Boundary-Optimal Triangulation Flooding

Richard J. Nowakowski
Department of Mathematics and Statistics, Dalhousie University
Halifax, NS B3H 3J5, Canada
rjn@mathstat.dal.ca

Norbert Zeh
Faculty of Computer Science, Dalhousie University
Halifax, NS B3H 1W5, Canada
nzeh@cs.dal.ca

Given a planar triangulation all of whose faces are initially white, we study the problem of colouring the faces black one by one so that the boundary between black and white faces as well as the number of connected black and white regions are small at all times. We call such a colouring sequence of the triangles a flooding. We prove that a flooding with boundary size $\mathcal{O}(\sqrt{n})$ and $\mathcal{O}(\log n)$ regions can be computed in $\mathcal{O}(n \log n)$ time. We also prove that it is in general impossible to guarantee boundary size $\mathcal{O}(n^{1-\epsilon})$, for any $\epsilon > 0$, and a number of regions that is $o(\log n)$, where n is the number of faces of the triangulation.

Keywords: Planar triangulations, separators, graph layout.

1. Introduction

We study the following *triangulation flooding problem* posed by Hurtado: Given a planar triangulation all of whose triangles are initially white, find an ordering of the triangles such that, when colouring the triangles black in this order, the number of connected monochromatic regions and the boundary between white and black regions are small at all times.[1]

In this paper, we provide an $\mathcal{O}(n \log n)$-time algorithm that finds an order of colouring the triangles such that, at all times, the boundary between the black and the white regions has size $\mathcal{O}(\sqrt{n})$, there are $\mathcal{O}(\log n)$ white regions, and there is only one black region. This is best possible up to constant factors: At the time when exactly half the faces are black, the boundary vertices form a $\frac{1}{2}$-separator. Since Lipton and Tarjan (Ref. 2) show that the minimum size of such a separator is $\Omega(\sqrt{n})$ in the worst case, we cannot hope to achieve a flooding with boundary size $o(\sqrt{n})$ in the worst case. As for the number of white regions, we show that, for any $\epsilon > 0$, there exists a family of triangulations such that, if a boundary size of $\mathcal{O}(n^{1-\epsilon})$ is desired, there have to be $\Omega(\log n)$ monochromatic regions at some point during the colouring process.

The triangulation flooding problem can be rephrased as a layout problem of the dual of the triangulation: Given a 3-regular planar graph, find a linear layout $(v_1, v_2, \ldots, v_n)$ of small cutwidth and such that, for every cut $(V_i, V \backslash V_i)$, the number of connected components of $G - E_i$ is small, where $V_i = \{v_1, v_2, \ldots, v_i\}$ and E_i is the set of edges with exactly one endpoint in V_i; that is, the edges in E_i cross the cut.

Layouts of small cutwidth have applications in network reliability (see Ref. 3), graph drawing (see Ref. 4), and rendering and stream processing of triangular meshes (see Refs. 5, 6, 7). The cutwidth of a graph is also closely related to its search number, which is relevant in VLSI design (see Ref. 8) and network security (see Refs. 9, 10, 11). See Ref. 12 for a comprehensive survey of graph layout problems and their applications. Next we discuss some results in this area that are closely related to our work.

It is well-known that every planar graph of maximal degree d has cutwidth $\mathcal{O}(\sqrt{dn})$.[13] For 3-regular planar graphs, this implies that their cutwidth is $\mathcal{O}(\sqrt{n})$. Our result shows that 3-regular planar graphs have linear layouts of cutwidth $\mathcal{O}(\sqrt{n})$ that are "well-behaved" in the sense that their cuts do not partition the graph into too many connected components.

The following model for rendering a triangular mesh is studied in Ref. 5: Vertices are pushed onto (and popped from) a vertex stack. To render a triangle, it is necessary that all three vertices of the triangle are on the stack. The question studied in Ref. 5 is to determine the stack size required to render any triangular mesh while pushing every vertex onto the stack only once. It is shown that a stack of size $\mathcal{O}(\sqrt{n})$ suffices. In order to obtain this result, a recursive separator decomposition is used, similar to the one used in Ref. 13 to compute a layout of small cutwidth. Our approach for finding a colouring sequence of the triangles also uses a recursive separator decomposition. The main difference is that we use simple-cycle separators (see Ref. 14) in our partition and that we combine these separators carefully to keep the number of monochromatic regions small.

In Refs. 6, 7, the problem of processing large triangular meshes is studied. More precisely, the processed meshes are several gigabytes in size and, therefore, cannot be processed in the main memory of commodity computers. The paradigm for processing such gigantic meshes proposed in Refs. 6, 7 is the following: Read the mesh from disk, one triangle at a time; process the triangles in memory; and write processed triangles to disk, one at a time. THe processing of triangles often depends on their neighbourhood or at least on their interaction with already processed neighbours. Hence, in order to be able to keep the necessary information in memory, it is beneficial to keep the boundary between processed an unprocessed triangles small at all times. This is very similar to our flooding problem if we consider black triangles to be those that have already been processed and white triangles to be the unprocessed ones. Minimizing the scatter of the flooding may help to reduce the amount of book-keeping that is required in these applications.

Finally, we want to point out the relationship between our results and the graph

search problem. In this problem, a number of searchers try to clean the edges of a graph all of whose edges are initially contaminated. In order to clean an edge, a searcher has to move along the edge; if there exists a path between a contaminated edge and a clean edge that is free of searchers, the clean edge is recontaminated. The *search number* of a graph is the minimal number of searchers that are sufficient to clean the graph. The *connected search number* is the minimal number of searchers that are sufficient to clean the graph and, in addition, guarantee that the graph spanned by clean edges is connected at all times. Determining the search number of a graph is NP-complete.[15] For graphs of maximal degree 3, it is known that the search number is equal to the graph's cutwidth.[16] Using the result of Ref. 13, this implies that the search number of a 3-regular planar graph is $\mathcal{O}(\sqrt{n})$. In fact, the proof in Ref. 13 immediately implies that the *connected* search number of a 3-regular planar graph is $\mathcal{O}(\sqrt{n})$. Our result proves that, for 3-regular planar graphs, $\mathcal{O}(\sqrt{n})$ searchers suffice to clean the graph while keeping the clean subgraph connected and keeping the number of maximal connected contaminated subgraphs small.

2. Preliminaries

In this section, we introduce the terminology used in this paper and discuss previous results that are used in our algorithms.

Let T be a given planar triangulation with n faces (triangles). A *colouring* c of T is an assignment of a colour $c(f) \in \{\text{black}, \text{white}\}$ to every face f of T. Colouring c is *trivial* if it colours all faces black or all faces white. The *boundary* ∂c of a non-trivial colouring c is the set of edges in T whose two adjacent triangles have different colours. We say that a colouring c is $t(n)$-*bounded* if $|\partial c| \leq t(n)$. Consider the connected components of $\mathbb{R}^2 \setminus \partial c$. Each of these components contains either only black or only white faces. Accordingly, we call such a region black or white. We define W and B to be the union of all white and black faces, respectively. We say that colouring c is $(b(n), w(n))$-*scattered* if B consists of at most $b(n)$ connected regions and W consists of at most $w(n)$ connected regions. The colouring is $s(n)$-*scattered* if the total number of monochromatic regions is at most $s(n)$; that is, a $(b(n), w(n))$-scattered colouring is also $(b(n) + w(n))$-scattered. See Fig. 1 for illustrations of these definitions.

The *black-count* of a colouring c is the number of faces f of T such that $c(f) =$ black. A *flooding* is a sequence $F = (c_0, c_1, \ldots, c_n)$ of colourings such that, for all $0 \leq i \leq n$, the black-count of colouring c_i is i and $c_i(f) =$ black implies that $c_{i+1}(f) =$ black. We say that a flooding F is $t(n)$-bounded, $(b(n), w(n))$-scattered, or $s(n)$-scattered if every non-trivial colouring in F is $t(n)$-bounded, $(b(n), w(n))$-scattered, or $s(n)$-scattered, respectively. See Fig. 2 for illustrations of these definitions.

If both regions W and B defined by colouring c are connected, the boundary of c is a simple cycle C. We call this cycle a *simple-cycle separator* of T. Given an assignment of weights to the faces of T such that the weights of all faces sum to 1,

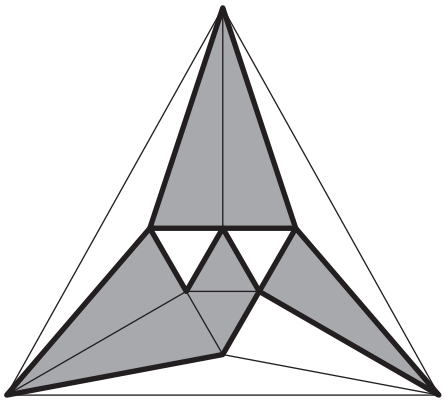

Fig. 1. A $(3, 3)$-scattered, 13-bounded colouring. There are 3 black and 3 white regions. The 13 boundary edges are shown in bold.

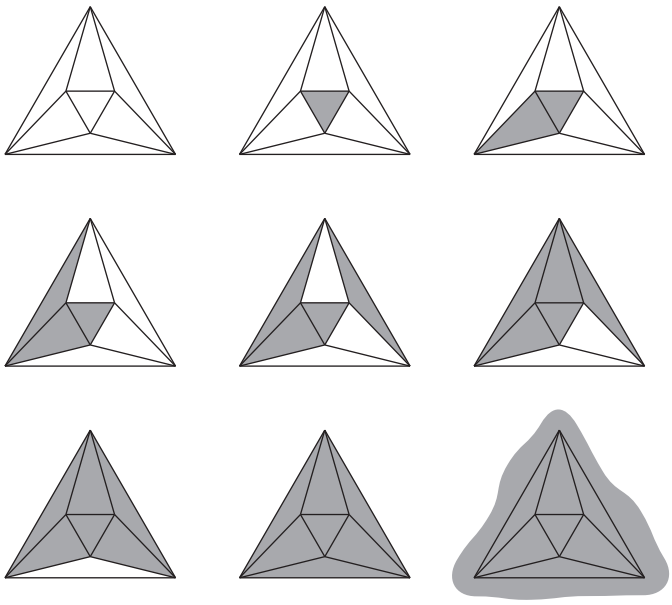

Fig. 2. A $(2, 2)$-scattered, 8-bounded flooding that starts with the central face and ends with the exterior face. The colouring with maximal scatter and largest boundary size is the one shown in the center (after colouring 4 faces).

we call the cycle C a *balanced simple-cycle separator* if the total weight of the faces in each of the regions W and B is at most $2/3$. The *size* of a simple-cycle separator is the number of edges on the cycle. Miller shows the following result:

Theorem 1. *Given a planar triangulation T with n vertices and a weight function*

that assigns non-negative weights to the faces of T such that the total weight of all faces is at most 1 and no face has weight more than 2/3, a balanced simple-cycle separator of size at most $\sqrt{8n}$ for T can be found in linear time.[14]

To obtain a separator as in Theorem 1, Miller uses a non-trivial refinement of the separator algorithm by Lipton and Tarjan (see Ref. 2). Lipton and Tarjan compute a breadth-first search (BFS) tree of the triangulation and then remove two appropriate levels to partition the tree into a top, middle, and bottom part. The top and bottom parts have weight at most 2/3 each, while the middle part may have weight exceeding 2/3. However, the middle part is guaranteed to have small diameter, so that a fundamental cycle w.r.t. an appropriate spanning tree can be used to partition the middle part into two pieces of weight at most 2/3 each. Since the removal of the separator produces two or more pieces, each of which has weight at most 2/3, these pieces can be grouped together to obtain two graphs of weight at most 2/3 each.

Miller's refinement of this procedure consists of two parts: First, instead of partitioning the graph into three pieces, as Lipton and Tarjan do, intuitively he merges faces of T to form larger faces until he obtains a subgraph T' of T that has diameter $\mathcal{O}(\sqrt{n})$ and each of whose faces has weight at most 2/3 and a boundary of length $\mathcal{O}(\sqrt{n})$. When merging two faces, the weight of the resulting face is the sum of the weights of the two merged faces. The final separator is a cycle in T'. To determine which faces to merge, he performs BFS in the face incidence graph of T and then uses a search procedure similar to Lipton and Tarjan's to identify the subset of faces to be merged as those corresponding to subtrees in the BFS-tree.

The second refinement is the manner in which the cycle separator in T' is found. While Lipton and Tarjan use a single fundamental cycle in a BFS-tree of the low-diameter graph, Miller identifies a collection of fundamental cycles and shows that the cycle space spanned by these fundamental cycles contains a cycle that is the desired separator.

3. New Results

We prove that every planar triangulation T has an $\mathcal{O}(\sqrt{n})$-bounded $(1, \mathcal{O}(\log n))$-scattered flooding. More strongly, given a face f_0 of T, there exists such a flooding $F = (c_0, c_1, \ldots, c_n)$ such that $c_1(f_0) =$ black; that is, f_0 is the first triangle coloured black, and every subsequent triangle is coloured black only after at least one of its neighbours has been coloured black. We show that such a flooding can be computed in $\mathcal{O}(n \log n)$ time. Section 4 is dedicated to proving this result.

We also prove that the result from Section 4 is best possible in a very strong sense: For any $0 < \epsilon < 1$, there exists a family of triangulations that do not have $\mathcal{O}(n^{1-\epsilon})$-bounded $o(\log n)$-scattered floodings. This result is presented in Section 5.

4. Floodings with Small Boundary and Low Scatter

In this section, we show how to efficiently find a flooding for a given triangulation that maintains a small boundary at all times, keeps the black region connected, and creates only few white regions at any time. The following theorem states this formally:

Theorem 2. *For every planar triangulation T with n faces and every face f_0 of T, there exists an $\mathcal{O}(\sqrt{n})$-bounded $(1, \mathcal{O}(\log n))$-scattered flooding $F = (c_0, c_1, \ldots, c_n)$ such that $c_1(f_0) =$ black. Such a flooding can be computed in $\mathcal{O}(n \log n)$ time.*

To prove Theorem 2, we compute a recursive partition $\mathcal{P}$ of T using simple-cycle and multi-path separators and use an ordering of the leaves of $\mathcal{P}$ to compute the order in which to colour the faces of T.

In Section 4.1, we define precisely what type of separators we use to obtain this partition and show how to obtain the partition $\mathcal{P}$. In Section 4.2, we show how to order the leaves of $\mathcal{P}$ and how to flood each of these leaves so that the resulting flooding satisfies the conditions of Theorem 2.

4.1. *A Partition into Partial Triangulations*

The partition $\mathcal{P}$ we compute recursively partitions T into partial triangulations. A *partial triangulation* is an embedded planar graph G with the following properties (see Fig. 3):

(i) Every face of G is marked as either *interior* or *exterior*.
(ii) All interior faces are triangles.
(iii) The union of all interior faces forms a connected region.
(iv) No two exterior faces share an edge.

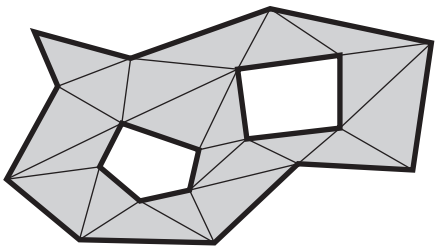

Fig. 3. A partial triangulation. The interior faces are shaded. The boundary is shown in bold.

The *boundary* ∂G of G is the set of edges on the boundaries of the exterior faces. For a partial triangulation G with parent G' in $\mathcal{P}$, the proper boundary $\partial^* G$ of G is the set of boundary edges that are not boundary edges of G'; that is, $\partial^* G = \partial G \setminus \partial G'$. We show how to compute a partition $\mathcal{P}$ of T into partial triangulations with the following properties:

(P1) The root of $\mathcal{P}$ is T.
(P2) Every leaf triangulation has at most $\sqrt{n}$ interior faces.
(P3) Every non-leaf triangulation has more than $\sqrt{n}$ interior faces.
(P4) Every partial triangulation in $\mathcal{P}$ whose parent has size h has size at most $\frac{2}{3}h$ and a proper boundary of size $\mathcal{O}(\sqrt{h})$.

An immediate consequence of Properties (P1), (P3), and (P4) is that the depth of partition $\mathcal{P}$ is $\mathcal{O}(\log n)$.

4.1.1. *Separators of Partial Triangulations*

To compute partition $\mathcal{P}$, we recursively partition T by applying the following lemma until all resulting partial triangulations have at most $\sqrt{n}$ interior faces:

Lemma 1. *Every partial triangulation G with $n \geq 2$ interior faces has a separator of one of the following two types:*

(i) A simple cycle C of length at most $\sqrt{8n+16}$ such that at most $2n/3$ interior faces are on either side of C.
(ii) A set of simple paths $P_1, P_2, \ldots, P_k$ of total length at most $\sqrt{8n+16}$ such that each path has its endpoints, but no internal vertices, in ∂G and each of the regions into which the interior of G is partitioned by these paths contains at most $2n/3$ faces.

Such a separator can be found in linear time.

Proof. We triangulate the exterior faces of G and give weight $1/n$ to every interior face of G and weight 0 to every triangle produced by triangulating the exterior faces. We use Theorem 1 to find a balanced simple-cycle separator C of the resulting triangulation T. Since the interior of G is connected, T has at most $n+2$ vertices. Hence, by Theorem 1, the cycle C has length at most $\sqrt{8n+16}$. If C has at most one vertex in ∂G, we have Case (i). Otherwise, let $P_1, P_2, \ldots, P_k$ be the maximal subpaths of C that have no internal vertices in ∂G. Since C is simple, every path P_i is simple and no two paths P_i and P_j, $i \neq j$, share an internal vertex. Every region in the partition of the interior of G is completely on one side of C and, thus, has size at most $2n/3$. Thus, we have Case (ii).

The complexity of this procedure is linear: The exterior faces of G are easily triangulated in linear time. By Theorem 1, finding the cycle C takes linear time. Assuming that the vertices of G are labelled as belonging to ∂G or as being interior, distinguishing between Cases (i) and (ii), and extracting paths $P_1, P_2, \ldots, P_k$ if we have the latter, takes a single scan of the edge list representing C. □

Given a partial triangulation G, a separator as in Lemma 1 partitions G into partial triangulations $G_1, G_2, \ldots, G_h$ as follows (see Fig. 4): Let $R_1, R_2, \ldots, R_h$ be the connected regions into which the interior of G is divided by the separator

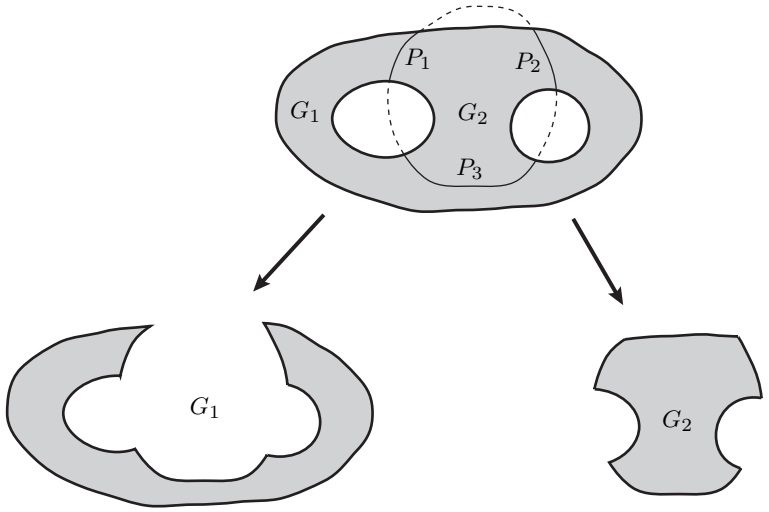

Fig. 4. The separator consisting of paths P_1, P_2, and P_3 partitions the interior of the partial triangulation into the two shown partial triangulations G_1 and G_2. The interior faces are shaded. The boundary edges are shown in bold. The diagonals of the exterior faces that belong to the cycle C from which the multi-path separator has been derived are dashed.

given by the lemma. Then the partial triangulation G_i has the faces of R_i as its interior faces. The exterior faces of G_i are bounded by the boundary edges of R_i. The following lemma is crucial for bounding the number of monochromatic regions produced by our flooding:

Lemma 2. *If G has an exterior face, then at most one of the partial triangulations $G_1, G_2, \ldots, G_h$ does not share an edge with an exterior face of G.*

Proof. Assume that there are two partial triangulations G_i and G_j, $i \neq j$, that do not share an edge with an exterior face of G. Then the edges in ∂G_i and ∂G_j belong to C. Since ∂G_i and ∂G_j consist of simple cycles, the simplicity of C implies that $\partial G_i = \partial G_j = C$. In other words, G_i is the interior of C and G_j is the exterior of C. But this is impossible because one of the two sides has to include an exterior face of G. □

4.1.2. *Computing the Partition*

Partition $\mathcal{P}$ is obtained by repeated application of Lemma 1. The root of $\mathcal{P}$ is T. Every partial triangulation G in $\mathcal{P}$ that has more than $\sqrt{n}$ interior faces is partitioned into partial triangulations $G_1, G_2, \ldots, G_k$ using Lemma 1. These are the children of G in $\mathcal{P}$. Each G_i is partitioned recursively if the number of its interior faces exceeds $\sqrt{n}$. Next we prove that partition $\mathcal{P}$ has the desired properties.

Lemma 3. *Partition $\mathcal{P}$ has Properties P1–P4.*

Proof. Properties P1–P3 are explicitly ensured. Observe that all proper boundary edges of a partial triangulation H in $\mathcal{P}$ belong to the separator used to partition the parent H' of H. By Lemma 1, this separator has size $\mathcal{O}(\sqrt{h})$, where h is the number of faces of H', and partitions H' into partial triangulations of size at most $2h/3$, one of which is H. This proves Property P4. □

For constructing the flooding, it is necessary to have the following auxiliary information about $\mathcal{P}$ available: for every partial triangulation G in $\mathcal{P}$, a complete list of the boundary edges and a classification of these edges as proper or inherited from the parent; for every partial triangulation G in $\mathcal{P}$, a *partition dual* G°, which has one vertex G_i° per child G_i of G and an edge (G_i°, G_j°) if G_i and G_j share at least one boundary edge; the dual T^* of T. Every proper boundary edge e between G_i and G_j stores a pointer to the edge (G_i°, G_j°) and information which of the endpoints of e^* belongs to G_i^* and which one belongs to G_j^*.

Lemma 4. *Partition $\mathcal{P}$ and the auxiliary information just described can be computed in $\mathcal{O}(n \log n)$ time.*

Proof. Computing the dual of G takes linear time. To analyze the cost of computing the partition $\mathcal{P}$ and the partition dual, we observe that the former is achieved by applying Lemma 1 to every non-leaf triangulation H in $\mathcal{P}$, and the latter is achieved by computing the dual of the interior faces of H and contracting edges that are not dual to proper boundary edges of the children of H. Both steps take $\mathcal{O}(|H|)$ time for every partial triangulation H in $\mathcal{P}$. The total size of all partial triangulations is $\mathcal{O}(n \log n)$ because there are $\mathcal{O}(\log n)$ levels in $\mathcal{P}$ and the partial triangulations at every level define a partition of T into face-disjoint partial triangulations. Hence, the total cost of computing $\mathcal{P}$ and the partition duals is $\mathcal{O}(n \log n)$. □

4.2. *Computing the Flooding*

Given partition $\mathcal{P}$, we show in this section how to order the leaves of $\mathcal{P}$ and how to flood each of them in turn, in order to obtain the desired flooding. The ordering of the leaves is discussed in Section 4.2.1. The procedure for flooding leaf triangulations is presented in Section 4.2.2. In Section 4.2.3, we argue that flooding the leaves of $\mathcal{P}$ in the order described in Section 4.2.1, and using the procedure from Section 4.2.2 to do so, produces the desired flooding.

4.2.1. *Ordering the Leaf Triangulations*

We obtain the ordering of the leaves of $\mathcal{P}$ as their left-to-right ordering defined by ordering the children of every internal node of $\mathcal{P}$. To order the children of all internal nodes of $\mathcal{P}$, we apply a recursive procedure that starts by ordering the children of the root and then orders the children of the nodes in the subtrees rooted at the children of the root. After ordering the children of all ancestors of a partial triangulation G in $\mathcal{P}$, we order the children of G as follows:

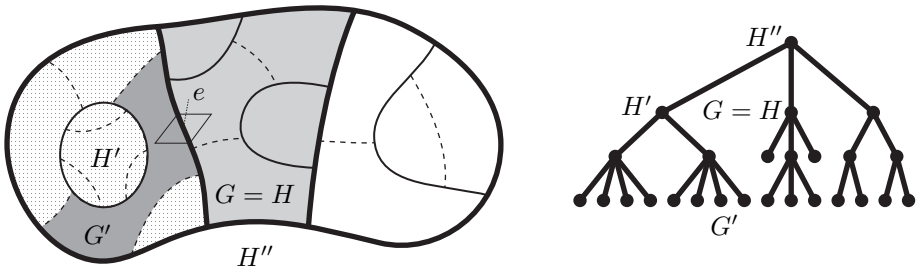

Fig. 5. Edge e is black w.r.t. G and white w.r.t. any descendant of H' that has e on its boundary.

We start by colouring the boundary edges of G black or white (see Fig. 5). The idea is to capture whether the triangle on the other side of this edge will be black or white by the time G will be flooded. Formally, we define this colouring as follows: Let e be a boundary edge of G, let f be the face of G incident to e, and let f' be the other face incident to e (the one that does not belong to G). Let G' be the leaf triangulation that contains f', let H'' be the lowest common ancestor of G and G' in $\mathcal{P}$, let H be the child of H'' that is an ancestor of G, and let H' be the child of H'' that is an ancestor of G'. Then edge e is black if H' precedes H in the ordering of the children of H'', and white otherwise. Note that the colour of an edge depends on the partial triangulation that is considered. If e is black w.r.t. G, then e is black w.r.t. every descendant of H that has e on its boundary and white w.r.t. every descendant of H' that has e on its boundary.

Our goal now is to order the children of G so that every child shares an edge with a triangle that has been coloured before and so that colouring the children of G in this order does not introduce too many white "holes" in intermediate colourings. The former is captured by the first condition in the following lemma; the latter is captured by the second condition.

Lemma 5. *If a partial triangulation G in $\mathcal{P}$ has f_0 as an interior face or has at least one black boundary edge, then the children of G can be arranged in an order $G_1, G_2, \ldots, G_h$ that satisfies the following two conditions:*

(i) If f_0 is an interior face of G, then f_0 is an interior face of G_1 and every child G_i, $i > 1$, shares an edge with a child G_j, $j < i$. If f_0 is not an interior face of G, then every child G_i of G shares an edge with a child G_j, $j < i$, or has a black boundary edge of G on its boundary.

(ii) There is at most one child G_i of G that does not share an edge with a child G_j, $j > i$, and does not have a white boundary edge of G on its boundary.

Proof. We partition the children of G into three groups $\mathcal{G}_1$, $\mathcal{G}_2$, and $\mathcal{G}_3$: If f_0 is an interior face of G, then $\mathcal{G}_2$ contains the child G_1 that contains f_0; all remaining faces are in $\mathcal{G}_3$. $\mathcal{G}_1$ is empty in this case. If f_0 is not an interior face of G, then $\mathcal{G}_3$ contains all children whose boundary edges are white or proper. $\mathcal{G}_2$ contains

all children that have at least one black boundary edge and at least one boundary edge that is white or is shared with a child in $\mathcal{G}_3$. Group $\mathcal{G}_1$ contains the remaining children, that is, those whose boundary edges are either black or shared with other children in $\mathcal{G}_1$ and $\mathcal{G}_2$. If groups $\mathcal{G}_2$ and $\mathcal{G}_3$ are empty, we move an arbitrary child from group $\mathcal{G}_1$ to group $\mathcal{G}_2$.

In the ordering of the children of G, the children in $\mathcal{G}_1$ precede the children in $\mathcal{G}_2$, which in turn precede the children in $\mathcal{G}_3$. The children in $\mathcal{G}_2$ are arranged in no particular order. The children in $\mathcal{G}_3$ are arranged so that each such child G_i shares an edge with a child G_j, $j < i$. Since $\mathcal{G}_1 \cup \mathcal{G}_2$ is non-empty, the connectivity of the interior of G implies the existence of such an ordering. The children in $\mathcal{G}_1$ are arranged so that each such child G_i shares an edge with a child G_j, $j > i$. Again, the non-emptiness of $\mathcal{G}_2 \cup \mathcal{G}_3$ and the connectivity of the interior of G imply the existence of such an ordering.

The constructed ordering satisfies Condition (i): If f_0 is in G, then Condition (i) is trivially satisfied by the constructed ordering. So assume that f_0 is not in G. Then every face in $\mathcal{G}_1 \cup \mathcal{G}_2$ has a black boundary edge and every child G_i in $\mathcal{G}_3$ shares an edge with a child G_j, $j < i$.

To see that Condition (ii) is satisfied, observe that, by Lemma 2, at most one child in $\mathcal{G}_3$ does not have a white boundary edge. Every child G_i in $\mathcal{G}_1$ shares an edge with a child G_j, $j > i$; every child in $\mathcal{G}_2$ shares an edge with a child G_j in $\mathcal{G}_3$ or has a white boundary edge. □

The next lemma shows that the children of all partial triangulations in $\mathcal{P}$ and, hence, the leaves of $\mathcal{P}$ can be ordered efficiently.

Lemma 6. *The children of all partial triangulations in $\mathcal{P}$ can be ordered in $\mathcal{O}(n \log n)$ time.*

Proof. Once we have computed the order of the children of G, we direct the edges of the partition dual of G so that edge (G_i°, G_j°) is directed from G_i° to G_j° if and only if G_i precedes G_j in the computed order of the children of G. Then we direct all edges of G^* that are dual to proper boundary edges of the children of G. For such an edge (f^*, g^*) with $f \in G_i$ and $g \in G_j$, we choose its direction from f^* to g^* if and only if the edge (G_i°, G_j°) in the partition dual of G is directed from G_i° to G_j°. These edge directions are used by the descendants of G to order their children.

In particular, after all ancestors of G have directed the edges of their duals to reflect the order of their children, we use this information as follows to compute the desired grouping of the children of G: For every child G_i, we inspect the edges in $\partial G_i \setminus \partial^* G_i$. Such an edge is black if its dual edge (f^*, g^*) with $g \in G_i$ is directed towards g^*. Otherwise, the edge is white. By inspecting all edges in $\partial G_i \setminus \partial^* G_i$, we determine whether G_i has at least one black and/or at least one white edge on its boundary. This information is sufficient to compute the grouping of the children of G as in Lemma 5. Next we compute a spanning tree of the partition dual of G and remove all vertices dual to children in $\mathcal{G}_2$. Since no vertex in $\mathcal{G}_1$ is adjacent to a

vertex in $\mathcal{G}_3$, every subtree in the resulting forest contains vertices from either $\mathcal{G}_1$ or $\mathcal{G}_3$, but not both. At least one of the vertices in every subtree must be adjacent to a vertex in $\mathcal{G}_2$. We choose this vertex to be the root. For a $\mathcal{G}_1$-subtree, we compute a postorder numbering and arrange its vertices in this order. The vertices in a $\mathcal{G}_3$-subtree are arranged according to a preorder numbering of the tree. Finally, we concatenate the sorted vertex lists of all $\mathcal{G}_1$-subtrees; followed by all vertices in $\mathcal{G}_2$; followed by the concatenation of the sorted vertex lists of all $\mathcal{G}_3$-subtrees.

The marking of the children of a partial triangulation G, depending on the colours of their non-proper boundary edges, requires an inspection of all boundary edges of G. This takes $\mathcal{O}(|G|)$ time. The computation of a spanning tree of the partition dual of G and the removal of the vertices in $\mathcal{G}_2$, including their incident edges, can also be carried out in $\mathcal{O}(|G|)$ time. Finally, it takes linear time to compute the numberings of the vertices of the different trees and to concatenate these sorted vertex lists. Since we repeat this process once for every partial triangulation, the cost is bounded by the total size of all partial triangulations, which is $\mathcal{O}(n \log n)$. □

Given the ordering of the leaves of $\mathcal{P}$ produced by top-down application of Lemma 5, we flood these partial triangulations in this order. The first leaf contains f_0 and is flooded starting from f_0. Every subsequent triangulation G is flooded starting from a face that has a black edge of G on its boundary. The following lemma shows that this is possible.

Lemma 7. *Every leaf triangulation, except the one containing f_0, has a black boundary edge.*

Proof. We prove by induction on the depth d that every level in $\mathcal{P}$ contains at most one triangulation with no black boundary edge, namely the one that contains f_0. This claim implies the lemma.

For $d = 0$, the claim is trivial because the root is the only triangulation at this level and contains f_0.

So assume that the claim holds for some level d. We want to show that it holds for level $d + 1$. Consider the partial triangulation G at level d that contains f_0. By Lemma 5(i), the child of G that contains f_0 is the first child in the ordering of the children of G. Any other child shares an edge with a child that precedes it in the ordering; thus, it has a black boundary edge.

For a partial triangulation G that does not contain f_0, the ordering is chosen so that every child either inherits a black boundary edge from G or shares an edge with a a child of G that precedes it. Hence, every child of G has a black boundary edge. □

4.2.2. *Flooding Leaf Triangulations*

Maintaining a small boundary when flooding a leaf of $\mathcal{P}$ is trivial because every leaf has at most $\sqrt{n}$ faces. The next lemma shows that it is possible to bound the scatter by $\log \sqrt{n}$.

Lemma 8. *A leaf triangulation G can be flooded starting from any face f of G so that, at all times, the boundary between black and white interior faces of G has size at most $\sqrt{n}+2$, the number of black regions interior to G is one, and the number of white regions interior to G is at most $\log \sqrt{n}$.*

Proof. We keep the black region connected by colouring f black and subsequently colouring a face f' black only if at least one of its adjacent faces in G is black. This keeps the black region connected. The bound on the boundary size follows immediately from the fact that G has at most $\sqrt{n}$ interior faces. To guarantee that there are never more than $\log \sqrt{n}$ white regions, we choose the order in which to colour triangles using the following recursive procedure: Let R be the current region to be coloured. The next face in R to be coloured partitions R into at most two regions. We flood each of them recursively, first the smaller one, then the bigger one. If we consider this recursive partition of the interior of G into white regions, then, at any time, only $\log \sqrt{n}$ ancestors of the current region can have siblings waiting to be flooded because each such sibling is of at least the same size as the corresponding ancestor of the current region. Since each sibling of such an ancestor is a connected region, the number of white regions cannot be greater than the number of these siblings. □

4.2.3. *Flooding T*

It remains to show that, by flooding the leaves of $\mathcal{P}$ in the order described in Section 4.2.1, and by using the procedure from Section 4.2.2 to do so, we obtain a flooding F that has the desired properties. Each of the following three lemmas establishes one of these properties.

Lemma 9. *Every colouring in F defines only one black region.*

Proof. Since we flood leaf triangulations in the order defined by recursive ordering of the children of every node, the first triangulation we flood contains f_0. For every subsequent triangulation G, a black boundary edge is one that has a black triangle on the other side by the time G is flooded. The flooding of G starts at a triangle incident to such an edge and, by Lemma 8, keeps the black region connected. Hence, the black region defined by any colouring in F is connected. □

Lemma 10. *The boundary size of every colouring in F is $\mathcal{O}(\sqrt{n})$.*

Proof. Consider the leaf G currently being flooded. Only ancestors of G can be bichromatic. Thus, the boundary of the black region is part of the separators com-

puted at these ancestors. Since the separator of an ancestor with n' internal faces has size $\mathcal{O}(\sqrt{n'})$, and the sizes of these ancestors are geometrically decreasing, the proper ancestors of G contribute $\mathcal{O}(\sqrt{n})$ to the boundary size of the colouring. By Lemma 8, G itself also contributes $\mathcal{O}(\sqrt{n})$ to the boundary size. □

Lemma 11. *Every colouring in F defines at most $\mathcal{O}(\log n)$ connected white regions.*

Proof. Consider the leaf triangulation G currently being flooded. As already observed, only ancestors of G can be bichromatic. It suffices to prove that every ancestor contributes at most one white region to the current colouring. This is sufficient because there are only $\mathcal{O}(\log n)$ such ancestors and, by Lemma 8, the flooding of G itself contributes only $\mathcal{O}(\log n)$ white regions to the current colouring.

We prove this claim by induction on the depths of these bichromatic ancestors. The separator used to partition the root is a cycle. In particular, the root has only two children. At the time when the first child is being flooded, the second child defines a white region.

Now consider the ancestor, G, of the current leaf that is at depth $d > 0$. By the induction hypothesis, G's ancestors contribute at most $d-1$ white regions. By Lemma 5, at most one child G_i of G has no white boundary edge and does not share an edge with a child G_j, $j > i$. Thus, at any time, every white child of G, except G_i, belongs to the same white region as a sibling of an ancestor of G; only G_i may introduce an additional white region. □

We have shown that partition $\mathcal{P}$ can be computed in $\mathcal{O}(n \log n)$ time and that the leaves of $\mathcal{P}$ can be ordered in the same amount of time. Once the leaf ordering is given, the flooding of every leaf triangulation is easily carried out in linear time; hence, the total time for flooding T is linear. This completes the proof of Theorem 2.

5. Families of Hard Triangulations

As already mentioned in the introduction, it is in general impossible to obtain an $o(\sqrt{n})$-bounded flooding for a given triangulation. In this section, we prove that the scatter of the flooding in Theorem 2 is also optimal, even if we relax the bound on the boundary size to $\mathcal{O}(n^{1-\epsilon})$, for any $0 < \epsilon < 1$, and we allow more than one black region, that is, we are only interested in the total number of monochromatic regions. The following theorem states this formally:

Theorem 3. *For any $0 < \epsilon < 1$, there exists a family of triangulations that do not have $\mathcal{O}(n^{1-\epsilon})$-bounded $o(\log n)$-scattered floodings.*

To prove Theorem 3, we show how to construct, for any pair of parameters n and ϵ, a triangulation T of size $\mathcal{O}(n)$ as in Theorem 3. Then we define a tree X that captures the structure of T; prove that the scatter of any $\mathcal{O}(n^{1-\epsilon})$-bounded flooding of T cannot be less than the minimal scatter of an edge flooding of X, defined below; and finally prove that X does not have an $o(\log n)$-scattered flooding.

5.1. *Construction of a Hard Triangulation*

To construct triangulation T, we place triangles in the plane and then triangulate the regions bounded by these triangles. Every such triangle Δ is said to be at a level (i, j). We compare levels lexicographically; that is, $(i, j) < (i', j')$ if either $i < i'$ or $i = i'$ and $j < j'$.

The first triangle we place is a bounding triangle at level $(0, 0)$. All subsequent triangles are placed inside this triangle. After placing the bounding triangle, we iteratively place two level-$(i+1, 0)$-triangles into every level-$(i, 0)$ triangle until the last level $(\ell, 0)$ we produce contains between n^α and $2n^\alpha$ triangles, where $\alpha = \epsilon/2$. Observe that $\ell \geq \log(n^\alpha) = \alpha \cdot \log n$. We call levels $(0, 0), (1, 0), \ldots, (\ell, 0)$ *branching levels*. Now we continue by placing triangles at *non-branching levels* (i, j), $j > 0$. Let $m = \lceil n^{1-\alpha} \rceil$. Then we place one level-(i, j) triangle Δ, for $0 \leq i \leq \ell$ and $1 < j < m$, into every level-$(i, j-1)$ triangle Δ' so that the level-$(i+1, 0)$ triangles contained in Δ' are also contained in Δ. We obtain our final triangulation T by triangulating the regions between triangles at consecutive levels (see Fig. 6). To avoid confusion, we refer to the nested triangles we place during the construction of T as *triangles* and to the triangular faces of T as *faces*. Our first lemma proves that T has the desired number of faces.

Lemma 12. *Triangulation T has $\mathcal{O}(n)$ faces.*

Proof. The number of triangles at level $(\ell, 0)$ is at most $2n^\alpha$. The number of triangles at all branching levels is at most twice that. Every triangle at a branching level $(i, 0)$ contains $m = \lceil n^{1-\alpha} \rceil$ triangles at non-branching levels (i, j). Hence, the total number of triangles is $\mathcal{O}(n^\alpha \cdot n^{1-\alpha}) = \mathcal{O}(n)$. Since every triangle contributes 3 vertices to the vertex set of T, T has $\mathcal{O}(n)$ vertices and, hence, by Euler's formula, $\mathcal{O}(n)$ faces. □

For two triangles Δ and Δ' at levels (i, j) and $(i, j+1)$, we call the region bounded by Δ and Δ' a *ring*. If Δ and Δ' are at levels $(i, 0)$ and $(i, m-1)$, respectively, we call the region bounded by Δ and Δ' a *tube*. The next lemma establishes that every $\mathcal{O}(n^{1-\epsilon})$-bounded colouring of T has to contain a sufficient number of monochromatic rings, which is the key to lower-bounding the scatter of any $\mathcal{O}(n^{1-\epsilon})$-bounded flooding of T by the scatter of edge floodings of certain trees.

Lemma 13. *Every $\mathcal{O}(n^{1-\epsilon})$-bounded colouring c of T colours at least one ring in every tube completely white or completely black.*

Proof. Assume that there is a tube in T none of whose rings is monochromatic; that is, every ring contains at least one black and at least one white face. Then there is at least one boundary edge between black and white faces in every ring in this tube. Since there are $\lceil n^{1-\alpha} - 1 \rceil = \omega(n^{1-\epsilon})$ rings per tube, this contradicts the assumption that c is $\mathcal{O}(n^{1-\epsilon})$-bounded. □

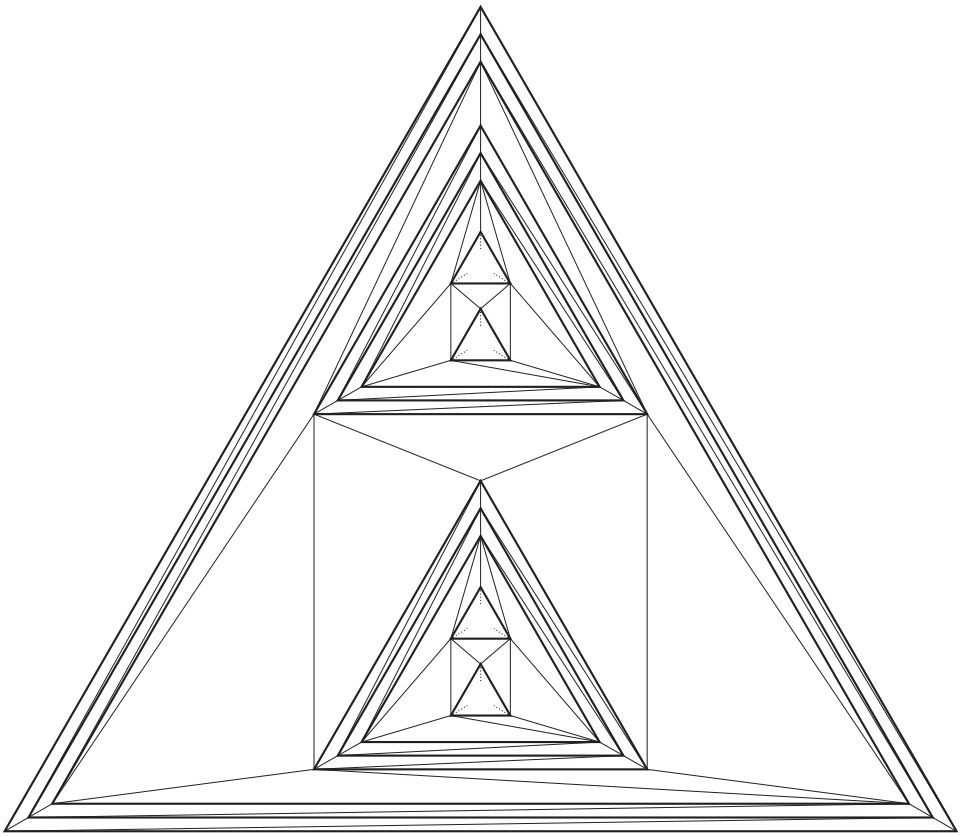

Fig. 6. The construction of a triangulation as in Theorem 3. The nested triangles in the hierarchy are shown in bold. Thin edges show a possible triangulation obtained from this hierarchy of triangles.

5.2. *The Tube Tree of T*

Next we define a tree X that captures the structure of T and such that the minimum scatter of an edge flooding of X is a lower bound on the scatter of any $\mathcal{O}(n^{1-\epsilon})$-bounded flooding of T. Intuitively, the edges of X correspond to the tubes of T. Hence, we call X the *tube tree* of T.

We begin by constructing a tree X_0: Tree X_0 contains one node per region that is bounded by a set of triangles and does not contain any triangle. There is an edge between two nodes if the corresponding regions have a common triangle on their boundaries. Tree X is obtained from X_0 by replacing every maximal path whose internal nodes have degree two with an edge. The nodes of X represent the exterior triangle of T, the regions that are bounded by three triangles, and the triangles that do not contain any other triangles. The edges of X represent the tubes of T. Fig. 7 shows the tube tree for the triangulation shown in Fig. 6.

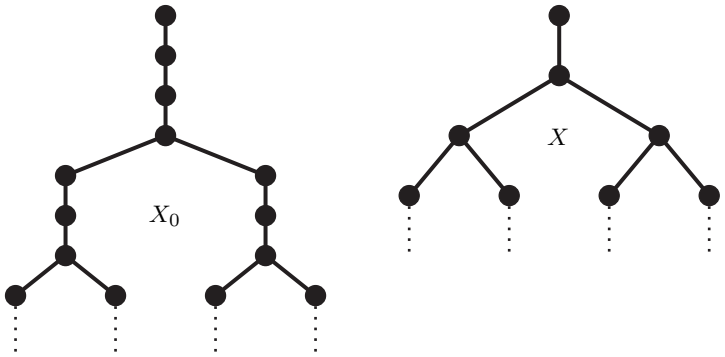

Fig. 7. The construction of the tube tree X for the triangulation T in Fig. 6.

For every $\mathcal{O}(n^{1-\epsilon})$-bounded colouring c of T, we define a colouring c' of the edges of X as follows: By Lemma 13, every tube of T contains either a black or a white ring, or both. We colour the corresponding edge of X black if there is a black ring in the tube and white if all rings in the tube are either white or bichromatic.

A *monochromatic subtree* of X under colouring c' is a maximal subtree all of whose edges have the same colour. We call an edge colouring of X $s(n)$-scattered if it defines at most $s(n)$ monochromatic subtrees of X. An edge flooding of X is defined analogously to a flooding of T, the only difference being that we colour edges. An edge flooding is $s(n)$-scattered if all its colourings are $s(n)$-scattered.

The next lemma proves that the number of monochromatic subtrees of X defined by colouring c' is a lower bound on the number of monochromatic regions of T defined by colouring c.

Lemma 14. *For every $\mathcal{O}(n^{1-\epsilon})$-bounded colouring c of T with k monochromatic regions, the corresponding colouring c' of X defines at most k monochromatic subtrees of X.*

Proof. Let k' be the number of monochromatic subtrees of X under colouring c'. We prove that $k \geq k'$. If there are at most two monochromatic subtrees, the lemma is trivial because c cannot define less than one region of T and, if X has two monochromatic subtrees, then T has two regions of different colours. So assume that there are at least three monochromatic subtrees in X. We prove that, for each of these subtrees, there is at least one monochromatic region in T.

Consider two black subtrees X_1 and X_2 and two edges $e_1 \in X_1$ and $e_2 \in X_2$. Since $X_1 \neq X_2$, there has to be at least one white edge e' on the path connecting e_1 and e_2. Since edges e_1 and e_2 are black, there is at least one black face in each of the tubes represented by these edges. Call these faces f_1 and f_2. Edge e' is white because there is no black ring in the tube represented by e'. Hence, by Lemma 13, this tube contains at least one white ring. Since e' is on the path from e_1 to e_2 in X, this ring separates f_1 from f_2. Therefore, f_1 and f_2 belong to different black

regions of T. This proves that, for every black subtree of X, there is at least one black region in T. An analogous argument shows that the number of white regions of T is at least the number of white subtrees of X. □

The following is an easy consequence of Lemma 14:

Corollary 1. *If X does not have an $s(n)$-scattered edge flooding, then T does not have an $\mathcal{O}(n^{1-\epsilon})$-bounded $s(n)$-scattered flooding.*

5.3. *A Lower Bound on the Scatter of Floodings of the Tube Tree*

By Corollary 1, it suffices to show that the tube tree X does not have an $o(\log n)$-scattered edge flooding. Observe that X has the shape of a complete binary tree whose root has an extra parent. In particular, X has 2^h nodes, for some integer h. We call X an *h-hanger*. The following lemma proves that X does not have an $\lfloor h/2 \rfloor$-scattered edge flooding. By the construction of X, the number of nodes of X is proportional to the number of branching triangles used in the construction of T. Their number is $\Theta(n^\alpha)$. Hence, $h/2 = \frac{1}{2}\log\Theta(n^\alpha) = \Theta(\log n)$.

Lemma 15. *An h-hanger does not have an $\lfloor h/2 \rfloor$-scattered edge flooding.*

Proof. We use induction on h to prove the lemma. If $h \leq 3$, the claim holds trivially because every colouring is at least 1-scattered and every flooding of a tree with more than one edge is at least 2-scattered. So assume that $h > 3$ and that the claim holds for $h-2$. Let $F = (c_0, c_1, \ldots, c_{2^h-1})$ be an edge flooding of an h-hanger H. By removing the root of H, its child, and the three edges incident to these vertices, we partition H into two complete subtrees; both subtrees can be partitioned into two $(h-2)$-hangers. We denote these hangers as H_1, H_2, H_3, H_4. Since the restriction of F to any H_j is an edge flooding of H_j with duplicate consecutive colourings, there have to be colourings $c_{i_1}, c_{i_2}, c_{i_3}, c_{i_4}$, $i_1 < i_2 < i_3 < i_4$, such that the restriction of c_{i_j} to H_j defines at least $\lfloor h/2 \rfloor$ monochromatic subtrees of H_j. If, for any colouring c_{i_j}, $H - H_j$ is not monochromatic, c_{i_j} defines at least $\lfloor h/2 \rfloor + 1$ monochromatic subtrees of H. Now we observe that, for colouring c_{i_2}, $H - H_2$ cannot be monochromatic. Indeed, c_{i_2} succeeds c_{i_1}, so H_1 contains at least one edge that is coloured black by c_{i_2}; c_{i_2} precedes c_{i_3}, so H_3 contains at least one edge that is coloured white by c_{i_2}. Hence, H does not have an $\lfloor h/2 \rfloor$-scattered flooding. □

By Lemma 15, the tube tree of T does not have an $o(\log n)$-scattered edge flooding. By Corollary 1, this implies that triangulation T does not have an $\mathcal{O}(n^{1-\epsilon})$-bounded $o(\log n)$-scattered flooding. This completes the proof of Theorem 3.

Acknowledgements

This work was partially supported by NSERC and MITACS. We would like to thank three anonymous referees for pointing out the relationship of the triangulation

flooding problem to the large number of problems discussed in the introduction.

References

1. F. Hurtado. Open problem posed at the 15th Canadian Conference on Computational Geometry. 2003.
2. R. J. Lipton and R. E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.
3. D. Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM Journal on Computing*, 29(2):492–514, 1999.
4. P. Mutzel. A polyhedral approach to planar augmentation and related problems. In *Proceedings of the 3rd Annual European Symposium on Algorithms*, volume 979 of *Lecture Notes in Computer Science*, pages 497–507, 1995.
5. R. Bar-Yehuda and C. Gotsman. Time/space tradeoffs for polygon mesh rendering. *ACM Transactions on Graphics*, 15(2):141–152, 1996.
6. M. Isenburg, P. Lindstrom, S. Gumhold, and J. Snoeyink. Large mesh simplification using processing sequences. In *Proceedings of IEEE Visualization*, pages 465–472, 2003.
7. M. Isenburg and S. Gumhold. Out-of-core compression of gigantic polygon meshes. In *Proceedings of SIGGRAPH*, pages 935–942, 2003.
8. T. Lengauer. Black-white pebble games and graph separation. *Acta Informatica*, 16(4):465–475, 1981.
9. L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an intruder by mobile agents. In *Proceedings of the 14th ACM Symposium on Parallel Algorithms and Architectures*, pages 200–209, 2002.
10. M. Franklin, Z. Galil, and M. Yung. Eavesdropping games: A graph theoretic approach to privacy in distributed systems. *Journal of the ACM*, 47(2):225–243, 2000.
11. E. H. Spafford and D. Zamboni. Intrusion detection using autonomous agents. *Computer Networks*, 34(4):547–570, 2000.
12. J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Computing Surveys*, 34(3):313–356, 2002.
13. K. Diks, H. N. Djidjev, O. Sykora, and I. Vrto. Edge separators of planar and outerplanar graphs with applications. *Journal of Algorithms*, 14:258–279, 1993.
14. G. L. Miller. Finding small simple cycle separators for 2-connected planar graphs. *Journal of Computer and System Sciences*, 32:265–279, 1986.
15. N. Megiddo, S. Hakimi, M. Garey, D. Johnson, and C. Papadimitriou. The complexity of searching a graph. *Journal of the ACM*, 35(1):18–44, 1988.
16. F. Makedon and H. Sudborough. Minimizing width in linear layout. In *Proceedings of the 10th International Colloquium on Automata, Languages, and Programming*, volume 154 of *Lecture Notes in Computer Science*, pages 478–490. Springer-Verlag, 1983.