Minimum Cycle and Homology Bases of Surface Embedded Graphs*

Glencora Borradaile[1], Erin Wolf Chambers[2], Kyle Fox[3], and Amir Nayyeri[4]

1 **Oregon State University, Corvallis, USA**
`glencora@eecs.orst.edu`
2 **St. Louis University, St. Louis, USA**
`echambe5@slu.edu`
3 **Duke University, Durham, USA**
`kylefox@cs.duke.edu`
4 **Oregon State University, Corvallis, USA**
`nayyeria@eecs.orst.edu`

Abstract

We study the problems of finding a minimum cycle basis (a minimum weight set of cycles that form a basis for the cycle space) and a minimum homology basis (a minimum weight set of cycles that generates the 1-dimensional ($\mathbb{Z}_2$)-homology classes) of an undirected graph embedded on an orientable surface of genus g. The problems are closely related, because the minimum cycle basis of a graph contains its minimum homology basis, and the minimum homology basis of the 1-skeleton of any graph is exactly its minimum cycle basis.

For the minimum cycle basis problem, we give a deterministic $O(n^\omega + 2^{2g}n^2)$-time algorithm. The best known existing algorithms for surface embedded graphs are those for general sparse graphs: an $O(n^\omega)$ time Monte Carlo algorithm [2] and a deterministic $O(n^3)$ time algorithm [27]. For the minimum homology basis problem, we give an $O(g^3 n \log n)$-time algorithm, improving on existing algorithms for many values of g and n.

1998 ACM Subject Classification F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Cycle basis, Homology basis, Topological graph theory

Digital Object Identifier 10.4230/LIPIcs.SoCG.2016.23

1 Introduction

Minimum cycle basis

Let $G = (V, E)$ be a connected simple undirected graph with n vertices and m edges. We define a *cycle* of G to be a subset $E' \subseteq E$ where each vertex $v \in V$ is incident to an even number of edges in E'. The *cycle space* of G is the vector space over cycles in G where addition is defined as the symmetric difference of cycles' edge sets. It is well known that the cycle space of G is isomorphic to $\mathbb{Z}_2^{m-n+1}$; in particular, the cycle space can be generated by the fundamental cycles of any spanning tree of G. A *cycle basis* is a maximal set of independent cycles. A *minimum cycle basis* is a cycle basis with a minimum number of edges

* This material is based upon work supported by the National Science Foundation under grants CCF-12-52833, CCF-10-54779, IIS-13-19573, CCF-11-61359, IIS-14-08846, CCF-15-13816, and IIS-14-47554; by an ARO grant W911NF-15-1-0408; and by Grant 2012/229 from the U.S.-Israel Binational Science Foundation.

(counted with multiplicity) or minimum total weight if edges are weighted[1]. Minimum cycle bases have applications in many areas such as electrical circuit theory [24, 9], structural engineering [8], surface reconstruction [29], and the analysis of algorithms [26].

Sets of independent cycles form a matroid, so the minimum cycle basis can be computed using the standard greedy algorithm. However, there may be an exponential number of cycles in G. Horton [21] gave the first polynomial time algorithm for the problem by observing that every cycle in the minimum cycle basis is the fundamental cycle of a shortest path tree. Several other algorithms have been proposed to compute minimum cycle bases in general graphs [10, 17, 3, 23, 27, 2]. The fastest of these algorithms are an $O(n^\omega)$ time Monte Carlo randomized algorithm of Amaldi *et al.* [2] and an $O(nm^2/\log n + n^2 m)$ time deterministic algorithm of Mehlhorn and Michail [27]. Here, $O(n^\omega)$ is the time it takes to multiply two $n \times n$ matrices using fast matrix multiplication.

For the special case of *planar graphs*, faster algorithms are known. Hartvigsen and Mardon [20] observed that the cycles in the minimum cycle basis nest, implicitly forming a tree; in fact, the edges of each cycle span an s, t-minimum cut between two vertices in the dual graph, and the Gomory-Hu tree [18] of the dual graph is precisely the tree of minimum cycle basis cycles in the primal. Hartvigsen and Mardon [20] gave an $O(n^2 \log n)$ time algorithm for the minimum cycle basis problem in planar graphs, and Amaldi *et al.* [2] improved their running time to $O(n^2)$. Borradaile, Sankowski, and Wulff-Nilsen [4] showed how to compute an *oracle* for the minimum cycle basis and dual minimum cuts in $O(n \log^4 n)$ time that is able to report individual cycles or cuts in time proportional to their size. Borradaile *et al.* [5] recently generalized the minimum cut oracle to graphs embeddable on surfaces of genus g. Their oracle takes $O(2^{O(g^2)} n \log^3 n)$ time to construct (improving upon the original planar oracle by a factor of $\log n$). Unfortunately, their oracle does not help in finding the minimum cycle basis in higher genus graphs, because there is no longer a bijection between cuts in the dual graph and cycles in the primal graph.

That said, it is not surprising that the cycle basis oracle has not been generalized beyond the plane. While cuts in the dual continue to nest in higher genus surfaces, cycles do not. In fact, the minimum cycle basis of a toroidal graph must *always* contain at least one pair of crossing cycles, because any cycle basis must contain cycles which are topologically distinct, in different *homology classes* of the surface.

Minimum homology basis

Given a graph G embedded in a surface Σ of genus g, the homology of G is an algebraic description of the topology of Σ and of G's embedding. In this paper, we focus on orientable surfaces and one-dimensional cellular homology over the coefficient ring $\mathbb{Z}_2$. Homology of this type allows for simplified definitions. We say a cycle η is *null-homologous* if η is the boundary of a subset of faces $F' \subseteq F$. Two cycles η and η' are *homologous* or in the same *homology class* if their symmetric difference $\eta \oplus \eta'$ is null-homologous. The homology classes form a vector space isomorphic to $\mathbb{Z}_2^{2g}$ known as the *homology space*. A *homology basis* of G is a set of $2g$ cycles in linearly independent homology classes, and the *minimum homology basis* of G is the homology basis with either the minimum number of edges or with minimum total weight if edges of G are weighted.

Erickson and Whittlesey [15] described an $O(n^2 \log n + gn^2 + g^3 n)$ time algorithm for computing the minimum homology basis. Like Horton [21], they apply the greedy matroid

[1] There is a notion of minimum cycle bases in directed graphs as well, but we focus on the undirected case in this paper.

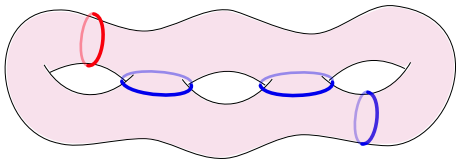

Figure 1 Two homologous cycles, one shown in red and the other in blue.

basis algorithm to a set of $O(n^2)$ candidate cycles. A set of 2^{2g} candidate cycles containing the minimum homology basis can be computed easily by applying the algorithms of Italiano *et al.* [22] or Erickson and Nayyeri [14] for computing the minimum homologous cycle in any specified homology class. These algorithms take $g^{O(g)}n \log\log n$ and $2^{O(g)}n \log n$ time respectively. Dey, Sun, and Wang [11] generalized these results to arbitrary simplicial complexes, and Busaryev *et al.* [6] improved the running time of their generalization from $O(n^4)$ to $O(n^\omega + n^2 g^{\omega-1})$. Note that all of the algorithms above either take quadratic time in n (or worse) *or* they have exponential dependency on g. In contrast, it is well understood how to find exactly one cycle of the minimum homology basis of G in only $O(g^2 n \log n)$ time, since the minimum weight non-separating cycle will always be in the basis [7, 13].

Our results

We describe new algorithms for computing the minimum cycle basis and minimum homology basis of the graph G. Our algorithm for minimum cycle basis is deterministic and runs in $O(n^\omega + 2^{2g}n^2)$ time, matching the running time of the randomized algorithm of Amaldi *et al.* [2] when g is sufficiently small. Our algorithm for minimum homology basis is also deterministic and runs in $O(g^3 n \log n)$ time assuming shortest paths are unique[2]. Ours is the first algorithm for minimum homology basis that has a running time simultaneously near-linear in n and polynomial in g.

At a high level both of our algorithms are based on the $O(nm^2 + n^2 m \log n)$ time algorithm of Kavitha *et al.* [23] who in turn use an idea of de Pina [10]. We compute our basis cycles one by one. Over the course of the algorithm, we maintain a set of *support vectors* that form the basis of the subspace that is *orthogonal* to the set of cycles we have already computed. Every time we compute a new cycle, we find the one of minimum weight that is *not* orthogonal to a chosen support vector S, and then update the remaining support vectors so they remain orthogonal to our now larger set of cycles. Using the divide-and-conquer approach of Kavitha *et al.* [23], we are able to maintain these support vectors in only $O(n^\omega)$ time total in our minimum cycle basis algorithm and $O(g^\omega)$ time total in our minimum homology basis algorithm. Our approaches for picking the minimum weight cycle not orthogonal to S form the more technically interesting parts of our algorithms and are unique to this work.

For our minimum cycle basis algorithm, we compute a collection of $O(2^{2g}n)$ cycles that contain the minimum cycle basis and then partition these cycles according to their homology classes. The cycles within a single homology class nest in a similar fashion to the minimum cycle basis cycles of a planar graph. Every time we compute a new cycle for our minimum cycle basis, we walk up the 2^{2g} trees of nested cycles and find the minimum weight cycle

[2] This assumption is only necessary to use the multiple-source shortest path data structure of Cabello, Chambers, and Erickson. It can be avoided with high probability by using randomization or deterministically by increasing the running time of our algorithm by a factor of $O(\log n)$ [7]. For simplicity, we will assume shortest paths are unique during presentation of our minimum homology basis algorithm.

not orthogonal to S in $O(n)$ time per tree. Overall, we spend $O(2^{2g}n^2)$ time finding these cycles; if any improvement is made on the time it take to update the support vectors, then the running time of our algorithm as a whole will improve as well.

Our minimum homology basis algorithm uses a covering space called the cyclic double cover. As shown by Erickson [13], the cyclic double cover provides a convenient way to find a minimum weight closed walk γ crossing an arbitrary non-separating cycle λ an odd number of times. We extend his construction so that we may consider not just one λ but any arbitrarily large collection of cycles. Every time we compute a new cycle in our minimum homology basis algorithm, we let S determine a set of cycles that must be crossed an odd number of times, build the cyclic double cover for that set, and then compute our homology basis cycle in $O(g^2 n \log n)$ time by computing minimum weight paths in the covering space[3].

The rest of the paper is organized as follows. We provide more preliminary material on surface embedded graphs in Section 2. In Section 3, we describe a characterization of cycles and homology classes using binary vectors. These vectors are helpful in formally defining our support vectors. We give a high level overview of our minimum cycle basis algorithm in Section 4 and describe how to pick individual cycles in Section 5. Finally, we give our minimum homology basis algorithm in Section 6.

2 Preliminaries

In this paper, we will be working with an edge-weighed undirected graph $G = (V, E)$ cellularly embedded on an orientable surface of genus g without boundary. To simplify the exposition, we assume G is connected and simple. We let F denote the faces of G and let f_∞ denote an arbitrary face we refer to as the *infinite face* for convenience. Let n, m, and ℓ be the number of vertices, edges, and faces of G respectively. The *Euler characteristic* χ of Σ is $2 - 2g$. By Euler's formula, $\chi = n - m + \ell$. We assume $g = O(n^{1-\varepsilon})$ for some constant $\varepsilon > 0$ (otherwise, our algorithms offer no improvement over previously known results.) This assumption implies $m = O(n)$ and $\ell = O(n)$. Embedded graphs can be *dualized*: G^* is the graph embedded on the same surface, with a vertex in G^* for every face in G and a face in G^* for every vertex of G. Two vertices in G^* are then adjacent if the corresponding faces are separated by an edge in G. We generally do not distinguish between edges in the primal and dual graphs.

A *spanning tree* of the graph G is a subset of edges of G which is a tree containing every vertex. Similarly, a *cotree* C is a spanning tree of the dual graph G^*. A *tree-cotree decomposition* of G is a partition of G into 3 edge disjoint subsets, (T, L, C), where T is a spanning tree of G, C is a cotree, and L is the set of leftover edges $E \setminus (T \cup C)$ [12]. Euler's formula implies that $|L| = 2g$.

A w, w'*-path* p is an ordered sequence of edges $\{u_1v_1, u_2v_2, \ldots, u_kv_k\}$ where $w = u_1$, $w' = v_k$, and $v_i = u_{i+1}$ for all positive $i < k$; a *closed path* is a path which starts and ends on the same vertex. A path is *simple* if it repeats no vertices (except the first and last). A path in the dual graph G^* is referred to as a *co-path* and (abusing terminology) a closed path in the dual is referred to as a *co-cycle*. We sometimes use *simple cycle* to mean

[3] In addition to the above results, we note that it is possible to improve the $g^{O(g)}n \log\log n$ time algorithm of Italiano *et al.* [22] for minimum homology basis to run in $2^{O(g)}n \log\log n$ time. However, this improvement is a trivial adaption of techniques used by Fox [16] to get a $2^{O(g)}n \log\log n$ time algorithm for minimum weight non-separating and non-contractible cycle in undirected graphs. We will not further discuss this improvement in the paper.

a simple closed path. Every member of the minimum cycle basis (and subsequently the minimum homology basis) is a simple cycle [21]. We let $\sigma(u, v)$ denote an arbitrary shortest (minimum weight) u, v-path in G. Let $p[u, v]$ denote the *subpath* of p from u to v. Given a u, v-path p and a v, w-path p', let $p \cdot p'$ denote their concatenation. Two paths p and p' *cross* if their embeddings in Σ cannot be be made disjoint through infinitesimal perturbations; more formally, they cross if there is a maximal (possibly trivial) common subpath p'' of p and p' such that, upon contracting p'' to a vertex v, two edges each of p and p' alternate in their embedded around v. Two closed paths cross if they have subpaths which cross.

Let γ be a closed path that does not cross itself. We define the operation of *cutting* along γ and denote it $G \not\leftarrow \gamma$. Graph $G \not\leftarrow \gamma$ is obtained by cutting along γ in the drawing of G on the surface, creating two copies of γ. The two copies of γ each form boundary components in the cut open surface.

Let $F' \subseteq F$ be a subset of faces and let η be the boundary of F'. We sometimes call F' a *cut* of G^* and say η *spans* the cut. A co-path p with edge $uv \in \eta$ *crosses* the cut at uv.

3 Cycle and Homology Signatures

We begin the presentation of our algorithms by giving a characterization of cycles and homology classes using binary vectors. These vectors will be useful in helping us determine which cycles can be safely added to our minimum cycle and homology bases. Let (T, L, C) be an arbitrary tree, co-tree decomposition of G; set L contains exactly $2g$ edges $e_1, \ldots, e_{2g}$. For each index $i \in \{1, \ldots, 2g\}$, let p_i denote the unique co-cycle in $C \cup \{e_i\}$. Let f_∞ be any designated face of G. Let $p_{2g+1}, \ldots, p_{m-n+1}$ be any collection of simple co-paths from f_∞ to each of the $m - n + 1 - 2g$ other faces of G.

For each edge e in G, we define its *cycle signature* $[e]$ as an $(m - n + 1)$-bit vector whose ith bit is equal to 1 if and only if e appears in p_i. The cycle signature $[\eta]$ of any cycle η is the bitwise exclusive-or of the signatures of its edges. Equivalently, the ith bit of $[\eta]$ is 1 if and only if η and p_i share an odd number of edges. Similarly, for each edge e in G, we define its *homology signature* $[e]_h$ as a $2g$-bit vector whose ith bit is equal to 1 if and only if e appears in p_i. The homology signature of cycles are defined similarly.

The following lemma is immediate.

▶ **Lemma 1.** *Let η and η' be two cycles. We have $[\eta \oplus \eta'] = [\eta] \oplus [\eta']$ and $[\eta \oplus \eta']_h = [\eta]_h \oplus [\eta']_h$.*

Cycle and homology signatures provide a convenient way to distinguish between cycles and their homology classes as shown by the following lemmas.

▶ **Lemma 2** (Erickson and Nayyeri [14, Corollary 3.3])**.** *Two cycles η and η' are homologous if and only if $[\eta]_h = [\eta']_h$.*

▶ **Lemma 3.** *Let η and η' be two homologous cycles and let $\eta \oplus \eta'$ be the boundary of a subset of faces F' such that $f_\infty \notin F'$ (set F' may be empty.) Cycle signatures $[\eta]$ and $[\eta']$ differ at bit i if and only if p_i is a co-path with exactly one endpoint in F'.*

Proof. By Lemma 2, the first $2g$ bits of $[\eta]$ and $[\eta']$ are identical. Now, consider any other bit i. The boundary of set F' is a cut in G^* that does not contain f_∞, and $\eta \oplus \eta'$ spans the cut. Suppose one endpoint of p_i lies in F'. Co-path p_i must cross the cut an odd number of times. In particular, p_i intersects exactly one of η and η' an odd number of times and the ith bits of $[\eta]$ and $[\eta']$ differ. Now, suppose instead that both endpoints of p_i lie outside F'. Here, p_i crosses the cut an even number of times. Co-path p_i either intersects both η and η' an even number of times or it intersects them both an odd number of times. Either way, the ith bits of $[\eta]$ and $[\eta']$ are equal. ◀

▶ **Lemma 4.** *Let η and η' be two cycles. We have $\eta = \eta'$ if and only if $[\eta] = [\eta']$.*

Proof. Necessity is trivial. Now, suppose $[\eta] = [\eta']$. Lemma 2 states that η and η' are $\mathbb{Z}_2$-homologous. In particular, $\eta \oplus \eta'$ forms the boundary of a subset F' of faces such that $f_\infty \notin F'$. However, subset F' must be empty by Lemma 3. We conclude $\eta = \eta'$. ◀

▶ **Corollary 5.** *Cycle signatures are an isomorphism between the cycle space and $\mathbb{Z}_2^{m-n+1}$, and homology signatures are an isomorphism between the first homology space and $\mathbb{Z}_2^{2g}$.*

4 Minimum Cycle Basis

Our algorithm for computing a minimum cycle basis is based on one of Kavitha, Mehlhorn, Michail and Paluch [23] which is in turn based on an algorithm of de Pina [10]. Our algorithm incrementally adds simple cycles $\gamma_1, \ldots, \gamma_{m-n+1}$ to the minimum cycle basis. In order to do so, it maintains a set of $(m-n+1)$-bit *support vectors* $S_1, \ldots, S_{m-n+1}$ with the following properties:

- The support vectors form a basis for $\mathbb{Z}_2^{m-n+1}$.
- When the algorithm is about to compute the jth simple cycle γ_j for the minimum cycle basis, $\langle S_j, [\gamma_{j'}] \rangle = 0$ for all $j' < j$.

Our algorithm chooses for γ_j the minimum weight cycle γ such that $\langle S_j, [\gamma] \rangle = 1$. Note that S_j must have at least one bit set to 1, because the set of vectors $S_1, \ldots, S_{m-n+1}$ forms a basis. Therefore, such a γ does exist; in particular, we could choose $[\gamma]$ to contain exactly one bit equal to 1 which matches any 1-bit of S_j. The correctness of choosing γ_j as above is guaranteed by the following lemma.

▶ **Lemma 6.** *Let S be an $(m+n-1)$-bit vector with at least one bit set to 1, and let η be the minimum weight cycle such that $\langle S, [\eta] \rangle = 1$. Then, η is a member of the minimum cycle basis.*

Proof. Let $\eta_1, \ldots, \eta_{2^{m-n+1}}$ be the collection of cycles ordered by increasing weight, and choose j such that $\eta_j = \eta$. For any subset Υ of $\{\eta_1, \ldots, \eta_{j-1}\}$, we have $\langle [\bigoplus_{\eta' \in \Upsilon} \eta'], S \rangle = 0$, where $\bigoplus$ is the symmetric difference of its operands. Therefore, η is independent of $\{\eta_1, \ldots, \eta_{j-1}\}$. Sets of independent cycles form a matroid, so η must be a member of the minimum weight cycle basis. ◀

4.1 Computing support sets

Our algorithm updates the support vectors and computes minimum cycle basis vectors in a recursive manner. Initially, each support vector S_i has only its ith bit set to 1. Borrowing nomenclature from Kavitha *et al.* [23], we define two procedures, extend(j, k) which extends the current set of basis cycles by adding k cycles starting with γ_j, and update(j, k) which updates support vectors $S_{j+\lfloor k/2 \rfloor}, \ldots, S_{j+k-1}$ so that for any j', j'' with $j + \lfloor k/2 \rfloor \leq j' < j+k$ and $1 \leq j'' < j + \lfloor k/2 \rfloor$, we have $\langle S_{j'}, [\gamma_{j''}] \rangle = 0$. Our algorithm runs extend$(1, m-n+1)$ to compute the minimum cycle basis.

We implement extend(j, k) in the following manner: If $k > 1$, then our algorithm recursively calls extend$(j, \lfloor k/2 \rfloor)$ to add $\lfloor k/2 \rfloor$ cycles to the partial minimum cycle basis. It then calls update(j, k) so that support vectors $S_{j+\lfloor k/2 \rfloor}, \ldots, S_{j+k-1}$ become orthogonal to the newly added cycles of the partial basis. Finally, it computes the remaining $\lceil k/2 \rceil$ basis cycles by calling extend$(j + \lfloor k/2 \rfloor, \lceil k/2 \rceil)$. If $k = 1$, then $\langle S_j, [\gamma_{j'}] \rangle = 0$ for all $j' < j$. Our algorithm is ready to find basis cycle γ_j. We describe an $O(2^{2g} n)$ time procedure to find γ_j in Section 5.

We now describe update(j,k) in more detail. Our algorithm updates each support vector $S_{j'}$ where $j+\lfloor k/2\rfloor \le j' < j+k$. The vector $S_{j'}$ becomes $S'_{j'} = S_{j'} + \alpha_{j'0}S_j + \alpha_{j'1}S_{j+1} + \cdots + \alpha_{j'(\lfloor k/2\rfloor-1)}S_{j+\lfloor k/2\rfloor-1}$ for some set of scalar bits $\alpha_{j'0}\ldots\alpha_{j'(\lfloor k/2\rfloor-1)}$. After updating, the set $S_1,\ldots,S_{m-n+1}$ remains a basis for $\mathbb{Z}_2^{m-n+1}$ regardless of the choices for the α bits. Further, $\langle S'_{j'}, [\gamma_{j''}]\rangle = 0$ for all $j'' < j$ for all choices of the α bits, because extend(j,k) is only called after its support vectors are updated to be orthogonal to all minimum basis cycles $\gamma_1,\ldots,\gamma_{j-1}$. However, it is non-trivial to guarantee $\langle S'_{j'}, [\gamma_{j''}]\rangle = 0$ for all j'' where $j \le j'' < j+\lfloor k/2\rfloor$. Kavitha *et al.* [23, Section 4] describe how to select the new vectors $S'_{j'}$ in $O(nk^{\omega-1})$ time using linear algebraic manipulations and fast matrix multiplication.

We can bound the running time of extend(j,k) using the following recurrence:

$$T(k) = \begin{cases} 2T(k/2) + O(nk^{\omega-1}) & \text{if } k > 1 \\ O(2^{2g}n) & \text{if } k = 1 \end{cases}$$

The total time spent in calls to extend(j,k) where $k>1$ is $O(nk^{\omega-1})$. The total time spent in calls to extend$(j,1)$ is $O(2^{2g}nk)$. Therefore, $T(k) = O(nk^{\omega-1} + 2^{2g}nk)$. The running time of our minimum cycle basis algorithm is $T(O(n)) = O(n^\omega + 2^{2g}n^2)$.

5 Selecting cycles

A *Horton cycle* is a simple cycle given by a shortest x,u-path, a shortest x,v-path, and the edge uv; in particular, the set of all Horton cycles is given by the set of $m-n+1$ elementary cycles for each of the n shortest path trees [21]. Thus, in graphs of fixed genus, there are $O(n^2)$ Horton cycles. A simple cycle γ of a graph G is *isometric* if for every pair of vertices $x,y \in \gamma$, γ contains a shortest x,y-path. Hartvigsen and Mardon prove that the cycles of any minimum cycle basis are all isometric [20]. Therefore, it suffices for us to focus on the set of isometric cycles to find the cycle γ_j as needed for Section 4.1.

Amaldi, Iuliano, Jurkiewicz, Mehlhorn, and Rizzi show how to, in a graph with n vertices and m edges, extract a set of distinct isometric cycles from a set of Horton cycles in $O(nm)$ time [2]. Each isometric cycle is identified by a shortest path tree's root and a non-tree edge.

Here, we show that there are, in fact, at most $O(2^{2g}n)$ isometric cycles in a graph of genus g (Section 5.1), and they can be partitioned into sets according to their homology classes. We can represent the isometric cycles in a given homology class using a tree that can be built in $O(n^2)$ time (Section 5.2). We then show that we can use these trees to find the minimum-cost cycle γ_j as needed for Section 4.1 in linear time per homology class of isometric cycles. We close with a discussion on how to improve the running time for computing and representing isometric cycles (Section 5.4). While these improvements do not improve the overall running time of our algorithm (by maintaining separate representations of the cycles according to their homology class, we require linear time per representation to process the support vector with respect to which γ_j is non-orthogonal; we also require $O(n^\omega)$ time to update the support vectors), it does further emphasize the bottleneck our algorithm faces in updating and representing the support vectors.

5.1 Isometric cycles in surface embedded graphs

Here we prove some additional structural properties that isometric cycles have in surface embedded graphs. To this end, we herein assume that shortest paths are unique. Hartvigsen and Mardon show how to achieve this assumption algorithmically when, in particular, all

pairs of shortest paths are computed, as we do [20]. We first prove a generalization of the following lemma for the planar case by Borradaile, Sankowski and Wulff-Nilsen.

▶ **Lemma 7** (Borradaile *et al.* [4, Lemma 1.4])**.** *Let G be a graph in which shortest paths are unique. The intersection between an isometric cycle and a shortest path in G is a (possibly empty) shortest path. The intersection between two distinct isometric cycles γ and γ' in G is a (possibly empty) shortest path; in particular, if G is a planar embedded graph, γ and γ' do not cross.*

▶ **Lemma 8.** *Two isometric cycles in a given homology class in a graph with unique shortest paths do not cross.*

Proof. Let γ and γ' be two isometric cycles in a given homology class. Suppose for a contradiction that γ and γ' cross. By the first part of Lemma 7, and the assumption that γ and γ' cross, $\gamma \cap \gamma'$ is a single simple path p. Therefore, γ and γ' cross exactly once.

Suppose γ and γ' are not null-homologous. Cutting the surface open along γ results in a connected surface with two boundary components which are connected by γ'. Cutting the surface further along γ' does not disconnect the surface. Therefore $\gamma \oplus \gamma'$ does not disconnect the surface, and so γ and γ' are not homologous, a contradiction.

If γ and γ' are null-homologous, then cutting the surface open along γ results in a disconnected surface in which $\gamma' \setminus p$ is a path, but between different components of the surface, a contradiction. ◀

▶ **Corollary 9.** *There are at most ℓ distinct isometric cycles in a given homology class in a graph with ℓ faces and unique shortest paths.*

Proof. Consider the set $\{C_1, C_2, \ldots\}$ of distinct isometric cycles in a given homology class other than the null homology class. We prove by induction that $\{C_1, C_2, \ldots, C_i\}$ cut the surface into non-trivial components, each of which is bounded by exactly two of $C_1, C_2, \ldots, C_i$; this is true for C_1, C_2 since they are homologous, distinct and do cross. C_{i+1} must be contained in one component, bounded by, say, C_j and C_k since C_{i+1} does not cross any other cycle. Cutting this component along C_{i+1} creates two components bounded by C_j, C_{i+1} and C_k, C_{i+1}, respectively. Since the cycles are distinct, these component must each contain at least one face. A similar argument holds for the set of null-homologous isometric cycles. ◀

Since there are 2^{2g} homology classes and $\ell = O(n)$, we get:

▶ **Corollary 10.** *There are $O(2^{2g}n)$ distinct isometric cycles in a graph of genus g with unique shortest paths.*

5.2 Representing isometric cycles in each homology class

We begin by determining the homology classes of each of the $O(2^{2g}n)$ isometric cycles in the following manner. Let p be a simple path, and let $[p]_h$ denote the bitwise exclusive-or of the homology signatures of its edges. Let r be the root of any shortest path tree T. Recall, $\sigma(r, v)$ denotes the shortest path between r and v. It is straightforward to compute $[\sigma(r, v)]_h$ for every vertex $v \in V$ in $O(gn)$ time by iteratively computing signatures in a leafward order. Then, the homology signature of any isometric cycle $\gamma = \sigma(r, u) \cdot uv \cdot \sigma(v, r)$ can be computed in $O(g)$ time as $[\sigma(r, u)]_h \oplus [uv]_h \oplus [\sigma(r, v)]_h$. We spend $O(2^{2g}gn) = O(2^{2g}n^2)$ time total computing homology signatures and therefore homology classes. For the remainder of this section, we consider a set of isometric cycles $\mathcal{C}$ in a single homology class.

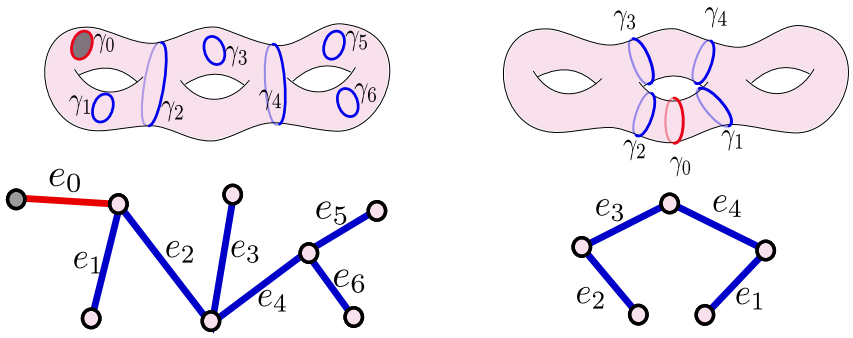

Figure 2 Two collections of homologous cycles and their generalized region trees. Left: The cycles are null-homologous. Right: The cycles lie in a non-trivial homology class.

Let $\gamma, \gamma' \in \mathcal{C}$ be two isometric cycles in the same homology class. The combination $\gamma \oplus \gamma'$ forms the boundary of a subset of faces. That is, $G \not\downarrow (\gamma \cup \gamma')$ contains at least two components. We represent the cycles in $\mathcal{C}$ by a tree $T_{\mathcal{C}}$ where each edge e of $T_{\mathcal{C}}$ corresponds to a cycle $\gamma(e) \in \mathcal{C}$ and each node v in $T_{\mathcal{C}}$ corresponds to a subset of faces $F(v)$; specifically, the nodes correspond to sets of faces in the components of $G \not\downarrow \mathcal{C}$. This tree generalizes the *region tree* defined by Borradaile, Sankowski and Wulff-Nilsen for planar graphs [4] to more general surface embedded graphs. We also designate a single representative cycle $\gamma(\mathcal{C})$ of $\mathcal{C}$ and pre-compute its cycle signature $[\gamma(\mathcal{C})]$ for use in our basis cycle finding procedure. See Figure 2.

We describe here the construction of $T_{\mathcal{C}}$. Initially, $T_{\mathcal{C}}$ is a single vertex with one (looping) edge to itself (we will guarantee $T_{\mathcal{C}}$ is a tree later). Let γ_0 be an arbitrary cycle in $\mathcal{C}$. We compute $G' = G \not\downarrow \gamma_0$. For the one vertex v of $T_{\mathcal{C}}$, we set $F(v)$ to be every face of G' and for the one edge e, we set $\gamma(e) = \gamma_0$.

We maintain the invariants that every component of G' is bound by at least two cycles of $\mathcal{C}$ (initially the cycle γ_0 is used twice), each vertex of $T_{\mathcal{C}}$ is associated with all faces in one component of G', and each edge e in $T_{\mathcal{C}}$ is associated with the cycle in $\mathcal{C}$ bounding the faces for the two vertices incident to e. Assuming these invariants are maintained, and because cycles in $\mathcal{C}$ do not cross, each cycle in $\mathcal{C}$ lies entirely within some component of G'. For each cycle $\gamma \in \mathcal{C} \setminus \{\gamma_0\}$, we set $G' := G' \not\downarrow \gamma$, subdivide the vertex associated with the faces of C's component, associate the two sets of faces created in G' with the two new vertices of $T_{\mathcal{C}}$, and associate the new edge of $T_{\mathcal{C}}$ with γ.

Let r be the vertex of $T_{\mathcal{C}}$ associated with f_∞. If cycles in $\mathcal{C}$ have trivial homology, then they each separate G, and $T_{\mathcal{C}}$ is a tree. We root $T_{\mathcal{C}}$ at r and let $\gamma(\mathcal{C})$ be an arbitrary cycle. Otherwise, let e be an arbitrary edge incident to r. We set $\gamma(\mathcal{C})$ to be $\gamma(e)$, remove e from $T_{\mathcal{C}}$, and root T at r. Observe that T has exactly one leaf other than r in this case.

Computing $G' \not\downarrow \gamma$ for one cycle γ takes $O(n)$ time. Therefore, we can compute $T_{\mathcal{C}}$ in $O(n^2)$ time total.

5.3 Selecting an isometric cycle from a homology class

Let S be an $(m - n + 1)$-bit support vector. We describe a procedure to compute $\langle S, [\gamma] \rangle$ for every isometric cycle γ in G in $O(2^{2g}n)$ time. Using this procedure, we can easily return the minimum weight cycle such that $\langle S, [\gamma] \rangle = 1$.

We begin describing the procedure for cycles in the trivial homology class. Let $\mathcal{C}$ be the collection of null-homologous isometric cycles computed above, and let $T_{\mathcal{C}}$ be the tree

computed for this set. Consider any edge e of $T_{\mathcal{C}}$. The first $2g$ bits of $[\gamma(e)]$ are equal to 0 [14, Lemma 3.2]. Cycle $\gamma(e)$ bounds a subset of faces F' such that $f_\infty \notin F'$. In particular, F' is the set of faces associated with vertices lying *below* e in $T_{\mathcal{C}}$. By Lemma 3, the ith bit of $[\gamma(e)]$ is 1 if and only if F' contains an endpoint of p_i. Therefore, the ith bit of $[\gamma(e)]$ is 1 if and only if the endpoint of p_i other than f_∞ is associated with a vertex lying below e in $T_{\mathcal{C}}$.

We compute $\langle S, [\gamma]\rangle$ for every cycle $\gamma \in \mathcal{C}$ in $O(n)$ time by essentially walking up $T_{\mathcal{C}}$ in the following manner. For each edge e in $T_{\mathcal{C}}$ going to a leaf v, we maintain a bit z initially equal to 0 and iterate over the faces in $F(v)$. Each face is the endpoint of some path p_i. If the ith bit of S is equal to 1 then we flip z. After going through all the faces, z is equal to $\langle S, [\gamma(e)]\rangle$.

We then iterate up the edges of $T_{\mathcal{C}}$ toward the root. For each edge e, we let v be the lower endpoint of e and set bit z equal to the symmetric difference over all $\langle S, \gamma(e')\rangle$ for edges e' lying below v. We then iterate over the faces of $F(v)$ as before and set $\langle S, [\gamma(e)]\rangle$ equal to z as before. We iterate over every face of G at most once during this procedure, so it takes $O(n)$ time total.

Now, consider the set of isometric cycles $\mathcal{C}$ for some non-trivial homology class. Consider any edge e of $T_{\mathcal{C}}$. Let F' be the subset of faces bound by $\gamma(\mathcal{C}) \oplus \gamma(e)$ such that $f_\infty \notin F'$. By Lemma 3, the ith bit of $[\gamma(e)]$ disagrees with the ith bit of $[\gamma(\mathcal{C})]$ if and only if path p_i has one endpoint in F'. By construction, $\gamma(\mathcal{C})$ lies on the boundary of $F(r)$ and $F(v)$ where r and v are the root and other leaf of $T_{\mathcal{C}}$ respectively. Root r is the only node of $T_{\mathcal{C}}$ associated with f_∞. We conclude the ith bit of $[\gamma(e)]$ disagrees with $[\gamma(\mathcal{C})]$ if and only if the endpoint of p_i other than f_∞ is associated with a vertex lying below e in $T_{\mathcal{C}}$.

We again walk up $T_{\mathcal{C}}$ to compute $\langle S, [\gamma]\rangle$ for every cycle $\gamma \in \mathcal{C}$. Recall, $[\gamma(\mathcal{C})]$ is precomputed and stored with $T_{\mathcal{C}}$. For each edge e of $T_{\mathcal{C}}$ in rootward order, let v be the lower endpoint of e. Let e' be the edge lying below e in $T_{\mathcal{C}}$ if it exists. If e' does not exist, we denote $\gamma(e')$ as $\gamma(\mathcal{C})$. We set z equal to $\langle S, \gamma(e')\rangle$. We then iterate over the faces of $F(v)$ as before, flipping z once for every bit i where p_i has an endpoint in $F(v)$ and bit i of S is equal to 1. We set $\langle S, \gamma(e)\rangle := z$. As before, we consider every face at most once, so the walk up $T_{\mathcal{C}}$ takes $O(n)$ time.

We have shown the following lemma, which concludes the discussion of our minimum cycle basis algorithm.

▶ **Lemma 11.** *Let G be a graph with n vertices cellulary embedded in a surface of genus g. We can preprocess G in $O(2^{2g}n^2)$ time so that for any $(m - n + 1)$-bit support vector S we can compute the minimum weight cycle γ such that $\langle S, \gamma\rangle = 1$ in $O(2^{2g}n)$ time.*

▶ **Theorem 12.** *Let G be a graph with n vertices, cellularly embedded in an orientable surface of genus g. We can compute a minimum weight cycle basis of G in $O(n^\omega + 2^{2g}n^2)$ time.*

5.4 Improving the time for computing and representing isometric cycles

Here we discuss ways in which we can improve the running time for finding and representing isometric cycles using known techniques, thereby isolating the bottleneck of the algorithm to updating the support vectors and computing γ_j.

The set and representation of isometric cycles can computed recursively using $O(\sqrt{gn})$ balanced separators (e.g. [1]) as inspired by Wulff-Nilsen [30]. Briefly, given a set S of $O(\sqrt{gn})$ separator vertices (for a graph of bounded genus), find all the isometric cycles in each component of $G \setminus S$ and represent these isometric cycles in at most 2^{2g} region trees per component, as described above. Merging the region trees for different components of $G \setminus S$ is relatively simple since different sets of faces are involved. It remains to compute the set

of isometric cycles that contain vertices of S and add them to their respective region trees. First note that a cycle that is isometric in G and does not contain a vertex of S is isometric in $G \setminus S$, but a cycle that is isometric in $G \setminus S$ may not be isometric in G, so indeed we are computing a superset of the set of isometric cycles via this recursive procedure. However, it is relatively easy to show that an isometric cycle of $G \setminus S$ can cross an isometric cycle of G at most once, so, within a given homology class, isometric cycles will nest and be, therefore, representable by a region tree.

To compute the set of isometric cycles that intersect vertices of S, we first compute shortest paths trees rooted at each of the vertices of S, generating the Horton cycles rooted at these vertices; this procedure takes $O(\sqrt{gn} \cdot n)$ time using the linear time shortest path algorithm for graphs excluding minors of sub-linear size [28]. We point out that the algorithm of Amaldi et al. [2] works by identifying Horton cycles that are not isometric and by identifying, among different Horton-cycle representations of a given isometric cycle, one representative; this can be done for a subset of Horton cycles, such as those rooted in vertices of S, and takes time proportional to the size of the representation of the Horton cycles (i.e., the $O(\sqrt{n})$ shortest path trees, or $O(\sqrt{g}n^{1.5})$).

For a given homology class of cycles, using the shortest-path tree representation of the isometric cycles, we can identify those isometric cycles in that homology class by computing the homology signature of root-to-node paths in the shortest path tree as before; this process can be done in $O(\sqrt{g}n^{1.5})$ time. We must now add these cycles to the corresponding region tree. Borradaile, Sankowski and Wulff-Nilsen [4] describe a method for adding n cycles to a region tree in $O(n \operatorname{poly} \log n)$ time that is used in their minimum cycle basis algorithm for planar graphs; this method will generalize to surfaces for nesting cycles. Therefore computing the homology classes of these isometric cycles and adding these isometric cycles to the region trees takes a total of $O(2^{2g}\sqrt{g}n^{1.5})$ time.

In total, this recursive method for computing and building a representation of a superset of the isometric cycles takes time given by the recurrence relation $T(n) = 2T(n/2) + O(2^{2g}\sqrt{g}n^{1.5})$ or $O(2^{2g}\sqrt{g}n^{1.5})$ time.

6 Homology Basis

At a high level, our algorithm for minimum homology basis is very similar to our algorithm for minimum cycle basis. As before, our algorithm incrementally adds simple cycles $\gamma_1, \ldots, \gamma_{2g}$ to the minimum homology basis by maintaining a set of $2g$ support vectors $S_1, \ldots, S_{2g}$ such that the following hold:

- The support vectors form a basis for $\mathbb{Z}_2^{2g}$.
- When the algorithm is about to compute the jth cycle γ_j for the minimum homology basis, $\langle S_j, [\gamma_{j'}]_h \rangle = 0$ for all $j' < j$.

Our algorithm chooses for γ_j the minimum weight simple cycle γ such that $\langle S_j, [\gamma]_h \rangle = 1$. The following lemma has essentially the same proof as Lemma 6.

▶ **Lemma 13.** *Let S be a $2g$-bit vector with at least one bit set to 1, and let η be the minimum weight cycle such that $\langle S, [\eta]_h \rangle = 1$. Then, η is a member of the minimum homology basis.*

As before, our algorithm updates the support vectors and computes minimum homology basis cycles in a recursive manner. We define extend(j, k) and update(j, k) as before, using homology signatures in place of cycle signatures when applicable. Our algorithm runs extend$(1, 2g)$ to compute the minimum homology basis.

The one crucial difference between our minimum cycle basis and minimum homology basis algorithms is the procedure we use to find each minimum homology basis cycle γ_j

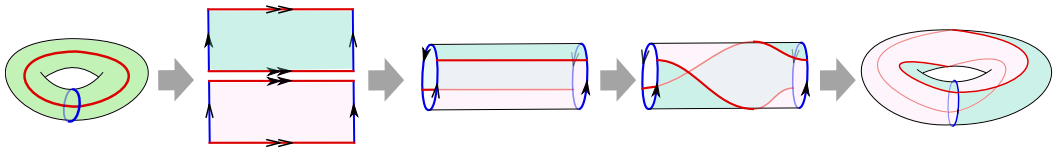

Figure 3 Constructing the cyclic double cover. Left to right: A pair of co-cycles Ψ on the torus Σ; the surfaces $(\Sigma', 0)$ and $(\Sigma', 1)$; identifying copies of one co-cycle; preparing to identify copies of the other co-cycle; the cyclic double cover.

given support vector S_j. This procedure takes $O(g^2 n \log n)$ time instead of $O(2^{2g} n)$ time and requires no preprocessing step. We describe the procedure in Sections 6.1 and 6.2.

The procedure update(j, k) takes only $O(gk^{\omega-1})$ time in our minimum homology basis algorithm, because signatures have length $O(g)$. Therefore, we can bound the running time of extend(j, k) using the following recurrence:

$$T(k) = \begin{cases} 2T(k/2) + O(gk^{\omega-1}) & \text{if } k > 1 \\ O(g^2 n \log n) & \text{if } k = 1 \end{cases}$$

The total time spent in calls to extend(j, k) where $k > 1$ is $O(gk^{\omega-1})$. The total time spent in calls to extend$(j, 1)$ is $O(g^2 kn \log n)$. Therefore, $T(k) = O(gk^{\omega-1} + g^2 kn \log n)$. The running time of our minimum homology basis algorithm[4] is $T(O(n)) = O(g^\omega + g^3 n \log n) = O(g^3 n \log n)$.

6.1 Cyclic double cover

In order to compute minimum homology basis cycle γ_j, we lift the graph into a *covering space* known as the *cyclic double cover*. Our presentation of the cyclic double cover is similar to that of Erickson [13]. Erickson describes the cyclic double cover relative to a single simple non-separating cycle; however, we describe it relative to an arbitrary *set* of non-separating *co-cycles* determined by a support vector S, similar to the homology cover construction of Erickson and Nayyeri [14].

Let S be a $2g$-bit support vector for the minimum homology basis problem as defined above. We define the cyclic double cover relative to S using a standard *voltage construction* [19, Chapter 4]. Let G_S^2 be the graph whose vertices are pairs (v, z), where v is a vertex of G and z is a bit. The edges of G_S^2 are ordered pairs $(uv, z) := (u, z)(v, z \oplus \langle S, [uv]_h \rangle)$ for all edges uv of G and bits z. Let $\pi : G_S^2 \to G$ denote the *covering map* $\pi(v, z) = v$. The *projection* of any vertex, edge, or path in G_S^2 is the natural map to G induced by π. We say a vertex, edge, or path p in G *lifts* to p' if p if the projection of p'. A closed path in G_S^2 is defined to be a face of G_S^2 if and only if its projection with regard to π is the boundary of a face of G. This construction defines an embedding of G_S^2 onto a surface Σ_S^2 (we will prove G_S^2 and Σ_S^2 are connected shortly).

We can also define G_S^2 in a more topologically intuitive way as follows. Let Ψ be a set of co-cycles which contains each co-cycle p_i for which the ith bit of S is equal to 1. Let Σ' be

[4] Our minimum homology basis algorithm can be simplified somewhat by having extend(j, k) recurse on extend$(j, 1)$ and extend$(j + 1, k - 1)$ and by using a simpler algorithm for update(j, k). This change will increase the time spent in calls to extend(j, k) where $k > 1$, but the time taken by calls with $k = 1$ will still be a bottleneck on the overall run time.

the surface obtained by cutting Σ along each cycle of Ψ. Note that Σ' may be disconnected. Each co-cycle $p_i \in \Psi$ appears as two copies on the boundary of Σ' denoted p_i^- and p_i^+ (note that p_i^- and p_i^+ may themselves be broken into multiple components if p_i crosses other co-cycles of Ψ). Create two copies of Σ' denoted $(\Sigma', 0)$ and $(\Sigma', 1)$, and let (p_i^-, z) and (p_i^+, z) denote the copies of p_i^- and p_i^+ in surface (Σ', z). For each co-cycle $p_i \in \Psi$, we identify $(p_i^+, 0)$ with $(p_i^-, 1)$ and we identify $(p_i^+, 1)$ with $(p_i^-, 0)$, creating the surface Σ_S^2 and the graph G_S^2 embedded on Σ_S^2. See Figure 3.

The first three of the following lemmas are immediate.

▶ **Lemma 14.** *Let γ be any simple cycle in G, and let s be any vertex of γ. Then γ is the projection of a unique path in G_S^2 from $(s, 0)$ to $(s, \langle S, [\gamma]_h \rangle)$.*

▶ **Lemma 15.** *Every lift of shortest path in G is a shortest path in G_S^2.*

▶ **Lemma 16.** *Let γ be the minimum weight simple cycle of G such that $\langle S, [\gamma]_h \rangle = 1$, and let s be any vertex of γ. Then γ is the projection of the shortest path in G_S^2 from $(s, 0)$ to $(s, 1)$.*

▶ **Lemma 17.** *The cyclic double cover G_S^2 is connected.*

Proof. There exists some simple cycle γ in G such that $\langle S, [\gamma]_h \rangle = 1$. Let s be any vertex of γ. Let v be any vertex of G. We show there exists a path from (v, z) to $(s, 0)$ in G_S^2 for both bits z. By Lemma 14, there is a path in G_S^2 from $(s, 0)$ to $(s, 1)$. There exists a path from v to s in G so there is a path from (v, z) to one of $(s, 0)$ or $(s, 1)$ in G_S^2. The other of $(s, 0)$ or $(s, 1)$ may be reached by following the lift of γ. ◀

Observe that G_S^2 has $2n$ vertices and $2m$ edges. Each co-cycle p_i shares an even number of edges with each face f of G. By Lemma 14, both lifts of f to G_S^2 are cycles; in particular both lifts are faces. We conclude G_S^2 contains 2ℓ faces. Surface Σ_S^2 has Euler characteristic $2n - 2m + 2\ell = 2\chi = 2 - 2(2g - 1)$. The genus of G_S^2 is exactly $2g - 1$.

6.2 Selecting homology basis cycles

Let S be any $2g$-bit support vector. We now describe our algorithm to select the minimum weight cycle γ such that $\langle S, [\gamma]_h \rangle = 1$. Our algorithm is based on one by Erickson and Nayyeri [14] for computing minimum weight cycles in arbitrary homology classes, except we use the cyclic double cover instead of their $\mathbb{Z}_2$-homology cover. We begin by computing a *greedy* tree-cotree decomposition (T^*, L^*, C^*) of G where T^* is a shortest path tree of an arbitrary vertex r and C^* is an arbitrary co-tree. Again, set L^* contains $2g$ edges $\{u_1v_1, \ldots, u_{2g}v_{2g}\}$. Let $\Lambda = \{\lambda_1, \ldots, \lambda_{2g}\}$ be the set of $2g$ loops where $\lambda_i = \sigma(r, u_1) \cdot u_1v_1 \cdot \sigma(v_1, r)$. Set Λ is called a *greedy system of loops*; cutting along each loop of Λ unfolds Σ into a topological disk [15]. Minimum homology basis cycle γ is non-separating; therefore, it crosses some loop of Λ at least once. Our algorithm constructs set Λ in $O(n \log n + gn)$ time[5]. Let G_S^2 be the cyclic double cover of G with regard to S. Our algorithm constructs G_S^2 in $O(gn)$ time.

Suppose our desired cycle γ crosses loop $\lambda_i \in \Lambda$. Simple cycle γ intersects one of two shortest paths, $\sigma(r, u_i)$ or $\sigma(v_i, r)$. Without loss of generality, it intersections $\sigma(r, u_i)$ at some vertex s. By Lemma 16, simple cycle γ is the projection of the shortest path in G_S^2 from $(s, 0)$ to $(s, 1)$. Let $\hat{\gamma}$ be this shortest path in G_S^2. Let $\hat{\sigma}$ be the lift of $\sigma(r, u_i)$ to G_S^2

[5] We only need to construct Λ once for the entire minimum homology basis algorithm, but constructing it once per basis cycle does not affect the overall run time.

that contains vertex $(s, 0)$. By Lemma 15, path $\hat{\sigma}$ is also a shortest path in G_S^2. If $\hat{\gamma}$ uses any other vertex (v, z) of $\hat{\sigma}$ other than $(s, 0)$, then it can use the entire subpath of $\hat{\sigma}$ between $(s, 0)$ and (v, z).

Now, consider the surface $\Sigma_S^2 \not\downarrow \hat{\sigma}$ which contains a single face bounded by two copies of $\hat{\sigma}$ we denote $\hat{\sigma}^-$ and $\hat{\sigma}^+$. For each vertex (v, z) on $\hat{\sigma}$, let $(v, z)^-$ and $(v, z)^+$ denote its two copies on $\hat{\sigma}^-$ and $\hat{\sigma}^+$ respectively. From the above discussion, we see $\hat{\gamma}$ is a shortest path in $\Sigma_S^2 \not\downarrow \hat{\sigma}$ from one of $(s, 0)^-$ or $(s, 0)^+$ to $(s, 1)$.

To find γ, we use the following generalization of Klein's [25] multiple-source shortest path algorithm:

▶ **Lemma 18** (Cabello *et al.* [7]). *Let G be a graph with n vertices, cellularly embedded in a surface of genus g, and let f be any face of G. We can preprocess G in $O(gn \log n)$ time and $O(n)$ space so that the length of the shortest path from any vertex incident to f to any other vertex can be retrieved in $O(\log n)$ time.*

Our algorithm iterates over the $O(g)$ shortest paths present in loops of Λ. For each such path σ, it computes a lift $\hat{\sigma}$ in G_S^2, cuts Σ_G^2 along $\hat{\sigma}$, and runs the multiple-source shortest path procedure of Lemma 18 to find the shortest path from some vertex $(s, z)^{\pm}$ on $\hat{\sigma}^{\pm}$ to $(s, z \oplus 1)$. Each shortest path it finds projects to a closed path γ' such that $\langle S, [\gamma']_h \rangle = 1$. By the above discussion, the shortest such projection can be chosen for γ. Running the multiple-source shortest path procedure $O(g)$ times on a graph of genus $O(g)$ takes $O(g^2 n \log n)$ time total. We conclude the discussion of our minimum weight homology basis algorithm.

▶ **Lemma 19.** *Let G be a graph with n vertices cellulary embedded in a surface of genus g. For any $2g$-bit support vector S we can compute the minimum weight cycle γ such that $\langle S, \gamma \rangle = 1$ in $O(g^2 n \log n)$ time.*

▶ **Theorem 20.** *Let G be a graph with n vertices, cellularly embedded in an orientable surface of genus g. We can compute a minimum weight homology basis of G in $O(g^3 n \log n)$ time.*

References

1. L. Aleksandrov and H. Djidjev. Linear algorithms for partitioning embedded graphs of bounded genus. *SIAM J. of Disc. Math.*, 9(1):129–150, 1996.
2. Edoardo Amaldi, Claudio Iuliano, Tomasz Jurkiewicz, Kurt Mehlhorn, and Romeo Rizzi. Breaking the $O(m^2 n)$ barrier for minimum cycle bases. In *Proc. 17th Ann. Euro. Symp. Algo.*, pages 301–312, 2009.
3. Franziska Berger, Peter Gritzmann, and Sven de Vries. Minimum cycle bases for network graphs. *Algorithmica*, 40(1):51–62, 2004.
4. G. Borradaile, P. Sankowski, and C. Wulff-Nilsen. Min st-cut oracle for planar graphs with near-linear preprocessing time. *ACM Trans. Algo.*, 11(3):16, 2015.
5. Glencora Borradaile, David Eppstein, Amir Nayyeri, and Christian Wulff-Nilsen. All-pairs minimum cuts in near-linear time for surface-embedded graphs. In *Proc. 32nd Ann. Int. Symp. Comput. Geom.*, 2016.
6. Oleksiy Busaryev, Sergio Cabello, Chao Chen, Tamal K. Dey, and Yusu Wang. Annotating simplicies with a homology basis and its applications. In *Proc. 13th Scandinavian Workshop on Algo. Theory*, pages 189–200, 2012.
7. Sergio Cabello, Erin W. Chambers, and Jeff Erickson. Multiple-source shortest paths in embedded graphs. *SIAM J. Comput.*, 42(4):1542–1571, 2013.

8 A. C. Cassell, J. C. de Henderson, and K. Ramachandran. Cycle bases of minimal measure for the structural analysis of skeletal structures by the flexibility method. *Proc. R. Soc. Lond. Ser. A*, 350:61–70, 1976.

9 L. O. Chua and Li-Kuan Chen. On optimally sparse cycle and coboundary basis for a linear graph. *IEEE Trans. Circuit Theory*, 20:495–503, 1973.

10 José Coelho de Pina. *Applications of Shortest Path Methods.* PhD thesis, University of Amsterdam, 1995.

11 Tamal K. Dey, Jian Sun, and Yusu Wang. Approximating loops in a shortest homology basis from point data. In *Proc. 26th Ann. Symp. Comput. Geom.*, pages 166–175, 2010.

12 David Eppstein. Dynamic generators of topologically embedded graphs. In *Proc. 14th Ann. ACM-SIAM Symp. Disc. Algo.*, pages 599–608, 2003.

13 Jeff Erickson. Shortest non-trivial cycles in directed surface graphs. In *Proc. 27th Ann. Symp. Comput. Geom.*, pages 236–243, 2011.

14 Jeff Erickson and Amir Nayyeri. Minimum cuts and shortest non-separating cycles via homology covers. In *Proc. 22nd Ann. ACM-SIAM Symp. Disc. Algo.*, pages 1166–1176, 2011.

15 Jeff Erickson and Kim Whittlesey. Greedy optimal homotopy and homology generators. In *Proc. 16th Ann. ACM-SIAM Symp. on Disc. Algo.*, pages 1038–1046, 2005.

16 Kyle Fox. Shortest non-trivial cycles in directed and undirected surface graphs. In *Proc. 24th Ann. ACM-SIAM Symp. Disc. Algo.*, pages 352–364, 2013.

17 Alexander Golynski and Joseph D. Horton. A polynomial time algorithm to find the minimum cycle basis of a regular matroid. In *Proc. 8th Scandinavian Workshop on Algo. Theory*, pages 200–209, 2002.

18 R. E. Gomory and T. C. Hu. Multi-terminal network flows. *J. SIAM*, 9(4):551–570, 1961.

19 Jonathan L. Gross and Thomas W. Tucker. *Topological graph theory.* Dover Publications, 2001.

20 D. Hartvigsen and R. Mardon. The all-pairs min cut problem and the minimum cycle basis problem on planar graphs. *SIAM J. Disc. Math.*, 7(3):403–418, 1994.

21 J. D. Horton. A polynomial-time algorithm to find the shortest cycle basis of a graph. *SIAM J. Comput.*, 16(2):358–366, 1987.

22 Giuseppe F. Italiano, Yahav Nussbaum, Piotr Sankowski, and Christian Wulff-Nilsen. Improved algorithms for min cut and max flow in undirected planar graphs. In *Proc. 43rd Ann. ACM Symp. Theory Comput.*, pages 313–322, 2011.

23 Telikepalli Kavitha, Kurt Mehlhorn, Dimitrios Michail, and Katarzyna E. Paluch. An $\tilde{O}(m^2n)$ algorithm for minimum cycle basis of graphs. *Algorithmica*, 52(3):333–349, 2008.

24 G. Kirchhoff. Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Poggendorf Ann. Physik*, 72:497–508, 1847. English transl. in Trans. Inst. Radio Engrs. CT-5 (1958), pp. 4–7.

25 Philip Klein. Multiple-source shortest paths in planar graphs. In *Proc. 16th Ann. ACM-SIAM Symp. Disc. Algo.*, pages 146–155, 2005.

26 Donald E. Knuth. *The Art of Computer Programming*, volume 1. Addison-Wesley, 1968.

27 K. Mehlhorn and D. Michail. Minimum cycle bases: Faster and simpler. *ACM Trans. Algo.*, 6(1):8, 2009.

28 S. Tazari and M. Müller-Hannemann. Shortest paths in linear time on minor-closed graph classes with an application to Steiner tree approximation. *Disc. Applied Math.*, 157(4):673–684, 2009.

29 Geetika Tewari, Craig Gotsman, and Steven J. Gortler. Meshing genus-1 point clouds using discrete one-forms. *Comput. Graph.*, 30(6):917–926, 2006.

30 C. Wulff-Nilsen. Minimum cycle basis and all-pairs min cut of a planar graph in subquadratic time. Technical Report arXiv:0912.1208, University of Copenhagen, 2009.