On Computing Edges That Are In All Minimum-Weight Triangulations*

Patrice Belleville
CS UBC Vancouver
patrice@cs.ubc.ca

Mark Keil
CS U. Saskatchewan
keil@cs.usask.ca

Michael McAllister
CS UBC Vancouver
mcallist@cs.ubc.ca

Jack Snoeyink
CS UBC Vancouver
snoeyink@cs.ubc.ca

1 Introduction

Given a set P of n points in the plane, we say that a triangle τ is *empty* if the intersection $\tau \cap P$ contains only the three vertices of τ. A *triangulation* of P is a set of empty triangles with disjoint interiors whose union covers the convex hull of P. A set P usually admits many triangulations; a *minimum-weight triangulation* (MWT) is one in which the sum of edge lengths is minimum. No general, polynomial-time algorithm for MWTs is known.

Keil [4] and Dickerson [3] have independently suggested an approach to identify edges that are *possible* in some MWT and *certain* in any MWT. Start with all edges that join two points in P. Edges of the convex hull of P are certain in any MWT. Any possible non-hull edge e must be incident on two empty triangles whose edges are also possible; if these triangles form a convex quadrilateral, then e must be the shorter diagonal (otherwise, a diagonal swap would reduce total edge length). We call such a pair of triangles a *certificate* for e.

Edges with no certificate are not in any MWT, neither can they take part in any certificate. On the other hand, if an edge e is possible (has a certificate) and crosses no possible edge, then e is *certain* to be in every MWT. This video shows our implementation of an algorithm that searches for a certificate for every edge and the resulting *possible* and *certain* edges for some examples.

2 Implementation

To search for a certificate for an edge e, we examine all pairs of empty triangles, one on each side of e. If no certificate is found, then e is impossible and any certificate that used e must be discarded. Our implementation identifies possible and certain edges in $O(n^2)$ space and $O(n^4)$ worst-case time. On an SGI Indy, 250 points take 25 seconds and 1000 take half an hour; observed time is proportional to $n^3 \log n$.

Around each point, we sort the points radially. To scan empty triangles left of edge e, suppose that the k points left of e appear in ccw order $p_1, p_2, \ldots, p_k$ around the first endpoint of e. Around the second endpoint they appear in order $p_{\sigma(1)}, p_{\sigma(2)}, \ldots, p_{\sigma(k)}$ for some permutation σ. Each maximum of a prefix of σ corresponds to an empty triangle; all empty triangles can be found by scanning both orders in $O(n)$ time. (If σ is a random permutation, then we expect $O(\log n)$ prefix maxima, which could explain our implementation's observed running time.)

For each side of each edge, therefore, storing two pointers into radially-sorted lists indicates the empty triangle being scanned. If we encounter a potential certificate, we stop scanning; if an edge of the certificate is later found to be impossible, we can resume. Inspecting all pairs of empty triangles adjacent to e takes $O(n^2)$ worst-case time. The constants in the scan are small and there is no need to store a potentially cubic number of triangles as in [3]. If an edge e is found to have no certificate, then we can check whether the adjacent empty triangles participate in any certificates that are now invalid. To speed future scans, we delete impossible edges from the radially-sorted lists. At the end, a final scan identifies the certain edges.

Input points are scaled to 20 bit positive integers so that all of the arithmetic can be performed correctly in 53-bit double precision. Colinearities and equal lengths are handled.

3 Examples

Both Keil and Dickerson observed that for small examples the certain edges formed a connected graph, if not a triangulation. A weaker conjecture would be that the vertices and certain edges bound polyg-

*Supported in part by NSERC, the Killam Foundation, and the B.C. Advanced Systems Institute

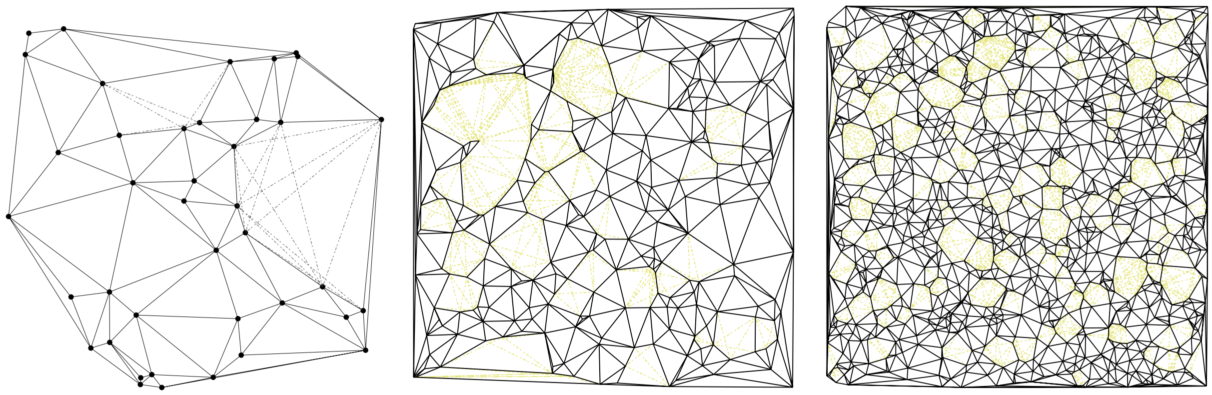

Figure 1: Examples with 40, 250 and 1000 points

onal regions that have a constant number of holes. If this were true, then the MWT computation could be completed in polynomial time by dynamic programming [5]. Using this implementation on its own and as a user macro in IPE has allowed us to construct point sets where the polygonal regions have linearly many holes.

First, note that by placing points on a circle so that any $60°$ sector contains at least three points, we can isolate a point in the center. Bose, Devroye, and Evans [2] show that this configuration occurs with a small, but constant, probability, so one can expect that the graph of certain edges for uniformly distributed points has a linear number of connected components. Since their constant of proportionality is less than 10^{-51}, one need not expect to observe this behavior on the screen.

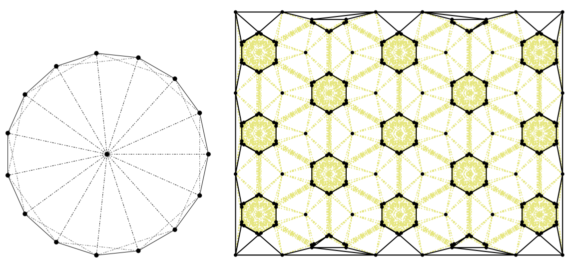

Figure 2: The complement of the certain edges can have many holes.

One can place this wheel configuration at the vertices of a hexagonal lattice and adjust it so that the graph of the certain edges consists of isolated vertices and isolated 18-gons (which look like hexagons in figure 2, but each side contain two points as well). This construction can tile the plane, forming a polygonal region with $2n/19 - o(n)$ holes.

4 Future research

Does knowledge of the possible and certain edges help in designing gadgets to show that MWT is NP-hard? Our attempts to do so have been frustrated by a natural extension of *light triangulations* [1] that use only possible edges. Or can other criteria reduce the number of possible edges?

Several optimizations could be applied to speed up the algorithm by constant factors. For example, we consider the edges from longest to shortest since long ones are unlikely to have certificates. If one could bound from above the length of the longest edge in an MWT, time and memory spent on longer edges could be recovered. Since memory is the bottleneck, we have not tried ideas that involve more storage per segment, such as precomputing all line equations. Starting with the subset of edges that miss a β-skeleton is also a good idea.

References

[1] O. Aichholzer, F. Aurenhammer, M. Taschwer, and G. Rote. Triangulations intersect nicely. In *Proc. 11th ACM SCG*, pages 220–229, 1995.

[2] P. Bose, L. Devroye, and W. Evans. Diamonds are not a minimum weight triangulation's best friend. Tech. Report 96–01, Dept. of CS, Univ. of British Columbia, Jan. 1996.

[3] M. Dickerson and M. Montague. The exact minimum weight triangulation. In *Proc. 12th ACM SCG*, this proceedings, 1996.

[4] M. Keil. Private communication, Jan. 1994.

[5] G. T. Klincsek. Minimal triangulations of polygonal domains. *Ann. Disc. Math.*, 9:121–123, 1980.