Shortest Cut Graph of a Surface with Prescribed Vertex Set

Éric Colin de Verdière*

Laboratoire d'informatique, École normale supérieure, CNRS, Paris, France
Email: Eric.Colin.de.Verdiere@ens.fr

Abstract. We describe a simple greedy algorithm whose input is a set P of vertices on a combinatorial surface $\mathscr{S}$ without boundary and that computes a shortest cut graph of $\mathscr{S}$ with vertex set P. (A cut graph is an embedded graph whose removal leaves a single topological disk.) If $\mathscr{S}$ has genus g and complexity n, the running-time is $O(n \log n + (g + |P|)n)$. This is an extension of an algorithm by Erickson and Whittlesey [*Proc. ACM-SIAM Symp. on Discrete Algorithms*, 1038–1046 (2005)], which computes a shortest cut graph with a single given vertex. Moreover, our proof is simpler and also reveals that the algorithm actually computes a minimum-weight basis of some matroid.

1 Introduction

A basic tool to compute with topological surfaces is the notion of *decomposition*: cutting a surface $\mathscr{S}$ into topologically simpler elements, so that $\mathscr{S}$ is entirely described by how these elements are assembled together. In the last decade, algorithms for computing several types of decompositions have been given. The most important type of decomposition is the notion of *cut graph*: a graph G embedded (drawn without crossing) on $\mathscr{S}$ such that $\mathscr{S} \setminus G$ is a topological disk.

When the surface is endowed with a metric, it is an asset to be able to compute a *shortest* decomposition of some kind. From a theoretical perspective, this provides a "canonical" decomposition of the surface. For applications, like parameterization or texture mapping, it is often necessary to make the surface planar, by cutting it along curves. The way the curves are chosen affects the quality of the parameterization; so shortest curves are desirable (where the "metric" can also be adapted to favor cutting along areas of high curvature, for example). See, e.g., the discussion by Erickson and Har-Peled [13] and references therein.

In this paper, we are interested in computing a shortest cut graph of a surface with a given vertex set. Before presenting our result, we survey related works.

* Supported by the *Agence Nationale de la Recherche* under the *Triangles* project of the *Programme blanc* ANR-07-BLAN-0319.

1.1 Previous Work

Most of the previous works in the algorithmic study of curves on surfaces consider the *combinatorial surface model* [8, 9], where the input surface $\mathscr{S}$ comes with a representation of a fixed graph embedding M. By definition, each edge of each subsequent graph embedding G is required to be a walk in M; however, these walks may share edges and vertices of M: it is only required that G be the limit of a sequence of real graph embeddings on the surface. Each edge of M bears a non-negative weight, used to measure the length of the graph embeddings. (See Section 2.4 for a more detailed description of this model.)

Erickson and Har-Peled [13] are the first to study the problem of computing a shortest cut graph of a surface. They prove that the problem is NP-hard, by reduction to a minimum Steiner tree problem, and provide a polynomial-time approximation algorithm.

So far, the only known algorithm to compute a shortest decomposition of a surface is that of Erickson and Whittlesey [14]. Specifically, a *system of loops* of a surface $\mathscr{S}$ without boundary is a cut graph G with a single vertex, its *basepoint*. Relying on ideas by Eppstein on tree-cotree decompositions [12], Erickson and Whittlesey provide a greedy algorithm to compute a shortest system of loops with a given basepoint on an orientable surface without boundary, with running-time $O(n \log n + gn)$, where n is the complexity of the combinatorial surface and g is its genus. (Actually, $O(n \log n)$ is sufficient to compute an implicit description of the system of loops, which has $O(gn)$ complexity.) In contrast, it is striking that no algorithm has been found to compute, even approximately, shortest decompositions of other kinds, like canonical systems of loops [18], pants decompositions [10], or octagonal decompositions [8].

The algorithm by Erickson and Whittlesey [14] has been used as a subroutine in other problems, like computing shortest non-contractible or non-separating cycles [2, 17]. Two other structural properties of the shortest system of loops prove useful in this context: each of the loops is as short as possible in its homotopy class, and is the concatenation of two shortest paths plus an edge.

1.2 Our Result

There is a natural generalization of this greedy algorithm which, instead of a system of loops, computes a cut graph whose vertex set is *exactly* a prescribed set of k points. The algorithm runs in $O(n \log n)$ time plus the output size, which is $O((g + k)n)$. In this paper, we prove that this generalization (on a possibly non-orientable surface) computes a shortest cut graph on $\mathscr{S}$ with that vertex set. This should be put in perspective with the NP-hardness of computing a shortest cut graph of a surface [13]: our result implies that this is easy once the vertex set has been "guessed".

Compared to the paper by Erickson and Whittlesey [14], our proof is also simpler and more natural. In particular, to prove that a greedy algorithm is optimal, a generic strategy is to express the underlying combinatorial structure as a matroid. Erickson and Whittlesey show that the partial homotopy bases and

the partial systems of loops do not form matroids, so they prove optimality of their result by more indirect means. We show, however, that their algorithm and its generalization fit within the matroid framework: they compute a minimum-weight independent set in some matroid, and that independent set turns out to be a shortest cut graph.

The main additional tool we need to introduce for our extension is *relative homology*, which provides an algebraic condition on whether a given set of paths separates the surface. We give an ad-hoc, self-contained description of relative homology that is sufficient for our purposes. In contrast, for the shortest system of loops [14], only standard homology was required.

When $\mathscr{S}$ has at least one boundary component, a variant of the problem is the computation of a shortest *system of arcs*: a family of simple, pairwise disjoint paths whose endpoints are on the boundary of the surface and that cut $\mathscr{S}$ into a disk. Our result implies that we can also compute a shortest system of arcs. While such systems of arcs have already been described and used for various purposes, such as computing shortest splitting cycles [5], shortest curves with prescribed homotopy [8], or minimum cuts [4], it was not known that they were as short as possible.

1.3 Overview

At a high level, the algorithm for computing a shortest cut graph with prescribed vertex set P of a surface $\mathscr{S}$ is simple enough to be presented here. We begin by growing disks around all vertices of P simultaneously, and consider the curves on which these disks collide. This is essentially the *Voronoi* diagram of P, except that both sides of a given "Voronoi" edge may be incident to the same cell (Figure 1(a)). Define the *weight* of a "Voronoi" edge to be the length of the dual "Delaunay" edge (the shortest path crossing that "Voronoi" edge and connecting the two points in P on either side of it). Compute a *maximum* spanning tree of the "Voronoi" diagram with respect to these weights (Figure 1(b)). Return the "Delaunay" edges whose dual "Voronoi" edges are not in that maximum spanning tree (Figure 1(c),(d)).

The rest of the paper is organized as follows. We describe some necessary background on matroids and topology of surfaces in Section 2. Then we state and prove our main result on shortest cut graphs with prescribed vertex set (Section 3). Finally, we discuss a few extensions.

2 Preliminaries

2.1 Matroids and Greedy Algorithms

A *matroid* is a pair (S, Σ) where S is a finite set and Σ is a non-empty collection of subsets of S satisfying the two following conditions: (1) if $I \in \Sigma$ and $J \subseteq I$, then $J \in \Sigma$; (2) if $I, J \in \Sigma$ and $|I| < |J|$, then $I \cup \{z\} \in \Sigma$ for some $z \in J \setminus I$. For example, if S is a finite vector space and Σ is the collection of all linearly

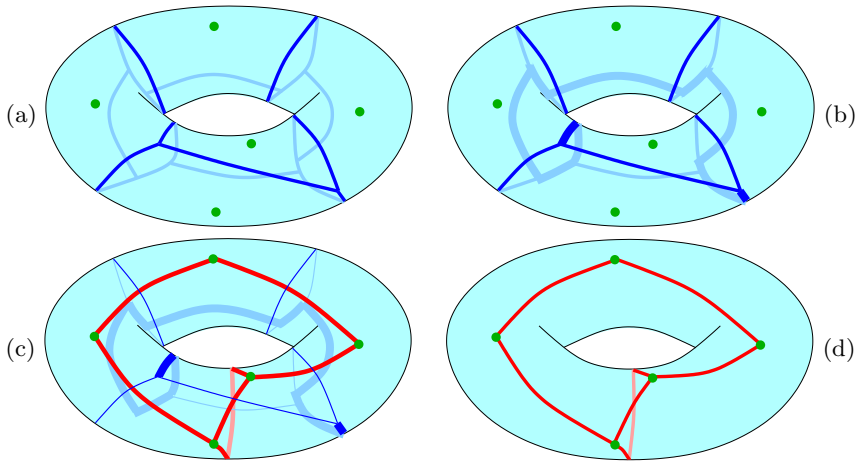

Fig. 1. Overview of the algorithm, on a torus. (a): The vertex set P and its "Voronoi" diagram; light curves lie on the back of the surface. (b): In bold, a maximum spanning tree (in this case, a path) of that "Voronoi" diagram, where the weight of an edge is the length of the dual "Delaunay" edge. Most of the path is on the back of the torus. (c): Here, the "Voronoi" edges not in the maximum spanning tree are replaced by their dual "Delaunay" edges. (d): Those "Delaunay" edges form the shortest cut graph with vertex set P.

independent subsets of S, then (S, Σ) is a matroid. We refer to the elements in Σ as the *independent sets*. A *basis* of (S, Σ) is an inclusionwise maximal independent set. By (2), all bases have the same cardinality.

Consider a weight function $w : S \to \mathbb{R}$. For every subset I of S, define $w(I) = \sum_{z \in I} w(z)$. It is well-known [11, Sect. 16.4] that the following *greedy algorithm* finds a basis I of (S, Σ) minimizing $w(I)$: maintain an initially empty independent set I; repeatedly choose $y \in S \setminus I$ with $I \cup \{y\}$ in Σ and with $w(y)$ as small as possible; stop when no such y exists.

2.2 Surfaces and Embeddings

We recall here standard definitions on topology of surfaces. For general background on topology, see for example Stillwell [19] or Hatcher [16].

In this paper, unless noted otherwise, $\mathscr{S}$ is a compact, connected surface *without boundary*; g denotes its Euler genus. Thus, if $\mathscr{S}$ is orientable, $g \geq 0$ is even, and $\mathscr{S}$ is a sphere with $g/2$ handles attached; if $\mathscr{S}$ is non-orientable, $\mathscr{S}$ is a sphere with $g \geq 1$ disks replaced by Möbius strips.

We consider *curves* drawn on $\mathscr{S}$. A *path* p is a continuous map from $[0, 1]$ to $\mathscr{S}$; it is *simple* if it is one-to-one, except that its endpoints $p(0)$ and $p(1)$ may coincide. Similarly, we say that two paths are *disjoint* if their images are disjoint, except that some of their endpoints may coincide. A *loop* with *basepoint* x is a

path with both endpoints equal to x. A *cycle* is a continuous map from the unit circle S^1 to $\mathscr{S}$. We often identify a curve with its image on $\mathscr{S}$.

Two paths p and q are *homotopic* if, informally, there is a continuous deformation between them that keeps their endpoints fixed. More precisely, a *homotopy* is a continuous map $h : [0,1] \times [0,1] \to \mathscr{S}$ such that $h(0,\cdot) = p$, $h(1,\cdot) = q$, and such that $h(\cdot,0)$ and $h(\cdot,1)$ are constant maps. Similarly, a *homotopy* between two cycles γ and δ is a continuous map $h : [0,1] \times S^1$ such that $h(0,\cdot) = \gamma$ and $h(1,\cdot) = \delta$. A loop or cycle homotopic to a constant loop or cycle is *contractible*.

An *embedding* of a graph G on $\mathscr{S}$ is a "crossing-free" drawing of G on $\mathscr{S}$: it maps the vertices of G to distinct points on $\mathscr{S}$ and the edges of G to simple paths on $\mathscr{S}$ that intersect only at common endpoints. Often, we will identify the graph G with its embedding on $\mathscr{S}$. A *face* of an embedding of G on $\mathscr{S}$ is a connected component of $\mathscr{S}$ minus (the image of) G. A graph is *cellularly embedded* on $\mathscr{S}$ if every face of the graph is an open disk. Given a graph embedding G, we denote by $\mathscr{S}\!\setminus\!\setminus G$ the surface $\mathscr{S}$ cut along G. In particular, if G is cellularly embedded, then $\mathscr{S}\!\setminus\!\setminus G$ is a disjoint union of closed disks.

A *cut graph* of $\mathscr{S}$ is a graph G embedded on $\mathscr{S}$ whose unique face is a disk (sometimes called a *polygonal schema* of $\mathscr{S}$). The proof of the following lemma is omitted (the first point follows from Euler's formula and the second one from the classification of surfaces with boundary):

Lemma 1. *The following properties hold:*

i. A cut graph with k vertices has $g + k - 1$ edges.
ii. Let G be a graph embedded on $\mathscr{S}$. Then every face of G is a disk if and only if no edge can be added to G without increasing its number of faces or creating a new vertex.

2.3 Homology

In this paper, we will use *1-dimensional singular homology for graphs embedded on surfaces without boundary, over $\mathbb{Z}/2\mathbb{Z}$, relatively to a finite set of points* [16, p. 115]. For convenience, we give a non-standard, self-contained presentation of this tool that is sufficient for our purposes. Note that our notion of homology differs from the one used in the previous papers in the area [1–3, 5, 6, 13, 14, 17]: here, *relative* homology is needed.

Let P be a finite set of points on $\mathscr{S}$. A *P-path* is a path intersecting P exactly at its endpoints. The definition of *non-oriented P-path* is similar, except that one P-path and its reversal are considered to be the same non-oriented P-path. If p is an arbitrary non-oriented P-path, we denote by $p^\flat$ and $p^\sharp$ the two oriented versions of P.

A *homology cycle* is a finite set of non-oriented P-paths. The homology cycles form a vector space over the field $\mathbb{Z}/2\mathbb{Z}$: addition is symmetric difference, multiplication by zero gives the empty set, and multiplication by one is the identity.

Let $(p_1, \ldots, p_m)$ be a sequence of non-oriented P-paths. Assume that there exists $(\tau_1, \ldots, \tau_m) \in \{\flat, \sharp\}^m$ such that the concatenation of $p_1^{\tau_1}, \ldots, p_m^{\tau_m}$ is a

contractible loop ℓ. Then $\{p_1\} + \cdots + \{p_m\}$ (the set of non-oriented P-paths appearing an odd number of times in $(p_1, \ldots, p_m)$) is called a *homology boundary*.

The homology boundaries form a vector subspace of the set of homology cycles. (Proof: Let S and S' be two homology boundaries. Let ℓ be a contractible loop as in the definition, witnessing the fact that S is a homology boundary, and similarly ℓ' for S'. Let q be a P-path from the basepoint of ℓ to the basepoint of ℓ'. Then the concatenation of ℓ, q, ℓ', and the reverse of q is a contractible loop witnessing the fact that the symmetric difference of S and S' is a homology boundary.)

Two homology cycles are *homologous* if their sum (symmetric difference) is a homology boundary. Thus, the homology cycles are partitioned into *homology classes*. Together, the homology classes form the *homology space*; formally, it is the vector space that is the quotient of the space of homology cycles Z by the space of homology boundaries B. Although Z and B have infinite dimension, it will turn out that the dimension of their quotient is finite.

Given a P-path p, we denote by $[p]$ its homology class (more precisely, the homology class of $\{\overline{p}\}$, where $\overline{p}$ is the non-oriented version of p). If p and p' are homotopic P-paths, then $[p] = [p']$: homology is a coarser relation than homotopy.

2.4 Endowing Surfaces with a Metric

Our algorithm needs a way to measure lengths of curves and graph embeddings. Like most earlier works in the area, our result is described in the *combinatorial surface* model [9,10]. Actually, most of the description below is in the equivalent, dual, *cross-metric setting* [8]. See Colin de Verdière and Erickson [8] for a more precise discussion, and also the connection with the informal description given in the introduction.

A *combinatorial surface* $(\mathscr{S}, M)$ is a surface $\mathscr{S}$ together with a fixed graph M cellularly embedded on $\mathscr{S}$; each edge of M has a non-negative *weight*. Let M^* be the *dual graph* of M, also embedded on $\mathscr{S}$: it has a vertex inside each face of M^*, and, for every edge e of M, an edge e^* that crosses e exactly once and no other edge of M. The weight of e^* equals, by definition, the weight of e. We only consider graph embeddings (in particular, curves) that are *regular* with respect to M^*; namely, every vertex of the embedding is in a face of M^*, and every edge of the embedding intersects the edges of M^* at finitely many points in their relative interior, where a crossing occurs. The *length* of a graph embedding on $(\mathscr{S}, M)$ is the sum of the weights of the edges of M^* crossed by the curve, counted with multiplicity. A graph embedding can be represented combinatorially by its *arrangement* with M^* on $\mathscr{S}$. The *complexity* of a combinatorial surface is the total number of vertices, edges, and faces of M (or, equivalently, M^*). In this paper, $(\mathscr{S}, M)$ is a combinatorial surface, and M^* is the dual graph of M.

3 Shortest Cut Graph with Prescribed Vertex Set

Our main result is the following:

Theorem 2. *Let $(\mathscr{S}, M)$ be a combinatorial surface, possibly non-orientable, without boundary. Let g be its genus and n be its complexity. Let P be a set of k vertices of M. We can compute a shortest cut graph of $(\mathscr{S}, M)$ with vertex set (exactly) P in $O(n \log n + (g+k)n)$ time.*

In particular, if $\mathscr{S}$ is orientable and P is a single vertex, this is the result by Erickson and Whittlesey [14, Theorem 3.9]. Again, we note that the $O((g+k)n)$ term is present because it is the worst-case complexity of the output, but an implicit representation of the solution can be found in $O(n \log n)$ time.

3.1 Homology Bases and Cut Graphs

In this paper, a *homology basis* is a set of P-paths $\{p_1, \dots, p_m\}$ such that $[p_1], \dots, [p_m]$ form a basis of the homology vector space of $\mathscr{S}$ with respect to P. (Note that this definition differs from the one by Erickson and Whittlesey [14].) We will see that the algorithm for Theorem 2 computes a shortest homology basis. The connection with cut graphs is the following.

Proposition 3. *Let G be a graph embedded on $\mathscr{S}$ whose vertices belong to P. The edges of G form a basis of the homology vector space if and only if G is a cut graph with vertex set P.*

This result is standard in the simpler case of Erickson and Whittlesey [14], and is indeed implicit in their work, but not in our case, where relative homology is used. The proof relies on the three following lemmas. The first two lemmas can be proved directly if one assumes the equivalence between simplicial and singular homology, but we provide alternate proofs that do not rely on this equivalence.

Lemma 4. *If G has at least two faces, then the edges of G are homologically dependent.*

Proof. Let f be an arbitrary face of G, and let S be the set of edges of G incident exactly once to f. Since G has at least two faces, the set S is non-empty. We next prove that S is a homology boundary, which concludes.

By Lemma 1(ii), we can find a (possibly empty) set $\{p_1, \dots, p_m\}$ of pairwise disjoint simple P-paths such that $f \setminus\setminus \{p_1, \dots, p_m\}$ is a disk. In other words, if we add to the edge set of G the P-paths $p_1, \dots, p_m$, obtaining a new graph G' with vertex set in P, then f corresponds to a single face f' of G' that is a disk. We may follow the boundary of f', recording the P-paths in order along that boundary. This boundary is contractible since f' is a disk. Furthermore, the set of non-oriented P-paths appearing an odd number of times on the boundary of f' is exactly S, so S is a homology boundary. □

Lemma 5. *If G has a single face, then the edges of G are homologically independent.*

Proof. Assume that a non-empty subset E' of edges of G forms a homology boundary: some P-paths can be concatenated to form a contractible loop ℓ, such that E' is exactly the set of non-oriented P-paths appearing an odd number of times in this concatenation. We actually view ℓ as a contractible cycle γ. Let e be a fixed edge of G; we shall prove that e does not belong to E'. Since this is valid for every e, this proves $E' = \emptyset$, a contradiction.

Because G has a single face, there exists a cycle δ that crosses e once, and crosses no other edge of G. Without loss of generality, we may assume that γ and δ meet at a finite number of points, where they actually cross. It is well-known [15, p. 79] that the parity of the number of intersection points between γ and δ depends only on the homotopy classes of γ and δ: intuitively, changing γ continuously only creates or removes pairs of intersection points with δ. Since γ is contractible, γ must thus cross δ an even number of times.

Recall that E' is the set of non-oriented P-paths appearing an odd number of times in γ. Hence, the P-paths in E' cross δ an even number of times in total. However, the only edge of E' crossing δ is e, since E' is a subset of edges of G. Thus e crosses δ an even number of times, and does not belong to E'. □

Lemma 6. *If G is a cut graph with vertex set P, then the edges of G generate the homology vector space.*

Proof. It suffices to prove that every P-path p is homotopic to the concatenation of some edges of G. We can assume that p crosses G regularly. So p is the concatenation of paths $p_1, \ldots, p_m$ intersecting G exactly at their endpoints. (Recall that every point in P is a vertex of G.) Since $\mathscr{S} \backslash\backslash G$ is a disk, every path p_i is homotopic to a path in the image of G. Thus p is homotopic to a path in the image of G, and therefore to a concatenation of edges of G. □

Proof of Proposition 3. By Lemmas 5 and 6, if G is a cut graph with vertex set P, then its edges form a basis of the homology vector space. Conversely, assume that the edges of G form a basis of the homology vector space; then G has a single face by Lemma 4. If G is not a cut graph with vertex set P, then an edge can be added to G while keeping $\mathscr{S} \backslash\backslash G$ connected, but without changing its vertex set (Lemma 1(ii)); these edges are homologically independent (Lemma 5). So the edges of G did not form an inclusionwise maximal homologically independent set; this is a contradiction. □

3.2 Algorithm (A): Shortest Homology Basis

The algorithm for Theorem 2 can be formulated in the following abstract form (let us call it (A)): Maintain an initially empty set I of non-oriented P-paths. At each step, add some shortest P-path in I homologically independent with the P-paths already in I. Stop when no such path exists.

Proposition 7. *(A) computes a shortest homology basis.*

Proof. Let S be the set of all homology classes, and Σ be the set of subsets of S that are linearly independent. Define the weight of $s \in S$ to be the minimum length of a P-path in s. (Of course, only P-paths that are regular with respect to M^* are considered. Furthermore, the weight is ∞ if there is no P-path in s.)

Consider the following greedy algorithm: Let I be an initially empty set of homology classes. Iteratively add to I a minimum-weight homology class linearly independent with those already in I; stop when no such homology class exists. Since S is a vector space of finite dimension (this follows from Proposition 3), (S, Σ) is a matroid, and this algorithm gives a minimum-weight basis of the homology vector space.

The regular P-paths redundantly generate the homology vector space, so this algorithm only chooses homology classes of P-paths. It therefore coincides with the algorithm (A). □

3.3 Algorithm (A′): Shortest Cut Graph

In this section, we prove that, under an additional restriction, the algorithm (A) computes a shortest cut graph with vertex set P.

Let F be a spanning forest of shortest paths in M, starting from every vertex of P simultaneously. In other words, each connected component of F is a tree containing exactly one vertex of P, and F contains a shortest path from every vertex of M to its nearest vertex in P. Let C^* be the graph obtained from M^* by removing each edge e^* whose primal edge e belongs to F; this graph C^* is the "Voronoi" diagram we alluded to in the introduction.

Lemma 8. *$\mathscr{S}\backslash\backslash C^*$ is a set of closed disks, each containing a single vertex in P.*

Proof. F contains all vertices of M, and thus $\mathscr{S}\backslash\backslash C^*$ is obtained by gluing the faces of M^* along the edges of F^*. For each tree in the forest F, we attach the corresponding faces of M^* according to this tree. Since attaching disks together in a tree-like fashion gives a disk, we indeed obtain that $\mathscr{S}\backslash\backslash C^*$ is a set of closed disks. Furthermore, every such disk contains a point in P. □

For each edge e^* in C^*, let $\sigma(e^*)$ be a shortest P-path crossing e^* exactly once and no other edge of C^*. Thus, each part of $\sigma(e^*)$ on either side of its crossing with e^* is a shortest path. So we can, without loss of generality, assume that all the paths $\sigma(e^*)$ are simple and pairwise disjoint. A P-path is *primitive* if it is of the form $\sigma(e^*)$ for some edge e^* of C^*.

The following proposition proves a structural property on the P-paths chosen by the algorithm. Similar arguments have been used in recent papers [1, 2, 14] while the core of the idea, the *3-path condition*, is older [20].

Proposition 9. *At each step, we may assume that (A) chooses primitive paths.*

Proof. Every P-path computed by (A) has to cross C^* at least once, for otherwise it is contractible and thus homologically trivial. Assume some path p crossing edges $e_1^*, \ldots, e_m^*$ of C^* is chosen at some step of the algorithm (A). This

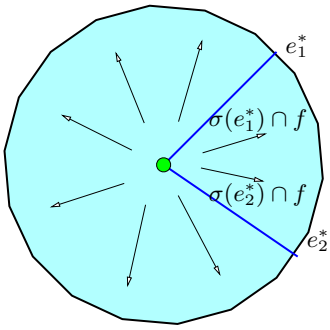

Fig. 2. The retraction in the proof of Lemma 11. In this example, $E^* = \{e_1^*, e_2^*\}$.

path is homotopic to the concatenation of $\sigma(e_1^*), \ldots, \sigma(e_m^*)$, since the faces of C^* are disks. Thus $[p] = [\sigma(e_1^*)] + \ldots + [\sigma(e_m^*)]$. Since p is homologically independent with the already chosen P-paths, one of the $\sigma(e_i^*)$ must be homologically independent with these P-paths. Moreover, $\sigma(e_i^*)$ is no longer than p, since p crosses e_i^* and since $\sigma(e_i^*)$ is a shortest P-path among those crossing e_i^*. Therefore, $\sigma(e_i^*)$ can be chosen by (A). □

We call (A′) the version of algorithm (A) that always chooses primitive paths. In particular, (A′) chooses P-paths that are simple and pairwise disjoint.

Corollary 10. *Algorithm (A′) computes a shortest cut graph with vertex set P.*

Proof. Since the P-paths chosen by (A′) are simple and pairwise disjoint, (A′) computes a homology basis that is a graph with vertex set included in P, and thus a cut graph with vertex set P (Proposition 3). Also by Proposition 3, any shorter cut graph with vertex set P would be a shorter homology basis, which is impossible (Proposition 7). □

3.4 Algorithm (A″): Concrete Algorithm

In this section, we describe an algorithm (A″), already sketched in the introduction, that will be shown to give the same result as (A′) and that can be efficiently implemented. The following lemma was noted and used by Erickson and Whittlesey [14, Section 3.4] in the particular case where P is a single vertex.

Lemma 11. *Let E^* be a set of edges of C^*. Then $C^* - E^*$ is connected if and only if $\mathscr{S} \setminus (P \cup \sigma(E^*))$ is connected. ($\sigma(E^*)$ is the set $\{\sigma(e^*) \mid e^* \in E^*\}$.)*

Proof. We will actually show that $Y := C^* - E^*$ is a *deformation retract* of $X := \mathscr{S} \setminus (P \cup \sigma(E^*))$: there is a continuous map $\varphi : [0,1] \times X \to X$ such that $\varphi(0, \cdot)$ is the identity, $\varphi(1, \cdot)$ has its image in Y and is the identity when restricted to Y. This in particular implies the result.

For every edge e^* of C^*, let $\tau(e^*)$ be the intersection of e^* with $\sigma(e^*)$. Consider a face f of C^*. We can retract $f \setminus (P \cup \sigma(E^*))$ onto $(\partial f) \setminus \tau(E^*)$ (Figure 2). Gluing these retractions together, we obtain a deformation retract of $\mathscr{S} \setminus (P \cup \sigma(E^*))$ to $C^* \setminus \tau(E^*)$. This in turn retracts to $C^* - E^*$. □

(A″) can be described as follows: For every edge e^* of C^*, define the *weight* of e^* to be the length of the P-path $\sigma(e^*)$. Compute a *maximum* spanning tree T^* of C^* with respect to these weights. Return $\{\sigma(e^*) \mid e^* \in C^* \setminus T^*\}$.

Proof of Theorem 2. We prove that Algorithm (A″) is equivalent to (A′), which computes a shortest cut graph (Corollary 10). In light of Proposition 3, the algorithm (A′) can be rephrased as follows: Maintain an initially empty set of edges I^* of C^*; iteratively add to I^* the minimum-weight edge e^* of $C^* - I^*$ such that $\mathscr{S} \setminus \sigma(I^* \cup \{e^*\})$ is connected. This latter condition is equivalent to having $C^* - (I^* \cup \{e^*\})$ connected (Lemma 11). Thus (A′) iteratively removes minimum-weight edges of C^* while keeping it connected. No such edge belongs to a maximum spanning tree of C^* [11, Exercise 23.1-5], so (A′) computes a maximum spanning tree and is therefore equivalent to (A″).

There remains to analyze the complexity of (A″). The graph C^*, together with the weights of its edges, can be computed in $O(n \log n)$ time using Dijkstra's algorithm. The maximum spanning tree T^* can also be computed in $O(n \log n)$ time using any textbook algorithm [11, Ch. 23]. The edges in $C^* - T^*$ form an implicit description of the cut graph; to compute it explicitly, one has to build the paths $\sigma(e^*)$, for each edge $e^* \in C^* - T^*$. Each such P-path has $O(n)$ complexity. Furthermore, the number of P-paths is $g + k - 1$ (Lemma 1(i)). □

In particular, each edge of a shortest cut graph with vertex set P enjoys the following useful properties: it is a primitive P-path, and is as short as possible in its homology class (and therefore in its homotopy class).

4 Conclusion

We sketch a few extensions of our result. It is sometimes useful to cut a surface $\mathscr{S}$ with b boundaries into a disk along a *system of arcs*, whose endpoints are on the boundary of $\mathscr{S}$ [4, 5, 8]. It follows from Theorem 2 that the algorithm of these papers computes the shortest such cut graph. Roughly, it works as follows: Fill each boundary of $\mathscr{S}$ with a disk, obtaining a surface without boundary $\overline{\mathscr{S}}$; let P be a set of points, one inside each disk; compute a shortest cut graph with vertex set P on $\overline{\mathscr{S}}$, where crossing a boundary of $\mathscr{S}$ corresponds to a large (symbolically infinite) weight; the restriction of that cut graph to $\mathscr{S}$ is the shortest system of arcs. The running-time is $O(n \log n + (g + b)n)$.

As also noted by Erickson and Whittlesey [14], the algorithm extends to piecewise-linear (PL) surfaces, obtained by assembling Euclidean polygons: the running-time becomes $O(n^2 + (g+k)n)$, because an algorithm to compute shortest paths on such surfaces has to be used [7]. Since C^* is a graph embedded on $\mathscr{S}$ with k faces, Euler's formula implies that it has $O(g + k)$ edges (if we remove

iteratively vertices of degree one together with their incident edges, which belong to any spanning tree of C^* anyway, and vertices of degree two, merging the two incident edges). Since shortest paths on PL surfaces may overlap for some time and then diverge, the output of the algorithm is a set of *non-crossing paths*, which can be transformed into a graph embedding by an arbitrarily small perturbation. Actually, everything extends to the case of *Riemannian surfaces*, except, of course, that we cannot in general compute shortest paths and the graph C^*.

Acknowledgments. I would like to thank Jeff Erickson for helpful discussions, and Francis Lazarus for comments on a preliminary version of this paper.

References

1. Cabello, S., Colin de Verdière, É., Lazarus, F.: Finding shortest non-trivial cycles in directed graphs on surfaces. In: Proc. ACM Symp. on Computational Geometry. pp. 156–165 (2010)
2. Cabello, S., Colin de Verdière, É., Lazarus, F.: Output-sensitive algorithm for the edge-width of an embedded graph. In: Proc. ACM Symp. on Computational Geometry. pp. 147–155 (2010)
3. Cabello, S., Mohar, B.: Finding shortest non-separating and non-contractible cycles for topologically embedded graphs. Disc. Comput. Geom. 37(2), 213–235 (2007)
4. Chambers, E., Erickson, J., Nayyeri, A.: Minimum cuts and shortest homologous cycles. In: Proc. ACM Symp. on Computational Geometry. pp. 377–385 (2009)
5. Chambers, E.W., Colin de Verdière, É., Erickson, J., Lazarus, F., Whittlesey, K.: Splitting (complicated) surfaces is hard. Comput. Geom.: Theory Appl. 41(1–2), 94–110 (2008)
6. Chambers, E.W., Erickson, J., Nayyeri, A.: Homology flows, cohomology cuts. In: Proc. ACM Symp. on Theory of Computing. pp. 273–282 (2009)
7. Chen, J., Han, Y.: Shortest paths on a polyhedron. Int. J. Comput. Geom. Appl. 6, 127–144 (1996)
8. Colin de Verdière, É., Erickson, J.: Tightening non-simple paths and cycles on surfaces. In: Proc. ACM-SIAM Symp. on Discrete Algorithms. pp. 192–201 (2006)
9. Colin de Verdière, É., Lazarus, F.: Optimal system of loops on an orientable surface. Disc. Comput. Geom. 33(3), 507–534 (2005)
10. Colin de Verdière, É., Lazarus, F.: Optimal pants decompositions and shortest homotopic cycles on an orientable surface. J. ACM 54(4), Article No. 18 (2007)
11. Cormen, Th.H., Leiserson, Ch.E., Rivest, R.R., Stein, C.: Introduction to algorithms. MIT Press, Cambridge, MA (2001)
12. Eppstein, D.: Dynamic generators of topologically embedded graphs. In: Proc. ACM-SIAM Symp. on Discrete Algorithms. pp. 599–608 (2003)
13. Erickson, J., Har-Peled, S.: Optimally cutting a surface into a disk. Disc. Comput. Geom. 31(1), 37–59 (2004)
14. Erickson, J., Whittlesey, K.: Greedy optimal homotopy and homology generators. In: Proc. ACM-SIAM Symp. on Discrete Algorithms. pp. 1038–1046 (2005)
15. Guillemin, V., Pollack, A.: Differential topology. Prentice-Hall (1974)
16. Hatcher, A.: Algebraic topology. Cambridge University Press (2002), `http://www.math.cornell.edu/~hatcher/`

17. Kutz, M.: Computing shortest non-trivial cycles on orientable surfaces of bounded genus in almost linear time. In: Proc. ACM Symp. on Computational Geometry. pp. 430–438 (2006)
18. Lazarus, F., Pocchiola, M., Vegter, G., Verroust, A.: Computing a canonical polygonal schema of an orientable triangulated surface. In: Proc. ACM Symp. on Computational Geometry. pp. 80–89 (2001)
19. Stillwell, J.: Classical topology and combinatorial group theory. Springer-Verlag, second edn. (1993)
20. Thomassen, C.: Embeddings of graphs with no short noncontractible cycles. J. Comb. Theory, Series B 48(2), 155–177 (1990)