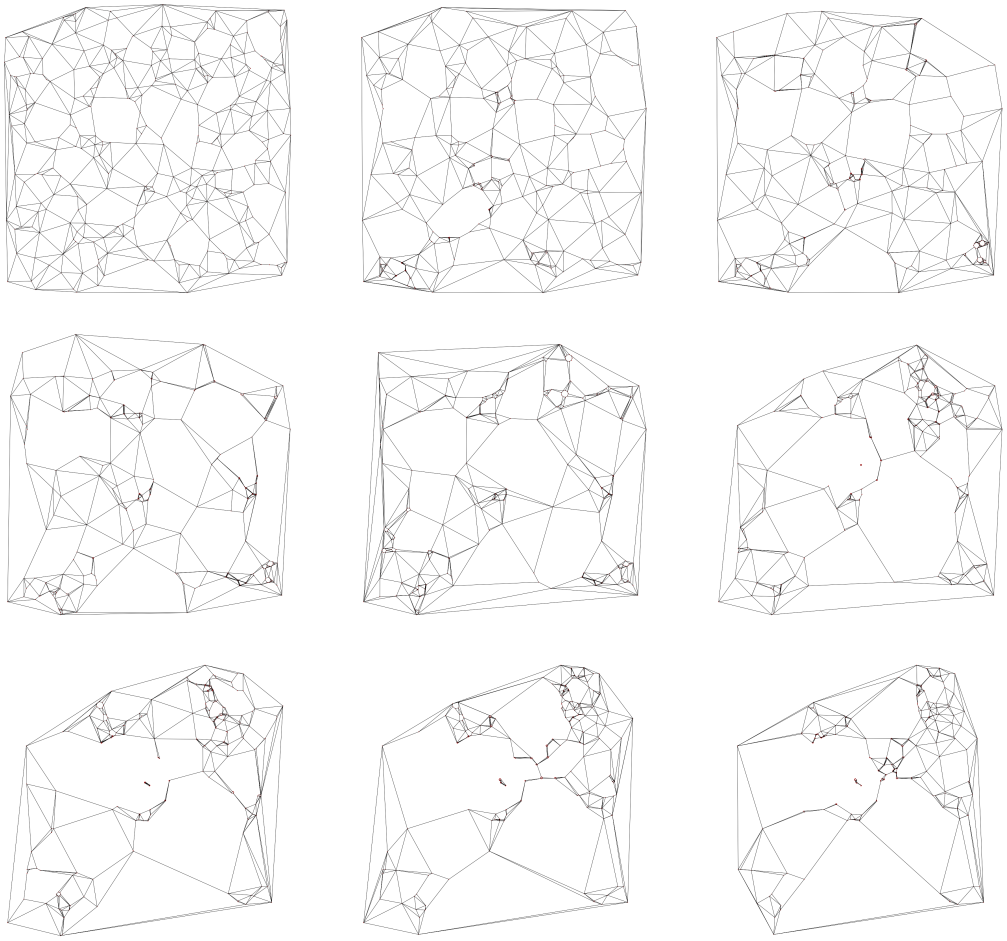

Figure 3 Evolving a point set to produce more complex LMT faces.

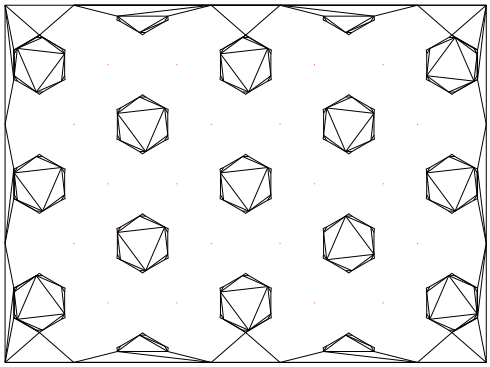

Figure 4 Instances for which the LMT-skeleton has a complex face with many connected components. Adapted from Belleville et al. [2].

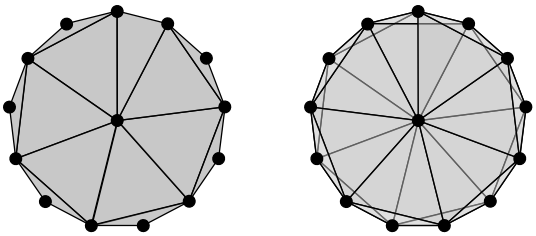

Figure 5 Instances for which the IP has fractional solutions. Adapted from Yousefi and Young [20].

instances with one complex face of the LMT-skeleton that produce fractional solutions when handled by the described integer program; see Figure 5. However, these instances are of quite different nature, so it is not clear that they can be combined for instances of reasonable size.

4 Experimental Results

We investigated the practical solvability of MWT instances, with a focus on constructing hard instances. All experiments were executed with CPLEX 12.9 on an Intel(R) Core(TM) i7-6700K CPU 4.00GHz with 4 cores and 8 threads utilizing an L3 Cache with 8MB, and a maximum amount of 64GB RAM. With the evolutionary strategy shown in Figure 2, we were able to generate many instances that contain at least one complex face. In order to generate the variables of the integer program (i.e. possible empty triangles within the complex face), we used a heuristic that performs well in practical scenarios. The heuristic does not guarantee to find empty triangles in every case. Therefore, we added a callback to the integer programs that verifies that all triangles of an integer solution are empty. In the upcoming figures, the optimization and verification time (callback time) are separated. We chose the instance size to be 306 points, which is comparable to the one shown in Figure 4. The goal was to produce complex faces that require large computational effort during the optimization of the integer program. After generating an instance with a complex face, we applied random local perturbations with respect to certain properties. The generation process was executed for several days, producing around $17,000$ instances.

We first studied the size of the complex boundary, which includes all edges on the boundary of the complex face, as well as hole boundary edges and antennas. Figure 6 (Left)

shows that the number of variables of the IP increases linearly with the size of the complex boundary. Moreover, the time to solve these instances (see Figure 6 (Right)) increases from 0.01 seconds to 0.25 seconds for complex faces with a boundary size of 300 edges.

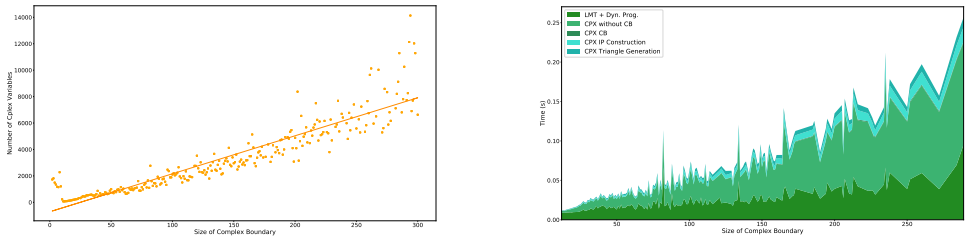

Figure 6 Results on instances that were generated with a large complex boundary size. **(Left)** Number of IP variables as a function of the size of the complex boundary. **(Right)** Runtime of the integer program. *LMT + Dyn. Prog.* refers to the construction of the LMT skeleton and triangulation of the empty faces. *CPX Triangle Generation* and *CPX IP Construction* represents the time that was necessary to generate the variables and constraints of the IP. *CPX (without) CB* refers to the runtime of the optimization and the empty triangle verification.

Next we investigated the number of complex faces in an instance. As shown in Figure 7 (Left), the number of IP variables for empty triangles grows linearly in the number of components. Figure 7 (Right) shows that runtimes for instances with < 5 complex components differ only by 0.1 seconds compared to instances with > 30 components.

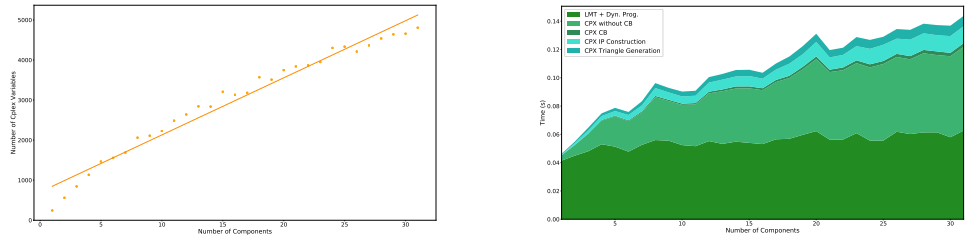

Figure 7 Results on instances that were generated with a large number of components. **(Left)** Number of IP variables as a function of the number of components. **(Right)** Runtime of the integer program. *LMT + Dyn. Prog.* refers to the construction of the LMT skeleton and triangulation of the empty faces. *CPX Triangle Generation* and *CPX IP Construction* represents the time that was necessary to generate the variables and constraints of the IP. *CPX (without) CB* refers to the runtime of the optimization and the empty triangle verification.

Despite the extensive length of the search, no instances with larger complex boundary sizes or more complex components were found. Therefore, we extended the instance from Figure 4 to produce instances with arbitrary numbers of complex boundary edges and connected components. Increasing the complex boundary size to more than $13,000$ edges showed that the runtime of the algorithm increases quadratically, see Figure 8. Further investigation showed that the optimal solution of the LP relaxation of the integer program for the produced instances was integral. Thus, only two of the three necessary malicious properties from the previous section could be established at once, so all instances could be

solved to provable optimality. In particular, only relatively degenerate instances similar to the one from Figure 5 seem to produce complex faces with non-integer LP solutions.

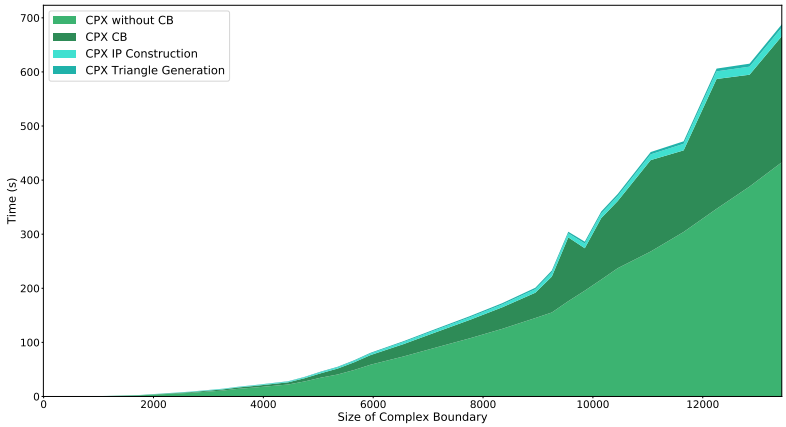

Figure 8 Runtime of the integer program for extensions of the instance in Figure 4. Note the moderate runtime despite the size: The largest IPs have more than 6,000,000 variables. *LMT + Dyn. Prog.* refers to the construction of the LMT skeleton and triangulation of the empty faces. *CPX Triangle Generation* and *CPX IP Construction* represents the time that was necessary to generate the variables and constraints of the IP. *CPX (without) CB* refers to the runtime of the optimization and the empty triangle verification.

5 Conclusions

Our systematic study for constructing practically difficult MWT instances showed that medium-sized point sets that simultaneously have three malicious properties seem hard to come by. This provides further evidence to the observation that the MWT problem is practically easier to solve than indicated by its theoretical complexity, as it shows that this practical solvability does not only depend on benign properties of special classes of instances, but remains intact even when we try to make instances deliberately difficult.

References

1 Ronald Beirouti and Jack Snoeyink. Implementations of the LMT Heuristic for Minimum Weight Triangulation. In *Proc. Symposium on Computational Geometry (SoCG)*, pages 96–105, 1998.

2 Patrice Belleville, Mark Keil, Michael McAllister, and Jack Snoeyink. On computing edges that are in all minimum-weight triangulations. In *Proc. Symposium on Computational Geometry (SoCG)*, pages V7–V8, 1996.

3 George B. Dantzig, Alan J. Hoffmann, and T.C. Hu. Triangulations (tilings) and certain block matrices. *Mathematical Programming*, 31:1–14, 1985.

4 Gautam Das and Deborah Joseph. Which triangulations approximate the complete graph? In *Proc. International Symposium on Optimal Algorithms*, pages 168–192, 1989.

5 Matthew Dickerson, J. Mark Keil, and Mark H. Montague. A Large Subgraph of the Minimum Weight Triangulation. *Discrete & Computational Geometry*, 18(3):289–304, 1997.

6 Matthew Dickerson and Mark H. Montague. A (Usually?) Connected Subgraph of the Minimum Weight Triangulation. In *Proc. Symposium on Computational Geometry (SoCG)*, pages 204–213, 1996.

7 Robert L. Scot Drysdale, Scott A. McElfresh, and Jack Snoeyink. On exclusion regions for optimal triangulations. *Discrete Applied Mathematics*, 109(1-2):49–65, 2001.

8 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

9 P. D. Gilbert. New results in planar triangulations. Master's thesis, University Illinois, 1979.

10 Peter D. Gilbert. New results on planar triangulations. Technical report, Illinois University at Urbana-Champaign, 1979.

11 Magdalene Grantson, Christian Borgelt, and Christos Levcopoulos. A Fixed Parameter Algorithm for Minimum Weight Triangulation: Analysis and Experiments. Technical Report LU-CS-TR: 2005-234, Lund University, Sweden, 2005.

12 Magdalene Grantson, Christian Borgelt, and Christos Levcopoulos. Minimum Weight Triangulation by Cutting Out Triangles. In *Proc. International Symposium on Algorithms and Computationa (ISAAC)*, pages 984–994, 2005.

13 Andreas Haas. Solving large-scale minimum-weight triangulation instances to provable optimality. In *Proc. Symposium on Computational Geometry (SoCG)*, pages 44:1–44:14, 2018.

14 Michael Hoffmann and Yoshio Okamoto. The minimum weight triangulation problem with few inner points. *Computational Geometry*, 34(3):149–158, 2006.

15 G.T. Klincsek. Minimal Triangulations of Polygonal Domains. *Annals of Discrete Mathematics*, 9:121–123, 1980.

16 Wolfgang Mulzer and Günter Rote. Minimum-weight triangulation is NP-hard. *Journal of the ACM*, 55(2):11, 2008.

17 Olga Ohrimenko, Peter J. Stuckey, and Michael Codish. Propagation = lazy clause generation. In Christian Bessière, editor, *Principles and Practice of Constraint Programming – CP 2007*, pages 544–558, 2007.

18 G. Reinelt. TSPLIB–A Traveling Salesman Problem Library. *ORSA Journal of Computing*, 3(4):376–384, 1991.

19 Chiu-fai Tsang. *Expected Case Analysis of [beta]-skeletons with Applications to the Construction of Minimum-weight Triangulations*. PhD thesis, Hong Kong University of Science and Technology, 1995.

20 Arman Yousefi and Neal E. Young. On a linear program for minimum-weight triangulation. *SIAM Journal on Computing*, 43(1):25–51, 2014.