Optimally Cutting a Surface into a Disk*

Jeff Erickson† Sariel Har-Peled

University of Illinois at Urbana-Champaign
{jeffe,sariel}@cs.uiuc.edu
http://www.cs.uiuc.edu/~{jeffe,sariel}

ABSTRACT

We consider the problem of cutting a set of edges on a polyhedral manifold surface, possibly with boundary, to obtain a single topological disk, minimizing either the total number of cut edges or their total length. We show that this problem is NP-hard, even for manifolds without boundary and for punctured spheres. We also describe an algorithm with running time $n^{O(g+k)}$, where n is the combinatorial complexity, g is the genus, and k is the number of boundary components of the input surface. Finally, we describe a greedy algorithm that outputs a $O(\log^2 g)$-approximation of the minimum cut graph in $O(g^2 n \log n)$ time.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Geometrical problems and computations*; G.2.m [**Discrete Mathematics**]: Miscellaneous—*Combinatorial topology*

General Terms

Algorithms

Keywords

computational topology, polyhedral 2-manifold, polygonal schema, cut graph, NP-hardness, approximation

1. INTRODUCTION

Several applications of three-dimensional surfaces require information about underlying topological structure in addition to geometry. In some cases, we wish to simplify the surface topology, to facilitate algorithms that can be performed only if the surface is a topological disk.

Applications when this is important include surface parameterization [14, 29] and texture mapping [2, 28]. In the texture mapping problem, one wish to find a continuous mapping from the texture, usually a two-dimensional rectangular image, to the surface. Unfortunately, if the surface is not a topological disk, no such map exists. In such a case, the only feasible solution is to cut the surface so that it becomes a topological disk. (Haker *et al.* [18] present an algorithm for directly texture mapping models with the topology of a sphere, where the texture is also embedded on a sphere.) Of course, when cutting the surface, one would like to find the best possible cut under various considerations. For example, one might want to cut the surface so that the resulting surface can be textured mapped with minimum distortion [14, 29]. To our knowledge, all current approaches for this cutting problem either rely on heuristics with no quality guarantees or require the user to perform this cutting beforehand [14, 28].

The problem of just cutting a surface into a topological disk is not trivial by itself. Lazarus *et al.* [25] presented and implemented two algorithms for computing a canonical polygonal schema of an orientable surface of complexity n and with genus g, in time $O(gn)$, simplifying an earlier algorithm of Vegter and Yap [34]. Computing such a schema requires finding $2g$ cycles, all passing through a common basepoint in $\mathcal{M}$, such that cutting along those cycles breaks $\mathcal{M}$ into a topological disk. Since these cycles must share a common point, it is easy to find examples where the overall size of those cycles is $\Omega(gn)$. Furthermore, those cycles share several edges and are visually unsatisfying.

For most applications, computing a canonical schema is overkill. It is usually sufficient to find a collection of edges whose removal transforms the surface into a topological disk. We call such a set of edges a *cut graph*; see Figure 1 for an example. Cut graphs have several advantages. First, they are compact. Trivially, any cut graph contains at most n edges of the surface mesh, much less than any canonical schema in the worst case, although we expect it to be much smaller in practice. Second, it is quite easy to construct a cut graph for an arbitrary polyhedral surface in $O(n)$ time, using a breadth-first search of the dual graph [9], or simply taking a maximal set of edges whose complement is connected [25]. Finally, the cut graph has an extremely simple structure: a tree with $O(g)$ additional edges. As such, it should be easier to manipulate algorithmically than other representations. For example, Dey and Schipper [9] describe fast algorithms

*See http://www.cs.uiuc.edu/~jeffe/pubs/schema.html for the most recent version of this paper.

†Partially supported by a Sloan Fellowship, NSF CAREER award CCR-0093348, and NSF ITR grant DMR-0121695.

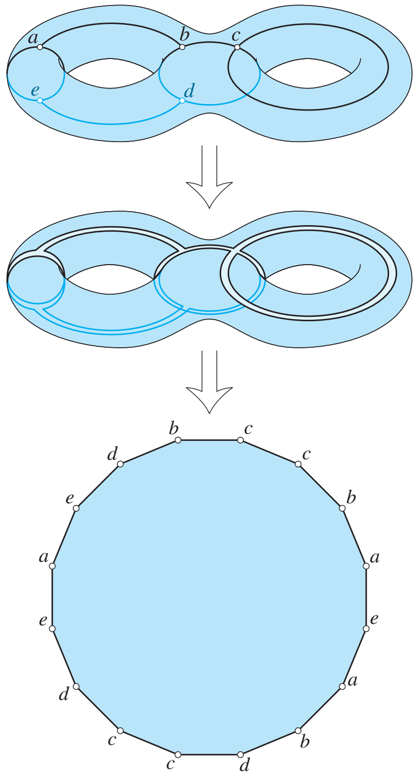

Figure 1. A cut graph for a two-holed torus and its induced (non-canonical) polygonal schema.

to determine whether a curve is contractible, or two curves are homotopic, using an arbitrary cut graph instead of a canonical schema.

In this paper, we investigate the question of how find the "best" such cutting of a surface, restricting ourselves to cuts along the edges of the given mesh. Specifically, we want to find the smallest subset of edges of a polyhedral manifold surface $\mathcal{M}$, possibly with boundary, such that cutting along those edges transforms $\mathcal{M}$ into a topological disk. We also consider the weighted version of this problem, where each edge has an arbitrary non-negative weight and we want to minimize the total weight of the cut graph. The most natural weight of an edge is its Euclidean length, but we could also assign weights to take problem-specific considerations into account. For example, if we want to compute a texture mapping for a specific viewpoint, we could make visible edges more expensive, so that the minimum cut graph would minimize the number of visible edges used in the cuts. Our algorithms do not require the edge weights to satisfy the triangle inequality.

We show that the minimum cut graph of any polyhedral manifold $\mathcal{M}$ with genus g and k boundary components can be computed in $n^{O(g+k)}$ time. We also show that the problem is NP-hard in general, even if g or k is fixed. Finally, we present a simple and efficient greedy approximation algorithm for this problem. Our algorithm outputs a cut graph whose weight is a factor $O(\log^2 g)$ larger than optimal, in $O(g^2 n \log n)$ time.[1] If $g = 0$, the approximation factor is exactly 2. We plan to implement this algorithm and present experimental results in the near future.

[1] To simplify notation, we define $\log x = \max\{1, \lceil \log_2 x \rceil\}$.

2. BACKGROUND

Before presenting our new results, we review several useful notions from topology and describe related results in more detail. We refer the interested reader to Hatcher [21], Munkres [27], or Stillwell [30] for further topological background and more formal definitions. For related computational results, see the recent surveys by Dey, Edelsbrunner, and Guha [7] and Vegter [33].

A *2-manifold with boundary* is a set $\mathcal{M}$ such that every point $x \in \mathcal{M}$ lies in a neighborhood homeomorphic to either the plane $\mathbb{R}^2$ or a closed halfplane. The points with only halfplane neighborhoods constitute the *boundary* of $\mathcal{M}$; the boundary consists of zero or more disjoint circles. This paper will consider only *compact* manifolds, where every infinite sequence of points has a convergent subsequence.

The *genus* of a 2-manifold $\mathcal{M}$ is the maximum number of disjoint non-separating cycles $\gamma_1, \gamma_2, \ldots, \gamma_g$ in $\mathcal{M}$; that is, $\gamma_i \cap \gamma_j = \varnothing$ for all i and j, and $\mathcal{M} \setminus (\gamma_1 \cup \cdots \cup \gamma_g)$ is connected. For example, a sphere and a disc have genus 0, a torus and a Möbius strip have genus 1, and a Klein bottle has genus 2.

A manifold is *orientable* if it has two distinct sides, and *non-orientable* if it has only one side. Although many geometric applications use only orientable 2-manifolds (primarily because non-orientable manifolds without boundary cannot be embedded in $\mathbb{R}^3$ without self-intersections) our results will apply to non-orientable manifolds as well. Every (compact, connected) 2-manifold with boundary is characterized by its orientability, its genus g, and the number k of boundary components [15].

A *polyhedral* 2-manifold is constructed by gluing closed simple polygons edge-to-edge into a *cell complex*: the intersection of any two polygons is either empty, a vertex of both, or an edge of both. We refer to the component polygons has *facets*. (Since the facets are closed, every polyhedral manifold is compact.) For any polyhedral manifold $\mathcal{M}$, the number of vertices and facets, minus the number of edges, is the *Euler characteristic* χ of $\mathcal{M}$. Euler's formula [13] implies that χ is an invariant of the underlying manifold, independent of any particular polyhedral representation; $\chi = 2 - 2g - k$ if the manifold is orientable, and $\chi = 2 - g - k$ if the manifold is non-orientable. Euler's formula implies that if $\mathcal{M}$ has v vertices, then $\mathcal{M}$ has at most $3v - 6 + 6g$ edges and at most $2v - 4 + 4g - k$ facets, with equality for orientable manifolds where every facet and boundary circle is a triangle. We let $n \le 6v - 10 + 10g - k$ denote the total number of facets, edges, and vertices in $\mathcal{M}$.

The 1-*skeleton* $\mathcal{M}_1$ of a polyhedral manifold $\mathcal{M}$ is the graph consisting of its vertices and edges. We define a *cut graph* G of $\mathcal{M}$ as a subgraph of $\mathcal{M}_1$ such that $\mathcal{M} \setminus G$ is homeomorphic to a disk.[2] The disk $\mathcal{M} \setminus G$ is known as a *polygonal schema* of $\mathcal{M}$. Each edge of G appears twice on the boundary of polygonal schema $\mathcal{M} \setminus G$, and we can obtain $\mathcal{M}$ by gluing together these corresponding boundary edges. Finding a cut graph of $\mathcal{M}$ with minimum total length is clearly equivalent to to finding a polygonal schema of $\mathcal{M}$ with minimum perimeter.

Any 2-manifold has a so-called *canonical* polygonal schema, whose combinatorial structure depends only on the genus g,

[2] Cut graphs are generalizations of the *cut locus* of a manifold $\mathcal{M}$, which is essentially the geodesic medial axis of a single point.

the number of boundary components k, and whether the manifold is orientable. The canonical schema of an orientable manifold is a $(4g+3k)$-gon with successive edges labeled

$$x_1, y_1, \bar{x}_1, \bar{y}_1, \ldots, x_g, y_g, \bar{x}_g, \bar{y}_g, z_1, e_1, \bar{z}_1, \ldots, z_k, e_k, \bar{z}_k;$$

for a non-orientable manifold, the canonical schema is a $(2g+3k)$-gon with edge labels

$$x_1, x_1, \ldots, x_g, x_g, z_1, e_1, \bar{z}_1, \ldots, z_k, e_k, \bar{z}_k.$$

Every pair of corresponding edges x and $\bar{x}$ is oriented in opposite directions. Gluing together corresponding pairs in the indicated directions recovers the original manifold, with the unmatched edges e_i forming the boundary circles. For a manifold $\mathcal{M}$ without boundary, a *reduced* polygonal schema is one where all the vertices are glued into a single point in $\mathcal{M}$; canonical schemata of manifolds without boundary are reduced. We emphasize that the polygonal schemata constructed by our algorithms are neither necessarily canonical nor necessarily reduced.

Dey and Schipper [9] describe an algorithm to construct a reduced, but not necessarily canonical, polygonal schema for any triangulated orientable manifold without boundary in $O(n)$ time. Essentially, their algorithm constructs an arbitrary cut graph G by depth-first search, and and then shrinks a spanning tree of G to a single point. (See also Dey and Guha [8].) Vegter and Yap [34] developed an algorithm to construct a canonical schema in optimal $O(gn)$ time and space. Two simpler algorithms with the same running time were later developed by Lazarus *et al.* [25]. The "edges" of the polygonal schemata produced by all these algorithms are (possibly overlapping) paths in the 1-skeleton of the input manifold. We will modify one of the algorithm of Lazarus *et al.* to construct and essential cycles and nearly-minimal cut graphs.

3. COMPUTING MINIMUM CUT GRAPHS IS NP-HARD

In this section, we prove that finding a minimum cut graph of a triangulated manifold is NP-hard. We consider two extreme cases: boundary but no genus, and genus but no boundary. Both reductions are from the *rectilinear Steiner tree* problem: Given a set P of n points from a $m \times m$ square grid in the plane, find the shortest connected set of horizontal and vertical line segments that contains every point in P. This problem is NP-hard, even if m is bounded by a polynomial in n [17]. Our reduction uses the *Hanan grid* of the points, which is obtained by drawing horizontal and vertical lines through each point, clipped to the bounding box of the points. At least one rectilinear Steiner tree of the points is a subset of the Hanan grid [20].

Theorem 3.1. *Computing the length of the minimum (weighted or unweighted) cut graph of a triangulated punctured sphere is NP-hard.*

Proof: Let P be a set of n points in the plane, with integer coordinates between 1 and m. We construct a punctured sphere in $O(n^2)$ time as follows. Assume that P lies on the xy-plane in $\mathbb{R}^3$. We modify the Hanan grid of P by replacing each terminal with a square of width $1/2n$, rotated 45 degrees so that its vertices lie on the neighboring edges.

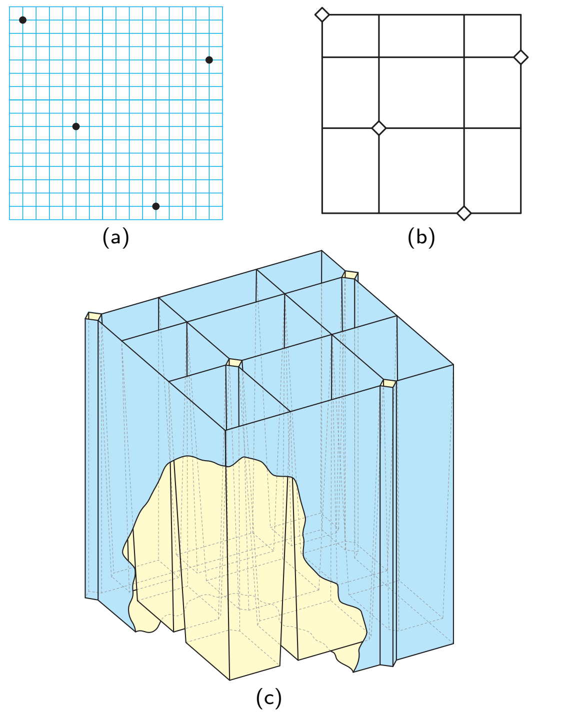

Figure 2. (a) A set of integer points. (b) The modified Hanan grid. (c) A cut-away view of the resulting punctured sphere.

These squares will form the punctures. We then attach a *basin* under each face f of the modified Hanan grid, by joining the boundary of f to a slightly scaled copy of f on the plane $z = -n^2$. We also attach a basin of depth n^2+1 to the boundary of the entire modified Hanan grid. The side facets of each basin are trapezoids. The basins are tapered so that adjacent basins intersect only on the modified Hanan grid. Triangulating this surface arbitrarily, we obtain a polyhedral sphere $\mathcal{M}$ with n punctures and overall complexity $O(n^2)$. See Figure 2.

Let G^* be a minimum weighted cut graph of $\mathcal{M}$. We easily observe that G^* contains only "long" edges from the modified Hanan grid and contains at least one vertex of every puncture. Thus, the edges of G^* are in one-to-one correspondence with the edge of a rectilinear Steiner tree of P.

For the unweighted case, we modify the original $m \times m$ integer grid instead of the Hanan grid. To create a punctured sphere, we replace each terminal point with a small diamond as above. We then fill in each modified grid cell with a triangulation, chosen so that the shortest path between any two points on the boundary of any cell stays on the boundary of that cell; this requires a constant number of triangles per cell. The resulting manifold $\mathcal{M}'$ has complexity $O(m^2)$. By induction, the shortest path between any two points on the modified grid lies entirely on the grid. Thus, any minimal unweighted cut graph of $\mathcal{M}'$ contains only edges from the modified grid. It follows that if the minimum unweighted cut graph of $\mathcal{M}'$ has r edges, the length of any rectilinear Steiner tree of P is exactly r. □

We can easily generalize the previous proof to manifolds with higher genus, with or without boundary, oriented or not, by attaching small triangulated tori or cross-caps to any subset of punctures.

Theorem 3.2. *Computing the length of the minimum (weighted or unweighted) cut graph of a triangulated manifold with boundary, with any fixed genus or with any fixed number of boundary components, is NP-hard.*

4. COMPUTING MINIMUM CUT GRAPHS ANYWAY

In this section, we describe an algorithm to compute the minimum cut graph of a polyhedral manifold in $n^{O(g+k)}$ time. For manifolds with constant Euler characteristic, our algorithm runs in polynomial time.

Our algorithm is based on the following characterization of the minimum cut graph as the union of shortest paths. A *branch point* of a cut graph is any vertex with degree greater than 2. A simple path in a cut graph from one branch point or boundary point to another, with no branch points in its interior, is called a *cut path*.

Lemma 4.1. *Let $\mathcal{M}$ be a polyhedral 2-manifold, possibly with boundary, and let G^* be a minimum cut graph of $\mathcal{M}$. Any cut path in G^* can be decomposed into two equal-length shortest paths in $\mathcal{M}_1$.*

Proof: Let G be an arbitrary cut graph of $\mathcal{M}$, and consider a cut path between two (not necessarily distinct) branch points a and c of G. Let b be the midpoint of this path, and let α and β denote the subpaths of β from b to a and from b to c, respectively. Note that β may lie in the interior of an edge of $\mathcal{M}_1$. Finally, suppose α is *not* the shortest path from b to a in $\mathcal{M}_1$. To prove the lemma, it suffices to show that G is not the shortest cut graph of $\mathcal{M}$.

Let α' be the true shortest path from b to a. Clearly, α' is not contained in G. Walking along α' from b to a, let s be the first vertex whose following edge is not in G, and let t be the first vertex in G whose preceding edge is not in G. (Note that s and t may be joined by a single edge in $\mathcal{M} \setminus G$.) Finally, let $\sigma' \subset \alpha'$ be the shortest path from b to s, and let $\tau' \subset \alpha'$ be the shortest path from s to t. Thus, τ' is the first maximal subpath of α' whose interior lies in $\mathcal{M} \setminus G$. See Figure 3.

The subpath τ' cuts $\mathcal{M} \setminus G^*$ into two smaller disks. We claim that some subpath τ of either α or β appears on the boundary of both disks and is longer than τ'. Our claim implies that cutting $\mathcal{M} \setminus G$ along τ' and regluing the pieces along τ gives us a new polygonal schema with smaller perimeter, and thus a new cut graph shorter than G. See Figure 3 for an example.

We prove our claim by exhaustive case analysis. First consider the case where the manifold $\mathcal{M}$ is orientable. We can subdivide the entire boundary of the disk $\mathcal{M} \setminus G$ into six paths labeled consecutively $\alpha, \beta, \gamma, \bar{\beta}, \bar{\alpha}, \delta$. Here, $\bar{\alpha}$ and $\bar{\beta}$ are the corresponding copies of α and β in the polygonal schema. Because $\mathcal{M}$ is orientable, α and $\bar{\alpha}$ have opposite orientations, as do β and $\bar{\beta}$. Either or both of γ and δ could be empty. See the lower left part of Figure 3. The subpath τ' can enter the interior of the disk $\mathcal{M} \setminus D$ from four of these six paths (α, β, $\bar{\alpha}$, and $\bar{\beta}$) and leave the interior of the disk through any of the six paths.

Suppose τ' enters the interior of $\mathcal{M} \setminus G$ from α; the other three cases are symmetric. Figure 4 shows the six essentially different ways for τ' to leave the interior of $\mathcal{M} \setminus D$. In

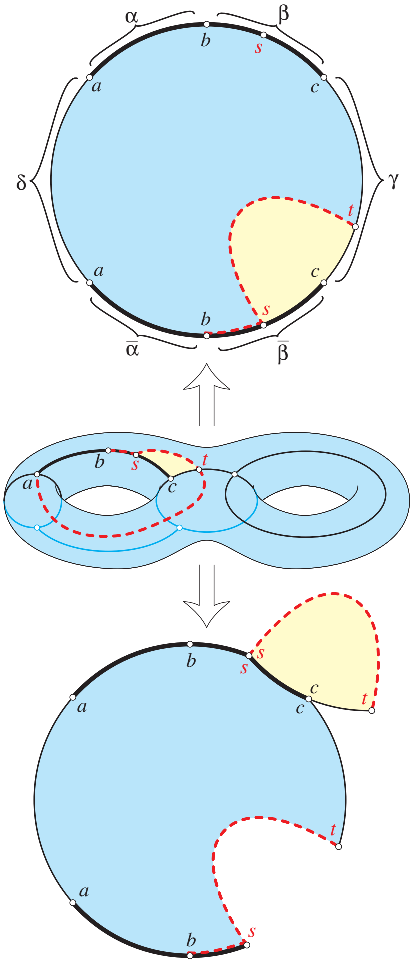

Figure 3. If the dashed path from a to b is shorter than the equal-length paths ab and bc in the cut graph, then the cut graph can be shortened by cutting and regluing.

each case, we easily verify that after cutting along τ', some subpath τ of either α or β is on the boundary of both disks. Specifically, if τ' leaves through α or $\bar{\alpha}$, then τ to be the subpath of α from s to t. If τ' leaves through β or $\bar{\beta}$, then τ is the subpath of β from b to t. If τ' leaves through γ, then $\tau = \beta$. Finally, if τ' leaves through δ, then τ is the subset of α from a to s.

If $\mathcal{M}$ is non-orientable, the path $\alpha\beta$ could appear either with the same orientation or with opposite orientations on the boundary of the disk $\mathcal{M} \setminus G$. If the orientations are opposite, the previous case analysis applies immediately. Otherwise, the boundary can be subdivided into six paths labeled consecutively $\alpha, \beta, \gamma, \alpha, \beta, \delta$. Without loss of generality, τ' enters the interior of $\mathcal{M} \setminus G$ from α and leaves through any of these six paths. The six cases are illustrated in Figure 5. Again, we easily verify that in each case, some subpath τ of either α or β is on the boundary of both disks. We omit further details. $\square$

For any cut graph G of a manifold $\mathcal{M}$, we define the corresponding *reduced* cut graph $\hat{G}$ has follows. We first augment the cut graph by adding all the boundary edges of $\mathcal{M}$. Next, we remove every vertex of degree one and its only edge, making the graph 2-edge-connected. Finally, we replace each

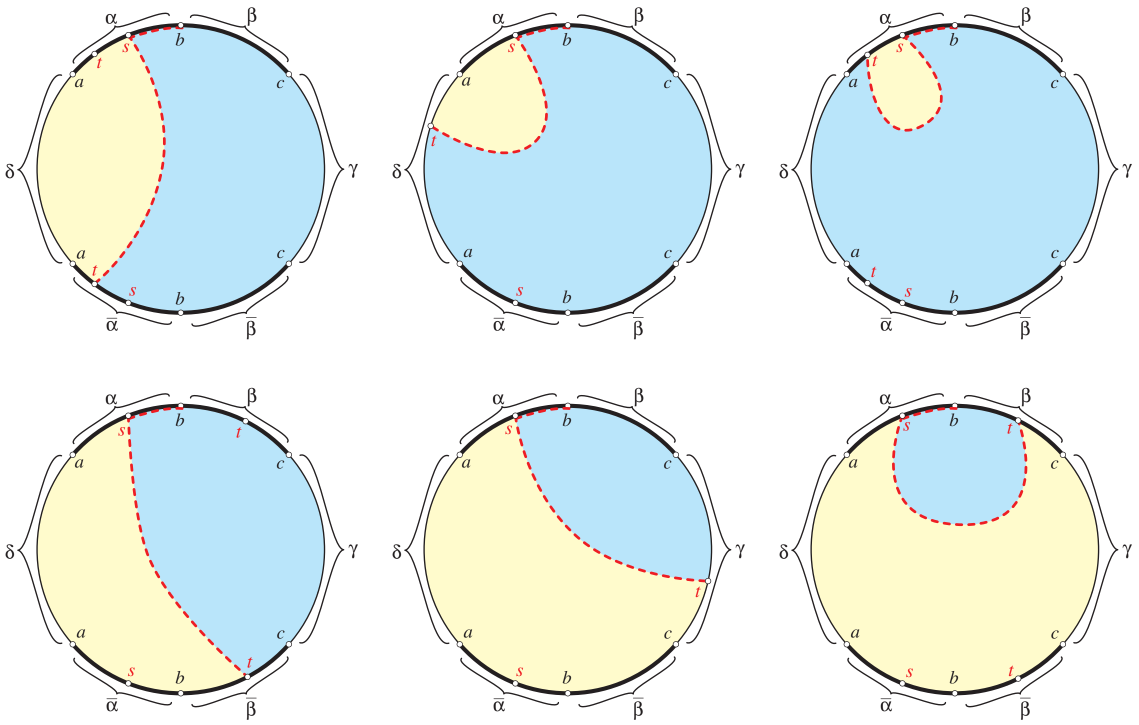

Figure 4. Six cases for the proof of Lemma 4.1 for orientable manifolds; all other cases are reflections of these. In each case, some subpath of α or β appears on the boundary of both sub-disks.

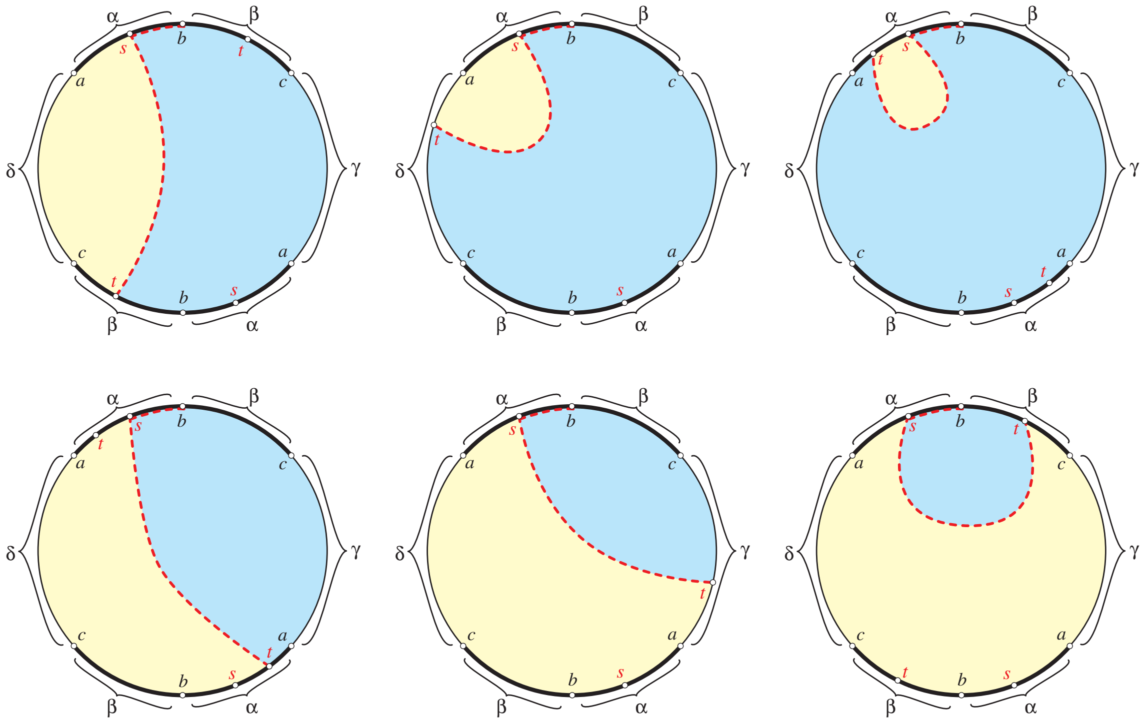

Figure 5. Six additional cases for the proof of Lemma 4.1 for non-orientable manifolds; all other cases are reflections or rotations of these. In each case, some subpath of α or β appears on the boundary of both sub-disks.

maximal path through degree-2 vertices with a single edge, so that each vertex in the reduced cut graph $\hat{G}$ has degree at least 3. Every vertex of $\hat{G}$ is either a branch point or a boundary point of G, and every edge of $\hat{G}$ corresponds to either a cut path or a boundary path in G. However, in general, not all branch points and cut paths are represented in $\hat{G}$.

Lemma 4.2. *Any reduced cut graph $\hat{G}$ of a manifold $\mathcal{M}$ has at most $4g+2k-2$ vertices and $6g+3k-3$ edges.*

Proof: Let v and e denote the number of vertices and edges in $\hat{G}$, respectively. If any vertex in $\hat{G}$ has degree $d \geq 4$, we can replace it with $d-3$ trivalent vertices and $d-3$ new edges of length zero. Thus, in the worst case, every vertex in $\hat{G}$ has degree exactly 3, which implies that $3v = 2e$. Since $\hat{G}$ is embedded in $\mathcal{M}$ with a single face, Euler's formula implies that $v - e + 1 = \chi = 2 - 2g - k$ if $\mathcal{M}$ is orientable, and $v - e + 1 = \chi = 2 - g - k$ if $\mathcal{M}$ is non-orientable. It follows that $v \leq 4g + 2k - 2$ and $e \leq 6g + 3k - 3$, as claimed. □

Our minimum cut graph algorithm exploits Lemma 4.1 by composing potential minimum cut graphs out of $O(g + k)$ shortest paths. Unfortunately, a single pair of nodes in $\mathcal{M}$ could be joined by $2^{\Omega(n)}$ shortest paths, in $2^{\Omega(g+k)}$ different isotopy classes, in the worst case. To avoid this combinatorial explosion, we can add a random infinitesimal weight $\varepsilon \cdot w(e)$ to each edge e. The Isolation Lemma of Mulmuley, Vazirani, and Vazirani [26] implies that if the weights $w(e)$ are chosen independently and uniformly from the integer set $\{1, 2, \ldots, n^2\}$, all shortest paths are unique with probability at least $1 - 1/n$; see also [6, 23].[3]

We are now finally ready to describe our minimum cut graph algorithm.

Theorem 4.3. *The minimum cut graph of a polyhedral 2-manifold $\mathcal{M}$ with genus g and k boundary components can be computed in time $n^{O(g+k)}$.*

Proof: We begin by computing the shortest path between every pair of vertices in $\mathcal{M}$ in $O(n^2 \log n)$ by running Dijkstra's single-source shortest path algorithm for each vertex [10, 22], breaking ties using random infinitesimal weights as described above. Once these shortest paths and midpoints have been computed, our algorithm enumerates by brute force every possible cut graph that satisfies Lemmas 4.1 and 4.2, and returns the smallest such graph.

Each cut graph is specified by a set V of up to $4g+2k-2$ vertices of $\mathcal{M}$, a set E of up to $6g + 3k - 3$ edges of $\mathcal{M}$, a trivalent multigraph $\hat{G}$ with vertices V, and a assignment of edges in E to edges in $\hat{G}$. Each edge (v, w) of $\hat{G}$ is assigned a unique edge $e \in E$ to define the corresponding cut path in $\mathcal{M}$. This cut path is the concatenation of the shortest path from v to e, e itself, and the shortest path from e to w. If the midpoint of this cut path is not in the interior of e, we declare the cut path invalid, since it violates Lemma 4.1. (Because shortest paths between vertices are unique, the midpoint of any cut path in the minimal cut graph must lie in the interior of an edge.) If all the cut paths are valid, we then check that every pair of cut paths is disjoint, except possibly at their endpoints, and that removing all the cut paths from $\mathcal{M}$ leaves a topological disk.

Our brute-force algorithm considers $\binom{n}{4g+2k-2}$ different vertex sets V, $\binom{n}{6g+3k-2}$ different edge sets E, $\binom{(4g+2k-2)^2}{6g+3k-2}$ different graphs $\hat{G}$, and $(6g + 3k - 2)!$ different edge assignments. Thus, $n^{O(g+k)}$ potential cut graphs are considered altogether. The validity of each potential cut graph can be checked in $O(n)$ time. □

[3]Alternately, if we choose $w(e)$ uniformly from the *real* interval $[0, 1]$, shortest paths are unique with probability 1. This may sound unreasonable, but recall that no polynomial-time algorithm is known to compare sums of square roots of integers in any model of computation that does not include square root as a primitive operation[5]. Thus, to compute Euclidean shortest paths in a geometric graph with integer vertex coordinates, we must either assume exact real arithmetic or (grudgingly) accept some approximation error [16].

5. APPROXIMATE MINIMUM CUT GRAPHS

In this section, we describe a simple polynomial-time greedy algorithm to construct an approximate minimum cut graph for any polyhedral manifold $\mathcal{M}$.

To handle manifolds with boundary, it will be convenient to consider the following simplified form. Given a manifold $\mathcal{M}$ with genus g and k boundary components, the corresponding *punctured* manifold $(\overline{\mathcal{M}}, P)$ consists of a manifold $\overline{\mathcal{M}}$ with the same genus as $\mathcal{M}$ but without boundary, and a set P of k points in $\overline{\mathcal{M}}$, called *punctures.* To construct $\overline{\mathcal{M}}$, we contract every boundary component of $\mathcal{M}$ to a single point, which becomes one of the punctures in P.[4] If any vertex of $\mathcal{M}$ has multiple edges to the same boundary component, $\overline{\mathcal{M}}$ contains only the edge with smallest weight, breaking ties using the Isolation Lemma as above. If $\mathcal{M}$ has no boundary, then $\overline{\mathcal{M}} = \mathcal{M}$ and $P = \varnothing$. This reduction is motivated by the following trivial observation.

Lemma 5.1. *The minimum cut graph of $\mathcal{M}$ has the same length as the minimum cut graph of $\overline{\mathcal{M}}$ that includes every puncture in P.*

A simple cycle γ in $\mathcal{M}$ is *essential* if it does not bound a disk or an annulus. In terms of punctured manifolds, a cycle in $\overline{\mathcal{M}}$ is essential if it does not bound a disk containing less than two punctures in P.

Our approximation algorithm works as follows. We repeatedly cut along a short essential cycle until our surface becomes a collection of punctured spheres, connect the punctures on each component by cutting along a minimum spanning tree, and finally (if necessary) reglue some previously cut edges to obtain a single disk. The resulting cut graph is composed of a subset of the edges of the short essential cycles and all the edges of the minimum spanning forest.

Before we describe our algorithm in more detail, we first describe how to execute each of the cutting operations and how the lengths of each type of cut compare to the minimum cut graph length.

5.1 Shortest Essential Cycles

We now describe an algorithm to compute the shortest essential cycle in the 1-skeleton $\mathcal{M}_1$ of a polyhedral 2-manifold $\mathcal{M}$. Although our most efficient approximation

[4]We could simulate this contraction by artificially assigning every boundary edge of $\mathcal{M}$ a weight of zero, although this would require a few simple changes in our algorithms.

algorithm for cut graphs does not use require the shortest essential cycle, we believe this algorithm is of independent interest. Our algorithm uses a combination of Dijkstra's single-source shortest path algorithm [10] and a modification of the canonical polygonal schema algorithm of Lazarus *et al.* [25].

The algorithm of Lazarus *et al.* builds a connected subset S of a triangulated manifold, starting with a single triangle and adding new triangles one at a time on the boundary. If a new triangle intersects the boundary of S in more than one component, the algorithm checks which of the following three cases holds: (1) $\mathcal{M} \setminus S$ is connected; (2) neither component of $\mathcal{M} \setminus S$ is a disk; or (3) one component of $\mathcal{M} \setminus S$ is a disk. In the first two cases, S contains an essential cycle. In case (1), the check runs in $O(n)$ time; in the other two cases, the running time of the check is proportional to the size of the smaller component of $\mathcal{M} \setminus S$. In case (3), the algorithm adds the disk component to S and continues searching the other component of $\mathcal{M} \setminus S$. If we run this algorithm until either case (1) or case (2) holds, the total running time is $O(n)$. See Lazarus *et al.* [25] for further details.

Lemma 5.2. *Let u be a vertex of a polyhedral 2-manifold $\mathcal{M}$. The shortest essential cycle in $\mathcal{M}_1$ that contains u can be computed in $O(n \log n)$ time.*

Proof: We find the shortest essential cycle through u by simulating a circular wave expanding from u. Whenever the wave touches itself, either we have the shortest essential cycle through u, or one component of the wave bounds a disk in $\mathcal{M}$ and we can continue expanding the other component.

We modify the algorithm of Lazarus *et al.* in three ways. First, S is no longer a set of triangles but a more general connected subset of vertices, edges, and facets of $\mathcal{M}$. Initially, S contains only the source vertex u. Second, we use Dijkstra's algorithm to determine the order for edges to be added. We add a facet to S only when all its vertices have been added to S, either directly or as part of another facet. We run the Lazarus topology check when S is no longer simply connected, that is, when we add a new edge vw with both endpoints on the boundary of S. (In the unweighted case, our algorithm behaves similarly to the 'wave traversal' algorithm of Axen and Edelsbrunner [1].) Our third change is that if $\mathcal{M} \setminus S$ is disconnected, we continue only if one component of $\mathcal{M} \setminus S$ is a disk *or an annulus*. In that case, we add the disk or annulus component of $\mathcal{M} \setminus S$ to S, discard the vertices of that component from the Dijkstra priority queue, and continue searching in the other component. Otherwise, we have found the shortest essential cycle through u, consisting of the shortest path from u to v, the edge vw, and the shortest path from w to u.

Altogether, Dijkstra's algorithm requires $O(n \log n)$ time. By our earlier discussion, the total time spent modifying the wave set S is only $O(n)$. Thus, the total running time of our algorithm is $O(n \log n)$. □

Running this algorithm once for every vertex of $\mathcal{M}$ gives us the following:

Corollary 5.3. *Let $\mathcal{M}$ be an polyhedral 2-manifold. The shortest essential cycle in $\mathcal{M}_1$ can be computed in $O(n^2 \log n)$ time.*

The following lemma relates the length of the shortest essential cycle to the length of the minimum cut graph.

Lemma 5.4. *Let G be any cut graph of a 2-manifold $\overline{\mathcal{M}}$ with genus g and no boundary. The shortest cycle in G contains $O((\log g)/g)$ of the total length of G.*

Proof: First consider the reduced cut graph $\hat{G}$, constructed by repeatedly contracting any edge with a vertex of degree less than three, as in Section 4. Every vertex in $\hat{G}$ has degree at least 3. Without loss of generality, assume that every vertex in $\hat{G}$ has degree exactly 3, splitting each high-degree vertex into a tree of degree-3 vertices if necessary, as in the proof of Lemma 4.2. A straightforward counting argument implies that any trivalent graph whose girth (minimum cycle length) is c must have at least $3\sqrt{2} \cdot 2^{c/2} - 2$ vertices if c is odd, and at least $2 \cdot 2^{c/2} - 2$ vertices if c is even [4]. By Lemma 4.2, $\hat{G}$ has at most $4g - 2$ vertices, so $\hat{G}$ must have a cycle $\hat{\gamma}$ with most $2(\lg g + 1) = O(\log g)$ edges.

Starting with $\hat{G}_0 = \hat{G}$, we inductively define a sequence of reduced graphs $\hat{G}_1, \hat{G}_2, \ldots$ as follows. For each $i > 0$, let $\hat{\gamma}_i$ denote the shortest cycle in $\hat{G}_{i-1}$. We obtain $\hat{G}_i$ by reducing the graph $\hat{G}_{i-1} \setminus \hat{\gamma}_i$, or equivalently, removing the vertices of $\hat{\gamma}_i$ and all their edges, and then contracting $|\hat{\gamma}_i|$ nearby length-2 paths to single edges. Our earlier argument implies that each cycle $\hat{\gamma}_i$ has at most $2(\lg g + 1)$ edges. Thus, for each i, we have $|E(\hat{G}_i)| = |E(\hat{G}_{i-1})| - 6(\lg g + 1)$. Since the original reduced cut graph $\hat{G}$ contains at least g edges, it follows that we can repeat this process at least $g/6(\lg g + 1)$ times.

Let γ_i denote the cycle in the original cut graph G corresponding to $\hat{G}_i$. By our construction, γ_i and γ_j are disjoint for all $i \neq j$, so we have a set of at least $g/6(\lg g + 1)$ disjoint cycles in G. At least one of these cycles has length at most $6(\lg g + 1)/g = O((\log g)/g)$ times the total length of G. □

Corollary 5.5. *For any 2-manifold $\overline{\mathcal{M}}$ with genus g and no boundary, the length of the shortest essential cycle is at most $O((\log g)/g)$ times the length of the minimum cut graph.*

5.2 Nearly-Shortest Essential Cycles

As we will argue shortly, computing short essential cycles is the bottleneck in our approximate cut graph algorithm. Fortunately, exact minimum essential cycles are not necessary. We can speed up our cut graph algorithm, without significantly increasing the approximation factor, by searching for an essential cycle at most twice as long as the shortest.

Our approximation algorithm works as follows. First, we compute a set of shortest paths (in fact, a cut graph) that intersects every essential cycle in the manifold $\overline{\mathcal{M}}$. Then we contract each shortest path π in this set to a point, and find the shortest essential cycle through that point, as described by Lemma 5.2.

Lemma 5.6. *Let π be a shortest path between two vertices in a polyhedral 2-manifold $\mathcal{M}$, and let γ^* be the shortest essential cycle in $\mathcal{M}_1$ that intersects π. In $O(n \log n)$ time, one can compute an essential cycle γ in $\mathcal{M}$ such that $|\gamma| \leq 2|\gamma^*|$.*

Proof: Let $\mathcal{M}'$ be the manifold obtained by contracting the shortest path π to a single vertex v. Because π has no cycles, $\mathcal{M}'$ has the same topological type as $\mathcal{M}$. Let γ' be the shortest essential cycle in $\mathcal{M}'$ that passes through v. Clearly, $|\gamma'| \leq |\gamma^*|$.

We construct a cycle γ in $\mathcal{M}$ by concatenating two paths α and β, where α contains the edges of γ' and β is the subpath of π between the endpoints of α. The sequence of edge contractions that transforms M to $\mathcal{M}'$ also transforms γ' to γ. Hence, γ' is an essential cycle of $\mathcal{M}$. Because β is a subpath of a shortest path, β is actually the shortest path between the endpoints of α, so $|\beta| \leq |\alpha| = |\gamma'|$. It follows that $|\gamma| = |\alpha| + |\beta| \leq 2|\gamma'| \leq 2|\gamma^*|$. □

This lemma suggests a natural algorithm for finding a short essential cycle: Compute a set of shortest paths that intersect every essential cycle, and then run the algorithm from Lemma 5.6 for each path in this set.

Lemma 5.7. *In $O(gn \log n)$ time, one can compute a set Π of $O(g)$ shortest paths on $\overline{\mathcal{M}}_1$ such that every essential cycle in $\overline{\mathcal{M}}_1$ intersects at least one path in Π.*

Proof: We use another variant of the algorithm of Lazarus *et al.* [25], replacing the simple breadth-first search with Dijkstra's shortest path algorithm. The cut graph G computed by our algorithm consists of a shortest-path tree T from the starting vertex v, plus an additional set E' of $O(g)$ edges.

Let Π be the set of $O(g)$ shortest paths from v to the endpoints of E', and let $G' = \Pi \cup E'$. We easily observe that G' is also a cut graph; that is, $\overline{\mathcal{M}} \setminus G$ is a topological disk. Thus, every essential cycle in $\overline{\mathcal{M}}$ intersects G'. Since every vertex of G' is also a vertex of Π, every essential cycle intersects at least one path in Π. □

Corollary 5.8. *Let $\overline{\mathcal{M}}$ be a polyhedral 2-manifold with genus g and no boundary, and let γ^* be its shortest essential cycle. In $O(gn \log n)$ time, one can compute an essential cycle γ of $\overline{\mathcal{M}}$ such that $|\gamma| \leq 2|\gamma^*|$.*

5.3 Puncture-Spanning Trees

A second component of our cut graph algorithm is computing the *minimum puncture-spanning tree* of a punctured manifold $(\overline{\mathcal{M}}, P)$: the minimum spanning tree of the punctures P in the shortest-path metric of $\overline{\mathcal{M}}_1$.

Lemma 5.9. *The minimum puncture-spanning tree of any punctured polyhedral 2-manifold $(\overline{\mathcal{M}}, P)$ can be computed in $O(n \log n)$ time.*

Proof: We simulate Prim's minimum spanning tree algorithm by adding shortest puncture-to-puncture paths one at a time in increasing order of length [31]. To compute the shortest paths, we simultaneously propagate wavefronts from all k punctures using Dijkstra's algorithm. Whenever two wavefronts (*i.e.*, two growing shortest-path trees) collide, we add a new edge to the evolving minimum spanning tree and merge those two wavefronts. To implement this algorithm efficiently, we maintain the wavefronts in a union-find data structure. The resulting running time is $O(n \log n)$. □

Lemma 5.10. *The length of the minimum puncture-spanning tree of any punctured manifold $(\overline{\mathcal{M}}, P)$ is at most twice the length of any cut graph of $(\overline{\mathcal{M}}, P)$.*

Proof: The *minimum Steiner tree* of P is the subgraph of $\overline{\mathcal{M}}_1$ of minimum total weight that includes every point in P. Since any cut graph of $(\overline{\mathcal{M}}, P)$ must touch every puncture, no cut graph is shorter than this minimum Steiner tree. On the other hand, the minimum spanning tree of P has at most twice the length of the minimum Steiner tree [24, 31]. □

5.4 Greedy Algorithm Analysis

We now have all the components of our greedy cut graph algorithm. At any stage of the algorithm, we have a (possibly disconnected) punctured manifold $(\overline{\mathcal{M}}, P)$. Our algorithm repeatedly cuts along a short essential cycle of $\overline{\mathcal{M}}$, using the algorithm of Corollary 5.8. This cut creates one or two new boundary circles, which we collapse to new punctures. When the manifold is reduced to a collection of punctured spheres, we cut along the minimum puncture-spanning tree of each component using the algorithm in Lemma 5.9. Finally, we reglue some cut edges to obtain a single topological disk.

Each essential cycle cut is either a *separating cut*, which breaks a component of $\overline{\mathcal{M}}$ into two smaller components, each with non-trivial topology, or a *reducing cut*, which decreases the genus of some component of $\overline{\mathcal{M}}$ by 1. The algorithm performs at most $g-1$ separating cuts and exactly g reducing cuts. Thus, the overall running time of our algorithm is

$$(2g-1) \cdot O(gn \log n) + O(n \log n) = O(g^2 n \log n).$$

It remains only to analyze how well our greedy cut graph approximates the true minimum cut graph G^*. For any graph X, let $|X|$ denote its total length. We split the algorithm into phases numbered from g down to 1. In the ith phase, we cut along the shortest essential cycle in every component of the manifold whose genus is exactly i. Some phases may include no cuts. Let $(\overline{\mathcal{M}}_i, P_i)$ denote the punctured manifold at the beginning of the ith phase, and let G_i^* denote the union of the minimum cut graphs of its components. Since collapsing edges cannot increase the minimum cut graph length, we have $|G_i^*| \leq |G_g^*| = |G^*|$ for all i.

Let $\overline{\mathcal{M}}_{ij}$ denote the jth component of $\overline{\mathcal{M}}_i$ with genus i; let G_{ij}^* denote its minimum cut graph; let γ_{ij} denote the short essential cycle found by Corollary 5.8. (We easily observe that *any* cut graph of $\overline{\mathcal{M}}_{ij}$ must intersect this cycle.) Lemma 5.4 and Corollary 5.8 imply that $|\gamma_{ij}| \leq O((\log i)/i) \cdot |G_{ij}^*|$ for all i and j. Thus, we can bound the total length of all cuts in phase i as follows.

$$\sum_j |\gamma_{ij}| \leq \sum_j O((\log i)/i) \cdot |G_{ij}^*| \leq O((\log i)/i) \cdot |G_i^*|$$

Summing over all g phases, we conclude that the total length of all cycle cuts is at most

$$\sum_{i=1}^{g} O((\log i)/i) \cdot |G_i^*| = O(\log^2 g) \cdot |G^*|.$$

Similarly, Lemma 5.10 implies that the minimum puncture-spanning forest has length at most $2|G^*|$. Finally, regluing previously cut edges to obtain a single disk only reduces the length of the final cut graph. Thus, the final cut graph computed by our algorithm has length at most $O(\log^2 g) \cdot |G^*|$.

Theorem 5.11. *Given a polyhedral 2-manifold $\mathcal{M}$ with genus g and k boundary components, an $O(\log^2 g)$-approximation of its minimum cut graph can be constructed in $O(g^2 n \log n)$ time.*

6. OPEN PROBLEMS

We have developed new algorithms to compute exact and approximate minimal cut graphs for manifold surfaces with arbitrary genus and arbitrary boundary complexity. Our approximation algorithm is particularly simple. We plan to implement this algorithm and present experimental results in a future version of this the paper.

Our results suggest several open problems, the most obvious of which is to improve the running times and approximation factors of our algorithms. Is the minimum cut graph problem *fixed-parameter tractable* [11]? That is, can we compute exact minimum cut graphs in time $f(g, k) \cdot n^{O(1)}$ for some function f? The similarity to the Steiner problem offers some hope here, since the minimum Steiner tree of k nodes in an n-node graph can be computed in $O(3^k n + 2^k n^2 + n^3)$ time [12, 19]. How well can we approximate the minimum cut graph in nearly-linear time? More generally, is there a simple, practical, $O(1)$-approximation algorithm, like the MST approximation of Steiner trees? Unfortunately, the general Steiner tree problem is MAXSNP-hard [3], so an efficient $(1 + \varepsilon)$-approximation algorithm for arbitrary $\varepsilon > 0$ seems unlikely.

The approximation algorithm of Theorem 5.11 is somewhat indirect. It computes a short cut graph by repeatedly computing a 'reasonable' cut graph and then extracting a short essential cycle that interacts with this cut graph. It is natural to conjecture that one can compute such a short cut graph directly, resulting in a faster algorithm. In particular, we conjecture that an approximately minimum cut graph can be computed in $O(gn \log n)$ time.

Finally, can our ideas be applied to other useful families of curves on manifolds, such as homology generators (families of $2g$ cycles that intersect in g pairs) and pants decompositions (maximal sets of pairwise disjoint essential cycles [32])?

Acknowledgments. We would like to thank Herbert Edelsbrunner for an enlightening initial conversation. We are also grateful to Noga Alon, John Hart, Benjamin Sudakov, and Kim Whittlesey for helpful discussions.

References

[1] U. Axen and H. Edelsbrunner. Auditory Morse analysis of triangulated manifolds. *Mathematical Visualization*, 223–236, 1998. Springer-Verlag.

[2] C. Bennis, J.-M. Vézien, G. Iglésias, and A. Gagalowicz. Piecewise surface flattening for non-distorted texture mapping. *Computer Graphics* 25:237–246, 1991. *Proc. SIGGRAPH '91.*

[3] M. Bern and P. Plassman. The Steiner problem with edge lengths 1 and 2. *Inform. Proc. Lett.* 32(4):171–176, 1989.

[4] N. Biggs. Constructions for cubic graphs with large girth. *Elec. J. Combin.* 5:A1, 1998.

[5] J. Blömer. Computing sums of radicals in polynomial time. *Proc. 32nd Annu. IEEE Sympos. Found. Comput. Sci.*, 670–677, 1991.

[6] S. Chari, P. Rohatgi, and A. Srinivasan. Randomness-optimal unique element isolation with applications to perfect matching and related problems. *SIAM J. Comput.* 24(5):1036–1050, 1995.

[7] T. Dey, H. Edelsbrunner, and S. Guha. Computational topology. *Advances in Discrete and Computational Geometry*, 109–143, 1999. Contemporary Mathematics 223, American Mathematical Society.

[8] T. K. Dey and S. Guha. Transforming curves on surfaces. *J. Comput. Sys. Sci.* 58:297–325, 1999.

[9] T. K. Dey and H. Schipper. A new technique to compute polygonal schema for 2-manifolds with application to null-homotopy detection. *Discrete Comput. Geom.* 14:93–110, 1995.

[10] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269–271, 1959.

[11] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Monographs in Computer Science. Springer-Verlag, 1999.

[12] S. Dreyfus and R. Wagner. The Steiner problem in graphs. *Networks* 1:195–207, 1971.

[13] D. Eppstein. Seventeen proofs of Euler's formula: $V - E + F = 2$. *The Geometry Junkyard*, May 2001. ⟨http://www.ics.uci.edu/~eppstein/junkyard/euler/⟩.

[14] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Comput. Aided Geom. Design* 14(4):231–250, 1997.

[15] G. K. Francis and J. R. Weeks. Conway's ZIP proof. *Amer. Math. Monthly* 106:393–399, 1999. ⟨http://new.math.uiuc.edu/zipproof/⟩.

[16] M. R. Garey, R. L. Graham, and D. S. Johnson. The complexity of computing Steiner minimal trees. *SIAM J. Appl. Math.* 32:835–859, 1977.

[17] M. R. Garey and D. S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM J. Appl. Math.* 32:826–834, 1977.

[18] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, and M. Halle. Conformal surface parameterization for texture mapping. *IEEE Trans. Visualizat. Comput. Graph.* 6(2):181–187, 2000.

[19] M. Hallett and T. Wareham. A compendium of parameterized compplexity results. *SIGACT News* 25(3):122–123, 1994. ⟨http://web.cs.mun.ca/~harold/W_hier/compendium.html⟩.

[20] M. Hanan. On Steiner's problem with rectilinear distance. *SIAM J. Appl. Math.* 14:255–265, 1966.

[21] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2001. ⟨http://www.math.cornell.edu/~hatcher/⟩.

[22] D. B. Johnson. Efficient algorithms for shortest paths in sparse networks. *J. Assoc. Comput. Mach.* 24(1):1–13, 1977.

[23] A. Klivans and D. A. Spielman. Randomness efficient identity testing of multivariate polynomials. *Proc. 33rd Annu. ACM Sympos. Theory Comput.*, 216–223, 2001.

[24] L. Kou, G. Markowsky, and L. Berman. A fast algorithm for Steiner trees. *Acta Inform.* 15:141–145, 1981.

[25] F. Lazarus, M. Pocchiola, G. Vegter, and A. Verroust. Computing a canonical polygonal schema of an orientable triangulated surface. *Proc. 17th Annu. ACM Sympos. Comput. Geom.*, 80–89, 2001.

[26] K. Mulmuley, U. V. Vazirani, and V. V. Vazirani. Mathing is as easy as matrix inversion. *Combinatorica* 7:105–113, 1987.

[27] J. R. Munkres. *Topology*, 2nd edition. Prentice-Hall, 2000.

[28] D. Piponi and G. Borshukov. Seamless texture mapping of subdivision surfaces by model pelting and texture blending. *Proc. SIGGRAPH 2000*, 471–478, 2000.

[29] A. Sheffer and E. de Sturler. Surface parameterization for meshing by triangulation flattening. *Proc. 9th International Meshing Roundtable*, 161–172, 2000. ⟨http://www.andrew.cmu.edu/user/sowen/abstracts/Sh742.html⟩.

[30] J. Stillwell. *Classical Topology and Combinatorial Group Theory*, 2nd edition. Graduate Texts in Mathematics 72. Springer-Verlag, 1993.

[31] H. Takahashi and A. Matsuyama. An approximate solution for the network Steiner tree problem. *Math. Japonica* 24:573–577, 1980.

[32] W. Thurston. *Three-Dimensional Geometry and Topology, Volume 1.* Princeton University Press, New Jersey, 1997.

[33] G. Vegter. Computational topology. *Handbook of Discrete and Computational Geometry*, chapter 28, 517–536, 1997. CRC Press LLC.

[34] G. Vegter and C. K. Yap. Computational complexity of combinatorial surfaces. *Proc. 6th Annu. ACM Sympos. Comput. Geom.*, 102–111, 1990.