River networks and watershed maps of triangulated terrains revisited[*]

Hee-Kap Ahn[†] Mark de Berg[‡] Otfried Cheong[§] Herman Haverkort[¶] A. Frank van der Stappen[‖]
Laura Toma[**]

Abstract

Triangulated surfaces are often used to represent terrains in geographic information systems. We investigate the complexity of river networks and watershed maps on such terrains under the assumption that water always follows the path of steepest descent. We show that the worst-case complexity is only $\Theta(n^2)$ if all triangles are non-obtuse or if all triangles are fat, that is, their minimum angles are bounded from below by a positive constant. Furthermore, we can compute the river networks and watershed maps by tracing paths in a directed acyclic graph representation of the triangulation—a property that can be exploited to do computations I/O-efficiently.

1 Introduction

Hydrologists, country planners etc. use terrain models while managing or monitoring water resources, possible flood areas, erosion and other natural processes. In such applications it is important that one can identify where rivers are, where the boundaries of their watersheds are (the areas that drain through them), and to which rivers they are tributaries. Construction works and natural processes in the terrain may change its shape and affect the river network and the boundaries of watersheds. To analyse or predict such changes by manual quantification of watersheds would be a tedious and time-consuming job, so we would rather compute river networks and watershed maps from the updated terrain model automatically.

Therefore several computational geometers have studied the structure, complexity and computation of river networks and watershed maps on triangulated terrains [3, 5, 7, 8]. To be able to discuss their and our results, we first give some definitions. A *terrain* is the graph in $\mathbb{R}^3$ of a continuous function $z = f(x, y)$ defined on a compact, connected subset of the xy-plane. A triangulated terrain or TIN is a terrain that consists of triangles. We assume that water always runs downhill in the direction of steepest descent (to keep the definitions simple we assume that the direction of steepest descent is always unique). The watershed of a point p is the set of all points in the terrain from which the water flows to p. The drainage area of p is the size of its watershed. The *river network* is the set of points with drainage area greater than zero, or in other words, the set of points whose watersheds are two-dimensional regions. A *watershed map* is a map of the boundaries between the watersheds of points of interest (such as river mouths and confluences).

Yu et al. distinguish three types of edges in the triangulation [8]: *confluent* edges or *channels* are edges that receive water from both adjacent triangles (because on both adjacent triangles, the direction of steepest descent is directed towards the edge); *transfluent* edges receive water from one adjacent triangle, which continues its way down the other triangle; and *diffluent* edges or *ridges* receive no water (because on both adjacent triangles, the direction of steepest descent is directed away from the edge).

De Berg et al. [3] observe that the river network of a triangulated terrain consists of confluent edges and paths of steepest descent that start from the lower ends of confluent edges. McAllister and Snoeyink [5, 7] analyse the complete topology of watershed maps. They find that watersheds are bounded by ridges and by paths of steepest descent that lead to the points for which we want to know the watershed boundaries and to certain vertices in the triangulation. Thus, in a triangulation with n vertices, both the river network and the watershed map consist of edges and vertices already present in the triangulation, plus $O(n)$ paths of steepest descent.

The map can therefore be computed by an algorithm based on tracing paths of steepest descent in the terrain. We can model this as a computation on a directed planar graph: let the *descent graph* $\mathcal{G}$ be the graph that contains a node $v(e)$ for every edge e of the triangulation, and a directed arc from $v(e)$ to $v(f)$ if and only if e and f are on the boundary of the

[*]We thank Sang-Won Bae, Hyo-Sil Kim, and the participants of the 2005 Carleton-Eindhoven workshop on computational geometry for discussions and the stimulating environment in which this research took place.

[†]Department of Computer Science, Korea Advanced Institute of Science and Technology, heekap@gmail.com

[‡]Department of Mathematics and Computer Science, Eindhoven University of Technology, mdberg@win.tue.nl

[§]Department of Computer Science, Korea Advanced Institute of Science and Technology, otfried@kaist.ac.kr. Otfried Cheong was supported by LG Electronics.

[¶]Department of Mathematics and Computer Science, Eindhoven University of Technology, cs.herman@haverkort.net

[‖]Department of Information and Computing Sciences, Utrecht University, frankst@cs.uu.nl

[**]Department of Computer Science, Bowdoin College, ltoma@bowdoin.edu

same triangle and there is a path of steepest descent across that triangle from e to f. Following a path of steepest descent (until it reaches a vertex or a confluent edge) now corresponds to following a path in $\mathcal{G}$. Computing a watershed map by tracing paths in this graph and connecting them properly can be done in $O(n \log n + k)$ time, where k is the complexity of the output (McAllister [6]).

Alas, the output may be quite big, at least in theory. De Berg et al. [3] describe how to construct a terrain with n vertices that has a descent graph with cycles, so that a path of steepest descent may return to the same edge of the triangulation several times. In fact their terrain contains $\Theta(n)$ rivers that each cross a set of $\Theta(n)$ edges $\Theta(n)$ times. Thus the river network has $\Theta(n^3)$ vertices and so has the watershed map. De Berg et al. provide some Dutch comfort by proving that the worst-case complexity of river networks (and, by the same arguments, watershed maps) is in fact not worse than $\Theta(n^3)$.

Problem. When computing watershed maps for large terrains that may not even fit in main memory, a cubic complexity of the watershed map would be disastrous.

On top of that, when the terrain does not fit in main memory, the computation may be slow. Accessing data stored on disk takes much longer than accessing data in main memory: in present-day hardware, the difference may be a factor 1 000 000. This is due to the latency of hard disks. Fortunately, once the disk and its read/write head have been moved into the correct position, reading a large block of data is almost as fast as reading just a few bytes. Hence, to amortize the latency, current computer systems transfer data between disk and memory in blocks: when we access the disk to read an edge of the triangulation, we do not read only one edge, but a block of several thousands.

To run McAllister's algorithm and trace paths of steepest descent without accessing the disk too often, we would have to group the nodes of the descent graph in blocks such that while following paths in this graph, we do not cross block boundaries too often. However, the best known methods for blocking planar graphs perform poorly [1]. Research into I/O-efficient algorithms (algorithms with optimized disk access patterns) has yielded another, much more efficient technique, called time-forward processing [2, 4], that can be used to process directed *acyclic* graphs efficiently. Unfortunately, as is apparent from the aforementioned construction by De Berg et al. [3], the descent graph $\mathcal{G}$ is not guaranteed to be acyclic.

However, the construction by De Berg et al. is highly contrived and we do not expect to encounter such artefacts in practice. This has led us to investigate what easily verifiable, realistic properties of a triangulation would guarantee that $\mathcal{G}$ is acyclic. Thus we kill two birds with one stone: find out under what conditions time-forward processing can be applied, and prove that the complexity of river networks and watershed maps in realistic triangulations is only $O(n^2)$.

Our results. We prove that the descent graph is acyclic if all triangles in the triangulation are non-obtuse (either in space, or in the projection on a horizontal plane).

Furthermore, we describe a method to cut the edges of *any* triangulation into segments such that the descent graph on those segments is acyclic. If all triangles in the triangulation are fat—that is, if their minimum angles, either in space or in the projection, are bounded from below by a positive constant—then the number of segments per edge is $O(1)$, and thus the total complexity of the descent graph is $O(n)$.

Therefore, for triangulations with only non-obtuse triangles or only fat triangles the complexity of any steepest descent path is $O(n)$, and the total complexity of river networks and watershed maps is therefore $O(n^2)$. We prove that this is optimal in the worst case by means of a lower-bound construction on a triangulated regular square grid.

Below we describe our results for descent graphs of fat triangulations and our lower-bound construction. We leave the results on non-obtuse triangulations for the full version of this paper.

2 Fat triangulations

We first describe our method to cut the edges of any triangulation into segments. For a vertex v in the triangulation, let $r_{\min}(v)$ be the distance to its nearest neighbour vertex in the triangulation, and let $d_{\min}(v)$ be the distance between v and the nearest edge in the triangulation that is not incident to v. We define the *neighbourhood* of a vertex v as the disk centered on v with radius $\min\{r_{\min}(v)/2, d_{\min}(v)\}$. The neighbourhood boundaries divide every edge in the triangulation into two *inner segments* (the segments inside the neighbourhoods of the endpoints) and at most one *outer segment* (the rest of the edge)—see Fig. 1. The neighbourhoods are pairwise disjoint.

We further subdivide the outer segments of the edges into a minimum number of smaller segments, called *free segments*, such that each segment's minimum circumscribed circle does not intersect any other edges—see Fig. 1 for an example. All segments are considered to include their upper endpoints and to be relatively open at their lower endpoints.

The *descent graph* $\mathcal{G}$ on these segments contains a node for every inner or free segment; the segment corresponding to node v is denoted by $segm(v)$, and its upper and lower endpoint are indicated by $up(v)$ and $lw(v)$, respectively. The descent graph contains a directed arc from node a to node b if and only if $segm(a)$ and $segm(b)$ are on the boundary of the same triangle T and there is a path of steepest descent across T from $segm(a)$ to $segm(b)$.

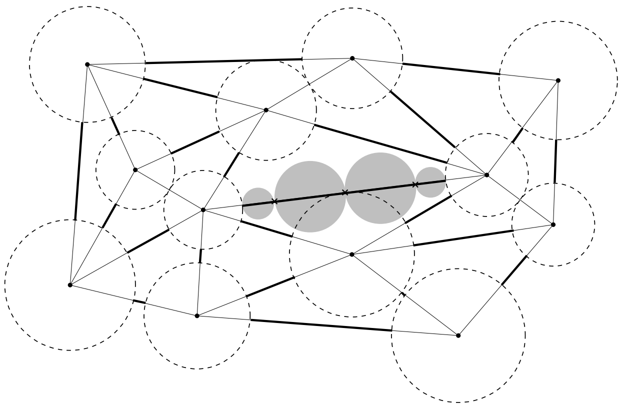

Figure 1: A part of a triangulation. Dotted circles delimit the vertex neighbourhoods. Inner edge segments are drawn with thin lines, outer edge segments are drawn with thick lines. For one outer edge, a subdivision into free segments is shown, with the minimum circumscribed circles of the segments (shaded disks).

For two nodes a and b in $\mathcal{G}$, we define $segm(a) \succ segm(b)$ if and only if $z(up(a)) > z(up(b))$, or $z(up(a)) = z(up(b))$ and $z(lw(a)) > z(lw(b))$, where $z(p)$ is the elevation of p.

Lemma 1 *If $\mathcal{G}$ contains an arc from a to b, then $segm(a) \succ segm(b)$.*

Proof. Without loss of generality, assume $segm(b)$ is oriented from east to west and $segm(a)$ lies on the triangle T to the north of $segm(b)$. Let L(owland) be the closed halfplane bounded from above by the contour line of T through $up(b)$. Let H(illside) be the open halfplane that contains $segm(b)$ and is bounded by the line of steepest descent through $lw(b)$ on T. Let N(orth) be the open halfplane bounded from below by the line that contains $segm(b)$. (See Fig. 2)

Assume, for the sake of contradiction, $segm(a) \not\succ segm(b)$. Then $segm(a)$ lies completely in L. Because there is flow across T from $segm(a)$ to $segm(b)$, there is at least one point in $segm(a)$ that lies in $H \cap N$, and hence, in $L \cap H \cap N$. By Thales' Theorem, that point lies inside the minimum circumscribed circle of $segm(b)$.

By definition, the minimum circumscribed circle of any free segment $segm(b)$ cannot intersect any other segments (or vertices) of the triangulation, and the minimum circumscribed circle of an inner segment can only intersect other inner segments in the same vertex neighbourhood. So $segm(a)$ and $segm(b)$ must be inner segments in the neighbourhood of some vertex v. By the assumption $segm(a) \not\succ segm(b)$, the endpoint of $segm(b)$ that is not v must lie at least as high as the endpoint of $segm(a)$ that is not v. Since $segm(a)$ and $segm(b)$ have the same length, this implies that $segm(b)$ does not descend more steeply (or ascend less steeply) from v than $segm(a)$. Thus the triangle shared by $segm(a)$ and $segm(b)$ is not tilted towards $segm(b)$, which contradicts that there would be a path of steepest descent from $segm(a)$ to $segm(b)$.

Therefore $segm(a) \succ segm(b)$. □

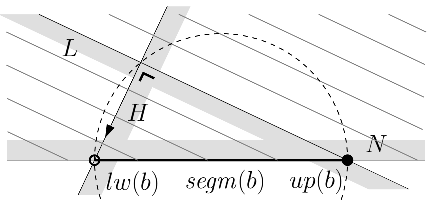

Figure 2: $L \cap H \cap N$ lies in the minimum circumscribed circle of $segm(b)$.

Let an *α-fat triangulation* be a triangulation that only contains triangles that have minimum angle at least α in the projection on a horizontal plane.

Theorem 2 *The descent graph $\mathcal{G}$ on the segments of an α-fat triangulation with n triangles is a planar directed acyclic graph with $O(n/\alpha^2)$ nodes.*

Proof. Sorting the nodes of $\mathcal{G}$ into $\succ$-order puts them into topological order (by Lemma 1); hence $\mathcal{G}$ is acyclic. To bound the number of nodes we use the fact that each edge of the triangulation is cut into at most $2\lceil 2.64/\alpha^2 \rceil$ segments. We omit the details from this abstract. □

Corollary 3 *Any path of steepest descent or ascent in an α-fat triangulation has $O(n/\alpha^2)$ vertices, and the total complexity of the river network and the watershed map in an α-fat triangulation is $O(n^2/\alpha^2)$.*

To allow tracing paths of steepest descent through channels and vertices, $\mathcal{G}$ also needs to include nodes representing the vertices of the triangulation. With a more refined definition of vertex neighbourhoods the theorem also holds if angles of triangles are measured in space instead of in the projection on the plane.

3 Lower bound

We describe a terrain, represented by a regular grid of $n/2$ triangulated squares, that has a river network with $\Theta(n^2)$ vertices. The basic ingredient of our construction is shown in Fig. 3. It is a terrain of 5×7 squares. The boundary is at roughly the same elevation all around, except for three lower, horizontal edges a, b and z (and the surrounding slopes that connect them to the higher boundary edges). All rivers that enter the terrain across the interior of a or b leave the terrain as separate rivers across the interior of z. The shaded areas drain through two more rivers that leave the terrain across the interior of z. All of the above properties can easily be maintained if the valleys (the triangles across which water flows from a or b to z) are simultaneously raised or lowered.

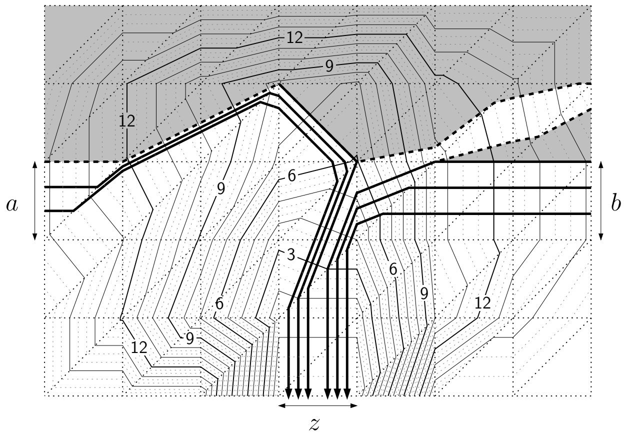

Figure 3: A contour map of a terrain of 5×7 triangulated squares. The rivers that enter the terrain across a or b leave the terrain as separate rivers across z.

Lemma 4 *For any $k > 0$ there is a rectangular terrain $T(k)$ of less than 168×2^k triangles such that at least 2^k different rivers flow out of the terrain across the middle edge of one of the long sides.*

Proof. Let $T(1)$ be the terrain shown in Fig. 3, with the valleys raised so that a and b are at the same elevation as the rest of boundary (except z). For $k > 1$, the terrain $T(k)$ consists of two appropriately rotated and mirrored copies of $T(k-1)$, connected by a copy of $T(1)$ with valleys appropriately lowered—see Fig. 4. We complete the terrain to a rectangle by padding it with triangles, such that the rivers that flow across z flow straight to the closest edge of the boundary. The claimed bounds are easily proven by induction. □

Theorem 5 *There is a terrain of n non-obtuse, fat, Delaunay triangles that has a river network with $\Theta(n^2)$ vertices.*

Proof. Take $T(\lfloor \log(n/336) \rfloor)$ and complete it to n triangles, such that all $\Omega(n)$ rivers that flow from it cross $\Theta(n)$ edges of the complementary triangles. □

4 Discussion

De Berg et al. [3] asked if one could prove an $O(n^2)$ bound on the complexity of river networks in Delaunay triangulations or other triangulations with well-shaped triangles. We showed that this bound indeed holds if all triangles are non-obtuse, or if the triangles are fat. The question if the same bound can be proven for a Delaunay triangulation is still open.

In the case of fat triangles, the constant in the $O(n^2)$ bound depends on the minimum angle of the triangles. A close look at the analysis reveals that the dependency seems to be quite local: only in the direct neighbourhood of small angles, will many nodes be put in the descent graph. Hence an occasional non-fat

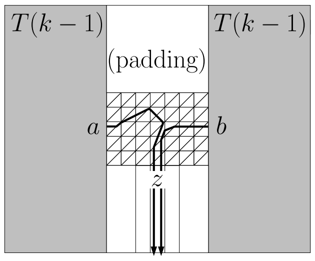

Figure 4: $T(k)$: a terrain of $O(2^k)$ triangles with $\Theta(2^k)$ rivers flowing out across a single edge.

triangle in the triangulation will not affect the bounds on the complexity of the river network significantly.

Our lower bound proves that the $O(n^2)$ worst-case bound cannot be improved under any of the conditions mentioned so far. However, because quadratic complexity would still be impractical, because our lower-bound construction is still quite contrived, and because Yu et al. [8] observed linear complexity in experiments on real data, we still wonder if other, easily verifiable conditions on triangulations may enable us to prove subquadratic bounds.

References

[1] P. K. Agarwal, L. Arge, T. M. Murali, K. R. Varadarajan and J. S. Vitter. I/O-efficient algorithms for contour-line extraction and planar graph blocking. *Symp. on Discrete Algorithms '98*, p117–126.

[2] L. Arge. The buffer tree: A technique for designing batched external data structures. *Algorithmica*, 37(1):1–24, 2003.

[3] M. de Berg, P. Bose, K. Dobrint, M. van Kreveld, M. Overmars, M. de Groot, T. Roos, J. Snoeyink and S. Yu. The complexity of rivers in triangulated terrains. *Canad. Conf. Comp. Geom. '96*, p325–330.

[4] Y.-J. Chiang, M. T. Goodrich, E. F. Grove, R. Tamassia, D. E. Vengroff and J. S. Vitter. External-memory graph algorithms. *Symp. on Discrete Algorithms '95*, p139–149.

[5] M. McAllister. *The computational geometry of hydrology data in geographic information systems.* PhD th., Univ. of British Columbia, 1999.

[6] M. McAllister. *A watershed algorithm for triangulated terrains. Canad. Conf. Comp. Geom. '99.*

[7] M. McAllister and J. Snoeyink. Extracting consistent watersheds from digital river and elevation data. *Ann. Conf. Amer. Soc. for Photogrammetry and Remote Sensing / Amer. Congr. on Surveying and Mapping '99.*

[8] S. Yu, M. van Kreveld and J. Snoeyink. Drainage Queries in TINs: from local to global and back again. *Symp. on Spatial Data Handling '96*, p13A.1–14.