Optimization for First Order Delaunay Triangulations *

Marc van Kreveld, Maarten Löffler, and Rodrigo I. Silveira

Department of Information and Computing Sciences,
Utrecht University, 3508TB Utrecht, the Netherlands,
{marc,loffler,rodrigo}@cs.uu.nl

Abstract. This paper discusses optimization of quality measures over first order Delaunay triangulations. Unlike most previous work, our measures relate to edge-adjacent or vertex-adjacent triangles instead of only to single triangles. We give efficient algorithms to optimize certain measures, whereas other measures are shown to be NP-hard. For two of the NP-hard maximization problems we provide for any constant $\varepsilon > 0$, factor $(1-\varepsilon)$ approximation algorithms that run in $2^{O(1/\varepsilon)} \cdot n$ and $2^{O(1/\varepsilon^2)} \cdot n$ time (when the Delaunay triangulation is given). For a third NP-hard problem the NP-hardness proof provides an inapproximability result. Our results are presented for the class of first-order Delaunay triangulations, but also apply to triangulations where every triangle has at most one flippable edge. One of the approximation results is also extended to k-th order Delaunay triangulations.

1 Introduction

Triangulation is a well-studied topic in computational geometry. The input is a point set or planar straight line graph in the plane, and the objective is to generate a subdivision where all faces are triangles, except for the outer face. In some cases extra points are allowed, in which case we speak of a Steiner triangulation. Since a point set (or planar straight line graph) allows many different triangulations, one can try to compute one that optimizes a criterion. For example, one could maximize the minimum angle used in any triangle, or minimize the total edge length (minimum weight triangulation). The former optimization is solved with the Delaunay triangulation in $O(n \log n)$ time for n points. The latter optimization is NP-hard [18].

Several other optimization measures exist. In finite element methods, triangular meshes with various quality constraints are used, and Steiner points may be used to achieve this; Bern and Plassmann [4] give a survey. Other optimization measures arise if the triangulation represents a terrain (called a polyhedral terrain in computational geometry): all vertices have a specified height, and the height of points on edges and on triangles is obtained by linear interpolation. Such a terrain representation is common in GIS and is called a TIN [6, 22].

* This research has been partially funded by the Netherlands Organisation for Scientific Research (NWO) under FOCUS/BRICKS grant number 642.065.503 (GADGET) and under the project GOGO.

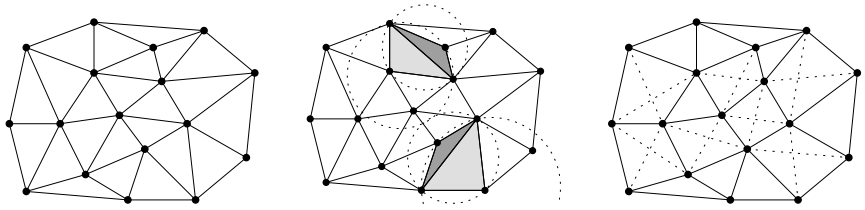


Fig. 1. (a) Delaunay triangulation (zero-th order). (b) Second order Delaunay triangulation (light grey triangles are first order, darker triangles are second order). (c) For first order Delaunay triangulations, one of every pair of dotted edges must be chosen.

Bern et al. [3] show that measures like maxmin triangle height, minmax slope, and minmax eccentricity of any triangle can be optimized with a technique called *edge insertion* in $O(n^3)$ or $O(n^2 \log n)$ time. Other measures such as minmax angle [9] and minmax edge length can also be optimized in polynomial time [8]. Interestingly, the Delaunay triangulation optimizes several measures simultaneously: maxmin angle, minmax circumscribed circle, minmax enclosing circle, and minimum integral of the gradient squared (e.g. [3]). For terrain modeling in GIS, Steiner points cannot be used because their elevation would not be known. Terrain modeling leads to a number of optimization criteria, both to yield good rendering of the terrain for visualization, and to make it suitable for modeling processes like water runoff and erosion [13, 17]. Slope characteristics are especially important. Furthermore, local minima and artificial dams, which may be artifacts due to the creation of the triangulation, should be avoided [7, 14, 21].

The *Delaunay triangulation* of a set P of points is defined as the triangulation where all vertices are points of P and the circumcircle of the three vertices of any triangle does not contain any other point of P. If no four points of P are cocircular, then the Delaunay triangulation is uniquely defined. Gudmundsson et al. [10] generalize this to *higher order Delaunay triangulations*. A triangulation is *k-th order Delaunay* if the circumcircle of the three vertices of any triangle contains at most k other points (see Figure 1). For higher order Delaunay triangulations, fewer results are known. Minimizing local minima in a terrain becomes NP-hard for orders higher than n^ε, for constant ε. Experiments showed that low order Delaunay triangulations can reduce the number of local minima significantly [7].

First order Delaunay triangulations have a special structure. All edges that are in any first order Delaunay triangulation form a subdivision that only has triangles and convex quadrilaterals (see Figure 1(c)). In the quadrilaterals, both diagonals are possible to obtain a first order Delaunay triangulation. We call these quadrilaterals and diagonals *flippable*. Due to this structure, measures like the number of local minima or extrema can be minimized in $O(n \log n)$ time. The same holds for minimizing the maximum area triangle, minimizing the total edge length, and various other measures [10]. On the other hand, minimizing the maximum vertex degree was only approximated by a factor of roughly 3/2.

Table 1. Optimization problems and complexity results for first order Delaunay triangulations. d is the maximum vertex degree in the Delaunay triangulation.

Triangles incident to	Opt. worst local measure (minmax)	Result	Opt. # occurrences	Result
edge	area ratio	$O(n \log n)$	max #convex edges	NP-hard
	angle of outward normals	$O(n \log n)$		
vertex	area ratio	$O(nd \log n)$	max #convex vertices	$O(n \log n)$
	angle of outward normals	$O(nd \log n)$	min #local minima	$O(n \log n)$ [10]
	vertex degree	NP-hard	min #mixed vertices	NP-hard

Many of the measures mentioned above are measures for single triangles. Exceptions are total edge length, number of local minima or extrema, and maximum vertex degree. In this paper, we consider measures that depend on pairs of triangles that are edge-adjacent, and measures that depend on groups of triangles that are vertex-adjacent. Note that a single flip in a first order Delaunay triangulation influences five pairs of edge-adjacent triangles and four vertex-adjacent groups. We consider objectives of the maxmin or minmax type, and objectives where the number of undesirable situations must be minimized. Examples of minmax objectives for edge-adjacent triangles include minimizing the maximum ratio of edge-adjacent triangle areas, which is relevant for numerical methods on meshes, or minimizing the maximum spatial angle of the normals of edge-adjacent triangles in polyhedral terrains, which is important for flow modeling. Geomorphologists classify parts of mountains or hills as footslopes, hillslopes, valley heads, etc. [13]. If we know that a part of a terrain is a valley head, we should maximize the number of convex edges or convex vertices in that part. A vertex of a polyhedral terrain is *convex* if there is a plane through that vertex such that all of its neighbors are on or below that plane, and at least one strictly below. A vertex is *mixed* if every plane containing it has neighbors strictly above and below the plane. We study maximization of convex edges, maximization of convex vertices, and minimization of mixed vertices.

Given a planar point set P with or without elevation, we study the complexity of optimizing measures over all first order Delaunay triangulations. Measures we consider and results are shown classified in Table 1. The optimization of other worst local measures for edge-adjacent triangles can also be solved in $O(n \log n)$ time with the same technique, like minimizing the largest minimum enclosing circle of any two edge-adjacent triangles. Our proof of NP-hardness of minimizing the maximum vertex degree justifies the factor 3/2 approximation algorithm given before in [10]. The proof yields inapproximability beyond a constant greater than 1 in polynomial time unless P=NP. It was already known that triangulating a biconnected planar graph while minimizing the maximum degree is NP-hard [15]. The NP-hard problems of maximizing convex edges and maximizing non-mixed vertices can be approximated within a factor $1 - \varepsilon$ in $2^{O(1/\varepsilon)} \cdot n$ and $2^{O(1/\varepsilon^2)} \cdot n$ time, if the Delaunay triangulation is given. The NP-hardness results show that, despite the simple structure of first order Delaunay triangulations, optimization of various measures is hard.

2 Exact algorithms

We start this section with a problem that turns out to be surprisingly easy to solve, namely, maximizing the number of convex vertices over all possible first order Delaunay triangulations. Let P be a set of n points in the plane, where each point has a height value. As observed before, if we take the Delaunay triangulation T of P, it has a number of edges that are in any first order Delaunay triangulation, and a number of flippable edges, and no two flippable edges bound the same Delaunay triangle [10]. The Delaunay triangulation and its flippable edges can be determined in $O(n \log n)$ time.

For any flippable quadrilateral, one diagonal is reflex and the other diagonal is convex in 3-dimensional space, unless the four vertices of the quadrilateral are co-planar. Consider a convex vertex v in T. If it is incident to a flippable quadrilateral where the convex diagonal is present, then v will remain convex if we use the reflex diagonal instead (regardless of which diagonal is incident to v). In other words: using only reflex edges in flippable quadrilaterals does not cause any vertex to become non-convex. At the same time, it may turn non-convex vertices into convex ones. It follows that the maximization problem on the given point set P can be solved in $O(n \log n)$ time.

2.1 Measures on edge-adjacent triangles

In this section we show how to optimize a measure function M defined for a triangulation T, over all first order Delaunay triangulations of P. The function M should be of the shape $M(T) = \max_{q \in T} \mu(q)$ for q a (not necessarily flippable) quadrilateral, and we wish to minimize $M(T)$ over all first order Delaunay triangulations T. We also use $\mu(e)$ for any edge e in a triangulation to denote $\mu(q)$, where e is the diagonal of q. A first order Delaunay triangulation has four types of edges: between two fixed triangles, between a fixed triangle and a flippable quadrilateral, between two flippable quadrilaterals, and flippable edges. As a consequence, there are only $O(n)$ possible values for $M(T)$, and we can determine and sort them in $O(n \log n)$ time.

We solve the $\min M(T)$ problem by transforming it into a series of 2-SAT instances. We will use 2-SAT to answer the following question: Is there a first order Delaunay triangulation T such that $M(T) \leq \mu_0$? Since there are $O(n)$ interesting values for μ_0, we can apply binary search to find the smallest one.

Let S be the subdivision that is the Delaunay triangulation of P with all flippable edges removed, and let μ_0 be given. For every edge e of S between a triangle and a quadrilateral, decide which of the two diagonals of the quadrilateral has $\mu(e) > \mu_0$. If neither does, then we can answer the question immediately with "no". If only one diagonal has $\mu(e) > \mu_0$, then we fix the other diagonal in S. Otherwise, we continue with the next edge between a triangle and a quadrilateral. This step may have made flippable quadrilaterals into two fixed triangles in S. Next we test the possible diagonals of each quadrilateral of S. If both diagonals give $\mu(.) > \mu_0$, then we answer with "no" again. If only one diagonal gives $\mu(.) > \mu_0$, then we fix the other diagonal to make two new triangles in S.

Next we test all edges of S between adjacent triangles. If any such edge does not satisfy $\mu(e) > \mu_0$, then we answer the question with "no" again.

It remains to solve the problem for edges between quadrilaterals of S. For every quadrilateral q we introduce a Boolean variable x_q, and let one diagonal choice represent TRUE and the other FALSE. Let e be an edge of S between two quadrilaterals q and r. For each choice of diagonals in q and r that gives $\mu(e) > \mu_0$, for example the one with TRUE in q and FALSE in r, we make a clause $(\neg x_q \vee x_r)$. We get at most four clauses for any edge between two quadrilaterals, so $O(n)$ clauses overall. The conjunction of all clauses is a 2-SAT instance, which we can solve in linear time with the algorithm of Aspvall et al. [1]. The binary search must try $O(\log n)$ values for μ_0 until we find the one minimizing $M(T)$. Hence, the whole algorithm takes $O(n \log n)$ time.

2.2 Measures on vertex-adjacent triangles

The algorithm described in the previous section can easily be extended to minimize measure functions of the form $M(T) = \max_{t,t' \in T} \mu(t, t')$ for t and t' triangles in T with a common vertex. The set of possible values of $M(T)$ induced by pairs of triangles incident to a vertex v is $\binom{d(v)}{2}$, where $d(v)$ denotes the degree of v. Since the sum of the degrees of all vertices is $O(n)$, the total number of possible values of $M(T)$ is at most $\sum_{v \in T} d(v)^2 = d \cdot \sum_{v \in T} d(v) = O(dn)$, where d is the maximum degree of any vertex in the triangulation.

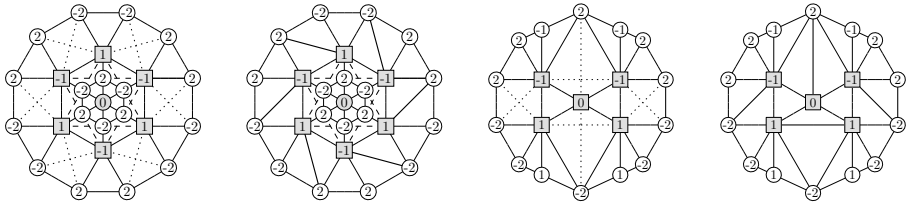
Theorem 1. *A first order Delaunay triangulation that minimizes the maximum area ratio of edge adjacent triangles can be computed in $O(n \log n)$ time. If the triangulation represents a polyhedral terrain, the same result holds for minimizing the maximum angle of outward normals. If we consider these ratio measures over pairs of vertex adjacent triangles, the algorithms take $O(nd \log n)$ time, where d is the maximum vertex degree in the Delaunay triangulation.*

3 NP-hardness results

We show NP-hardness for three different optimization problems on first order Delaunay triangulations. The proof for the first problem, minimization of the number of mixed vertices in a terrain, is treated in detail. The other two NP-hardness results are only stated; the proofs can be found in the full paper [23].

3.1 Mixed vertices

In a terrain, we call a vertex *mixed* if every plane through it has neighboring vertices above and below the plane. In some types of terrains, such vertices are uncommon, so we may want to minimize their number. Given a set of points with height information, we study the problem of constructing a first order Delaunay triangulation of this point set such that the number of mixed vertices is minimal. This problem is NP-hard. We prove this by reduction from planar 3-SAT [16].



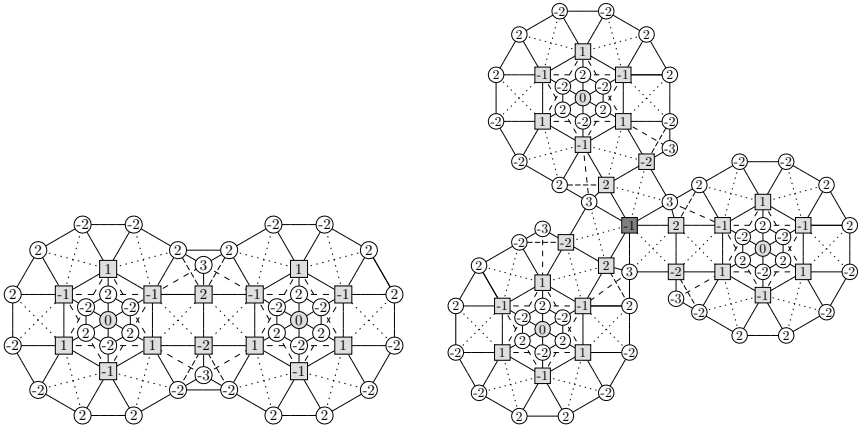
(a) (b) (c) (d)

Fig. 2. (a) A fan. (b) One of the solutions: a left-turning fan. Similarly a right-turning fan is possible. (c) An inverter gadget. (d) One of the two solutions.

We represent the variables of a 3-SAT instance by *fan gadgets*, see Figure 2(a). A fan gadget consists of 25 points with elevations. In the figure, all possible first order Delaunay edges are shown. Solid edges are in every first order Delaunay triangulation; dashed and dotted edges are flippable. The square nodes and the dotted edges are the most important part. We observe that a square vertex is mixed if and only if both incident dotted edges are in the triangulation.

We construct the gadget in such a way that the state of the round vertices does not depend on any of the dotted edges. The white round vertices are always non-mixed, even if all incident edges would be in the triangulation; the grey round vertices are always mixed, already if only the fixed edges are in the triangulation. Hence the number of mixed vertices is only affected by square vertices, and can only be minimal if there are never two dotted edges at the same square vertex. A fan gadget therefore has two possible states, see Figure 2(b).

We can connect fans together to form larger chains that are all in the same state, see Figure 3(a). We turn two more vertices into squares, and if the left



(a) (b)

Fig. 3. (a) Connecting variables. (b) Three variables come together in a clause.

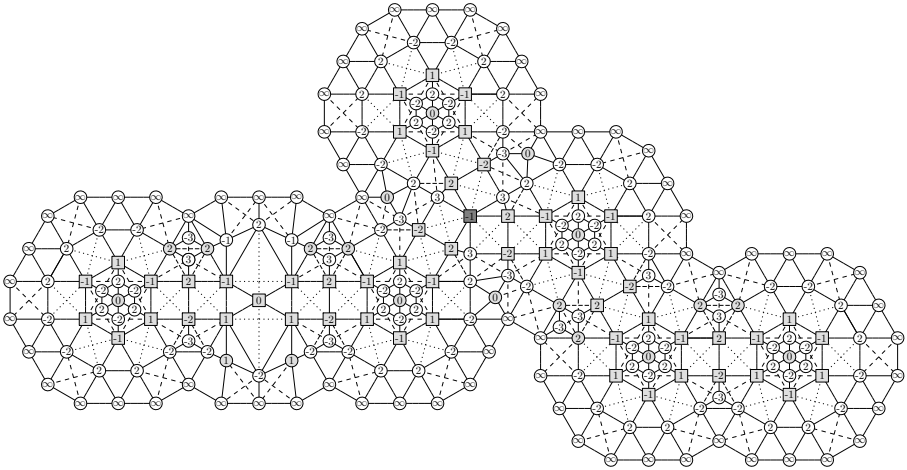


Fig. 4. A coating to shield the construction from the outside.

fan is left-turning, the right fan must also be left-turning and the other way around. We can connect up to three fans to an existing fan, so chains can also split. We also make negations in chains using the inverter gadget in Figure 2(c). Here, if the leftmost square has its positive sloping diagonal in the triangulation, the rightmost square must have its negative sloping diagonal and the other way around, see Figure 2(d). We use an inverter gadget in a chain to negate a variable.

We represent the clauses occurring in the 3-SAT instance by a special clause vertex, see Figure 3(b). Here three fan chains come together at one square vertex in a darker shade of grey. This vertex has a slightly different property than the other square vertices. A clause vertex is mixed if and only if all three incident dotted edges are in the triangulation.

So, the clause can be satisfied if at least one of the three fans is *not* right-turning, and by including inverters at the appropriate places this can represent any Boolean clause of a 3-SAT formula. With these gadgets we can build the whole planar 3-SAT instance. Finally, we need to triangulate the remaining gaps, so we need to ensure that the vertices on the boundary really have a fixed value. We add an extra layer of sufficiently high vertices (labeled ∞ in Figure 4). These vertices are all non-mixed, and the properties of the vertices that are not on the boundary can be checked locally.

Theorem 2. *Minimizing the number of mixed vertices over all first order Delaunay triangulations is NP-hard.*

3.2 Maximum vertex degree and convex edges

In the full paper [23] we also give NP-hardness proofs for minimizing the maximum vertex degree and maximizing the number of convex edges in a polyhedral terrain. The reductions are from planar 3-SAT and planar MAX-2-SAT [11].

Theorem 3. *The problems of minimizing the maximum vertex degree and maximizing the number of convex edges over all first order Delaunay triangulations are NP-hard.*

4 Approximation algorithms

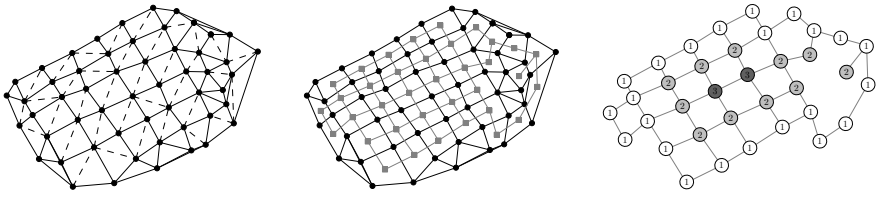
The problems of optimizing the number of convex edges or mixed vertices and minimizing the maximum vertex degree were shown NP-hard; hence it is of interest to develop approximation algorithms for them. For the last problem there is already a 1.5-approximation [10], and our NP-hardness proof shows that no polynomial time approximation scheme exists unless P=NP. For the other two problems we present polynomial time approximation schemes. We also sketch an extension to k-th order Delaunay triangulations for maximizing convex edges.

The general idea is as follows. First we transform the problem into a graph problem on some planar graph that can be obtained from the Delaunay triangulation without flippable edges. The resulting graph is partitioned into layers of outerplanarity at most λ. For each choice of i, where $0 \leq i < \lambda$, we delete every $(j\lambda + i)$-th layer of vertices, where $j = 0, 1, 2, \ldots$. The resulting "thick" layers are independent. For each thick layer, we compute a tree decomposition of width at most $3\lambda - 1$ and solve the problem optimally on this decomposition in $2^{O(\lambda)}n$ time, using dynamic programming. Finally, the union of the solutions of all the thick layers for a given i yields a solution to the original problem. We simply choose i such that the size of the solution is the maximal, and return the corresponding triangulation as output. Such an approach gives a $(1 - \varepsilon)$-approximation if λ is chosen suitably, depending on the problem and ε [2, 12].

4.1 Maximizing the number of convex edges

We build a graph G that has a vertex (called q-vertex) for each flippable quadrilateral, and an edge between two q-vertices if and only if their corresponding quadrilaterals share an edge. The rest of the input (all the fixed triangles) are not explicitly represented, see Figure 5(b). Each q-vertex has two possible states, *convex* or *reflex*, depending on the choice of the diagonal. It also has a value that depends on its state and represents the number of convex edges among the flippable edge and any edges that the quadrilateral shares with fixed triangles when the q-vertex is in that state (from 0 to 5). Furthermore, every edge in G has a value that depends on the states of both incident q-vertices. The goal of the algorithm is to find a state for each q-vertex such that the sum of the values (total number of convex edges) is maximized.

To create the independent thick layers from the graph we will remove the edges that connect two consecutive layers $j\lambda + i$ and $j\lambda + i + 1$ in G, where $j = 0, 1, 2, \ldots$, for all choices of $0 \leq i < \lambda$. The layers created after removing one set of layers of edges are independent, so if we optimize them separately and then join them by adding the removed edges, the number of convex edges after the join cannot decrease. Some edges are not considered for every i, but only in



(a) (b) (c)

Fig. 5. (a) Initial triangulation (solid edges are fixed). (b) Graph (in gray) where each vertex represents a flippable quadrilateral. (c) The same graph showing the outerplanarity layers.

$\lambda - 1$ out of λ solutions. We get a $(1-\varepsilon)$-approximation algorithm by taking $\lambda = \lceil \frac{1}{\varepsilon} \rceil$, due to the pigeonhole principle [2, 12].

Once we have the thick layers, each layer is solved optimally by using a tree decomposition approach. Since each layer is a λ-outerplanar graph, a tree decomposition with treewidth at most $3\lambda - 1$ can be computed in time linear in the number of nodes of the graph [5]. Once we have this decomposition we can apply one of the standard techniques to deal with problems on graphs of small treewidth. The technique consists of building tables of partial solutions in the nodes of the tree decomposition [5, 19].

Definition 1. *(from [19], originally in [20]) Let $G = (V, E)$ be a graph. A* tree decomposition *of G is a pair $\langle \{X_i | i \in I\}, T \rangle$ where each X_i is a subset of V, called a* bag*, and T is a tree with the elements of I as nodes. The following three properties must hold:*

1. $\bigcup_{i \in I} X_i = V$;
2. *for every edge $\{u, v\} \in E$, there is an $i \in I$ such that $\{u, v\} \subseteq X_i$; and*
3. *for all $i, j, k \in I$, if j is on the path between i and k in T then $X_i \cap X_k \subseteq X_j$.*

The width *of $\langle \{X_i | i \in I\}, T \rangle$ equals $\max\{|X_i| \mid i \in I\} - 1$. The* treewidth *of G is the minimum ω such that G has a tree decomposition of width ω.*

We will make T rooted by choosing any node to be the root. For each bag X_i, we will store a table A_i ($i \in I$). Tables will be created in a bottom up fashion as follows. For each bag X_i, the table A_i has 2^{n_i} rows and $n_i + 1$ columns, where $n_i = |X_i|$. Each row represents a *coloring*, which is an assignment of a state (*reflex*/*convex*) to each q-vertex (flippable quadrilateral) in X_i. All the different possible colorings for the bag are represented in the table. Furthermore, for each coloring C_j an extra value $m_i(C_j)$ is stored, containing the number of convex edges in an optimal triangulation of the point set induced by the subtree rooted at X_i that includes the current coloring as a subset. The details on how to compute these values are presented below.

Step 1: Table initialization. For every table A_i and each coloring C_j, we set $m_i(C_j)$ to the number of convex edges for that assignment: The sum of the values of each q-vertex (that will vary according to its state), plus 1 for each edge

with both incident q-vertices in X_i if their states define a convex edge between the corresponding quadrilaterals (with diagonals chosen).

Step 2: Table update. Next the tree is traversed, starting from the leaves, finishing at the root. For each node, the column m_i of A_i is updated based on its children. Let i be the parent of node j. Bags X_i and X_j have some q-vertices in common. We sort both tables first by the columns of the shared q-vertices, and second by m_i. Then we scan A_i row by row, and for each coloring C_l we update $m_i(C_l)$ based on the highest value that $m_j()$ has for that combination of the shared q-vertices. For later reconstruction of the triangulation we also store a pointer to the corresponding row in A_j. When a node X_i has several children, we update A_i against each child, one at a time, in the same way. Once the root node is updated, the number of convex edges in an optimal triangulation will be in the last column of one of the rows of its table. The final triangulation can be computed by following the pointers in the tables.

The correctness of the method follows from the definition and properties of tree decompositions, and the arguments are identical to the ones that hold for other well-known problems where the same technique has been used, such as vertex cover or dominating set (see [19]).

The running time is dominated by the computation and merging of the tables. The sorting of each table can be done in time $O(2^\omega\omega)$ (because all but one column have only two states). The time for updating a table based on another one is linear in the size of the largest one, so $O(2^\omega)$. The number of tables is linear in the number of nodes $|I|$ of tree T, hence the total running time is $O(2^\omega\omega \cdot |I|)$. Since the graph is λ-outerplanar we can compute a tree decomposition of width $\omega \leq 3\lambda - 1$ and $|I| = O(n)$ nodes [5, 19]. We apply this algorithm to the λ different values of i to get an approximation scheme, so the worst-case running time is $O(\lambda 2^\omega\omega \cdot |I|) = O(\lambda^2 8^\lambda \cdot n) = O(\frac{1}{\varepsilon^2} 8^{\frac{1}{\varepsilon}} \cdot n) = 2^{O(1/\varepsilon)} \cdot n$.

Theorem 4. *For any $\varepsilon > 0$, a $(1-\varepsilon)$-approximation algorithm for maximizing the number of convex edges over all first order Delaunay triangulations exists that takes $2^{O(1/\varepsilon)} \cdot n$ time (if the Delaunay triangulation is given).*

4.2 Maximizing the number of non-mixed vertices

Using a similar approach as above, we can also maximize the number of non-mixed vertices of a terrain. Because the mixed/non-mixed state of a vertex is determined by a large (possibly non-constant) number of neighboring quadrilaterals, several adaptations are needed. We now construct a graph with vertices for both the vertices and the quadrilaterals of the terrain. We remove the graph vertices that represent terrain vertices that have a fixed state, a high degree, or that can always be satisfied without disturbing the others. Of the remaining graph, we create λ-thick layers again, and we compute a tree decomposition of every layer, which we blow up such that every vertex contains all its neighbors in some bag. We solve the problem in each layer optimally by dynamic programming. More details are in the full paper [23]. We achieve:

Theorem 5. *For any $\varepsilon > 0$, a $(1-\varepsilon)$-approximation algorithm for maximizing the number of non-mixed vertices over all first order Delaunay triangulations exists that takes $2^{O(1/\varepsilon^2)} \cdot n$ time (if the Delaunay triangulation is given).*

4.3 Maximizing the number of convex edges, k-th order

The approximation algorithm for maximizing convex edges extends to k-th order Delaunay triangulations. To assure that every k-order Delaunay edge with its incident triangles is considered as a potentially convex edge in enough subproblems, we need layers with thickness proportional to k/ε. To use tree decompositions with bounded treewidth for maximizing convex edges, we also need to assure that the four vertices involved in two adjacent k-order Delaunay triangles appear in some bag of the tree decomposition. We show in the full paper [23]:

Lemma 1. *If $\langle \{X_i | i \in I\}, T \rangle$ is a tree decomposition of the Delaunay triangulation of a set of points with width ω, then a tree decomposition of width at most $2^{O(k)}\omega^2$ exists where every pair of adjacent k-th order Delaunay triangles appears in some bag.*

The number of states of a bag is exponential in the treewidth, and combining two bags takes time nearly linear in their number of states. This leads to:

Theorem 6. *For any $\varepsilon > 0$, a $(1-\varepsilon)$-approximation algorithm for maximizing the number of convex edges over all k-th order Delaunay triangulations exists that takes $2^{2^{O(k)}/\varepsilon^2} \cdot n$ time (if the Delaunay triangulation is given).*

5 Discussion

We analyzed the algorithmic complexity of optimizing various measures that apply to triangulations, and terrains represented by triangulations. The class of triangulations over which optimization is done is the first order Delaunay triangulations. We gave efficient algorithms for four measures, NP-hardness proofs for three other measures, and polynomial time approximation schemes for two measures that were shown NP-hard. One approximation algorithm could be extended to k-th order Delaunay triangulations.

Other measures related to terrain modeling in GIS may be of interest to optimize. Also, certain measures that have efficient, optimal algorithms for first order Delaunay triangulations may become harder for second and higher order Delaunay triangulations. These are interesting topics for further research. It is also unknown how to generalize the approximation algorithm for maximizing non-mixed vertices to higher order Delaunay triangulations. Finally, improving on the doubly-exponential dependency on the order k in the approximation algorithm for maximizing convex edges is worthwhile.

Acknowledgements. The authors thank Hans Bodlaender and René van Oostrum for helpful discussions.

References

1. B. Aspvall, M. Plass, and R. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Inf. Proc. Lett.*, 8:121–123, 1979.
2. B. S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41:153–180, 1994.
3. M. Bern, H. Edelsbrunner, D. Eppstein, S. Mitchell, and T. S. Tan. Edge insertion for optimal triangulations. *Discrete Comput. Geom.*, 10(1):47–65, 1993.
4. M. Bern and P. Plassmann. Mesh generation. In J. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 291–332. Elsevier, Amsterdam, 1997.
5. H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.
6. L. de Floriani, P. Magillo, and E. Puppo. Applications of computational geometry in Geographic Information Systems. In J. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*, pages 333–388. Elsevier, Amsterdam, 1997.
7. T. de Kok, M. van Kreveld, and M. Löffler. Generating realistic terrains with higher-order Delaunay triangulations. *Comput. Geom. Th. Appl.*, 36:52–65, 2007.
8. H. Edelsbrunner and T. S. Tan. A quadratic time algorithm for the minmax length triangulation. *SIAM J. Comput.*, 22:527–551, 1993.
9. H. Edelsbrunner, T. S. Tan, and R. Waupotitsch. $O(N^2 \log N)$ time algorithm for the minmax angle triangulation. *SIAM J. Sci. Stat. Comput.*, 13:994–1008, 1992.
10. J. Gudmundsson, M. Hammar, and M. van Kreveld. Higher order Delaunay triangulations. *Comput. Geom. Theory Appl.*, 23:85–98, 2002.
11. L. J. Guibas, J. E. Hershberger, J. S. B. Mitchell, and J. S. Snoeyink. Approximating polygons and subdivisions with minimum link paths. *Internat. J. Comput. Geom. Appl.*, 3(4):383–415, Dec. 1993.
12. D. S. Hochbaum and W. Maass. Approximation schemes for covering and packing problems in image processing and VLSI. *J. ACM*, 32:130–136, 1985.
13. R. J. Huggett. *Fundamentals of Geomorphology.* Routledge, 2003.
14. S. K. Jenson and C. M. Trautwein. Methods and applications in surface depression analysis. In *Proc. Auto-Carto 8*, pages 137–144, 1987.
15. G. Kant and H. L. Bodlaender. Triangulating planar graphs while minimizing the maximum degree. *Inform. Comput.*, 135:1–14, 1997.
16. D. Lichtenstein. Planar formulae and their uses. *SIAM J. Cmp.*, 11:329–343, 1982.
17. D. R. Maidment. GIS and hydrologic modeling. In M. Goodchild, B. Parks, and L. Steyaert, editors, *Environmental modeling with GIS*, pages 147–167. Oxford University Press, New York, 1993.
18. W. Mulzer and G. Rote. Minimum weight triangulation is NP-hard. In *Proc. 22nd Annu. ACM Sympos. Comput. Geom.*, pages 1–10, 2006.
19. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms.* Oxford University Press, New York, 2006.
20. N. Robertson and P. D. Seymour. Graph minors II. Algorithmic aspects of tree width. *J. Algorithms*, 7:309–322, 1986.
21. D. M. Theobald and M. F. Goodchild. Artifacts of TIN-based surface flow modelling. In *Proc. GIS/LIS*, pages 955–964, 1990.
22. M. van Kreveld. Geographic Information Systems. In J. E. Goodmann and J. O'Rourke, editors, *Handbook of Discrete and Computational Geometry*, chapter 58, pages 1293–1314. Chapman & Hall/CRC, Boca Raton, 2004.
23. M. van Kreveld, M. Löffler, and R. I. Silveira. Optimization for first order Delaunay triangulations. Technical Report UU-CS-2007-011, Utrecht University, Institute of Information and Computing Sciences, 2007.