# MINIMUM CUTS IN SURFACE GRAPHS*

ERIN W. CHAMBERS†, JEFF ERICKSON‡, KYLE FOX§, AND AMIR NAYYERI¶

**Abstract.** We describe algorithms to efficiently compute minimum $(s,t)$-cuts and global minimum cuts of undirected surface-embedded graphs. Given an edge-weighted undirected graph $G$ with $n$ vertices embedded on an orientable surface of genus $g$, our algorithms can solve either problem in $g^{O(g)}n\log\log n$ or $2^{O(g)}n\log n$ time, whichever is better. When $g$ is a constant, our $g^{O(g)}n\log\log n$ time algorithms match the best running times known for computing minimum cuts in planar graphs.

Our algorithms for minimum cuts rely on reductions to the problem of finding a minimum-weight subgraph in a given $\mathbb{Z}_2$-homology class, and we give efficient algorithms for this latter problem as well. If $G$ is embedded on a surface with genus $g$ and $b$ boundary components, these algorithms run in $(g+b)^{O(g+b)}n\log\log n$ and $2^{O(g+b)}n\log n$ time. We also prove that finding a minimum-weight subgraph homologous to a single input cycle is NP-hard, showing it is likely impossible to improve upon the exponential dependencies on $g$ for this latter problem.

**1. Introduction.** Planar graphs have been a natural focus of study for algorithms research for decades, both because they accurately model many real-world networks, and because they often admit simpler and/or more efficient algorithms for many problems than general graphs. Most planar-graph algorithms either apply immediately or have been quickly generalized to larger families of graphs, such as graphs of higher genus, graphs with forbidden minors, or graphs with small separators. Examples include minimum spanning trees [94, 104]; single-source and multiple-source shortest paths [20, 48, 53, 73, 85, 86, 92, 116]; graph and subgraph isomorphism [43, 44, 64, 75, 96]; and approximation algorithms for the traveling salesman problem, Steiner trees, and other NP-hard problems [10, 12, 13, 17, 37, 44, 62].

The classical minimum cut problem and its dual, the maximum flow problem, are stark exceptions to this general pattern. Flows and cuts were introduced in the 1950s as tools for studying transportation networks, which are naturally modeled as planar graphs [68]. Ford and Fulkerson's seminal paper [55] includes an algorithm to compute maximum flows in planar networks where the source and target lie on the same face. A long series of results eventually led to planar minimum-cut algorithms that run in near-linear time, first for undirected graphs [58, 70, 78, 106] and later for directed graphs [73, 79, 99].

In contrast, prior to our work, almost nothing was known about computing minimum cuts in even mild generalizations of planar graphs; in particular, except for the work reported in this paper, we are unaware of any algorithm to compute minimum-cuts in non-planar graphs that does not require first computing a maximum flow.

This paper describes the first algorithms to compute minimum cuts in surface-embedded graphs of fixed genus in near-linear time. Specifically, we describe two algorithms to compute minimum $(s,t)$-cuts in undirected surface graphs, the first in $g^{O(g)}n\log\log n$ time, and the second in $2^{O(g)}n\log n$ time. We also extend our algorithms to find *global* minimum cuts in undirected surface graphs in the same asymptotic time bounds. For all our algorithms, the input consists of an undirected $n$-vertex graph, with arbitrary positive real edge weights, embedded on an orientable surface of genus $g$. (Some of our results do generalize to non-orientable surfaces; we will mention these generalizations in context.)

Our algorithms are based on a natural generalization of the duality between cuts and cycles in planar graphs, first proposed by Whitney [119] and first exploited to compute minimum cuts in planar graphs by Itai and Shiloach [77]. A set $C$ of the edges crossing a nontrivial partition $(S,T)$ of $V$ is an $(s,t)$-cut if $s\in S$ and $t\in T$. If $G$ is embedded on a surface, then the corresponding edges $C^*$ in the dual graph $G^*$ separate the *faces* of $G^*$ into two disconnected subcomplexes, one containing the dual face $s^*$ and the other containing the dual face $t^*$.

* Portions of this work were presented in preliminary form, by different subsets of the authors, at the 25th Annual Symposium on Computational Geometry [27], the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms [50], and the 24th Annual ACM-SIAM Symposium on Discrete Algorithms [49].

†Department of Computer Science, Saint Louis University; erin.chambers@slu.edu. Supported in part by NSF grants CCF-1054779, CCF-1614562, DBI-1759807, CCF-1907612, and CCF-2106672. Portions of this work were done while this author was a student at the University of Illinois at Urbana-Champaign.

‡Department of Computer Science, University of Illinois, Urbana-Champaign; jeffe@illinois.edu. Supported in part by NSF grants CCF-0915519, CCF-1408763, and DMS-0528086.

§Department of Computer Science, University of Texas at Dallas; kyle.fox@utdallas.edu. Supported in part by the Department of Energy Office of Science Graduate Fellowship Program (DOE SCGF), made possible in part by the American Recovery and Reinvestment Act of 2009, administered by ORISE-ORAU under contract no. DE-AC05-06OR23100. Portions of this work were done while this author was a student at the University of Illinois at Urbana-Champaign.

¶School of Electrical Engineering and Computer Science, Oregon State University, nayyeria@eecs.oregonstate.edu. Supported in part by NSF grants CCF-1065106, CCF-0915519, and DMS-0528086. Portions of this work were done while this author was a student at the University of Illinois at Urbana-Champaign.

We formalize this characterization in terms of *homology*, a standard equivalence relation from algebraic topology; specifically, we use cellular homology with coefficients in $\mathbb{Z}_2$. Briefly, two subgraphs of a surface graph are *homologous*, or in the same *homology class*, if and only if their symmetric difference is the boundary of a subset of faces. In light of this characterization, finding minimum $(s,t)$-cuts in surface graphs becomes a special case of finding the minimum-weight subgraph of a surface graph in a given homology class. Indeed, both of our algorithms for computing minimum $(s,t)$-cuts solve this more general problem, which is sometimes called *homology localization* [31, 32].

Unlike in planar graphs, where every minimal cut is dual to a simple cycle [119], the dual of a minimum cut in a surface graph may consist of several disjoint cycles. More generally, the minimum-weight subgraph in any homology class may be disconnected, even when the homology class is specified by a simple cycle; see Figure 2.5. Dealing with disconnected "cycles" is a significant complication in our algorithms.

Before describing our results in further detail, we first review several related results; technical terms are more precisely defined in Section 2.

### 1.1. Past results.

**Minimum cuts in planar graphs.** For any two vertices $s$ and $t$ in a graph $G$, an ***(s, t)-cut*** is a subset of the edges of $G$ that intersects every path from $s$ to $t$. A *minimum* $(s,t)$-cut is an $(s,t)$-cut with the smallest number of edges, or with minimum total weight if the edges of $G$ are weighted.

Minimum cuts in planar graphs were already studied by Ford and Fulkerson [55], who observed that when $s$ and $t$ lie on a common face of a planar graph $G$, the minimum $(s,t)$-cut is dual to a shortest path in the dual graph $G^*$. It follows immediately that the minimum $(s,t)$-cut in such a graph can be computed in $O(n \log n)$ time using Dijkstra's algorithm [77]; Hassin [69] showed that the maximum $(s,t)$-flow can be computed in $O(n)$ additional time.

Itai and Shiloach [77] generalized Ford and Fulkerson's observation to arbitrary planar networks. They showed that the minimum $(s,t)$-cut in any planar graph $G$ is dual to the minimum-cost cycle that separates faces $s^*$ and $t^*$ in $G^*$, and moreover that this separating cycle intersects any shortest path from a vertex of $s^*$ to a vertex of $t^*$ exactly once. Thus, one can compute the minimum $(s,t)$-cut by slicing the dual graph $G^*$ along a shortest path $\pi$ from $s^*$ to $t^*$; duplicating every vertex and edge of $\pi$; and then computing, for each vertex $u$ of $\pi$, the shortest path between the two copies of $u$ in the resulting planar graph. Applying Dijkstra's shortest-path algorithm at each vertex of $\pi$ immediately yields a running time of $O(n^2 \log n)$.

Reif [106] improved the running time of this algorithm to $O(n \log^2 n)$ using a divide-and-conquer strategy. Reif's algorithm was extended by Hassin and Johnson to compute the actual maximum flow in $O(n \log n)$ additional time, using a carefully structured dual shortest-path computation [70]. The running time was improved to $O(n \log n)$ by Frederickson [58], and more recently to $O(n \log \log n)$ by Italiano *et al.* [78], by using a balanced separator decomposition to speed up the shortest-path computations.

Janiga and Koubek [79] attempted to adapt Reif's $O(n \log^2 n)$-time algorithm to directed planar graphs; however, their algorithm has a subtle error [80] which may lead to an incorrect result when the minimum $(t,s)$-cut is smaller than the minimum $(s,t)$-cut.

Henzinger *et al.* [73] generalized Frederickson's technique to obtain an $O(n)$-time planar shortest-path algorithm; using this algorithm in place of Dijkstra's algorithm improves the running times of both Reif's and Janiga and Koubek's algorithms to $O(n \log n)$. The same improvement can also be obtained using more recent multiple-source shortest path algorithms by Klein [86]; Cabello, Chambers, and Erickson [20]; and Erickson, Fox, and Lkhamsuren [48].

Minimum $(s,t)$-cuts in directed planar graphs can also be computed in $O(n \log n)$ time using the planar maximum-flow algorithms of Weihe [118] (after filtering out useless edges [52]) and Borradaile and Klein [8, 14, 15].

A ***cut*** (without specified $s$ and $t$) is a subset of edges of $G$ that separate $G$ into two non-empty sets of vertices. A *global minimum* cut is a cut of minimum size, or minimum total weight if the edges of $G$ are weighted. Equivalently, a global minimum cut is an $(s,t)$-minimum cut of smallest total weight, minimized over all pairs of vertices $s$ and $t$. Chalermsook, Fakcharoenphol, and Nanongkai [24] gave the first algorithm for computing global minimum cuts that relies on planarity; their algorithm runs in $O(n \log^2 n)$ time. Their algorithm was improved by Łącki and Sankowski [88] who achieved an $O(n \log \log n)$ running time. Mozes *et al.* recently achieved the same $O(n \log \log n)$ running time for global minimum cuts in *directed* planar graphs [100], using techniques reported in a preliminary version of the current paper [50], specifically, the $\mathbb{Z}_2$-homology covers described in Section 5.

**Generalizations of planar graphs.** Surprisingly little is known about the complexity of computing maximum flows or minimum cuts in generalizations of planar graphs. In particular, we know of no previous algorithm to compute minimum cuts in non-planar graphs that does not first compute a maximum flow.

By combining a technique of Miller and Naor [97] with the planar directed flow algorithm of Borradaile and Klein [8, 14, 15, 46], one can compute maximum (single-commodity) flows in a planar graph with $k$ sources and sinks in $O(k^2 n \log n)$ time. More recently, Borradaile *et al.* [16] described an algorithm to compute maximum flows in planar graphs with an arbitrary number of sources and sinks in $O(n \log^3 n)$ time. An algorithm of Hochstein and Weihe [74] computes maximum flows in planar graphs with $k$ additional edges in $O(k^3 n \log n)$ time, using a clever simulation of Goldberg and Tarjan's push-relabel algorithm [61]. Borradaile *et al.* [16] extend Hochstein and Weihe's framework to compute maximum flows in planar graphs with $k$ apices in $O(k^3 n \log^3 n)$ time.

Chambers and Eppstein [26] describe an algorithm to compute maximum flows in $O(n \log n)$ time if the input graph forbids a fixed minor that can be drawn in the plane with one crossing. Another related result is the algorithm of Hagerup *et al.* [66] to compute maximum flows in graphs of constant treewidth in $O(n)$ time.

Imai and Iwano [76] describe a max-flow algorithm that applies to graphs of positive genus, but not to arbitrary sparse graphs. Their algorithm computes minimum-cost flows in graphs with small balanced separators, using a combination of nested dissection [92, 103], interior-point methods [117], and fast matrix multiplication. Their algorithm can be adapted to compute maximum flows (and therefore minimum cuts) in any graph of constant genus in time $O(n^{1.595} \log C)$, where $C$ is the sum of integer edge weights. However, this algorithm is slower than more recent and more general algorithms [41, 60].

Chambers, Erickson, and Nayyeri [28] describe maximum flow algorithms that are tailored specifically for graphs of constant genus. Given a graph embedded on an orientable surface of genus $g$, their algorithms compute a maximum flow in $O(g^8 n \log^2 n \log^2 C)$ time where $C$ is the sum of integer edge weights and in $g^{O(g)} n^{3/2}$ arithmetic operations when edge weights are arbitrary positive real numbers. Their key insight is that it suffices to optimize the *homology class* (with coefficients in $\mathbb{R}$) of the flow, rather than directly optimizing the flow itself.

Euler's formula implies that a simple $n$-vertex graph embedded on a surface of genus $O(n)$ has $O(n)$ edges. The fastest known combinatorial maximum-flow algorithm for sparse graphs, due to Orlin [102], runs in $O(n^2/\log n)$ time. The fastest algorithm known for sparse graphs with small integer capacities, due to Goldberg and Rao [60] and Lee and Sidford [90], run in time $O(n^{3/2} \operatorname{polylog}(n, U))$, where $U$ is an upper bound on the integer edge weights. Mądry [93] describes a faster algorithm for unit capacity graphs that runs in $O(n^{10/7} \operatorname{polylog} n)$ time when the graph is sparse.

The fastest algorithm known to compute global minimum cuts in arbitrary weighted undirected graphs is a Monte Carlo randomized algorithm of Karger [81], which runs in $O(m \log^3 n)$ time but fails with small probability. A more recent deterministic algorithm of Henzinger, Rao, and Wang [72], based on breakthrough techniques of Kawarabayashi and Thorup [82, 83], computes global minimum cuts in *unweighted* graphs in $O(m \log^2 n \log^2 \log n)$ time. The fastest deterministic algorithms known for global minimum cuts in arbitrary weighted graphs run in $O(nm + n^2 \log n)$ time for undirected graphs [57, 101, 113] and in $O(mn \log(n^2/m))$ time for directed graphs [67].

For further background on maximum flows, minimum cuts, and related problems, we refer the reader to monographs by Ahuja *et al.* [2] and Schrijver [110].

**Optimal homology representatives.** Homology is a topological notion of equivalence with nice algebraic properties. Two subgraphs of a surface graph $G$ are *homologous*, or in the same *homology class*, if their difference is the sum of face boundaries, where summation is defined over some coefficient ring. Our minimum-cut algorithms all reduce to the problem of finding a subgraph of *minimum weight* in a given homology class (over the ring $\mathbb{Z}_2$). Several authors have considered variants of this problem, which is often called *homology localization*.

Most interesting variants of homology localization are NP-hard. Chambers *et al.* [25] prove that finding the shortest *splitting* cycle is NP-hard; a cycle is splitting if it is non-self-crossing, non-contractible, and null-homologous. A simple modification of their reduction (from Hamiltonian cycle in planar grid graphs) implies that finding the shortest *simple cycle* in a given homology class is NP-hard. Chen and Freedman [30, 31] proved a similar hardness result for general simplicial complexes; however, the complexes output by their reduction are never manifolds. Recently, Grochaw and Tucker-Foltz [63] proved that homology localization in surface graphs, over a sufficiently large finite coefficient ring, is equivalent to Unique Games; in particular, there is no PTAS for *any* finite coefficient ring unless the Unique Games Conjecture is false.

On the other hand, for homology with real or integer coefficients, homology localization in surface graphs is equivalent (via duality) to a minimum-cost flow problem and hence can be solved in polynomial time [28, 114].

Chambers *et al.* [28] describe an algorithm to find optimal circulations in a given homology class in near-linear time, given a graph with integer coefficients on an orientable surface of fixed genus. Sullivan [114] and Dey *et al.* [38] prove similar results for higher-dimensional orientable manifolds.

**1.2. New results and organization.** In Section 3, we describe two techniques to preprocess a graph on a surface with boundary, so that the homology class of any subgraph can be computed quickly. These are both natural generalizations of known methods for measuring homology in surfaces *without* boundary based on tree-cotree decompositions [25, 45, 51]. In particular, we describe how to construct a *system of arcs*—a collection of $O(g+b)$ boundary-to-boundary paths that cut the surface into a disk—in $O((g+b)n)$ time. This generalization is essential for our algorithms, as our dual homology characterization of minimum $(s,t)$-cuts removes the dual faces $s^*$ and $t^*$, leaving a surface with two boundary components.

Intuitively, the homology class of any cycle is determined by its pattern of crossings with any system of arcs, and the homology class of more complicated subgraphs can be computed by decomposing them into cycles. However, there are several subtle issues that must be resolved to formalize and apply this intuition. First, it is unclear how to modify the topological definition of "crossing" as transverse intersection to apply in our combinatorial setting, because the arcs in a system may share edges with each other and with the even subgraphs whose homology we want to measure. Second, even when crossings are well-defined, there is more than one way to extract homology from the crossing pattern.

The first way we resolve these issues is by perturbing the system of arcs within a small neighborhood of the input graph. Specifically, we replace the vertices and edges of the input graph with small disks and ribbons, and perturb cycles and arcs within the resulting structure, which we call a *ribbon graph* [42, 89, 91]. We also perturb any cycle whose homology we want to measure within the ribbon graph so that it intersects the perturbed arcs transversely. Finally, the homology of a perturbed cycle is determined by the *sequence* of perturbed arcs that it crosses. See Sections 2.4 and 3.1.

We then use this formulation in Section 4 to develop our first algorithm to compute minimum-weight subgraphs in a given homology class. Our algorithm first computes a *greedy system of arcs*; each arc in this system consists of two shortest paths in the input graph. Using an exchange argument, we prove that the minimum-weight subgraph in any homology class crosses each arc in the greedy system at most $O(g+b)$ times. Our algorithm enumerates all possible sequences of crossings consistent with this upper bound, and finds the shortest subgraph consistent with each crossing sequence, by reducing to a planar minimum cut problem. The resulting algorithm runs in $(g+b)^{O(g+b)} n \log\log n$ time.

The second way we resolve crossing subtleties is to build our system of arcs in the dual graph $G^*$; cycles in $G$ intersect these dual arcs only transversely. To avoid the need for perturbing the dual arcs apart, we show that the homology of a cycle in $G$ is determined by the *number* of times it crosses each of the dual arcs in the system. See Section 3.3.

We use this dual formulation in our second algorithm to compute minimum-weight homologous subgraphs in Section 5. Our algorithm computes the shortest *cycle* in *every* homology class, by constructing and searching a certain covering space of the surface that we call the $\mathbb{Z}_2$*-homology cover*, using an extension [48] of the multiple-source shortest path algorithm of Cabello *et al.* [20]. We then assemble the minimum-weight *even subgraph* in any desired homology class from these $\mathbb{Z}_2$-minimal cycles using dynamic programming. The resulting algorithm runs in $2^{O(g+b)} n \log n$ time. Our second algorithm is simpler, and its running time has better (but still exponential) dependence on the topological parameters $g$ and $b$, but at the expense of slightly worse dependence on $n$.

In Section 6, we prove that finding a minimum-weight even subgraph in a given homology class in NP-hard, which implies that our algorithms' exponential dependence on $g$ is almost certainly unavailable. Unlike Chen and Freedman [32], our reduction is done on a 2-manifold, and unlike Chambers *et al.* [25], our target subgraph does not need to be a simple cycle.

Finally, in Section 7, we describe our algorithms for computing global minimum cuts. Both algorithms ultimately reduce computing a global minimum cut to $2^{O(g)}$ instances of computing minimum $(s,t)$-cuts; thus, our algorithms have the same asymptotic running times as the minimum $(s,t)$-cut algorithms from Sections 4 and 5.

We note with some amusement that our algorithms solve a problem with a well-known polynomial-time solution by reducing it to an exponential number (in $g$) of instances of an NP-hard (but fixed-parameter tractable) problem! The authors of this paper are divided on whether to conjecture that minimum cuts in surface graphs can be computed in time $O(g^c n \operatorname{polylog} n)$ for some small constant $c$, or that the problem is "fixed-parameter quadratic" with respect to genus, just as diameter and radius are fixed-parameter quadratic with respect to treewidth [1].

Fomin *et al.* [54] raise similar questions about the fixed-parameter efficiency of flows and cuts with respect to treewidth.

**2. Notation and Terminology.** We begin by recalling several useful definitions related to surface-embedded graphs. For further background, we refer the reader to Gross and Tucker [65] or Mohar and Thomassen [98] for topological graph theory, and to Hatcher [71] or Stillwell [112] for surface topology and homology.

**2.1. Surfaces and curves.** A ***surface*** (more formally, a *2-manifold with boundary*) is a compact Hausdorff space in which every point has an open neighborhood homeomorphic to either the plane $\mathbb{R}^2$ or a closed halfplane $\{(x,y) \in \mathbb{R}^2 \mid x \geq 0\}$. The points with halfplane neighborhoods make up the ***boundary*** of the surface; every component of the boundary is homeomorphic to a circle. A surface is ***non-orientable*** if it contains a subset homeomorphic to the Möbius band, and ***orientable*** otherwise. In this paper, we consider only compact and connected surfaces.

A ***path*** in a surface $\Sigma$ is a continuous function $p : [0,1] \to \Sigma$. A ***loop*** is a path whose endpoints $p(0)$ and $p(1)$ coincide; we refer to this common endpoint as the ***basepoint*** of the loop. An ***arc*** is a path whose endpoints lie on the boundary of $\Sigma$, but that is otherwise disjoint from the boundary of $\Sigma$. A ***cycle*** is a continuous function $\gamma : S^1 \to \Sigma$; the only difference between a cycle and a loop is that a loop has a distinguished basepoint. We say a loop $\ell$ and a cycle $\gamma$ are ***equivalent*** if, for some real number $\delta$, we have $\ell(t) = \gamma(t+\delta)$ for all $t \in [0,1]$. We collectively refer to paths, loops, arcs, and cycles as ***curves***. A curve is ***simple*** if it is injective, except for the endpoints of a loop; we usually do not distinguish between simple curves and their images in $\Sigma$. A simple curve $p$ is ***separating*** if $\Sigma \setminus p$ is disconnected.

The ***reversal*** $rev(p)$ of a path $p$ is defined by setting $rev(p)(t) = p(1-t)$. The ***concatenation*** $p \cdot q$ of two paths $p$ and $q$ with $p(1) = q(0)$ is the path created by setting $(p \cdot q)(t) = p(2t)$ for all $t \leq 1/2$ and $(p \cdot q)(t) = q(2t-1)$ for all $t \geq 1/2$.

The ***genus*** of a surface $\Sigma$ is the maximum number of disjoint simple cycles in $\Sigma$ whose complement is connected. Up to homeomorphism, there is exactly one orientable surface with any genus $g \geq 0$ and any number of boundary cycles $b \geq 0$, and exactly one non-orientable surface with any positive genus $g > 0$ and any number of boundary cycles $b \geq 0$. The ***Euler characteristic*** $\chi$ of a surface with genus $g$ and $b$ boundary components is $2-2g-b$ if the surface is orientable, and $2-g-b$ otherwise.

**2.2. Graph embeddings.** An ***embedding*** of an undirected graph $G = (V,E)$ on a surface $\Sigma$ is an injective continuous function from $G$ to $\Sigma$; in particular, an embedding maps vertices of $G$ to distinct points in $\Sigma$ and edges of $G$ to simple, interior-disjoint paths in $\Sigma$ that intersect vertices only at their endpoints. The ***faces*** of the embedding are maximal connected subsets of $\Sigma$ that are disjoint from the image of the graph. We may denote an edge $uv \in E$ as $f|g$ if it is incident to faces $f$ and $g$. An embedding is ***cellular*** if each of its faces is homeomorphic to the plane; in particular, in any cellular embedding, each component of the boundary of $\Sigma$ must be covered by a cycle of edges in $G$.

We also refer to the complex of vertices, edges, and faces induced by a cellular embedding as a ***combinatorial surface***. Every combinatorial surface with boundary can be obtained from a combinatorial surface without boundary by deleting the interiors of one or more faces. See Kettner [84] for an overview and comparison of several standard data structures for combinatorial surfaces.

Euler's formula implies that any cellularly embedded graph with $n$ vertices, $m$ edges, and $f$ faces lies on a surface with Euler characteristic $\chi = n - m + f$, which implies that $m = O(n+g)$ and $f = O(n+g)$ if the graph is simple. To simplify our presentation, we implicitly assume throughout the paper that $g = O(\log n)$, since otherwise our minimum-cut algorithms are slower than textbook maximum-flow algorithms. This assumption implies that the overall complexity of an embedding is $O(n)$.

We redundantly use the term ***arc*** to refer to a walk in the graph whose endpoints are boundary vertices, but that is otherwise disjoint from the boundary. Likewise, we use the term ***cycle*** to refer to a closed walk in the graph. Note that arcs and cycles may traverse the same vertex or edge more than once.

**2.3. Duality.** Any undirected graph $G$ embedded on a surface $\Sigma$ without boundary has a ***dual graph*** $G^*$, which has a vertex $f^*$ for each face $f$ of $G$, and an edge $e^*$ for each edge $e$ in $G$ joining the vertices dual to the faces of $G$ that $e$ separates. The dual graph $G^*$ has a natural cellular embedding in $\Sigma$, whose faces corresponds to the vertices of $G$. See Figure 2.1.

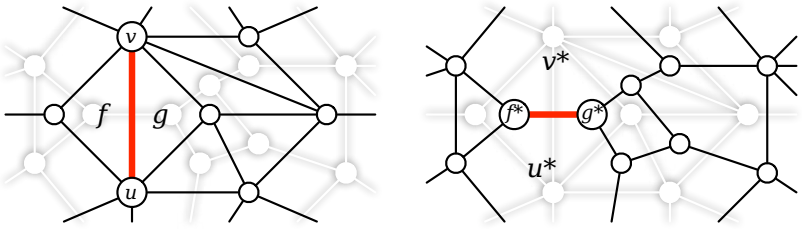


**Fig. 2.1.** Graph duality. One edge $uv$ and its dual $(uv)^* = f^*g^*$ are emphasized.

Any undirected graph $G$ embedded on a surface $\Sigma$ with boundary has a ***dual graph*** $G^*$, defined as follows.[1] The dual graph $G^*$ has a vertex $f^*$ for each face $f$ of $G$, *including the boundary cycles*, and an edge $e^*$ for each edge $e$ in $G$ (including boundary edges) joining the vertices dual to the faces that $e$ separates. For each boundary cycle $\delta$ of $G$, we refer to the corresponding vertex $\delta^*$ of $G^*$ as a ***dual boundary vertex***. The dual graph $G^*$ has a natural cellular embedding in the surface $\boldsymbol{\Sigma^\bullet}$ obtained from $\Sigma$ by gluing a disk to each boundary cycle; each face of this embedding corresponds to a vertex of $G$. See Figure 2.2. (Duality can be extended to directed graphs [28], but our results do not require this extension.)

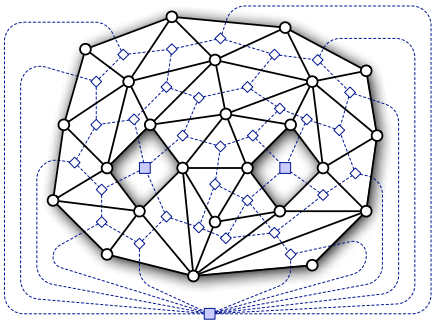

**Fig. 2.2.** A cellularly embedded graph $G$ (solid lines) on a pair of pants (the surface of genus 0 with 3 boundaries), and its dual graph $G^*$ (dashed lines). Dual boundary vertices are indicated by squares.

For any subgraph $F = (U, D)$ of $G = (V, E)$, we write $\boldsymbol{G \setminus F}$ to denote the edge-complement $(V, E \setminus D)$. Also, when the graph $G$ is fixed, we abuse notation by writing $F^*$ to denote the subgraph of $G^*$ corresponding to a subgraph $F$ of $G$; each edge in $F^*$ is the dual of a unique edge in $F$. In particular, we have the identity $(G \setminus F)^* = G^* \setminus F^*$. Further, we may sometimes use $D$ to refer to an edge set or the subgraph $F = (V, D)$, but it should be clear which we mean from context.

**2.4. Perturbations and Crossings.** Our algorithms manipulate cycles and arcs in combinatorial surfaces that can share or repeat graph vertices and edges, but that can be perturbed on the surface to avoid self-intersections and other degeneracies. To accommodate these perturbations, we represent our combinatorial surfaces as ***ribbon graphs*** [42, 89]. Ribbon graphs are also known as *band decompositions* [42] and *fat graphs* [89], and they are closely related to the *graph-encoded map* data structure [91] for representing both orientable and non-orientable combinatorial surfaces. Intuitively, ribbon graphs allow us to represent multiple traversals of the same edge as parallel paths in a narrow ribbon around that edge. Our ribbon-graph representation is (roughly) equivalent to the *cross-metric surface* representation used in several other papers [25, 34, 36]; however, the cross-metric surface representation is somewhat awkward to work with on surfaces with boundary, which is our main setting. We note that our use of ribbon graphs closely resembles the "side information" maintained by Kutz's algorithm [87, Section 4.1], which will be a key ingredient in Section 4.

The ribbon graph $G^\square$ of an embedded graph $G$ is constructed by expanding each vertex $v$ of $G$ into a closed disk $v^\square$ called a *vertex region*, and expanding each edge $e$ of $G$ into a narrow rectangle $e^\square$ called an *edge ribbon*. For any vertex $v$ and edge $e$, the intersection $v^\square \cap e^\square$ is a simple path on the boundary of both regions if $v$ is an endpoint of $e$; otherwise, the vertex regions and edge ribbons are pairwise disjoint. Each face $f$ of the original

[1] Our definition differs slightly from the one proposed by Erickson and Colin de Verdière [35].

embedding $G$ (including any faces deleted to create surface boundaries) contains exactly one component of $\Sigma \setminus G^\square$; we call this component the *face region* $f^\square$. See Figure 2.3.

Any finite collection $C$ of cycles and arcs in a combinatorial surface, each represented as a walk in the underlying graph $G$, can be continuously deformed into general position within the ribbon graph $G^\square$, so that all (self-)intersections are transverse and occur only within vertex regions. (Formally, this perturbation is a *homotopy* within $G^\square$; see Section 2.6 below.) In particular, each deformed curve visits the vertex regions and edge ribbons in $G^\square$ in the same order that the original curve visits the corresponding vertices and edges of $G$.

The intersection of each perturbed curve with any vertex region $v^\square$ consists of simple boundary-to-boundary paths in $v^\square$. In particular, any arc that ends at vertex $v$ on a boundary face $f$ is perturbed into an arc in $G^\square$ that ends at the boundary segment $v^\square \cap f^\square$. We also require minimal intersection within vertex regions, so any two boundary-to-boundary paths within the same vertex region intersect at most once. We refer to any deformation of $C$ that meets these criteria as a ***ribbon perturbation*** of $C$.

None of the cycles and arcs we consider in this paper contain *spurs*—subpaths that traverse an edge followed immediately by its reversal. Thus, the intersection of the perturbed cycles and arcs and any edge ribbon is a collection of disjoint paths that traverse the ribbon from one end to the other. Thus, to represent any ribbon perturbation of $C$, it suffices to record the following information:

- The alternating sequence of vertices and edges traversed by each cycle in $C$;
- The starting boundary face, alternating sequence of vertices and edges, and ending boundary face of each arc in $C$;
- The ordering of perturbed curve segments traversing each ribbon $e^\square$; and
- The cyclic order of intersections of the perturbed curves with the boundary of each vertex region $v^\square$.

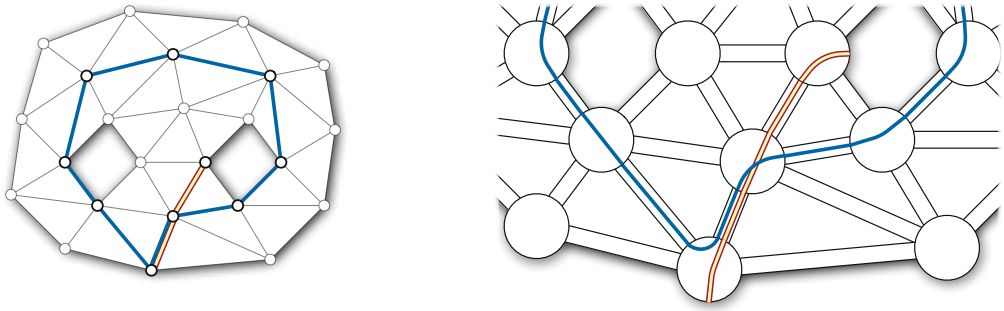

**Fig. 2.3.** Left: A cycle and an arc that share an edge in $G$. Right: A closeup on the ribbon graph $G^\square$, with a ribbon perturbation of the cycle and the arc that cross inside a vertex region.

An arc or cycle in a combinatorial surface is ***weakly simple*** if it can be perturbed into a *simple* path or cycle in the ribbon graph; algorithms to detect if a cycle is weakly simple have been studied extensively of late, in both the plane and on orientable or non-orientable surfaces [3, 29]. Similarly, an arc or cycle $\alpha$ and another arc or cycle $\beta$ are ***non-crossing*** if they have disjoint perturbations within the ribbon graph; otherwise, we say that $\alpha$ ***crosses*** $\beta$. More generally, we say that a collection $C$ of cycles and arcs is ***non-crossing*** if there is a single ribbon perturbation of $C$ in which all perturbed curves are pairwise disjoint.

**2.5. Even subgraphs and cycle decompositions.** An ***even subgraph*** is a subgraph of $G$ in which every node has even degree, or equivalently, the symmetric difference of cycles. A ***cycle decomposition*** of an even subgraph $H$ is a set of edge-disjoint, non-crossing, weakly simple cycles whose union is $H$.

LEMMA 2.1. *Every even subgraph of an embedded graph has a cycle decomposition.*

**Proof:** Let $H$ be an even subgraph of $G$. We can decompose $H$ into cycles by specifying, at each vertex $v$, which pairs of incident edges of $H$ are consecutive. Any pairing that does not create a crossing at $v$ is sufficient. For example, if $e_1, e_2, \ldots, e_{2d}$ are the edges of $H$ incident to $v$, indexed in clockwise order around $v$, we could pair edges $e_{2i-1}$ and $e_{2i}$ for each $i$. □

We emphasize that each cycle in a cycle decomposition may visit vertices multiple times; indeed, some even subgraphs of $G$ cannot be decomposed into strictly simple cycles in $G$.

***Slicing*** a combinatorial surface along a cycle or arc modifies both the surface and the embedded graph. For any combinatorial surface $S = (\Sigma, G)$ and any *simple* cycle or arc $\gamma$ in $G$, we define a new combinatorial surface

$S \setminus\!\setminus \gamma$ by taking the topological closure of $\Sigma \setminus \gamma$ as the new underlying surface; the new embedded graph contains two copies of each vertex and edge of $\gamma$, each bordering a new boundary. We can also slice along any *weakly* simple arc or cycle $\gamma$ by considering any simple ribbon perturbation $\tilde{\gamma}$ of $\gamma$. In particular, $\tilde{\gamma}$ partitions the vertex regions and edge ribbons of $G^{\square}$ into the vertex regions and edge ribbons of the ribbon graph of $S \setminus\!\setminus \gamma$. Our representation of ribbon perturbations allows us to compute this new ribbon graph in time proportional to the combinatorial length of $\gamma$.

We define the ***projection*** of a curve in $S \setminus\!\setminus \gamma$ as the natural mapping of points (or vertices and edges) to $S$.

**2.6. Homotopy and homology.** Two paths $p$ and $q$ in $\Sigma$ are ***homotopic*** if one can be continuously deformed into the other without changing their endpoints. More formally, a ***homotopy*** between $p$ and $q$ is a continuous map $h : [0,1] \times [0,1] \to \Sigma$ such that $h(0,\cdot) = p$, $h(1,\cdot) = q$, $h(\cdot,0) = p(0) = q(0)$, and $h(\cdot,1) = p(1) = q(1)$. Homotopy defines an equivalence relation over the set of paths with any fixed pair of endpoints.

Similarly, two cycles $\alpha$ and $\beta$ in $\Sigma$ are ***freely homotopic*** if one can be continuously deformed into the other. More formally, a free homotopy between $\alpha$ and $\beta$ is a continuous map $h : [0,1] \times S^1 \to \Sigma$ such that $h(0,\cdot) = \alpha$ and $h(1,\cdot) = \beta$. Free homotopy defines an equivalence relation over the set of cycles in $\Sigma$. We omit the word "free" when it is clear from context.

A cycle is ***contractible*** if it is homotopic to a constant map. Given a weight function on the edges of $G$, we say a path or cycle is ***tight*** if it has minimum total weight (counting edges with multiplicity) for its homotopy class.

Homology is a coarser equivalence relation than homotopy, with nicer algebraic properties. Like several earlier papers [30, 31, 39, 40, 47, 56], we will consider only one-dimensional cellular homology with coefficients in the finite field $\mathbb{Z}_2$; this restriction allows us to radically simplify our definitions. Fix a cellular embedding of an undirected graph $G$ on a surface with genus $g$ and $b$ boundaries. A ***boundary subgraph*** is the boundary of the union of a subset of faces of $G$; for example, on a surface with no boundary, every separating cycle is a boundary subgraph. Two even subgraphs are ***homologous***, or in the same ***homology class***, if their symmetric difference is a boundary subgraph. Boundary subgraphs are also called ***null-homologous***. Any two homotopic cycles are homologous, but homologous cycles are not necessarily homotopic; see Figure 2.4. Moreover, the homology class of a cycle can contain even subgraphs that are not cycles; see Figure 2.5. We call an even subgraph ***$\mathbb{Z}_2$-minimal*** if it is the minimum-weight subgraph in its homology class.

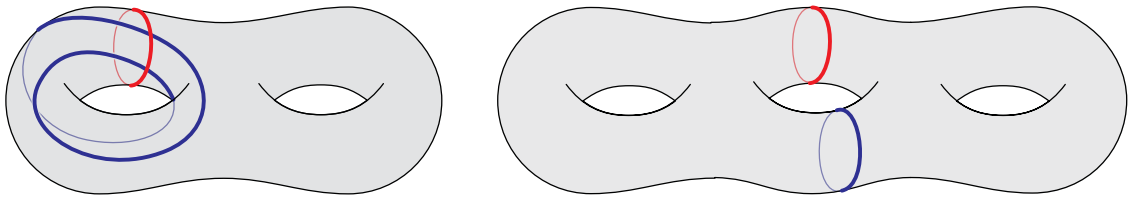

**Fig. 2.4.** Homologous pairs of cycles that are not homotopic. (Lighter portions of the curves are on the back side of the surface.)

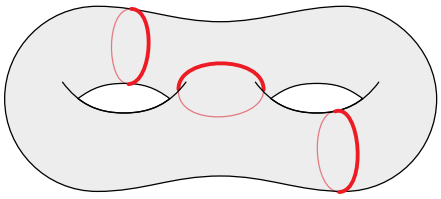

**Fig. 2.5.** Each cycle is homologous to the union of the other two.

Homology classes define a vector space $\mathbb{Z}_2^{\beta}$, called the first homology group, whose rank $\beta$ is called the first ***Betti number***. The first Betti number $\beta$ of any surface is related to its Euler characteristic $\chi$ as follows: $\beta = 2 - \chi$ if the surface has no boundary and $\beta = 1 - \chi$ otherwise. Equivalently, we have $\beta = 2g$ for orientable surfaces without boundary, $\beta = 2g + b - 1$ for orientable surfaces with boundary, $\beta = g$ for non-orientable surfaces without boundary, and $\beta = g - b + 1$ for non-orientable surfaces with boundary.

**2.7. Duality between cuts and even subgraphs.** A crucial component of our minimum $(s,t)$-cut algorithms is an equivalence between $(s,t)$-cuts and even subgraphs of the *dual graph* contained in a particular homology

class. This equivalence was first observed in planar graphs by Whitney [119] and was later used to compute minimum cuts in planar graphs by Itai and Shiloach [77]. We formalize the same equivalence on surface graphs in the following lemma:

LEMMA 2.2. *Let $G$ be an edge-weighted graph embedded on a surface $\Sigma$ without boundary, and let $s$ and $t$ be vertices of $G$. A subgraph $X$ is an $(s,t)$-cut in $G$ if and only if $X^*$ is an even subgraph of $G^*$ homologous with the boundary of $s^*$ in the surface $\Sigma \setminus (s^* \cup t^*)$. In particular, $X$ is a minimum-weight $(s,t)$-cut in $G$, if and only if $X^*$ is a minimum-weight even subgraph of $G^*$ homologous with the boundary of $s^*$ in $\Sigma \setminus (s^* \cup t^*)$.*

**Proof:** Let $\partial s^*$ denote the boundary of $s^*$, and let $\Sigma'$ denote the surface $\Sigma \setminus (s^* \cup t^*)$.

Let $X$ be an arbitrary $(s,t)$-cut in $G$. By definition, there is a non-trivial partition $V = S \cup T$, such that $s \in S$ and $t \in T$, where $X$ is the set of edges crossing $S$ and $T$. Thus, the dual subgraph $X^*$ partitions the faces of $G^*$ into two disjoint (possibly non-connected) subsets, $S^*$ and $T^*$, respectively containing faces $s^*$ and $t^*$. In particular, $X^*$ is the boundary of the union of the faces in $S^*$, which implies that $X^*$ is null-homologous in $\Sigma$. The symmetric difference $X^* \oplus \partial s^*$ is the boundary of the union of $S^* \setminus \{s^*\}$, which is a subset of the faces of $\Sigma'$. Thus, $X^* \oplus \partial s^*$ is null-homologous in $\Sigma'$. We conclude that $X^*$ and $\partial s^*$ are homologous in $\Sigma'$.

Conversely, let $X^*$ be an arbitrary even subgraph of $G^*$ homologous to $\partial s^*$ in $\Sigma'$. The subgraph $X^* \oplus \partial s^*$ is null-homologous in $\Sigma'$. This immediately implies that $X^*$ is null-homologous in $\Sigma$; moreover, faces $s^*$ and $t^*$ are on opposite sides of $X^*$. Any path from $s$ to $t$ in the original graph $G$ must traverse at least one edge of $X$. We conclude that $X$ is an $(s,t)$-cut. □

**3. Characterizing Homology.** Throughout the paper, we fix an *undirected* graph $G = (V, E)$, a positive weight function $w\colon E \to \mathbb{R}$, and a cellular embedding of $G$ on a (possibly non-orientable) surface $\Sigma$ of genus $g$ with $b$ boundary cycles. Except where explicitly indicated otherwise, we assume without loss of generality that $b > 0$; otherwise, we can remove an arbitrary face of $G$ from $\Sigma$ without affecting its homology at all. Let $\delta_1, \ldots, \delta_b$ denote the boundary cycles of $\Sigma$, and let $\beta$ denote the first Betti number of $\Sigma$. Recall that $\beta = 2g + b - 1$ if $\Sigma$ is orientable and $\beta = g + b - 1$ otherwise.

In this section, we describe two standard methods for preprocessing a combinatorial surface with boundary in $O(\beta n)$ time, so that the $\mathbb{Z}_2$-homology class of any even subgraph $H$ can be computed in $O(\beta)$ time per edge. These are both straightforward generalizations of standard methods for measuring homology in surfaces *without* boundary based on tree-cotree decompositions [25, 45, 51]. Tree-cotree decompositions were formalized by Eppstein [45] to compute homology generators, although they were studied earlier by other authors [7, 107, 115]. We give these full details here for completeness, and because as far as we are aware, no detailed description appears elsewhere in the literature for the first method. We note that a preliminary version of the current work [50] was the first detailed description of the second method; see also Chambers *et al.* [9] for an alternative description of the second method.

Both of the methods we will describe characterize the homology class of any even subgraph $H$ using a vector of $\beta$ bits. The vectors are computed using one of two natural generalizations of tree-cotree decompositions to surfaces with boundary. In the first method, the vector is based on the crossings between a cycle decomposition of $H$ and a set of $\beta$ primal arcs. By carefully selecting these arcs, we can bound the number of times any $\mathbb{Z}_2$-minimal even subgraph can cross any of these arcs; this bound is necessary for the algorithm given in Section 4. In the second method, the vector is based on the crossings between $H$ and a set of $\beta$ *dual* arcs. The second method is somewhat easier to describe and implement than the first, so we use the second method in the algorithm given in Section 5.

**3.1. Forest-cotree decompositions.** The first method begins by computing a set $A$ of $\beta$ arcs, each of which is the concatenation of two shortest paths in $G$ plus a single edge. Following previous papers [25, 33, 35], we construct a ***greedy system of arcs***, using a variant of Erickson and Whittlesey's algorithm to construct optimal systems of loops [51]. Our algorithm uses a natural generalization of tree-cotree decompositions [45] to surfaces with boundary.

A ***forest-cotree decomposition*** (see Figure 3.1) of a combinatorial surface $S = (\Sigma, G)$ is any partition $(\partial G, F, L, C)$ of the edges of $G$ into four edge-disjoint subgraphs with the following properties:

- $\partial G$ is the set of all boundary edges of $G$.
- $F$ is a spanning forest of $G$, that is, an acyclic subgraph of $G$ that contains every vertex.
- Each component of $F$ contains a single boundary vertex.
- $C^*$ is a spanning tree of $G^* \setminus (\partial G)^*$, that is, a subtree of $G^*$ that contains every vertex *except* the dual

boundary vertices $\delta_i^*$.
- Finally, $L$ is the set of leftover edges $E \setminus (\partial G \cup F \cup C^*)$

LEMMA 3.1. *In any forest-cotree decomposition $(\partial G, F, L, C)$ of any combinatorial surface with boundary and with first Betti number $\beta$, the set $L$ contains exactly $\beta$ edges.*

**Proof:** Recall that $n$, $m$, and $f$ respectively denote the number of vertices, edges, and faces of $G$. Let $d_1, \dots, d_b$ be the number of edges on the boundary components. We immediately have $|\partial G| = \sum_i d_i$. Because our boundary cycles are disjoint, contracting each boundary cycle to a single vertex transforms $F$ into a spanning forest of $G/\partial G$ with $b$ components. It follows that $|F| = (n - \sum_i (d_i - 1)) - b$. Finally, $C^*$ is a spanning tree of the dual graph, so it has $f - 1$ edges. As $(\partial G, F, L, C)$ is a partition of the edges, we have

$$\begin{aligned} m &= |\partial G| + |F| + |C| + |L| \\ &= \textstyle\sum_i d_i + (n - \sum_i (d_i - 1)) - b + f - 1 + |L|. \\ &= n + f - 1 + |L|. \end{aligned}$$

We conclude that $|L| = m - n - f + 1 = 1 - \chi = \beta$. □

Fix a forest-cotree decomposition $(\partial G, F, L, C)$, and arbitrarily index $L = \{e_1, e_2, \dots, e_\beta\}$. For each edge $e_i \in L$, the subgraph $F \cup \{e_i\}$ contains a single nontrivial arc $a_i$, which is either a simple path between distinct boundary cycles, or a nontrivial walk from a boundary cycle back to itself; in the second case, $a_i$ may traverse some edges of $F$ twice. We refer to the collection $\{a_1, a_2, \dots, a_\beta\}$ as a ***system of arcs***. See Figure 3.1.

LEMMA 3.2. *The arcs $a_1, a_2, \dots, a_\beta$ are weakly simple and weakly disjoint.*

**Proof:** Each arc $a_i$ consists of an edge in $L$ plus two (possibly overlapping) paths in $F$. We know the edges from $L$ are disjoint and paths in the forest are simple, so it suffices to describe a consistent way to perturb forest paths apart. We use an Euler tour on each component of $F$ to define consistent perturbations, as follows. For each edge $e_i$ in $L$, we subdivide $e_i$ into a path of length 3 and add the first and third edges on that path to the forest $F$, yielding a new augmented forest $F'$. Root each component of $F'$ at its (unique) boundary vertex. We then compute an Euler tour for each component of $F'$, which for each tree in $F'$ gives a total ordering of the leaves based on when they are encountered in the tour. We can then perturb all root-to-leaf paths in each component of $F'$ to become disjoint, where the order of the paths out of the root is the same as the total ordering of the leaves in the Euler tour. Adding the middle thirds of the edges from $L$ to these disjoint paths gives us a perturbation of $a_1, a_2, \dots, a_\beta$ into disjoint simple arcs. □

LEMMA 3.3. *Slicing along the arcs $a_1, a_2, \dots, a_\beta$ transforms $\Sigma$ into a topological disk.*

**Proof:** Let $\{\tilde{a}_1, \tilde{a}_2, \dots, \tilde{a}_\beta\}$ be any ribbon perturbation of the arcs $a_1, a_2, \dots, a_\beta$, such that the arcs $\tilde{a}_i$ are simple and pairwise disjoint. Lemma 3.2 implies that such a perturbation exists. Slicing along any perturbed arc $\tilde{a}_i$ increases the Euler characteristic of the surface by 1. Because the perturbed arcs $\tilde{a}_i$ are simple and disjoint, slicing along all $\beta$ perturbed arcs increases the Euler characteristic to 1. If the resulting sliced surface were disconnected, then the faces of $G$ incident to some edge in $L$ would lie in different components, but this is impossible, because all faces are connected in the sliced surface via the cotree $C^*$. We conclude that the sliced surface is connected and has Euler characteristic 1, so it must be a disk. □

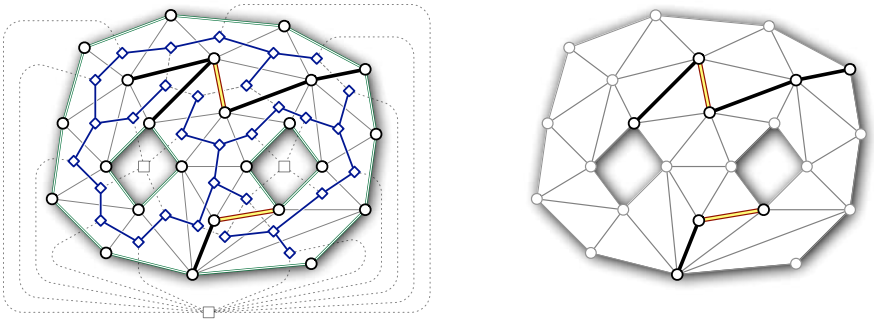

**Fig. 3.1.** Left: A forest-cotree decomposition of the graph in Figure 2.2; thick doubled lines indicate edges in $L$. Right: The resulting system of arcs. Compare with Figure 3.4.

We can easily construct an *arbitrary* forest-cotree decomposition, and thus an arbitrary system of arcs, in $O(n)$ time using whatever-first search, but our algorithms require a decomposition with a particular forest $F$ and a particular dual spanning tree $C^*$. Let $G/\partial G$ denote the graph obtained from $G$ by *contracting* the entire subgraph $\partial G$—both vertices and edges—to a single vertex $x$. Using the algorithm of Henzinger *et al.* [73] with Aleksandrov and Djidjev's linear-time algorithm for partitioning embedded graphs [4], we compute the single-source shortest-path tree $T$ in $G/\partial G$ rooted at $x$ in $O(n)$ time.[2] Let $F$ be the subgraph of $G$ corresponding to $T$. Each component of $F$ is a tree of shortest paths from a boundary vertex to a subset of the non-boundary vertices of $G$.

Now for each edge $e$ that is *not* in the forest $F$ or the boundary subgraph $\partial G$, let $\ell(e)$ denote the length of the unique arc in the subgraph $F \cup \{e\}$. We can easily compute $\ell(e)$ for each non-forest edge $e$ in $O(n)$ time. Finally, let $C^*$ denote the maximum spanning tree of $G^* \setminus (F \cup \partial G)^*$ with respect to the arc lengths $\ell(e)$.

For each edge $e_i \in L$, let $\sigma_i$ and $\tau_i$ denote the unique directed paths in $F$ from the boundary of $G$ to the endpoints of $e_i$, and let $P := \{\sigma_1, \dots, \sigma_\beta, \tau_1, \dots, \tau_\beta\}$. By construction of $F$, every element of $P$ is a (possibly empty) shortest path. For each index $i$, let $a_i = \sigma_i \cdot e_i \cdot \mathit{rev}(\tau_i)$. The ***greedy system of arcs*** is the set $A := \{a_1, a_2, \dots, a_\beta\}$.

Exchange arguments by Erickson and Whittlesey [51] and Colin de Verdière [33] imply that every arc in the greedy system is tight, and moreover that the greedy system of arcs has minimum total length among all systems of arcs.[3]

**3.2. Crossing parity vectors.** Fix a system of arcs $a_1, a_2, \dots, a_\beta$. In this section we define the ***crossing parity vector*** of any even subgraph $H$, with respect to the fixed system of arcs, and prove that this vector characterizes the homology class of $H$. Intuitively, the crossing parity vector is a vector of $\beta$ bits, whose $i$th bit is equal to 1 if and only if $H$ crosses arc $a_i$ an odd number of times. However, some care is required to ensure that this intuitive notion is actually consistent. Rather than working directly with $H$, we formally define the crossing parity vector of an arbitrary *ribbon perturbation* of an arbitrary *cycle decomposition* of $H$, and then argue that the resulting bit vector is the same for any cycle decomposition and any ribbon perturbation thereof.

First, fix a single cycle $\gamma$ in $G$ and an index $i$. Let $\tilde{\gamma}$ and $\tilde{a}_i$ be any ribbon perturbation of the cycle $\gamma$ and the arc $a_i$, as defined in Section 2.4. By definition, all intersections between $\tilde{\gamma}$ and $\tilde{a}_i$ are transverse crossings within vertex regions of the ribbon graph $G^\square$, and each vertex region $v^\square$ contains at most one such crossing. We define the crossing parity $\bar{x}_i(\gamma)$ to be 1 if $\tilde{\gamma}$ intersects $a_i$ an odd number of times, and 0 otherwise.

Consider two ribbon perturbations $\{\tilde{\gamma}, \tilde{a}_i\}$ and $\{\tilde{\gamma}', \tilde{a}_i'\}$. These two pairs of curves are homotopic in $G^\square$; that is, the pair of curves $\tilde{\gamma}$ and $\tilde{a}_i$ can be continuously deformed to the pair of curves $\tilde{\gamma}'$ and $\tilde{a}_i'$ within the ribbon graph. Classical topological arguments [5, 6, 105] imply that any homotopy between (pairs of) curves can be decomposed into a finite sequence of elementary *homotopy moves*, of three different types, as shown in Figure 3.2. Straightforward case analysis implies that any homotopy move preserves the parity of the number of crossings between the two deforming curves. It follows that the crossing parity $\bar{x}_i(\gamma)$ is independent of the choice of ribbon perturbation of $\gamma$ and $a_i$.

**Fig. 3.2.** Homotopy moves.

Now consider an even subgraph $H$. We define the *crossing parity* $\bar{x}_i(H)$ as the sum of the crossing parities of the cycles in any cycle decomposition of $H$. Again, we claim that this bit is independent of the choice of cycle decomposition. Consider two cycle decompositions $\gamma_1, \dots, \gamma_k$ and $\delta_1, \dots, \delta_l$ of $H$. Because the cycles in both decompositions traverse the same subset of edges, there are ribbon perturbations $\{\tilde{\gamma}_1, \dots, \tilde{\gamma}_k, \tilde{a}_i\}$ and $\{\tilde{\delta}_1, \dots, \tilde{\delta}_l, \tilde{a}_i\}$ that include identical perturbations $\tilde{a}_i$ of arc $a_i$ and that have identical intersections with each edge ribbon.

Consider the restriction of these two ribbon perturbations to a single vertex region $v^\square$, as shown in Figure 3.3. The intersections $\tilde{\gamma}_i \cap v^\square$ are simple, pairwise-disjoint, boundary-to-boundary paths in $v^\square$. The intersections $\tilde{\delta}_i \cap v^\square$ are also simple, pairwise-disjoint, boundary-to-boundary paths in $v^\square$. Moreover, these two sets of paths share identical endpoints. It follows that the symmetric difference $(\bigcup_i \tilde{\delta}_i \oplus \bigcup_i \tilde{\gamma}_i) \cap v^\square$ is the union of (not necessarily

[2] This running time requires that $g = O(n^{1-\varepsilon})$ for some constant $\varepsilon > 0$. But recall that we are assuming $g = o(\log n)$, since otherwise our minimum-cut algorithms are slower than textbook algorithms for arbitrary graphs.

[3] Specifically, Colin de Verdière's argument implies that the greedy system of arcs is a minimum-length basis in $G$ for the first relative homology group $H_1(\Sigma, \partial\Sigma)$ [33, Section 3]. Thus, each arc in the greedy system is as short as possible in its *relative homology* class.

simple) closed curves in $\nu^\square$. Any simple closed curve in $\nu^\square$ intersects any arc $\tilde{a}_i$ an even number of times. It follows that these two ribbon perturbations either cross $\tilde{a}_i$ an even number of times, or both cross $\tilde{a}_i$ an odd number of times. We conclude that $\bar{x}_i(H)$ is independent of the cycle decomposition of $H$.

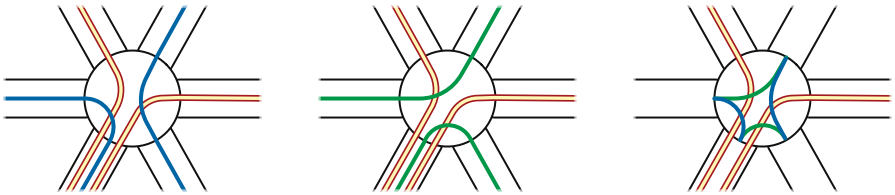

**Fig. 3.3.** Local view of ribbon perturbations of two cycle decompositions of the same even subgraph, and their symmetric difference. The doubled lines describe a perturbation of a single weakly simple arc $a_i$.

Finally, the crossing parity vector $\bar{x}(H)$ is defined as the vector $(\bar{x}_1(H), \bar{x}_2(H), \ldots, \bar{x}_\beta(H))$.

LEMMA 3.4. *Two even subgraphs are $\mathbb{Z}_2$-homologous if and only if their crossing parity vectors (with respect to the same system of arcs) are equal.*

**Proof:** Any arc crosses any facial cycle an even number of times; thus, the crossing parity vector of a facial cycle is the zero vector. Every boundary subgraph is the symmetric difference of facial cycles; thus, the crossing parity vector of any boundary subgraph is also the zero vector. Every pair of even subgraphs $H$ and $H'$ satisfies the identity $\bar{x}(H \oplus H') = \bar{x}(H) \oplus \bar{x}(H')$. In particular, if $\bar{x}(H \oplus H')$ is the zero vector, then $\bar{x}(H) = \bar{x}(H')$. We conclude that if two even subgraphs are homologous, then their crossing parity vectors are equal.

On the other hand, suppose $H$ is an even subgraph such that $\bar{x}(H) = 0$. We claim that there is a subset $S$ of faces whose boundary is $H$. Let $\tilde{H} \cup \tilde{A}$ be an arbitrary ribbon perturbation of (a cycle decomposition of) the even subgraph $H$ and the system of arcs $A$. By definition, every arc $\tilde{a}_i \in \tilde{A}$ intersects $\tilde{H}$ an even number of times.

The perturbed cycles and arcs $\tilde{H} \cup \tilde{A}$ partition the underlying surface into several regions. Lemma 3.3 implies that slicing the surface $\Sigma^\square$ along the perturbed arcs $\tilde{A}$ yields a topological disk $D = \Sigma^\square \setminus\setminus \tilde{A}$. The perturbed cycles in $\tilde{H}$ appear as disjoint cycles and boundary-to-boundary paths in $D$; it follows that we can consistently color the regions of $D$ red and blue, so that regions that share a boundary curve in $\tilde{H}$ have opposite colors. Because $\tilde{H}$ intersects each arc $\tilde{a}_i$ an even number of times, every region that contains part of the boundary of the original surface $\Sigma$ has the same color, without loss of generality red. It follows that any two regions that share a boundary curve in $\tilde{A}$ have the same color.

Every face region in the ribbon graph $\Sigma^\square$ is either entirely red or entirely blue, so we can pull the coloring back to the faces of $\Sigma$. In the resulting face coloring, the faces on either side of each edge in $H$ have opposite colors; the faces on either side of an edge not in $H$ have the same color; and all faces incident to boundary edges are red. It follows that $H$ is the boundary of the blue faces and is therefore null-homologous.

The identity $\bar{x}(H \oplus H') = \bar{x}(H) \oplus \bar{x}(H')$ now implies that if two even subgraphs have equal crossing parity vectors, they must be homologous. □

LEMMA 3.5. *We can compute the crossing parity vector of any even subgraph, with respect to any fixed system of arcs, in $O(\beta n)$ time.*

**Proof:** Let $a_1, \ldots, a_\beta$ be the fixed system of arcs. We can compute a cycle decomposition $\gamma_1, \ldots, \gamma_r$ of $H$ in $O(1)$ time per edge, by following the proof of Lemma 2.1. Finally, we can compute the number of crossings between (any ribbon perturbation of) any cycle $\gamma_i$ and any arc $a_j$ in time proportional to the number of edges in $\gamma_i$. □

**3.3. Homology signatures via tree-coforest decompositions.** Our second method associates a vector of $\beta$ bits with each edge $e$, called the ***signature*** of $e$; the homology class of any even subgraph is characterized by the bit-wise exclusive-or of the signatures of its edges.

Again, our construction is based on one of two natural generalizations of tree-cotree decompositions [45] to surfaces with boundary; the other generalization is used for computing crossing parity vectors as described above. We define a ***tree-coforest decomposition*** of $G$ to be any partition $(T, L, F)$ of the edges of $G$ into three edge-disjoint subgraphs with the following properties:

- $T$ is a spanning tree of $G$.
- $F^*$ is a spanning *forest* of $G^*$, that is, an acyclic subgraph that contains every vertex.
- Each component of $F^*$ contains a single dual boundary vertex.

- Finally, $L$ is the set of leftover edges $E \setminus (T \cup F)$.

LEMMA 3.6. *In any tree-coforest decomposition $(T, F, L)$ of any combinatorial surface with with boundary and with first Betti number $\beta$, the set $L$ contains exactly $\beta$ edges.*

**Proof:** Recall that $n$, $m$, and $f$ respectively denote the number of vertices, edges, and faces of $G$. We immediately have $|T| = n-1$ and $|F^*| = f$. Because $T$, $F$, and $L$ partition the edges, we conclude that $|L| = m-(n-1)-f = m-n-f+1 = 1-\chi = \beta$. □

Arbitrarily index the edges in $L$ as $e_1, \ldots, e_\beta$. For each edge $e_i \in L$, adding the corresponding dual edge $e_i^*$ to the forest $F^*$ creates a dual arc $\alpha_i$, which is either a simple path between distinct boundary vertices, or a nontrivial loop from a boundary vertex back to itself; in the second case, $\alpha_i$ may traverse some edges of $F^*$ twice. (The arguments in Lemmas 3.2 and 3.3 imply that the dual arcs $\alpha_i$ are weakly simple and weakly disjoint; however, our algorithms do not use these properties.) We call the set $\{\alpha_1, \alpha_2, \ldots, \alpha_\beta\}$ a ***system of dual arcs***. See Figure 3.4.

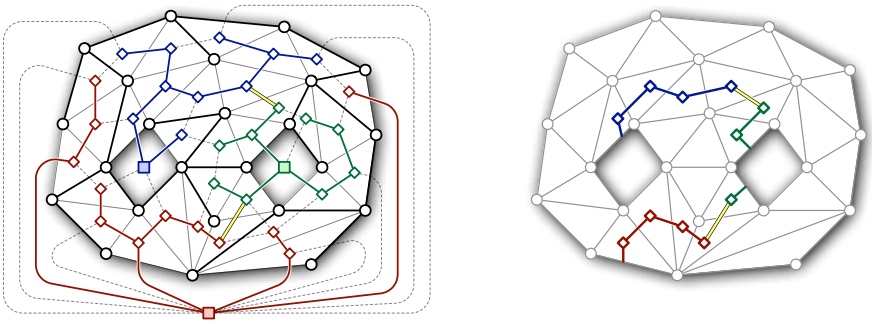

**Fig. 3.4.** Left: A tree-coforest decomposition of the graph in Figure 2.2; doubled lines indicate edges in $L$. Right: The resulting system of dual arcs. Compare with Figure 3.1.

Finally, for each edge $e$ in $G$, we define its signature $\boldsymbol{[e]}$ to be the $\beta$-bit vector whose $i$th bit is equal to 1 if and only if $e$ crosses $\alpha_i$ (that is, if $\alpha_i$ traverses the dual edge $e^*$) an odd number of times. The signature $[H]$ of an even subgraph $H$ is the bitwise exclusive-or of the signatures of its edges. Similarly, the signature $[\gamma]$ of a cycle $\gamma$ is the bitwise exclusive-or of the signatures of the edges that $\gamma$ traverses an odd number of times.

Let $\boldsymbol{h \oplus h'}$ denote the bitwise exclusive-or of two homology signatures $h$ and $h'$, or equivalently, their sum as elements of the homology group $(\mathbb{Z}_2)^\beta$. The identities $[H \oplus H'] = [H] \oplus [H']$ and $[\gamma \cdot \gamma'] = [\gamma] \oplus [\gamma']$ follow directly from the definitions.

LEMMA 3.7. *We can preprocess $G$ in $O(\beta n)$ time, so that the signature $[\gamma]$ of any cycle can be computed in $O(\beta)$ time per edge.*

**Proof:** A tree-coforest decomposition can be computed in $O(n)$ time as follows. First construct a graph $G'$ by identifying all the dual boundary vertices in $G^*$ to a single vertex. Compute a spanning tree of $G'$ by whatever-first search; the edges of this spanning tree define an appropriate dual spanning forest $F^*$. Construct the subgraph $G \setminus F$ and compute a spanning tree $T$ via whatever-first search. Finally, let $L = G \setminus (T \cup F)$. With the decomposition in hand, it is straightforward to compute each path $\alpha_i$ in $O(n)$ time, and then compute each edge signature in $O(\beta)$ time. □

For each edge $e_i \in L$, let $\gamma_i$ denote the fundamental cycle obtained by adding $e_i$ to $T$.

LEMMA 3.8. *The set of cycles $\{\gamma_1, \ldots, \gamma_\beta\}$ form a basis of the first homology group; precisely, these cycles lie in linearly independent homology classes that span the group.*

**Proof:** Consider any non-empty subset $\Gamma = \{\gamma_{i_1}, \ldots, \gamma_{i_k}\}$ of these cycles, and let $\gamma_i$ be an arbitrary member of this subset. Dual arc $\alpha_i$ crosses the subset exactly once, at the dual edge $e_i^*$. Therefore, even subgraph $\bigoplus_{\gamma \in \Gamma} \gamma$ does not bound the union of a subset of primal faces/dual vertices. □

A proof of our next lemma appears in Borradaile *et al.* [9, Corollary 3.5]. They cite an earlier version [50] of the current paper for the lemma statement, so we present a slightly simplified version of their proof for completeness.

LEMMA 3.9. *An even subgraph $H$ of $G$ is null-homologous in $\Sigma$ if and only if $[H] = 0$.*

**Proof:** Let $H$ be an even subgraph of $G$. Let $\Gamma = \{\gamma_{i_1}, \ldots, \gamma_{i_k}\} \subseteq \{\gamma_1, \ldots, \gamma_\beta\}$ be such that $\bigoplus_{\gamma \in \Gamma} \gamma$ is homologous to $H$. Then by definition, $H \oplus \bigoplus_{\gamma \in \Gamma} \gamma$ is the boundary of the union of a subset $Y$ of faces of $G$. The boundary of any

face $f$ is contractible in $\Sigma$ and therefore has signature 0. It follows immediately that $[H] = [\bigoplus_{\gamma\in\Gamma}\gamma] \oplus [\bigoplus_{f\in Y}\partial f] = [\bigoplus_{\gamma\in\Gamma}\gamma] \oplus \bigoplus_{f\in Y}[\partial f] = [\bigoplus_{\gamma\in\Gamma}\gamma]$. The $i$th bit of $[\bigoplus_{\gamma\in\Gamma}\gamma]$ is equal to 1 if and only if $\gamma_i \in \Gamma$, because dual arc $\alpha_i$ crosses no other member of $\{\gamma_1, \ldots, \gamma_\beta\}$. Therefore, $[H] = 0$ if and only if the homologous even subgraph $\bigoplus_{\gamma\in\Gamma}\gamma$ is empty. □

The following corollaries are now immediate.

COROLLARY 3.10. *Two even subgraphs $H$ and $H'$ of $G$ are $\mathbb{Z}_2$-homologous in $\Sigma$ if and only if $[H] = [H']$.*

COROLLARY 3.11. *Two cycles $\gamma$ and $\gamma'$ in $G$ are $\mathbb{Z}_2$-homologous in $\Sigma$ if and only if $[\gamma] = [\gamma']$.*

**4. Crossing Bounds and Triangulations.** In this section, we describe an algorithm to compute a minimum-weight even subgraph homologous with any specified even subgraph $H$ in $(g+b)^{O(g+b)}n \log\log n$ time, when the input graph is embedded on an ***orientable*** surface. In fact, our algorithm can be modified easily to compute a minimum-weight representative in *every* homology class in the same asymptotic running time; there are exactly $2^{2g+b-1}$ such classes; recall that we assume non-empty boundary. Lemma 2.2 implies our algorithm can be used to find a minimum $(s,t)$-cut in $G^*$ in the same amount of time.

Our algorithm closely resembles the algorithm of Chambers *et al.* [25] for computing a shortest splitting cycle; in fact, our algorithm is somewhat simpler. Our algorithm is based on the key observation (Lemma 4.1) that the shortest even subgraph in any homology class crosses any shortest path at most $O(g+b)$ times. The first stage of our algorithm cuts the underlying combinatorial surface into a topological disk by a greedy system of arcs, as described in Section 3.1. Next, we enumerate all possible ways for an even subgraph to intersect each of the greedy arcs at most $O(g+b)$ times; we quickly discard any crossing pattern that does not correspond to an even subgraph in the desired homology class. Each crossing pattern is realized by several (free) *homotopy* classes of sets of non-crossing cycles; we show how to enumerate these homotopy classes in Section 4.2. Then within each homotopy class, we find a minimum-length set of non-crossing cycles with each crossing pattern, essentially by reducing to a *planar* instance of the minimum-cut problem. The union of those cycles is an even subgraph in the desired homology class; we return the lightest such subgraph as our output.

**4.1. Crossing bound.** Our main technical lemma for this section establishes an upper bound on the number of crossings between members of a greedy system of arcs and the minimum-weight even subgraph in any homology class. Crossing-number arguments were first used by Cabello and Mohar [23] to develop the first subquadratic algorithms for shortest non-contractible and non-separating cycles in undirected surface embedded graphs; their arguments are the foundation of all later improvements of their algorithm [18, 20, 87]. Our proof is quite similar to the argument of Chambers *et al.* [25] that some shortest *splitting* cycle crosses any shortest path $O(g+b)$ times. However, our new proof is somewhat different, because we work explicitly with ribbon perturbations and the structure we seek is a true subgraph, which need not be connected, rather than a single weakly simple closed walk.

As mentioned in Section 3.1, we cannot consistently define when a shortest path crosses a $\mathbb{Z}_2$-minimal even subgraph. Instead, we upper-bound the smallest possible number of crossings between an entire arc in the greedy system and a ribbon perturbation of a cycle decomposition of a $\mathbb{Z}_2$-minimal even subgraph in each homology class. We emphasize that different cycle decompositions and different ribbon decompositions of the same even subgraph can have different numbers of crossings with the same arc.

LEMMA 4.1. *Let $G$ be an undirected graph with positively weighted edges, embedded on a surface with genus $g$ and $b > 0$ boundary components. Let $A = \{a_1, a_2, \ldots, a_\beta\}$ be a greedy system of arcs. Let $H$ be an even subgraph of $G$. There is a $\mathbb{Z}_2$-minimal even subgraph $H'$ homologous to $H$, a cycle decomposition $\{\gamma_1, \ldots, \gamma_r\}$ of $H'$, and a ribbon perturbation $\{\tilde{a}_1, \ldots, \tilde{a}_\beta, \tilde{\gamma}_1, \ldots, \tilde{\gamma}_r\}$, such that for each index $i$, the total number of crossings between the perturbed arc $\tilde{a}_i$ and the perturbed cycles $\tilde{\gamma}_1, \ldots, \tilde{\gamma}_r$ is at most $12g + 4b - 6$.*

**Proof:** Fix a $\mathbb{Z}_2$-minimal even subgraph $H'$ homologous to $H$, a cycle decomposition $\gamma_1, \gamma_2, \ldots, \gamma_r$ of $H'$, and a ribbon perturbation $\{\tilde{a}_1, \ldots, \tilde{a}_\beta, \tilde{\gamma}_1, \ldots, \tilde{\gamma}_r\}$ such that the total number $X$ of intersections between perturbed arcs $\tilde{a}_i$ and perturbed cycles $\tilde{\gamma}_j$ is as small as possible.

Recall from Section 3.1 that each arc $a_i$ is the concatenation of a shortest path $\sigma_i$ in the forest $F$, a single leftover edge $e_i \in L$, and the reversal of a shortest path $\tau_i$ in $F$. Let $\tilde{\sigma}_i$ and $\tilde{\tau}_i$ denote the components of $\tilde{a}_i \setminus e_i^\square$ containing $\sigma_i$ and $\tau_i$, respectively. Both $\tilde{\sigma}_i$ and $\tilde{\tau}_i$ are paths within the ribbon decomposition $\Sigma^\square$ with endpoints on the boundaries of vertex regions. Because perturbed paths intersect only in vertex regions, every point of intersection between $\tilde{\gamma}_j$ and $\tilde{a}_i$ is either a point in $\tilde{\gamma}_j \cap \tilde{\sigma}_i$ or a point in $\tilde{\gamma}_j \cap \tilde{\tau}_i$.

Let $\sigma$ be one of the shortest paths $\sigma_i$ or $\tau_i$ for some index $i$, and let $\tilde{\sigma}$ be the corresponding path $\tilde{\sigma}_i$ or $\tilde{\tau}_i$ in

the ribbon perturbation. For each index $j$, let $x_j$ denote the number of points in $\tilde{\sigma} \cap \tilde{\gamma}_j$, and let $x = x_1 + x_2 + \cdots x_\beta$. If we contract $\tilde{\sigma}$ to a single point $\tilde{v}$ on the boundary of $\Sigma^\square$, each cycle $\tilde{\gamma}_j$ is contracted to the union of $x_j$ simple loops, which are pairwise disjoint except at their common basepoint $\tilde{v}$. Altogether, we obtain a set $\mathcal{L}$ of $x$ simple interior-disjoint loops in $\Sigma^\square$.

Our key claim is that these $x$ loops lie in distinct nontrivial homotopy classes. This claim implies that $\mathcal{L}$ defines an embedding of a single-vertex graph with $x$ edges onto the surface $\Sigma^\square$, where every face of the embedding is bounded by at least three edges. Euler's formula now implies that $x \le 6g + 2b - 3$ [25, Lemma 2.1], completing the proof of the lemma.

We prove our key claim by contradiction, using a pair of exchange arguments.

***No contractible loops.*** For the sake of argument, suppose $\mathcal{L}$ contains a contractible loop. Let $\ell$ be an *innermost* contractible loop, meaning there are no other contractible loops in the disk bounded by $\ell$. This loop is the contraction of some subpath $\tilde{\gamma}(p,q)$ of some cycle $\tilde{\gamma}$ in the perturbed cycle decomposition; the endpoints $p$ and $q$ of this subpath lie on $\tilde{\sigma}$. Let $u$ and $v$ be the vertices of $G$ such that $p \in u^\square$ and $q \in v^\square$. Fix points $p^\flat \in u^\square \cap \tilde{\gamma}$ and $q^\sharp \in v^\square \cap \tilde{\gamma}$ such that the subpath $\tilde{\pi} = \tilde{\gamma}(p^\flat, q^\sharp)$ strictly contains $\tilde{\gamma}(p,q)$ but has no more intersections with any perturbed arc $\tilde{a}_i$ or perturbed cycle $\tilde{\gamma}_j$. Let $\pi$ be the path from $u$ to $v$ in $G$ determined by the sequence of vertex regions and edge ribbons traversed by $\tilde{\pi}$.

Now let $\tilde{\rho}$ be a path from $q^\sharp$ to $p^\flat$ that closely parallels $\tilde{\sigma}(q,p)$; specifically, $\tilde{\rho}$ traverses the same sequence of vertex regions and edge ribbons as $\tilde{\sigma}(q,p)$, without crossing any perturbed arc $\tilde{a}_i$ or perturbed cycle $\tilde{\gamma}_j$. See Figure 4.1. Let $\rho$ be the path from $v$ to $u$ obtained by pulling $\tilde{\rho}$ back to $G$. Because $\rho$ is a subpath of the shortest path $\sigma$, it is actually a *shortest* path from $v$ to $u$.

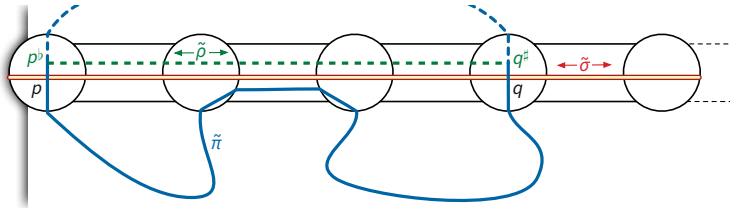


**Fig. 4.1.** Exchange argument to remove a contractible loop.

Because $\ell$ is contractible, the cycle $\tilde{\delta} = \tilde{\pi} \cdot \tilde{\rho}$ is also contractible. Thus, the corresponding closed walk $\delta = \pi \cdot \rho$ in $G$ is contractible and therefore null-homologous. It follows that the subgraph $H^- = H' \oplus \delta$ is homologous with $H'$ and thus with $H$. Because $\rho$ is a shortest path, the weight of $H^-$ cannot be larger than the weight of $H'$, and because $H'$ is $\mathbb{Z}_2$-minimal, the weight of $H^-$ cannot be smaller than the weight of $H'$. We conclude that the subgraphs $H^-$ and $H'$ have equal weight; both are $\mathbb{Z}_2$-minimal.[4]

The cycle $\tilde{\gamma}' = \tilde{\gamma} \oplus \tilde{\delta}$ intersects $\sigma$ strictly fewer times than $\tilde{\gamma}$, and does not cross any arc $\tilde{a}_i$ more times than $\tilde{\gamma}$. Because $\ell$ is an innermost contractible loop in $\mathcal{L}$, the cycle $\tilde{\gamma}'$ is simple and disjoint from all other cycles $\tilde{\gamma}_j$ in the perturbed cycle decomposition. Thus, replacing $\tilde{\gamma}$ with $\tilde{\gamma}'$ gives us a perturbed cycle decomposition of $H^-$ with fewer than $X$ crossings. But this contradicts our definition of $X$. We conclude that no loop in $\mathcal{L}$ is contractible.

***No homotopic loops.*** Now suppose for the sake of argument that $\mathcal{L}$ contains homotopic pairs of loops. Let $\ell$ and $\ell'$ be homotopic loops in $\mathcal{L}$ with no other homotopic loops between them. These two loops are contractions of subpaths $\tilde{\gamma}(p,q)$ and $\tilde{\gamma}'(r,s)$ of (not necessarily distinct) cycles $\tilde{\gamma}$ and $\tilde{\gamma}'$ in the perturbed cycle decomposition, with all endpoints $p,q,r,s$ on the path $\tilde{\sigma}$. As in the previous argument, we extend these subpaths within the vertex regions containing their endpoints, to obtain paths $\tilde{\pi} = \tilde{\gamma}(p^\flat, q^\sharp)$ and $\tilde{\gamma}'(r^\flat, s^\sharp)$. Let $\pi$ and $\pi'$ be the paths obtained by pulling $\tilde{\pi}$ and $\tilde{\pi}'$ back to $G$.

Let $\tilde{\rho}$ be a path from $r^\sharp$ to $r^\flat$ that closely follows $\tilde{\sigma}$, and similarly let $\tilde{\rho}'$ be a path from $q^\flat$ to $s^\sharp$ that closely follows $\tilde{\sigma}$. Let $\rho$ and $\rho'$ be the paths obtained by pulling $\tilde{\rho}$ and $\tilde{\rho}'$ back to $G$. Because $\rho$ and $\rho'$ are subpaths of $\sigma$, they are shortest paths in $G$.

Because $\ell$ and $\ell'$ are homotopic, the cycle $\tilde{\delta} = \tilde{\pi} \cdot \tilde{\rho} \cdot rev(\tilde{\pi}') \cdot \tilde{\rho}'$ is contractible, which implies that the corresponding closed walk $\delta = \pi \cdot \rho \cdot rev(\pi') \cdot \rho'$ in $G$ is contractible and therefore null-homologous. It follows that the subgraph $H^- = H' \oplus \delta$ is homologous to $H'$ and therefore to $H$. Moreover, $H^-$ must be $\mathbb{Z}_2$-minimal,

[4] If shortest paths in $G$ are unique, we can actually conclude at this point that $H^- = H'$.

because $\rho$ and $\rho'$ are shortest paths. Finally, exchanging $\tilde{\pi}$ and $\tilde{\pi}'$ with $\tilde{\rho}$ and $\tilde{\rho}'$ transforms our perturbed cycle decomposition of $H'$ into a perturbed cycle decomposition of $H^-$ with fewer than $X$ crossings, violating our definition of $X$. We conclude that no two loops in $\mathcal{L}$ are homotopic. □

**4.2. Triangulations and crossing sequences.** Our algorithm for computing minimum-weight even subgraph in a given homology class follows a strategy first used by Kutz to compute shortest non-contractible cycles [87]; in fact our algorithm uses Kutz's algorithm as a subroutine.

Our algorithm begins begin by constructing a greedy system of arcs $a_1, \ldots, a_\beta$ for the input combinatorial surface $\Sigma$; we also compute a ribbon perturbation $\{\tilde{a}_1, \ldots, \tilde{a}_\beta\}$ of this greedy systen into pairwise-disjoint arcs, as described in Lemma 3.2. Lemma 3.3 implies that slicing the combinatorial surface $\Sigma^\square$ along these perturbed arcs yields a disk $D^\square$, which we call a *polygonal schema*. The boundary of $D^\square$ alternates between perturbed arcs $\tilde{a}_i$ and boundary paths of $\Sigma^\square$; each perturbed arc appears twice on the boundary of $D^\square$. Replacing each copy of each perturbed arc on the boundary of $D^\square$ with a single edge, and contracting each boundary path of $\Sigma^\square$ to a single point, yields a $2\beta$-gon that we call the *abstract polygonal schema*. See Figure 4.2 for an illustrative example.

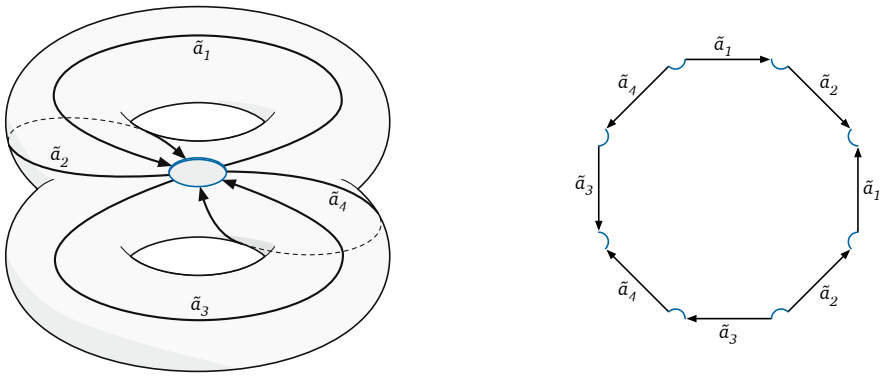


**Fig. 4.2.** A system of arcs on a surface with genus 2 and one boundary cycle, and the polygonal schema obtained by slicing along the arcs.

We next dualize the abstract polygonal schema, replacing each boundary edge with a vertex, and connecting vertices which correspond to adjacent edges in the primal schema. Thus, the dual polygonal schema is a $2\beta$-gon with two vertices corresponding to each perturbed greedy arc $\tilde{a}_i$. Any collection of disjoint simple cycles in $\Sigma^\square$ corresponds to a *weighted triangulation* [25] of this dual polygonal schema, which includes an edge between two vertices of the dual abstract polygonal schema if and only if some cycle consecutively crosses the corresponding pair of perturbed greedy arcs. Each triangulation edge is weighted by the number of times such a crossing occurs in our collection. See Figure 4.3 for an illustration of this correspondence. We note that the crossing parity vector of the collection of cycles can then be computed directly from this weighted triangulation.

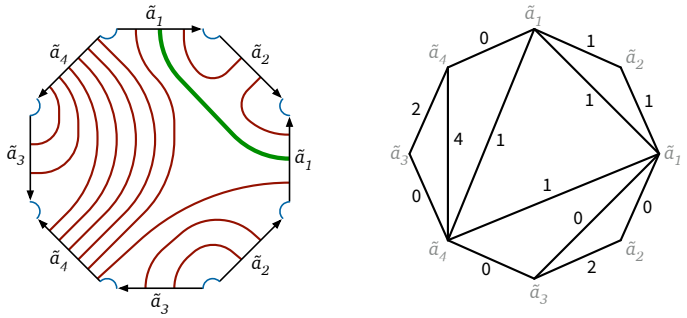


**Fig. 4.3.** Two disjoint simple cycles on a surface of genus 2 with one boundary, and the corresponding weighted triangulation.

Conversely, we call a weighted triangulation *valid* if corresponding vertices are incident to edges of equal total weight, and the weight of each edge is between 0 and $12g + 4b - 6$. Altogether, there are $(g + b)^{O(g+b)}$ different valid weighted triangulations. Each valid weighted triangulation corresponds to a collection of simple disjoint cycles in $\Sigma^\square$, which is unique up to homotopy. Lemma 4.1 implies that every homology class contains a $\mathbb{Z}_2$-minimal

even subgraph $H'$, such that some ribbon perturbation of some cycle decomposition of $H'$ is consistent with a valid weighted triangulation.

For each valid weighted triangulation, we can compute a corresponding collection of abstract cycles in $O((g+b)^2)$ time by brute force. In the same time, we can also compute the *sequence* of crossings of each abstract cycle with our perturbed greedy arcs. An algorithm of Kutz [87] computes the shortest cycle in $G$ with a given crossing sequence of length $x$ in $O(xn \log n)$ time, by gluing together $x$ copies of the polygonal schema $D^\square$ into an annulus and computing the shortest generating cycle of that annulus via Frederickson's planar minimum-cut algorithm [58]. Italiano *et al.* [78] point out that their recent $O(n \log\log n)$-time improvement in computing minimum $(s,t)$-cuts in planar graphs can be used instead of Frederickson's algorithm. Thus, for each weighted triangulation, we can compute the shortest corresponding set of cycles in $\Sigma^\square$, and therefore the minimum-weight corresponding even subgraph of $G$, in $O((g+b)^2 n \log\log n)$ time.

Now suppose we are given an even subgraph $H$. In $O(gn)$ time, we can compute the crossing parity vector $\bar{x}(H)$ by decomposing $H$ into cycles, perturbing the cycles within the ribbon graph $\Sigma^\square$, and counting crossings with the perturbed greedy arcs $\tilde{a}_i$ (modulo 2). To compute the minimum-weight even subgraph $\mathbb{Z}_2$-homologous with $H$, we enumerate all valid weighted triangulations with the correct crossing parity vector, compute a minimum-weight even subgraph corresponding to each triangulation, and return the smallest even subgraph found.

THEOREM 4.2. *Let $G$ be an undirected graph with positively weighted edges, embedded on an **orientable** surface with genus $g$ and $b$ boundary components, and let $H$ be an even subgraph of $G$. We can compute the minimum-weight even subgraph homologous with $H$ in $(g+b)^{O(g+b)} n \log\log n$ time.*

COROLLARY 4.3. *Let $G$ be an undirected graph with positively weighted edges, embedded on an **orientable** surface with genus $g$ (possibly with boundary), and let $s$ and $t$ be vertices of $G$. We can compute the minimum-weight $(s,t)$-cut in $G$ in $g^{O(g)} n \log\log n$ time.*

**4.3. Non-orientable Surfaces.** Kutz's reduction to the planar minimum-cut problem is the only component of our homology localization algorithm that requires the underlying surface to be orientable. If the underlying surface is not orientable, then gluing a cycle of copies of the polygonal schema $D^\square$ according to a valid crossing sequence could produce a Möbius band instead of an annulus. The fastest algorithm known for computing a shortest generating cycle in a combinatorial Möbius band runs in $O(n \log n)$ time, using Klein's multiple-source shortest path algorithm [86]; no improvement similar to the $O(n \log\log n)$-time algorithm of Italiano *et al.* [78] is known. The resulting algorithm for computing minimum-weight homologous subgraphs runs in $(g+b)^{O(g+b)} n \log\log n$ time; because this is subsumed by our later results, we omit further details.

Nevertheless, we can extend Corollary 4.3 to non-orientable surface graphs with no penalty in the running time. A simple cycle $\gamma$ on a surface is ***one-sided*** if some neighborhood of $\gamma$ is a Möbius band, and ***two-sided*** otherwise.

LEMMA 4.4. *Let $G$ be an undirected graph with positively weighted edges, embedded on a non-orientable surface $\Sigma$ with two boundary cycles $s^*$ and $t^*$, and let $H$ be the even subgraph of $G$ dual to a minimum $(s,t)$-cut in $G^*$. In any ribbon perturbation of any cycle decomposition of $H$, every cycle is two-sided.*

**Proof:** Consider any ribbon perturbation $\{\tilde{\gamma}_1, \ldots, \tilde{\gamma}_r\}$ of any cycle decomposition of $H$. Slicing the underlying surface along the perturbed cycles $\tilde{\gamma}_i$ separates it into two components, one containing the boundary cycle $s^*$, and the other containing the boundary cycle $t^*$. Color the first component red and second blue. If any cycle $\tilde{\gamma}_i$ has the same color on both sides, we can safely delete it from the cycle decomposition; the smaller set of cycles still separate the red and blue components, which implies that the subgraph $H \setminus \gamma_i$ is the dual of an $(s,t)$-cut, contradicting our assumption that $H$ is the dual of the *minimum* $(s,t)$-cut. We conclude that every cycle $\tilde{\gamma}_i$ is two-sided; specifically, it has red points on one side and blue points on the other. □

The previous lemma implies that if our minimum-cut algorithm can ignore weighted subgraphs that induce one-sided cycles. Specifically, whenever the algorithm glues copies of $D^\square$ according to some crossing sequence, if the resulting surface is a Möbius band, we ignore the weighted triangulation that produced it. All other aspects of the algorithm are unchanged.

COROLLARY 4.5. *Let $G$ be an undirected graph with positively weighted edges, embedded on a **possibly non-orientable** surface with genus $g$ (possibly with boundary), and let $s$ and $t$ be vertices of $G$. We can compute the minimum-weight $(s,t)$-cut in $G$ in $g^{O(g)} n \log\log n$ time.*

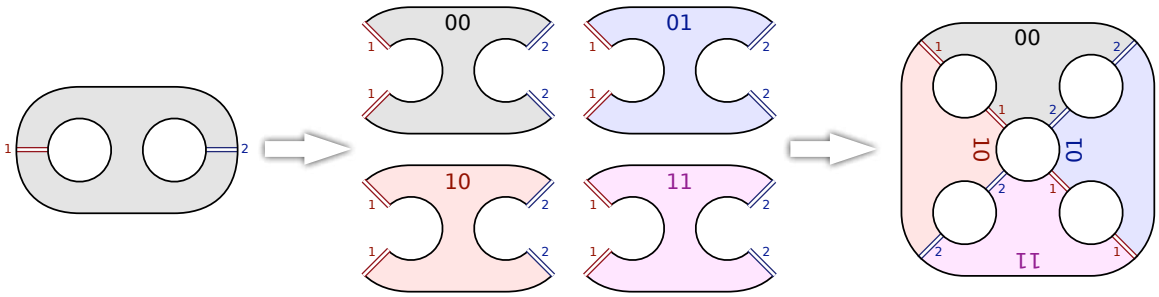


**Fig. 5.1.** Constructing the $\mathbb{Z}_2$-homology cover of a pair of pants (a genus zero surface with three boundaries).

**5. The $\mathbb{Z}_2$-Homology Cover.** At a very high level, our algorithm in Section 4 find minimum-weight homologous subgraphs by enumerating possible *homotopy* classes of the cycles in a cycle decomposition, and then finding the shortest cycle in each possible homotopy class by searching a finite portion of the universal cover of the surface $\Sigma$. In this section, we describe a more direct algorithm, which finds the shortest cycle in each *homology* class, by constructing and searching a space which we call the $\mathbb{Z}_2$*-homology cover*. Specifically, given a homology signature $h \in (\mathbb{Z}_2)^\beta$, our algorithm computes the shortest cycle with signature $h$ in $2^{O(\beta)} n \log n$ time, using a generalization of Klein's multiple-source shortest path algorithm [86] for planar graphs to higher-genus embedded graphs [20, 48]. In fact, because there are only $2^\beta$ homology classes, we can compute the shortest cycle in *every* homology class in the same running time. We then assemble the minimum-weight *even subgraph* in any given homology class from these $\mathbb{Z}_2$-minimal cycles using dynamic programming.

In the preliminary version of this section [50], we described an algorithm to compute shortest non-separating cycles in a directed surface graph in $g^{O(g)} n \log n$ time, improving (for fixed $g$) an earlier algorithm of Cabello *et al.* [21] that runs in $O(g^{1/2} n^{3/2} \log n)$ time. Using similar techniques but with different covering spaces, Erickson [47] and Fox [56] described even faster algorithms that find shortest non-separating cycles in $O(g^2 n \log n)$ time and shortest non-contractible cycles in $O(g^3 n \log n)$ time. In light of these improvements, we omit discussion of our non-separating cycle algorithm from this paper.

We emphasize that all results in this section apply to both orientable and non-orientable surfaces.

**5.1. Definition and construction.** We begin by computing homology signatures for the edges of $G$ in $O(\beta n)$ time, as described in Section 3.3. After computing homology signatures for each edge, the $\mathbb{Z}_2$-homology cover of a combinatorial surface can be defined using a standard *voltage construction* [65, Chapter 4], as follows.

We first define the covering graph $\overline{G}$. For simplicity, we regard every edge $uv$ of $G$ as a pair of oppositely oriented *darts* $u{\to}v$ and $v{\to}u$. The vertices of $\overline{G}$ are all ordered pairs $(v,h)$ where $v$ is a vertex of $G$ and $h$ is an element of $(\mathbb{Z}_2)^\beta$. The darts of $\overline{G}$ are the ordered pairs $(u{\to}v,h) := (u,h){\to}(v,h \oplus [uv])$ for all edges $u{\to}v$ of $G$ and all homology classes $h \in (\mathbb{Z}_2)^\beta$, and the reversal of any dart $(u{\to}v,h)$ is the dart $(v{\to}u,h \oplus [uv])$.

Now let $\pi\colon \overline{G} \to G$ denote the covering map $\pi(v,h) = v$; this map projects any cycle in $\overline{G}$ to a cycle in $G$. To define a cellular embedding of $\overline{G}$, we declare a cycle in $\overline{G}$ to be a face if and only if its projection is a face of $G$. The combinatorial surface defined by this embedding is the $\mathbb{Z}_2$-homology cover $\overline{\Sigma}$.

Our construction can be interpreted more topologically as follows. Let $\alpha_1, \ldots, \alpha_\beta$ denote the system of dual arcs used to define the homology signatures $[e]$. The surface $D := \Sigma \setminus (\alpha_1 \cup \cdots \cup \alpha_\beta)$ is a topological disk. Each arc $\alpha_i$ appears on the boundary of $D$ as two segments $\alpha_i^+$ and $\alpha_i^-$. For each signature $h \in (\mathbb{Z}_2)^\beta$, we create a disjoint copy $(D,h)$ of $D$; for each index $i$, let $(\alpha_i^+,h)$ and $(\alpha_i^-,h)$ denote the copies of $\alpha_i^+$ and $\alpha_i^-$ in the disk $(D,h)$. For each index $i$, let $b_i$ denote the $\beta$-bit vector whose $i$th bit is equal 1 and whose other $\beta-1$ bits are all equal to 0. The $\mathbb{Z}_2$-homology cover $\overline{\Sigma}$ is constructed by gluing the $2^\beta$ copies of $D$ together by identifying boundary paths $(\alpha_i^+,h)$ and $(\alpha_i^-,h \oplus b_i)$, for every index $i$ and homology class $h$. See Figure 5.1 for an example.

LEMMA 5.1. *The combinatorial surface $\overline{\Sigma}$ has $\overline{n} = 2^\beta n$ vertices, genus $\overline{g} = O(2^\beta \beta)$, and $\overline{b} = O(2^\beta b)$ boundaries, and it can be constructed in $O(2^\beta n)$ time.*

**Proof:** Let $m$ and $f$ denote the number of edges and faces of $\Sigma$, respectively. Recall that the Euler characteristic of $\Sigma$ is $\chi = n - m + f = 2 - 2g - b = 1 - \beta$. The combinatorial surface $\overline{\Sigma}$ has exactly $\overline{n} = 2^\beta n$ vertices, $2^\beta m$ edges, and $2^\beta f$ faces, so its Euler characteristic is $\overline{\chi} = 2^\beta(1-\beta)$.

If $b > 1$, then each boundary cycle $\delta_i$ has a non-zero homology signature; at least one arc $\alpha_j$ has exactly

one endpoint on $\delta_i$. Thus, $\overline{\Sigma}$ has exactly $\overline{b} = 2^{\beta-1}b$ boundary cycles, each of which is a double-cover (in fact, the $\mathbb{Z}_2$-homology cover) of some boundary cycle $\delta_i$. It follows that $\overline{\Sigma}$ has genus $\overline{g} = 1-(\overline{\chi}+\overline{b})/2 = 2^{\beta-2}(4g+b-4)+1$. (Somewhat surprisingly, $\overline{\Sigma}$ may have positive genus even when $\Sigma$ does not!) On the other hand, when $b = 1$, the boundary cycle $\delta_1$ is null-homologous, so $\overline{\Sigma}$ has $\overline{b} = 2^\beta b$ boundary cycles, and thus $\overline{\Sigma}$ has genus $\overline{g} = 1-(\overline{\chi}+\overline{b})/2 = 2^\beta(g-1)+1$.

After computing the homology signatures for $\Sigma$ in $O(\beta n)$ time, following Lemma 3.7, it is straightforward to construct $\overline{\Sigma}$ in $O(\overline{n}) = O(2^\beta n)$ time. □

Each edge in $\overline{G}$ inherits the weight of its projection in $G$. Now consider an arbitrary path $p$ in $G$, with (possibly equal) endpoints $u$ and $v$. A straightforward induction argument implies that for any homology class $h \in (\mathbb{Z}_2)^\beta$, the path $p$ is the projection of a unique path from $(u,h)$ to $(v, h \oplus [p])$, which we denote $\boldsymbol{(p,h)}$. Moreover, this lifted path has the same length as its projection. The following lemmas are now immediate.

LEMMA 5.2. *Every lift of a shortest path in $G$ is a shortest path in $\overline{G}$.*

LEMMA 5.3. *A loop $\ell$ in $G$ with basepoint $v$ is $\mathbb{Z}_2$-minimal if and only if, for every homology class $h \in (\mathbb{Z}_2)^\beta$, the lifted path $(\ell, h)$ is a shortest path in $\overline{G}$ from $(v,h)$ to $(v, h \oplus [\ell])$.*

**5.2. Computing $\mathbb{Z}_2$-minimal cycles.** The results in the previous section immediately suggest an algorithm to compute the shortest cycle in a given $\mathbb{Z}_2$-homology class $h$ in time $2^{O(\beta)}n^2$: construct the $\mathbb{Z}_2$-homology cover, and then compute the shortest path from $(v,0)$ to $(v,h)$, for every vertex $v$ in the original graph. In this section, we describe a more complex algorithm that runs in time $2^{O(\beta)}n\log n$. Recall that any path $\sigma$ from $u$ to $v$ in $G$ is the projection of a unique path $(\sigma, 0)$ from $(u,0)$ to $(v,[\sigma])$ in $\overline{G}$.

LEMMA 5.4. *Let $\gamma$ be a $\mathbb{Z}_2$-minimal cycle in $G$ in homology class $h$, and let $\sigma$ be any shortest path in $G$ that intersects $\gamma$. There is a $\mathbb{Z}_2$-minimal cycle $\gamma'$ in homology class $h$, which is the projection of a shortest path $(\gamma', h)$ in $\overline{G}$ that starts with a subpath of $(\sigma, 0)$ but does not otherwise intersect $(\sigma, 0)$.*

**Proof:** Let $v$ be the vertex of $\sigma \cap \gamma$ closest to the starting vertex of $\sigma$, and let $(v,h)$ be the corresponding vertex of the lifted path $(\sigma, 0)$. Think of $\gamma$ as a loop based at $v$. Lemma 5.3 implies that the lifted path $(\gamma, h)$ is a shortest path from $(v,h)$ to $(v, h \oplus [\gamma])$.

Now let $(w,h')$ be the last vertex along $(\gamma,h)$ that is also a vertex of $(\sigma,0)$. Let $(\gamma',h)$ be the path obtained from $(\gamma,h)$ by replacing the subpath from from $(v,h)$ to $(w,h')$ with the corresponding subpath of $(\sigma,0)$. By construction, $(\gamma',h)$ starts with a subpath of $(\sigma,0)$ but does not otherwise intersect $(\sigma,0)$. Because both $(\gamma,h)$ and $(\sigma,0)$ are shortest paths in $\overline{\Sigma}$, the new path $(\gamma',h)$ has the same length as $(\gamma,h)$. Thus, the projected cycle $\gamma'$ has the same length and homology class as $\gamma$, which implies that $\gamma'$ is $\mathbb{Z}_2$-minimal. □

We emphasize that the modified cycle $\gamma'$ may intersect $\sigma$ arbitrarily many times; however, all such intersections lift to intersections between $(\gamma',h)$ and lifts of $\sigma$ other than $(\sigma,0)$.

Our algorithm uses a generalization of Klein's multiple-source shortest path algorithm [86] to higher-genus embedded graphs, first developed by Chambers *et al.* [20] and later slightly improved by Erickson *et al.* [48].

LEMMA 5.5 ([20,48]). *Let $G$ be a graph with non-negatively weighted edges, cellularly embedded on a surface of genus $g$ (possibly non-orientable and possibly with boundary), and let $f$ be an arbitrary face of $G$. We can preprocess $G$ in $O(g^2 n\log n)$ time and $O(gn\log n)$ space so that the shortest-path distance from any vertex incident to $f$ to any other vertex can be retrieved in $O(\log n)$ time.*

THEOREM 5.6. *Let $G$ be an undirected graph with positively weighted edges, cellularly embedded on a (possibly non-orientable) surface $\Sigma$ with first Betti number $\beta$, and let $\gamma$ be a cycle in $G$ with $k$ edges. A shortest cycle in $\Sigma$ that is $\mathbb{Z}_2$-homologous with $\gamma$ can be computed in $O(\beta k + 8^\beta \beta^3 n\log n)$ time.*

**Proof:** We begin by computing homology signatures for the edges of $G$ in $O(\beta n)$ time, using a dual system of arcs, as described in Section 3.3. In $O(\beta k)$ time, we then compute the homology signature $[\gamma]$. If $[\gamma] = 0$, we can immediately return the empty walk, so assume otherwise. We then construct the $\mathbb{Z}_2$-homology cover $\overline{G}$ in $O(2^\beta n\log n)$ time, using the same system of dual arcs, as described in Section 5.1.

To exploit Lemma 5.4, we need a set $S$ of shortest paths in $G$ that are guaranteed to intersect every cycle with non-trivial homology. Somewhat counterintuitively, we construct $S$ essentially by building the *greedy primal* system of arcs, as described in Section 3.1. Specifically, we build the greedy forest-cotree decomposition $(\partial G, F, C^*, L)$, and let $S$ be the set of paths in the forest $F$ from the boundary of $\Sigma$ to the endpoints of edges in $L$. Lemma 3.1 immediately implies that $S$ contains $2\beta$ shortest paths. Lemma 3.3 implies that every cycle with non-trivial homology—in fact, every non-contractible cycle—shares at least one vertex with at least one path in $S$. We

emphasize that our algorithm in this section does not need ribbon perturbations; in particular, every path in $S$ is simple, and our algorithm considers each path in $S$ in isolation.

Then for each each shortest path $\sigma \in S$, we look for a $\mathbb{Z}_2$-minimal cycle homologous to $\gamma$ that intersects $\sigma$ and has the structure described in Lemma 5.4. Lemma 5.2 implies that $\sigma$ is the projection of a shortest path $(\sigma, 0)$ in $\overline{G}$; let us write $(\sigma, 0) = (v_0, 0) \to (v_1, h_1) \to \cdots \to (v_t, h_t)$. We construct the combinatorial surface $\overline{\Sigma} \setminus\!\setminus (\sigma, 0)$ by splitting the path $(\sigma, 0)$ into two parallel paths from $(v_0, 0)$ to $(v_t, h_t)$, which we denote $(\sigma, 0)^+$ and $(\sigma, 0)^-$. For each index $1 \le i \le t-1$, let $(v_i, h_i)^+$ and $(v_i, h_i)^-$ denote the copies of vertex $(v_i, h_i)$ on the paths $(\sigma, 0)^+$ and $(\sigma, 0)^-$, respectively. The paths $(\sigma, 0)^+$ and $(\sigma, 0)^-$ bound a new common face $f_{(\sigma,0)}$ in $\overline{\Sigma} \setminus\!\setminus (\sigma, 0)$.

Lemma 5.4 implies that if any $\mathbb{Z}_2$-minimal cycle homologous to $\gamma$ intersects $\sigma$, then some $\mathbb{Z}_2$-minimal cycle homologous to $\gamma$ is the projection of a shortest path in $\overline{\Sigma} \setminus\!\setminus (\sigma, 0)$ from some vertex $(v_i, h_i)^\pm$ to the corresponding vertex $(v_i, h_i \oplus [\gamma])$. To compute these shortest paths, we implicitly compute the shortest path in $\overline{\Sigma} \setminus\!\setminus (\sigma, 0)$ from every vertex on the boundary of $f_{(\sigma,0)}$ to *every* vertex of $\overline{\Sigma} \setminus\!\setminus (\sigma, 0)$, using Lemma 5.5. The resulting algorithm runs in $O(\overline{g}^2\, \overline{n} \log \overline{n}) = O(8^\beta \beta^3\, n \log n)$ time, by Lemma 5.1. □

By running this algorithm $2^\beta$ times, we can compute the shortest cycle in $\Sigma$ in *every* $\mathbb{Z}_2$-homology class, in $O(16^\beta \beta^3\, n \log n)$ time.

**5.3. Minimum cuts from the homology cover.** We now apply our algorithm for computing $\mathbb{Z}_2$-minimal cycles to the problem of computing $\mathbb{Z}_2$-minimal even subgraphs in undirected surface embedded graphs. Theorem 5.6 immediately implies that we can compute a minimum-weight *cycle* in every $\mathbb{Z}_2$-homology class in $O(16^\beta \beta^3\, n \log n)$ time. However, the minimum weight *even subgraph* in a given homology class may not be (the carrier of) a $\mathbb{Z}_2$-minimal cycle. In particular, if all edge weights are strictly positive, and some $\mathbb{Z}_2$-minimal cycle $\gamma$ traverses any edge more than once, then every minimum-weight even subgraph homologous to $\gamma$ *must* be disconnected. However, any *connected* $\mathbb{Z}_2$-minimal even subgraph is the carrier of a $\mathbb{Z}_2$-minimal cycle, and the components of any $\mathbb{Z}_2$-minimal even subgraph are themselves $\mathbb{Z}_2$-minimal even subgraphs. Thus, we can assemble a $\mathbb{Z}_2$-minimal even subgraph in any homology class from a subset of the $\mathbb{Z}_2$-minimal cycles we have already computed. The following lemma puts an upper bound on the number of cycles we need.

LEMMA 5.7. *Every $\mathbb{Z}_2$-minimal even subgraph of $G$ has at most $g + b - 1$ components.*

**Proof:** Let $H$ be an even subgraph of $G$ with more than $g + b - 1$ components. Each component has a cycle decomposition, so $H$ must have a cycle decomposition $\gamma_1, \ldots, \gamma_r$ consisting of $r > g + b - 1$ elements. Let $\Sigma^\bullet$ be the surface obtained from $\Sigma$ by gluing a disk to each boundary component; $\Sigma^\bullet$ is a surface of genus $g$ with no boundary but with $b$ designated faces.

Now consider the surface $\Sigma' = \Sigma^\bullet \setminus (\gamma_1 \cup \cdots \cup \gamma_r)$. The definition of genus implies that $\Sigma'$ cannot be connected; indeed, $\Sigma'$ must have at least $b+1$ components. So by the pigeonhole principle, some component $\Sigma''$ of $\Sigma'$ contains none of the $b$ designated faces. Thus, the boundary of $\Sigma''$ is null-homologous in $\Sigma^\bullet$, and therefore in $\Sigma$.

We conclude that some subgraph $H'$ of $H$ is null-homologous. Because all edges in $H'$ have positive weight, we conclude that $H$ is not $\mathbb{Z}_2$-minimal. □

THEOREM 5.8. *Let $G$ be an undirected graph with positively weighted edges, embedded on a (possibly non-orientable) surface with first Betti number $\beta$. A minimum-weight even subgraph of $G$ in each $\mathbb{Z}_2$-homology class can be computed in $O(16^\beta \beta^3\, n \log n)$ time.*

**Proof:** Our algorithm computes a minimum-weight cycle $\gamma_h$ in every $\mathbb{Z}_2$-homology class $h$ in $O(16^\beta \beta^3\, n \log n)$ time, via Theorem 5.6, and then assembles these $\mathbb{Z}_2$-minimal cycles into $\mathbb{Z}_2$-minimal even subgraphs using dynamic programming.

For each homology class $h \in (\mathbb{Z}_2)^\beta$ and each integer $1 \le k \le g + b - 1$, let $\boldsymbol{C(h, k)}$ denote the minimum total weight of any set of at most $k$ cycles in $G$ whose homology classes sum to $h$. Lemma 5.7 implies that the minimum weight of any even subgraph in homology class $h$ is exactly $C(h, g + b - 1)$. This function obeys the following straightforward recurrence:

$$C(h, k) = \min \left\{ C(h_1, k-1) + C(h_2, 1) \;\middle|\; h_1 \oplus h_2 = h \right\}.$$

This recurrence has two base cases: $C(0, k) = 0$ for any integer $k$, and for any homology class $h$, the value $C(h, 1)$ is just the length of $\gamma_h$. A standard dynamic programming algorithm computes $C(h, g + b - 1)$ for all $2^\beta$ homology classes $h$ in $O(4^\beta \beta)$ time. We can then assemble the actual minimum-weight even subgraphs in each homology class in $O(\beta n)$ time. The total time for this phase of the algorithm is $O(4^\beta \beta + 2^\beta \beta n)$, which is dominated by the time to compute all the $\mathbb{Z}_2$-minimal cycles. □

COROLLARY 5.9. *Let $G$ be an undirected graph with positively weighted edges, embedded on a surface with genus $g$ (possibly non-orientable and possibly with boundary), and let $s$ and $t$ be vertices of $G$. We can compute the minimum-weight $(s,t)$-cut in $G$ in $O(256^g g^3 n \log n)$ time.*

**6. NP-Hardness.** In this section, we show that finding the minimum-cost even subgraph in a given homology class is NP-hard, even when the underlying surface has no boundary. Our proof closely follows a reduction of McCormick *et al.* [95] from MIN2SAT to a special case of MAXCUT.

THEOREM 6.1. *Computing the minimum-weight even subgraph in a given homology class on a surface without boundary is equivalent to computing a minimum-weight cut in an embedded edge-weighted graph $G$, where negative-weight edges of $G$ are dual to an even subgraph in $G^*$.*

**Proof:** Fix a graph $G$ embedded on a surface $\Sigma$ without boundary, together with a positive weight function $c : E \to \mathbb{R}^+$. For any even subgraph $H$ of $G$, let $c(H) = \sum_{e \in H} c(e)$, and let $\textsc{MinHom}(H, c)$ denote the even subgraph of minimum weight in the homology class of $H$ with respect to weight function $c$.

Consider the *residual weight* function $c_H : E \to \mathbb{R}$ defined by setting $c_H(e) = c(e)$ for each edge $e \notin H$, and $c_H(e) = -c(e)$ for each edge $e \in H$. For any subgraph $H'$ of $G$, we have $c(H') = c_H(H \oplus H') + c(H)$, which immediately implies that $\textsc{MinHom}(H, c) = H \oplus \textsc{MinHom}(\emptyset, c_H)$.

Every boundary subgraph of $G$ is dual to a cut in the dual graph $G^*$. Thus, we have reduced our problem to computing the minimum cut in $G^*$ with respect to the weight function $c_H$, which is NP-hard as stated in Lemma 6.2. Since the empty set is a valid cut with zero cost, the cost of the minimum cut is never positive. In particular, $H$ is the minimum-cost even subgraph in its homology class if and only if the cut in $G^*$ with minimum residual cost has zero cost.

In fact, our reduction is reversible. Suppose we want to find the minimum cut in an embedded graph $G = (V, E)$ with respect to the cost function $c : E \to \mathbb{R}$, where every face of $G$ is incident to an even number of edges with negative cost. Let $H = \{e \in E \mid c(e) < 0\}$ be the subgraph of negative-cost edges, and let $X$ denote the (possibly empty) set of edges in the minimum cut of $G$. Consider the *absolute cost* function $|c| : E^* \to \mathbb{R}$ defined as $|c|(e^*) = |c(e)|$. Then $(H \oplus X)^*$ is the even subgraph of $G^*$ of minimum absolute cost that is homologous to $H^*$. □

We now prove that this special case of the minimum cut problem is NP-hard, by reduction from MINCUT in graphs with negative edges. This problem includes MAXCUT as a special case (when every edge has negative cost), but many other special cases are also NP-hard [95].

LEMMA 6.2 (McCormick *et al.* [95]). MINCUT *is strongly NP-hard.*

The output of our reduction is a simple triangulation; the reduction can be simplified if graphs with loops and parallel edges are allowed.

LEMMA 6.3. *Computing a minimum-weight cut in an embedded edge-weighted graph $G$ whose negative-weight edges are dual to an even subgraph in $G^*$ is strongly NP-hard.*

**Proof:** Let $n$ be the number of vertices of $G$ and $c : E \to \mathbb{R}$ be the edge weight function. We begin by computing a cellular embedding of $G$ on some orientable surface, by imposing an arbitrary cyclic order on the edges incident to each vertex. (We can compute the maximum-genus orientable cellular embedding in polynomial time [59].) Alternatively, we can add zero-length edges to make the graph complete and then use classical results of Ringel, Youngs, and others [108, 109] to compute a minimum-genus orientable embedding of $K_n$ in polynomial time. Once we have an embedding, we add vertices and zero-cost edges to obtain a triangulation.

Let $C$ be the sum of the absolute values of the edge costs: $C := \sum_e |c(e)|$. A ***cocycle*** of embedded graph $G$ is a subset of edges forming a cycle in the dual $G^*$. We locally modify both the surface and the embedding to transform each negative-weight edge into a cocycle, as follows. Therefore, in the end, the set of negative-weight edges are dual to an even subgraph.

We transform the edges one at a time; after each iteration, the embedding is a simple triangulation. (Our reduction can be simplified if a simple graph is not required.) For each edge $uv$ with $c(uv) < 0$, remove $uv$ to create a quadrilateral face. Triangulate this face as shown in Figure 6.1; we call the new faces $uu_1u_2$ and $vv_1v_2$ *endpoint triangles*. Assign cost $C$ to the edges of the endpoint triangles and cost zero to the other new edges. Glue a new handle to the endpoint triangles, and triangulate the handle with a cycle of six edges, each with cost $c(uv)/6$. These six edges form a cocycle of cost $c(uv)$, which we call an *edge cocycle*, in the new embedding. Each iteration adds 5 vertices and 21 edges to the graph and increases the genus of the underlying surface by 1.

Let $G'$ denote the transformed graph and $c' : E(G') \to \mathbb{R}$ its associated cost function. The minimum cut in $G'$

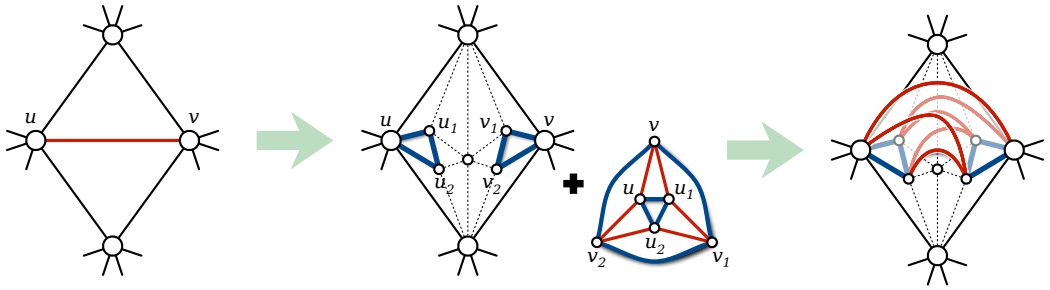


**Fig. 6.1.** Adding a handle to transform a negative edge into a negative cocycle. Thick (blue) edges have cost $C$; dashed edges have cost zero; and red edges have cost $c(uv)/6$.

cannot contain any edge of an endpoint triangle. Thus, for each edge cocycle, either all six edges cross the cut, or none of them cross the cut. It follows that the minimum cut in $G'$ corresponds to a cut with equal cost in the original graph $G$. Conversely, any cut in $G$ can be transformed into a cut in $G'$ of equal cost. Thus, computing the minimum cut in $G'$ is equivalent to computing the minimum cut in $G$, which is NP-hard by Lemma 6.2. □

THEOREM 6.4. *Given an even subgraph $H$ of an edge-weighted graph $G$ embedded on a surface without boundary, computing the minimum-weight even subgraph homologous to $H$ is strongly NP-hard.*

Our reduction can be modified further to impose other desirable properties on the output instances, for example, that the graph is unweighted, every vertex has degree 3, or the input subgraph $H$ is a simple cycle.

Finally, we emphasize that the NP-hardness of this problem relies crucially on the fact that we are using homology with coefficients taken from the finite field $\mathbb{Z}_2$. The corresponding problem for homology with real or integer coefficients is a minimum-cost circulation problem, and thus can be solved in polynomial time. Chambers, Erickson, and Nayyeri [28] show that this circulation problem can be solved in near-linear time for graphs of constant genus and polynomially bounded integer edge capacities using very different techniques.

**7. Global Minimum Cut.** Finally, we describe our algorithm to compute *global* minimum cuts in surface-embedded graphs, where no source and target vertices are specified in advance. Unlike previous sections, we begin our exposition assuming that the underlying surface of the input graph does *not* have boundary, because filling in any boundaries with disks does not change the minimum cut. We also assume without loss of generality that no edge of the input graph has the same face on both sides; we can enforce this assumption if necessary by adding infinitesimal-weight edges.

As in previous sections, it is convenient to work in the dual graph. We cannot apply Lemma 2.2 directly, but the following lemma similarly characterizes global minimum cuts in surface graphs in terms of homology in the dual graph. Suppose we have a graph embedded in a surface with a single boundary component. A ***separating subgraph*** is any non-empty boundary subgraph, or equivalently, the boundary of the union of a *non-empty* set of faces.

LEMMA 7.1. *Let $G$ be an undirected edge-weighted graph embedded on a surface $\Sigma$ without boundary, and let $s$ be an arbitrary vertex of $G$. A subgraph $X$ is a global minimum cut in $G$ if and only if $X^*$ is a minimum-weight separating subgraph of $G^*$ in $\Sigma \setminus s^*$.*

**Proof:** Let $X$ be an arbitrary cut in $G$. The cut partitions the vertices of $G$ into two disjoint subsets $S$ and $T$ with $s \in S$. Therefore, the dual subgraph $X^*$ partitions the faces of $G^*$ into two disjoint subsets $S^*$ and $T^*$ with $s^* \in S^*$. Further, $X^*$ is the boundary of the union of faces in $T^*$, implying that $X^*$ is a boundary subgraph of $\Sigma$ and therefore separating.

Conversely, let $X^*$ be any separating subgraph of $G^*$. Subgraph $X^*$ is the boundary of a nonempty subset of the faces $T^*$ of $G^*$. Let $t^*$ be a face in $T^*$. Any path from $s$ to $t$ in the primal graph $G$ must traverse at least one edge of $X$. We conclude that $X$ is a cut (in particular, an $(s,t)$-cut). □

In light of this lemma, the remainder of this section describes an algorithm to find a minimum-weight separating subgraph in a given surface-embedded graph $G$ with positive edge weights. Graph $G$ is embedded in a surface $\Sigma$ with *exactly one boundary component* $s^*$.

Let $X$ be a minimum-weight separating subgraph. Surface $\Sigma \setminus X$ has exactly one component not incident to $s^*$; otherwise, the boundary of any one of these components is a smaller separating subgraph. Abusing terminology

slightly, call the separating subgraph $X$ ***contractible*** if this component of $\Sigma \setminus X$ is a disk, and ***non-contractible*** otherwise. If $X$ is contractible, then $X$ is actually a shortest (weakly) simple contractible cycle of $G$ in the surface $\Sigma$; otherwise, $X$ can be decomposed into one or more simple cycles, each of which is non-contractible. See Figure 7.1.

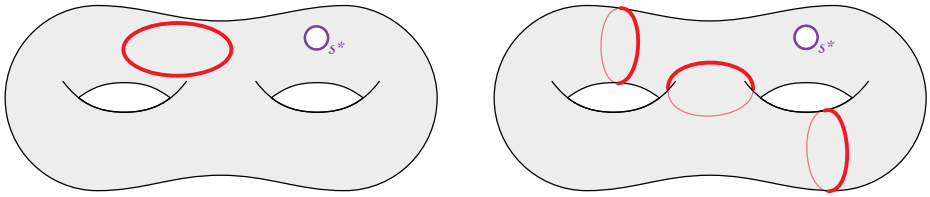

**Fig. 7.1.** Two types of minimum-weight separating subgraphs: a simple contractible cycle and otherwise.

Thus, in principle, we can find a minimum-weight separating subgraph by first computing a shortest contractible cycle, then computing a minimum-weight separating collection of non-contractible cycles, and finally returning the lighter of these two subgraphs. Unfortunately, we do not know how to solve either of these subproblems in our stated time bounds, so our algorithm takes a more subtle approach.

In Section 7.1, we describe an algorithm that computes a minimum-weight separating subgraph *if* any minimum-weight separating subgraph is contractible. Similarly, in Section 7.2, we describe an algorithm that computes a minimum-weight separating subgraph *if* any minimum-weight separating subgraph is non-contractible. In both cases, if no minimum-weight separating subgraph satisfies the corresponding condition, the algorithm still returns a boundary subgraph, but this subgraph could be empty or have large weight. By running both subroutines and returning the best result, we are guaranteed to find a minimum-weight separating subgraph in $G$, no matter which category it falls into.

For ease of exposition, we explicitly consider only the case where the underlying surface is orientable; we briefly discuss the non-orientable setting at the end of this section.

**7.1. Contractible.** First we consider the case where some minimum-weight separating subgraph $X$ is contractible. This case is nearly handled using a result of Cabello *et al.* [22, Theorem 5.4] wherein they describe an algorithm for finding a shortest contractible cycle enclosing a non-empty set of faces in $O(T_{min-cut}(n) + n \log n)$ time where $T_{min-cut}(n)$ is the time needed to find a minimum cut in a planar graph of size $n$. Using the minimum cut algorithm of Łącki and Sankowski [88], we may assume $T_{min-cut}(n) = O(n \log\log n)$. However, we still need to open the black box to reduce the $n \log n$ term in the running time. In the interest of completeness and to avoid having to repeatedly reprove lemmas and theorems from Cabello *et al.* [22], we describe how to handle the contractible case directly using tools developed earlier in the current paper.

We begin by borrowing a result of Cabello [19, Lemma 4.1]. Recall that an arc or cycle is *tight* if it has minimum-weight among all arcs or cycles in its homotopy class.

LEMMA 7.2 (Cabello [19]). *Let $\alpha$ be a tight arc or tight cycle on $G$. There exists a shortest simple contractible cycle that does not cross $\alpha$.*

Cabello [19] uses this observation to compute a shortest simple contractible cycle in a surface-embedded graph; unfortunately, his algorithm runs in $O(n^2 \log n)$ time.

We use the slicing operation ($\setminus\setminus$) along tight cycles and arcs in $G$. The following lemma implies it is safe for our algorithm to find minimum-weight separating subgraphs in sliced copies of $\Sigma$.

LEMMA 7.3. *Let $\alpha$ be an arbitrary simple cycle or arc in $G$. Let $\Sigma' = \Sigma \setminus\setminus \alpha$ and let $G' = G \setminus\setminus \alpha$. Finally, let $H'$ denote a boundary subgraph in $G'$ and let $H$ denote the set of edges that appear an odd number of times in the projection of $H'$. Subgraph $H$ bounds the natural projection to $G$ of the faces bound by $H'$ in $G'$.*

**Proof:** Let $F'$ be the subset of faces bound by $H'$ in $G'$. Let $F$ be natural mapping of $F'$ into $G$. We will argue that $H$ is the boundary of $F$, proving the lemma.

Consider any edge $e$ of $G$. Suppose $e$ is not in $\alpha$. In this case, $G'$ contains one copy $e'$ of $e$, and $e' \in H'$ if and only if $e \in H$. Edge $e'$ being incident to exactly one face of $F'$ is therefore equivalent to $e$ being incident to exactly one face of $F$.

Now suppose $e$ is in $\alpha$. Graph $G'$ contains two copies of $e$ denoted $e_1$ and $e_2$, each incident to one face denoted $f_1$ and $f_2$, respectively. If neither or both of $f_1$ and $f_2$ are in $F'$, then $H'$ includes neither or both of $e_1$ and $e_2$. Therefore, $H'$ contains the two copies of $e$ an even number of times total, implying $e \notin H$. If one, but not

both, of $f_1$ and $f_2$ are in $F'$, then $H'$ includes exactly one of $e_1$ or $e_2$. In turn, $H'$ contains the two copies of $e$ an odd number of times total, meaning $e \in H$.

In all cases, an edge $e$ is in $H$ if and only if exactly one incident face to $e$ is in $F$. □

We now present our algorithm for finding a minimum-weight separating subgraph if that subgraph happens to be a contractible cycle.

LEMMA 7.4. *There exists an $O(gn \log\log n)$-time algorithm that computes a minimum-weight separating subgraph if any such subgraph is a simple contractible cycle. If not, the algorithm either returns some separating subgraph (that may not be minimum weight) or nothing.*

**Proof:** The algorithm computes a greedy system of arcs $A$ in $O(n)$ time as described in Section 3.1. Observe both endpoints of each arc lie on $s^*$. Let $G'$ denote the planar graph $G \setminus\setminus A$; this graph has $O(gn)$ vertices.

We now run the algorithm of Łącki and Sankowski [88] to compute the shortest simple cycle $\gamma'$ of $G'$ in $O(gn \log\log n)$ time. Let $H'$ be the subgraph of $G'$ containing the edges of $\gamma'$. Subgraph $H'$ separates a non-empty subset of faces $F'$ from the boundary of $G'$. By multiple instantiations of Lemma 7.3, subgraph $H'$ projects to a boundary subgraph $H$. Because $s^*$ is a boundary component, we see $H$ separates the natural projection of $F'$ from $s^*$.

Now, suppose some minimum weight separating subgraph of $G$ is a simple contractible cycle. Lemma 7.2 implies that for any arc $\alpha \in A$, there exists a shortest simple contractible cycle $\sigma$ that does not cross $\alpha$. The cycle $\sigma$ appears as a simple contractible cycle in $G \setminus\setminus \alpha$. Any contractible cycle in $G \setminus\setminus \alpha$ is contractible in $G$, so $\sigma$ is a *shortest* contractible cycle in $G \setminus\setminus \alpha$ as well. Therefore, by repeated applications of Lemma 7.2, we may assume $\sigma$ does not cross any arc of $A$, and it appears as a simple cycle in $G'$ that separates at least one face of $G'$ from the boundary. We emphasize that our algorithm does not necessarily compute $\sigma$. However, $\sigma$ cannot be shorter than $H$, and our algorithm returns a minimum-weight separating subgraph. □

**7.2. Non-contractible.** Now suppose some minimum-weight separating subgraph $X$ is non-contractible. At a high level, our algorithm for this case computes a set $F$ of faces, such that some minimum-weight separating subgraph of $G$ separates $s^*$ from at least one face in $F$. (Equivalently, $F^*$ is a set of vertices of $G^*$, such that the global minimum cut in $G^*$ is an $(s,t)$-cut for some $t \in F^*$.) Then for each face in $F$, we compute a minimum-weight subgraph separating $s^*$ from that face using one of our earlier algorithms.

Throughout this section, we assume without loss of generality that every edge of $G$ lies on the boundary of two distinct faces of $G$. We can enforce this assumption if necessary by adding $O(n)$ infinite-weight edges to $G$.

The following lemma can be seen as the main technical take-away from this section. After its appearance in a preliminary version of our work [49], it was generalized by Borradaile *et al.* [11] for their construction of a minimum $(s,t)$-cut oracle for surface embedded graphs.

LEMMA 7.5. *Let $X$ be a minimum-weight separating subgraph. Let $\gamma$ be a closed walk in $G$ that lies in the closure of the component of $\Sigma \setminus X$ not incident to $s^*$, and let $H$ be a shortest even subgraph homologous to $\gamma$. There is a minimum weight separating subgraph $X'$ (possibly $X$) such that $H$ lies in the closure of the component of $\Sigma \setminus X'$ not incident to $s^*$.*

**Proof:** If $\gamma$ is null-homologous, then $H$ is empty and the lemma is trivial, so assume otherwise. If $H$ lies in the closure of the component of $\Sigma \setminus X$ not incident to $s^*$, then we are done, so assume otherwise. See Figure 7.2.

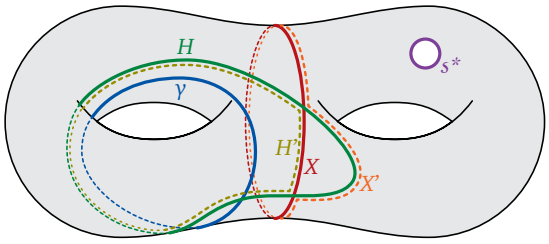


**Fig. 7.2.** The setting of Lemma 7.5. A $\mathbb{Z}_2$-minimal even subgraph $H$ is separated from face $f$ by a minimum weight separating subgraph $X'$.

Recall that $X$ bounds the union of one non-empty component of faces not incident to $s^*$. Call the faces in this component *far* and the rest *near*. Similarly, the even subgraph $H \oplus \gamma$ is null-homologous and therefore bounds a subset of faces of $G$. Call the faces in this subset *white* and the rest *black*. (If $H = \gamma$, then every face of $G$ is black.)

Let $X'$ be the boundary of the union of the far faces and white faces in $G$. There is at least one far face, so subgraph $X'$ is separating. For each edge $e$ of $H$, either $e$ is incident to a white face or it is also an edge of $\gamma$. Either way, $e$ lies in the closure of the component of $\Sigma \setminus X'$ not incident to $s^*$.

It remains to argue that $X'$ is a minimum-weight separating subgraph of $G$.

For any subgraph $A$ of $G$, let $w(A)$ denote the sum of the weights of the edges of $A$. Because both $X'$ and $X$ are null-homologous, the even subgraph $H' = H \oplus X' \oplus X$ is homologous to $H$, and therefore to $\gamma$. We immediately have $w(H') \geq w(H)$, because $H$ is $\mathbb{Z}_2$-minimal.

We now prove that $w(X') + w(H') \leq w(H) + w(X)$ by bounding the contribution of each edge $e \in E(G)$ to both sides of the inequality. Both $X'$ and $H'$ are subgraphs of $X \cup H$; moreover, $X' \oplus H' = X \oplus H$. There are three cases to consider.

- If $e \notin X \cup H$, then $e$ contributes 0 to both sides of the inequality.
- If $e \in X \oplus H$, then $e \in X' \oplus H'$. In this case, $e$ contributes $w(e)$ to both sides of the inequality.
- If $e \in X \cap H$, then $e$ contributes exactly $2w(e)$ to the right side of the inequality. Trivially, $e$ contributes at most $2w(e)$ to the left side.

We conclude that $X'$ is also a minimum-weight separating subgraph. □

LEMMA 7.6. *There is a $g^{O(g)} n \log\log n$-time algorithm that computes a minimum-weight separating subgraph of $G$ if any minimum-weight separating subgraph of $G$ is non-contractible. If every minimum-weight separating subgraph of $G$ is contractible, the algorithm returns a separating subgraph that may not have minimum weight.*

**Proof:** In a preprocessing phase, we construct a homology basis from a tree-coforest decomposition in $O(gn)$ time; see Lemma 3.8. Then we enumerate all $2^{2g} - 1$ non-trivial homology classes by considering subsets of cycles in this homology basis. For each non-trivial homology class $h$, we perform the following steps:

- Compute a minimum-weight subgraph $H_h$ in homology class $h$, in $g^{O(g)} n \log\log n$ time, as described by Theorem 4.2.
- Fix an arbitrary edge $e$ of $H_h$. By assumption, $e$ lies on the boundary of two distinct faces $f_L$ and $f_R$. In particular, at least one of these faces is not $s^*$.
- If $f_L \neq s^*$, compute a minimum-weight subgraph $X_h$ of $G$ that separates $s^*$ and $f_L$, in $g^{O(g)} n \log\log n$ time, using the minimum $(s,t)$-cut algorithm of Section 4. Otherwise, $X_h$ is undefined.
- If $f_R \neq s^*$, compute a minimum-weight subgraph $X'_h$ of $G$ that separates $s^*$ and $f_R$, in $g^{O(g)} n \log\log n$ time, again using the minimum $(s,t)$-cut algorithm of Section 4. Otherwise, $X'_h$ is undefined.

Altogether we compute between $2^{2g} - 1$ and $2^{2g+1} - 2$ separating subgraphs of $G$ (some of which may coincide); the output of our algorithm is the smallest of these separating subgraphs. The overall running time of our algorithm is $2^{O(g)} \cdot g^{O(g)} n \log\log n = g^{O(g)} n \log\log n$.

It remains to prove that our algorithm is correct. Let $X$ be any minimum-weight separating subgraph of $G$ such that the component of $\Sigma \setminus X$ not incident to $s^*$ is not a disk. Let $\Sigma'$ be the closure of the component of $\Sigma \setminus X$ that does not contain $s^*$. If $\Sigma'$ has genus, then it contains a cycle $\gamma$ that is not separating in $\Sigma'$ or $\Sigma$. Otherwise, $\Sigma'$ not being a disk implies it has multiple boundary components. We may assume that each of these boundary components is non-separating in $\Sigma$. Otherwise, we could remove exactly one non-separating component to find a separating subgraph that weighs no more than $X$. Let $\gamma$ be any one of these boundary components. In both cases, $\gamma$ lies in $\Sigma'$ and is non-separating in $\Sigma$. Let $h$ be the homology class of $\gamma$ in $\Sigma$, and let $H_h$ be any minimum-weight even subgraph of $G$ that is homologous with $\gamma$ in $\Sigma$. By Lemma 7.5, there exists a minimum-weight separating subgraph $X'$ such that $H_h$ lies in the closure of the component of $\Sigma \setminus X'$ not incident to $s^*$. Every edge of $H_h$ is on the boundary of at least one face $f'$ in the closure of the component of $\Sigma \setminus X'$ containing $H_h$. Further, $X'$ must be a minimum-weight even subgraph separating $s^*$ and $f'$. We conclude that when our algorithm considers homology class $h$, either $X_h$ or $X'_h$ is a minimum-weight separating subgraph of $G$. □

Modifying the previous algorithm to use results of Section 5, instead of the corresponding results in Section 4, immediately gives us the following:

LEMMA 7.7. *There is a $2^{O(g)} n \log n$-time algorithm that computes a minimum-weight separating subgraph of $G$ if any minimum-weight separating subgraph of $G$ is non-contractible. If every minimum-weight separating subgraph of $G$ is contractible, the algorithm returns a separating subgraph that may not have minimum weight.*

**7.3. Summing up.** Finally, to compute the minimum-weight separating subgraph in $G$, we run both algorithms described in Lemmas 7.4 and 7.6. If either algorithm returns nothing, the other algorithm returns a minimum-weight separating subgraph of $G$. Otherwise, both algorithms return non-empty separating subgraphs of $G$, and

the smaller of those two subgraphs is a minimum-weight separating subgraph of $G$. We conclude:

COROLLARY 7.8. *Let $G$ be an undirected graph with positively weighted edges, embedded on an **orientable** surface with genus $g$ (possibly with boundary). We can compute a global minimum cut in $G$ in either $g^{O(g)}n \log\log n$ time or $2^{O(g)}n \log n$ time.*

Most of the results in this section extend directly to non-orientable surfaces. The only exception is our algorithm for the non-contractible case (Lemma 7.6), which computes a minimum-weight subgraph in *every* homology class. When the underlying surface is non-orientable, we cannot use our crossing-sequence algorithm in Section 4 to solve this subproblem, for the reasons spelled out in Section 4.3. However, we can still use our homology-cover algorithm from Section 5.

THEOREM 7.9. *Let $G$ be an undirected graph with positively weighted edges, embedded on a **non-orientable** surface with genus $g$ and exactly one boundary component. We can compute a minimum-weight separating subgraph in $G$ in $2^{O(g)}n \log n$ time.*

COROLLARY 7.10. *Let $G$ be an undirected graph with positively weighted edges, embedded on a **non-orientable** surface with genus $g$ (possibly with boundary). We can compute a global minimum cut in $G$ in $2^{O(g)}n \log n$ time.*

*Acknowledgments.* The authors would like to thank Chandra Chekuri and Aparna Sundar for helpful discussions on some of the preliminary work included here. We would also like to thank the anonymous reviewers, both for our earlier extended abstracts [27, 49, 50] and for this paper, for many helpful comments and suggestions.

## REFERENCES

[1] A. ABBOUD, V. V. WILLIAMS, AND J. R. WANG, *Approximation and fixed parameter subquadratic algorithms for radius and diameter in sparse graphs*, in Proc. 27th Ann. ACM-SIAM Symp. Discrete Algorithms, 2016, pp. 377–391, https://doi.org/10.1137/1.9781611974331.ch28.

[2] R. K. AHUJA, T. L. MAGNANTI, AND J. ORLIN, *Network Flows: Theory, Algorithms, and Applications*, Prentice Hall, 1993.

[3] H. A. AKITAYA, G. ALOUPIS, J. ERICKSON, AND C. D. TÓTH, *Recognizing weakly simple polygons*, Discrete Comput. Geom., 58 (2017), pp. 785–821.

[4] L. ALEKSANDROV AND H. DJIDJEV, *Linear algorithms for partitioning embedded graphs of bounded genus*, SIAM J. Discrete Math., 9 (1996), pp. 129–150, https://doi.org/10.1137/S0895480194272183.

[5] J. W. ALEXANDER, *Combinatorial analysis situs*, Trans. Amer. Math. Soc., 28 (1926), pp. 301–326.

[6] J. W. ALEXANDER AND G. B. BRIGGS, *On types of knotted curves*, Ann. Math., 28 (1926–1927), pp. 562–586.

[7] N. BIGGS, *Spanning trees of dual graphs*, J. Comb. Theory, 11 (1971), pp. 127–131, https://doi.org/10.1016/0095-8956(71)90022-0.

[8] G. BORRADAILE, *Exploiting Planarity for Network Flow and Connectivity Problems*, PhD thesis, Brown University, May 2008, http://www.cs.brown.edu/research/pubs/theses/phd/2008/glencora.pdf.

[9] G. BORRADAILE, E. W. CHAMBERS, K. FOX, AND A. NAYYERI, *Minimum cycle and homology bases of surface-embedded graphs*, J. Comput. Geom., 8 (2017), pp. 58–79, https://doi.org/10.20382/jocg.v8i2a4.

[10] G. BORRADAILE, E. D. DEMAINE, AND S. TAZARI, *Polynomial-time approximation schemes for subset-connectivity problems in bounded-genus graphs*, Algorithmica, 68 (2014), pp. 287–311, https://doi.org/10.1007/s00453-012-9662-2.

[11] G. BORRADAILE, D. EPPSTEIN, A. NAYYERI, AND C. WULFF-NILSEN, *All-pairs minimum cuts in near-linear time for surface-embedded graphs*, in Proc. 32nd Int. Symp. Comput. Geom., vol. 51 of Leibniz Int. Proc. Informatics, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2016, pp. 22:1–22:16, https://doi.org/10.4230/LIPIcs.SoCG.2016.22.

[12] G. BORRADAILE, C. KENYON-MATHIEU, AND P. N. KLEIN, *A polynomial-time approximation scheme for Steiner tree in planar graphs*, in Proc. 18th Ann. ACM-SIAM Symp. Discrete Algorithms, 2007, pp. 1285–1294.

[13] G. BORRADAILE, C. KENYON-MATHIEU, AND P. N. KLEIN, *Steiner tree in planar graphs: An $O(n \log n)$ approximation scheme with singly-exponential dependence on epsilon*, in Proc. 10th Workshop on Algorithms and Data Structures, 2007, pp. 275–286.

[14] G. BORRADAILE AND P. KLEIN, *An $O(n \log n)$-time algorithm for maximum $st$-flow in a directed planar graph*, in Proc. 17th Ann. ACM-SIAM Symp. Discrete Algorithms, 2006, pp. 524–533.

[15] G. BORRADAILE AND P. KLEIN, *An $O(n \log n)$ algorithm for maximum $st$-flow in a directed planar graph*, J. ACM, 56 (2009), pp. 9:1–30.

[16] G. BORRADAILE, P. N. KLEIN, S. MOZES, Y. NUSSBAUM, AND C. WULFF-NILSEN, *Multiple-source multiple-sink maximum flow in directed planar graphs in near-linear time*, SIAM J. Comput., 46 (2017), pp. 1280–1303, https://doi.org/10.1137/15M1042929.

[17] G. BORRADAILE, H. LE, AND C. WULFF-NILSEN, *Minor-free graphs have light spanners*, in Proc. 58th IEEE Symp. Found. Comput. Sci., 2017, pp. 767–778, https://doi.org/10.1109/FOCS.2017.76.

[18] S. CABELLO, *Many distances in planar graphs*, in Proc. 17th Ann. ACM-SIAM Symp. Discrete Algorithms, 2006, pp. 1213–1220.

[19] S. CABELLO, *Finding shortest contractible and shortest separating cycles in embedded graphs*, ACM Trans. Algorithms, 6 (2010), pp. 24:1–24:18, https://doi.org/10.1145/1721837.1721840.

[20] S. CABELLO, E. W. CHAMBERS, AND J. ERICKSON, *Multiple-source shortest paths in embedded graphs*, SIAM J. Comput., 42 (2013), pp. 1542–1571.

[21] S. CABELLO, É. COLIN DE VERDIÈRE, AND F. LAZARUS, *Finding shortest non-trivial cycles in directed graphs on surfaces*, in Proc. 26th Ann. Symp. Comput. Geom., 2010, pp. 156–165, https://doi.org/10.1145/1810959.1810988.

[22] S. CABELLO, M. DEVOS, J. ERICKSON, AND B. MOHAR, *Finding one tight cycle*, ACM Trans. Algorithms, 6 (2010), pp. 61:1–61:13, https://doi.org/10.1145/1824777.1824781.

[23] S. Cabello and B. Mohar, *Finding shortest non-separating and non-contractible cycles for topologically embedded graphs*, Discrete Comput. Geom., 37 (2007), pp. 213–235, https://doi.org/10.1007/s00454-006-1292-5.

[24] P. Chalermsook, J. Fakcharoenphol, and D. Nanongkai, *A deterministic near-linear time algorithm for finding minimum cuts in planar graphs*, in Proc. 15th Ann. ACM-SIAM Symp. Discrete Algorithms, 2004, pp. 828–829.

[25] E. W. Chambers, É. Colin de Verdière, J. Erickson, F. Lazarus, and K. Whittlesey, *Splitting (complicated) surfaces is hard*, Comput. Geom. Theory Appl., 41 (2008), pp. 94–110, https://doi.org/10.1016/j.comgeo.2007.10.010.

[26] E. W. Chambers and D. Eppstein, *Flows in one-crossing-minor-free graphs*, J. Graph Algorithms Appl., 17 (2013), pp. 201–220, https://doi.org/10.7155/jgaa.00291.

[27] E. W. Chambers, J. Erickson, and A. Nayyeri, *Minimum cuts and shortest homologous cycles*, in Proc. 25th Ann. Symp. Comput. Geom., 2009, pp. 377–385, https://doi.org/10.1145/1542362.1542426.

[28] E. W. Chambers, J. Erickson, and A. Nayyeri, *Homology flows, cohomology cuts*, SIAM J. Comput., 41 (2012), pp. 1605–1634.

[29] H.-C. Chang, J. Erickson, and C. Xu, *Detecting weakly simple polygons*, in Proc. 26th ACM-SIAM Symp. Discrete Algorithms, 2015, pp. 1655–1670, https://doi.org/10.1137/1.9781611973730.110.

[30] C. Chen and D. Freedman, *Quantifying homology classes II: Localization and stability*. Preprint, 2007.

[31] C. Chen and D. Freedman, *Quantifying homology classes*, in Proc. 25th Ann. Symp. Theoretical Aspects Comput. Sci., no. 08001 in Dagstuhl Seminar Proceedings, 2008, pp. 169–180, https://doi.org/10.4230/LIPIcs.STACS.2008.1343, http://drops.dagstuhl.de/opus/volltexte/2008/1343/.

[32] C. Chen and D. Freedman, *Hardness results for homology localization*, in Proc. 21st Ann. ACM-SIAM Symp. Discrete Algorithms, 2010, pp. 1594–1604, https://doi.org/10.1137/1.9781611973075.129.

[33] É. Colin de Verdière, *Shortest cut graph of a surface with prescribed vertex set*, in Proc. 18th Ann. Europ. Symp. Algorithms, vol. 6347 of Lecture Notes Comput. Sci., 2010, pp. 100–111.

[34] É. Colin de Verdière and J. Erickson, *Tightening non-simple paths and cycles on surfaces*, in Proc. 17th Ann. ACM-SIAM Symp. Discrete Algorithms, 2006, pp. 192–201.

[35] É. Colin de Verdière and J. Erickson, *Tightening non-simple paths and cycles on surfaces*, SIAM J. Comput., 39 (2010), pp. 3784–3813.

[36] É. Colin de Verdière and F. Lazarus, *Optimal pants decompositions and shortest homotopic cycles on an orientable surface*, J. ACM, 54 (2007).

[37] E. D. Demaine, M. Hajiaghayi, and B. Mohar, *Approximation algorithms via contraction decomposition*, in Proc. 18th Ann. ACM-SIAM Symp. Discrete Algorithms, 2007, pp. 278–287.

[38] T. K. Dey, A. N. Hirani, and B. Krishnamoorthy, *Optimal homologous cycles, total unimodularity, and linear programming*, SIAM J. Comput., 40 (2011), pp. 1026–1044.

[39] T. K. Dey, K. Li, and J. Sun, *On computing handle and tunnel loops*, in IEEE Proc. Int. Conf. Cyberworlds, 2007, pp. 357–366.

[40] T. K. Dey, K. Li, J. Sun, and D. Cohen-Steiner, *Computing geometry-aware handle and tunnel loops in 3D models*, ACM Trans. Graphics, 27 (2008), pp. 1–9. Proc. SIGGRAPH 2008.

[41] S. I. Diatch and D. A. Spielman, *Faster lossy generalized flow via interior point algorithms*, in Proc. 40th Ann. ACM Symp. Theory Comput., 2008, pp. 451–460.

[42] J. A. Ellis-Monaghan and I. Moffatt, *Graphs on Surfaces - Dualities, Polynomials, and Knots*, Springer Briefs in Mathematics, Springer, 2013, https://doi.org/10.1007/978-1-4614-6971-1, https://doi.org/10.1007/978-1-4614-6971-1.

[43] D. Eppstein, *Subgraph isomorphism in planar graphs and related problems*, J. Graph Algorithms Appl., 3 (1999), pp. 1–27.

[44] D. Eppstein, *Diameter and treewidth in minor-closed graph families*, Algorithmica, 27 (2000), pp. 275–291.

[45] D. Eppstein, *Dynamic generators of topologically embedded graphs*, in Proc. 14th Ann. ACM-SIAM Symp. Discrete Algorithms, 2003, pp. 599–608.

[46] J. Erickson, *Maximum flows and parametric shortest paths in planar graphs*, in Proc. 21st Ann. ACM-SIAM Symp. Discrete Algorithms, 2010, pp. 794–804.

[47] J. Erickson, *Shortest non-trivial cycles in directed surface graphs*, in Proc. 27th Ann. Symp. Comput. Geom., 2011, pp. 236–243.

[48] J. Erickson, K. Fox, and L. Lkhamsuren, *Holiest minimum-cost paths and flows in surface graphs*, in Proc. 50th Ann. ACM Symp. Theory Comput., 2018, pp. 1319–1332, https://doi.org/10.1145/3188745.3188904.

[49] J. Erickson, K. Fox, and A. Nayyeri, *Global minimum cuts in surface embedded graphs*, in Proc. 23rd Ann. ACM-SIAM Symp. Discrete Algorithms, 2012, pp. 1309–1318.

[50] J. Erickson and A. Nayyeri, *Minimum cuts and shortest non-separating cycles via homology covers*, in Proc. 22nd Ann. ACM-SIAM Symp. Discrete Algorithms, 2011, pp. 1166–1176, https://doi.org/10.1137/1.9781611973082.88.

[51] J. Erickson and K. Whittlesey, *Greedy optimal homotopy and homology generators*, in Proc. 16th Ann. ACM-SIAM Symp. Discrete Algorithms, 2005, pp. 1038–1046.

[52] J. Fakcharoenphol, B. Laekhanukit, and P. Sukprasert, *Finding all useless arcs in directed planar graphs*. Preprint, May 2018.

[53] J. Fakcharoenphol and S. Rao, *Planar graphs, negative weight edges, shortest paths, and near linear time*, J. Comput. Syst. Sci., 72 (2006), pp. 868–889.

[54] F. V. Fomin, D. Lokshtanov, S. Saurabh, M. Pilipczuk, and M. Wrochna, *Fully polynomial-time parameterized computations for graphs and matrices of low treewidth*, ACM Trans. Algorithms, 14 (2018), pp. 34:1–34:45, https://doi.org/10.1145/3186898.

[55] L. R. Ford and D. R. Fulkerson, *Maximal flow through a network*, Canad. J. Math., 8 (1956). First published as Research Memorandum RM-1400, The RAND Corporation, Santa Monica, California, November 19, 1954.

[56] K. Fox, *Shortest non-trivial cycles in directed and undirected surface graphs*, in Proc. 24th Ann. ACM-SIAM Symp. Discrete Algorithms, 2013, pp. 352–364, https://doi.org/10.1137/1.9781611973105.26.

[57] A. Frank, *On the edge-connectivity algorithm of Nagamochi and Ibaraki*, EGRES Quick-Proof QP-2009-01, Egerváry Research Group, Eötvös University,, Budapest, 2009, http://bolyai.cs.elte.hu/egres/www/qp-09-01.html. Written at Laboratoire Artemis, IMAG, Université J. Fourier, Grenoble, March 1994.

[58] G. N. Frederickson, *Fast algorithms for shortest paths in planar graphs with applications*, SIAM J. Comput., 16 (1987), pp. 1004–1004.

[59] M. L. Furst, J. L. Gross, and L. A. McGeoch, *Finding a maximum-genus graph imbedding*, J. Assoc. Comput. Mach., 35 (1988), pp. 523–534.

[60] A. V. Goldberg and S. Rao, *Beyond the flow decomposition barrier*, J. ACM, 45 (1998), pp. 783–797.

[61] A. V. Goldberg and R. E. Tarjan, *A new approach to the maximum-flow problem*, J. Assoc. Comput. Mach., 35 (1988), pp. 921–940, https://doi.org/http://doi.acm.org/10.1145/48014.61051.

[62] M. Grigni and P. Sissokho, *Light spanners and approximate TSP in weighted graphs with forbidden minors*, in Proc. 13th Ann. ACM-SIAM Symp. Discrete Algorithms, 2002, pp. 852–857.

[63] J. A. Grochow and J. Tucker-Foltz, *Computational topology and the Unique Games Conjecture*, in Proc. 34th Int. Symp. Comput. Geom., no. 99 in Leibniz Int. Proc. Informatics, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2018, pp. 43:1–43:16, https://doi.org/10.4230/LIPIcs.SoCG.2018.43.

[64] M. Grohe, *Isomorphism testing for embeddable graphs through definability*, in Proc. 32nd ACM Symp. Theory Comput., 2000, pp. 63–72.

[65] J. L. Gross and T. W. Tucker, *Topological graph theory*, Dover Publications, 2001.

[66] T. Hagerup, J. Katajainen, N. Nishimura, and P. Ragde, *Characterizing multiterminal flow networks and computing flows in networks of small treewidth*, J. Comput. Syst. Sci., 57 (1998), pp. 366–375.

[67] J. Hao and J. B. Orlin, *A faster algorithm for finding the minimum cut in a directed graph*, J. Algorithms, 17 (1994), pp. 424–446, https://doi.org/10.1006/jagm.1994.1043.

[68] T. E. Harris and F. S. Ross, *Fundamentals of a method for evaluating rail net capacities*, Memorandum RM-1573, The RAND Corporation, Santa Monica, California, October 24, 1955. Cited in [111].

[69] R. Hassin, *Maximum flow in $(s, t)$ planar networks*, Inform. Proc. Lett., 13 (1981), p. 107.

[70] R. Hassin and D. B. Johnson, *An $O(n \log^2 n)$ algorithm for maximum flow in undirected planar networks*, SIAM J. Comput., 14 (1985), pp. 612–624.

[71] A. Hatcher, *Algebraic Topology*, Cambridge Univ. Press, 2002, http://www.math.cornell.edu/~hatcher/AT/ATpage.html.

[72] M. Henzinger, S. Rao, and D. Wang, *Local flow partitioning for faster edge connectivity*, in Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, 2017, pp. 1919–1938, https://doi.org/10.1137/1.9781611974782.125.

[73] M. R. Henzinger, P. Klein, S. Rao, and S. Subramanian, *Faster shortest-path algorithms for planar graphs*, J. Comput. Syst. Sci., 55 (1997), pp. 3–23.

[74] J. M. Hochstein and K. Weihe, *Maximum s-t-flow with k crossings in $O(k^3 n \log n)$ time*, in Proc. 18th Ann. ACM-SIAM Symp. Discrete Algorithms, 2007, pp. 843–847.

[75] J. E. Hopcroft and J. K. Wong, *Linear time algorithm for isomorphism of planar graphs (preliminary report)*, in Proc. 6th Ann. ACM Symp. Theory Comput., 1974, pp. 172–184.

[76] H. Imai and K. Iwano, *Efficient sequential and parallel algorithms for planar minimum cost flow*, in Proc. SIGAL Int. Symp. Algorithms, no. 450 in Lecture Notes Comput. Sci., Springer-Verlag, 1990, pp. 21–30.

[77] A. Itai and Y. Shiloach, *Maximum flow in planar networks*, SIAM J. Comput., 8 (1979), pp. 135–150.

[78] G. F. Italiano, Y. Nussbaum, P. Sankowski, and C. Wulff-Nilsen, *Improved algorithms for min cut and max flow in undirected planar graphs*, in Proc. 43rd Ann. ACM Symp. Theory Comput., 2011, pp. 313–322.

[79] L. Janiga and V. Koubek, *Minimum cut in directed planar networks*, Kybernetika, 28 (1992), pp. 37–49.

[80] H. Kaplan and Y. Nussbaum, *Minimum $s-t$ cut in undirected planar graphs when the source and the sink are close*, in Proc. 28th Int. Symp. Theoretical Aspects Comput. Sci., T. Schwentick and C. Dürr, eds., vol. 9 of Leibniz Int. Proc. Informatics, Dagstuhl, Germany, 2011, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, pp. 117–128, https://doi.org/http://dx.doi.org/10.4230/LIPIcs.STACS.2011.117, http://drops.dagstuhl.de/opus/volltexte/2011/3004.

[81] D. R. Karger, *Minimum cuts in near-linear time.*, J. ACM, 47 (2000), pp. 46–76.

[82] K. Kawarabayashi and M. Thorup, *Deterministic edge connectivity in near-linear time*, J. ACM, 66 (2018), pp. 4:1–4:50, https://doi.org/10.1145/3274663.

[83] K.-i. Kawarabayashi and M. Thorup, *Deterministic global minimum cut of a simple graph in near-linear time*, in Proc. 47th Ann. ACM Symp. Theory Comput., 2015, pp. 665–674, https://doi.org/10.1145/2746539.2746588.

[84] L. Kettner, *Using generic programming for designing a data structure for polyhedral surfaces*, Comput. Geom. Theory Appl., 13 (1999), pp. 65–90, https://doi.org/10.1016/S0925-7721(99)00007-3.

[85] P. Klein, S. Mozes, and O. Weimann, *Shortest paths in directed planar graphs with negative lengths: A linear-space $O(n \log^2 n)$-time algorithm*, ACM Trans. Algorithms, 6 (2010), p. article 30.

[86] P. N. Klein, *Multiple-source shortest paths in planar graphs*, in Proc. 16th Ann. ACM-SIAM Symp. Discrete Algorithms, 2005, pp. 146–155.

[87] M. Kutz, *Computing shortest non-trivial cycles on orientable surfaces of bounded genus in almost linear time*, in Proc. 22nd Ann. Symp. Comput. Geom., 2006, pp. 430–438, https://doi.org/10.1145/1137856.1137919.

[88] J. Łącki and P. Sankowski, *Min-cuts and shortest cycles in planar graphs in $O(n \log \log n)$ time*, in Proc. 19th Ann. Europ. Symp. Algorithms, C. Demetrescu and M. M. Halldórsson, eds., vol. 6942 of Lecture Notes Comput. Sci., Springer, 2011, pp. 155–166.

[89] S. K. S. K. Lando, *Graphs on surfaces and their applications*, Encyclopaedia of mathematical sciences; v. 141, Springer, Berlin ; New York, 2004.

[90] Y. T. Lee and A. Sidford, *Path finding methods for linear programming: Solving linear programs in $\tilde{o}$(vrank) iterations and faster algorithms for maximum flow*, in Proc. 55th IEEE Symp. Found. Comput. Sci., 2014, pp. 424–433, https://doi.org/10.1109/FOCS.2014.52.

[91] S. Lins, *Graph-encoded maps*, J. Comb. Theory Ser. B, 32 (1982), pp. 171–181.

[92] R. J. Lipton, D. J. Rose, and R. E. Tarjan, *Generalized nested dissection*, SIAM J. Numer. Anal., 16 (1979), pp. 346–358, https://doi.org/10.1137/0716027.

[93] A. Mądry, *Navigating central path with electrical flows: From flows to matchings, and back*, in Proc. 54th IEEE Symp. Found. Comput. Sci., 2013, pp. 253–262, https://doi.org/10.1109/FOCS.2013.35.

[94] M. Mareš, *Two linear time algorithms for MST on minor closed graph classes*, Archivum Mathematicum, 40 (2004), pp. 315–320.

[95] S. T. McCormick, M. R. Rao, and G. Rinaldi, *Easy and difficult objective functions for max cut*, Math. Program., Ser. B, 94 (2003), pp. 459–466.

[96] G. L. Miller, *Isomorphism testing for graphs of bounded genus*, in Proc. 12th Ann. ACM Symp. Theory Comput., 1980, pp. 225–235.

[97] G. L. Miller and J. Naor, *Flow in planar graphs with multiple sources and sinks*, SIAM J. Comput., 24 (1995), pp. 1002–1017.

[98] B. Mohar and C. Thomassen, *Graphs on Surfaces*, Johns Hopkins Univ. Press, 2001.

[99] S. Mozes, C. Nikolaev, Y. Nussbaum, and O. Weimann, *Minimum cut of directed planar graphs in $O(n \log \log n)$ time*. Preprint, February 2015.

[100] S. Mozes, K. Nikolaev, Y. Nussbaum, and O. Weimann, *Minimum cut of directed planar graphs in $O(n \log\log n)$ time*, in Proc. 29th Ann. ACM-SIAM Symp. Discrete Algorithms, 2018, pp. 477–494, https://doi.org/10.1137/1.9781611975031.32.
[101] H. Nagamochi and T. Ibaraki, *Computing edge-connectivity in multigraphs and capacitated graphs*, SIAM J. Discrete Math., 5 (1992), pp. 54–66.
[102] J. B. Orlin, *Max flows in $O(nm)$ time, or better*, in Proc. 45th Ann. ACM Symp. Theory Comput., 2013, pp. 765–774.
[103] V. Y. Pan and J. H. Reif, *Fast and efficient parallel solution of sparse linear systems*, SIAM J. Comput., 22 (1993), pp. 1227–1250.
[104] D. Pe'er, *On minimum spanning trees*, master's thesis, Hebrew University, 1998, http://www.math.ias.edu/~avi/STUDENTS/dpthesis.pdf.
[105] K. Reidemeister, *Elementare Begründung der Knotentheorie*, Abh. Math. Sem. Hamburg, 5 (1927), pp. 24–32.
[106] J. Reif, *Minimum $s$-$t$ cut of a planar undirected network in $O(n \log^2 n)$ time*, SIAM J. Comput., 12 (1983), pp. 71–81.
[107] R. B. Richter and H. Shank, *The cycle space of an embedded graph*, J. Graph Theory, 8 (1984), pp. 365–369, https://doi.org/10.1002/jgt.3190080304.
[108] G. Ringel, *Map Color Theorem*, Springer-Verlag, 1974.
[109] G. Ringel and J. W. T. Youngs, *Solution of the Heawood map-coloring problem*, Proc. Nat. Acad. Sci. USA, 60 (1968), pp. 438–445.
[110] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, no. 24 in Algorithms and Combinatorics, Springer-Verlag, 2003.
[111] A. Schrijver, *On the history of combinatorial optimization (till 1960)*, in Handbook of Discrete Optimization, K. Aardal, G. Nemhauser, and R. Weismantel, eds., Elsevier, 2005, pp. 1–68.
[112] J. Stillwell, *Classical Topology and Combinatorial Group Theory*, no. 72 in Graduate Texts in Mathematics, Springer-Verlag, 2nd ed., 1993.
[113] M. Stoer and F. Wagner, *A simple min-cut algorithm*, J. ACM, 44 (1997), pp. 585–591, https://doi.org/10.1145/263867.263872.
[114] J. M. Sullivan, *A Crystalline Approximation Theorem for Hypersurfaces*, PhD thesis, Princeton Univ., October 1990, http://torus.math.uiuc.edu/jms/Papers/thesis/thesis.pdf.
[115] G. Taubin and J. Rossignak, *Geometric compression through topological surgery*, ACM Trans. Graphics, 17 (1998), pp. 84–115, https://doi.org/10.1145/274363.274365.
[116] S. Tazari and M. Müller-Hannemann, *Shortest paths in linear time on minor-closed graph classes, with an application to Steiner tree approximation*, Discrete Appl. Math., 157 (2009), pp. 673–684.
[117] P. M. Vaidya, *Speeding-up linear programming using fast matrix multiplication*, in Proc. 30th Ann. Symp. Found. Comput. Sci., 1989, pp. 332–337.
[118] K. Weihe, *Maximum $(s,t)$-flows in planar networks in $O(|V|\log|V|)$-time*, J. Comput. Syst. Sci., 55 (1997), pp. 454–476.
[119] H. Whitney, *Planar graphs*, Fund. Math., 21 (1933), pp. 73–84.