Computing the Shortest Essential Cycle*

Jeff Erickson

Department of Computer Science
University of Illinois at Urbana-Champaign

Pratik Worah†

Department of Computer Science
University of Chicago

Submitted to *Discrete & Computational Geometry* — July 29, 2009

Revised and resubmitted — January 6, 2010

Abstract

An essential cycle on a surface is a simple cycle that cannot be continuously deformed to a point or a single boundary. We describe algorithms to compute the shortest essential cycle in an orientable combinatorial surface in $O(n^2 \log n)$ time, or in $O(n \log n)$ time when both the genus and number of boundaries are fixed. Our results correct an error in a paper of Erickson and Har-Peled [*DCG* 2004].

1 Introduction

Cutting surfaces into topologically simpler components is a common technique in combinatorial and algorithmic topology. For example, algorithms that repeatedly cut a given surface along short, topologically nontrivial cycles have been used for removing topological noise from graphical models [21], finding short cut graphs for surface parametrization [18], computing shortest paths in a given homotopy class [12], approximating optimal traveling salesman tours in surface-embedded graphs [14], probabilistically embedding high-genus graphs into planar graphs [26, 2], drawing abstract graphs in the plane with the fewest possible crossings [28], and testing isomorphism between graphs of fixed genus [27]. These and other applications have motivated a series of algorithms for computing optimal cycles with various topological properties [35, 31, 18, 7, 4, 29, 5, 8, 6, 9].

Cutting a surface along non-contractible cycles decomposes the surface into components with genus zero, but those components may have an unbounded number of boundary cycles. Further simplifying those components requires cutting along *essential* cycles, which are simple cycles that cannot be continuously deformed either to a point or to a boundary cycle. Repeatedly cutting along essential cycles decomposes the surface into *pairs of pants*: surfaces with genus zero and three boundary cycles. Pants compositions are a standard tool in Riemannian geometry and low-dimensional topology; see, for example, [20, 23, 34]. Colin de Verdiére and Lazarus describe a polynomial-time algorithm to compute the shortest pants decomposition in a given homotopy class [13]. The algorithm of Colin de Verdiére and Erickson to compute the shortest path is a given homotopy class uses pants decompositions in its preprocessing phase [12]. Eppstein [17] and Poon and Thite [33] describe algorithms to approximate the shortest pants decomposition for the punctured plane.

The fastest algorithm known for computing shortest non-contractible cycles was described by Erickson and Har-Peled [18]. In the same paper, the authors claim that a simple modification of their algorithm

*Research reported in this paper was partially supported by NSF grant DMS-0528086. See http://www.cs.uiuc.edu/~jeffe/pubs/essential.html for the most recent version of this paper.

†This work was done while the author was affiliated with the University of Illinois, Urbana-Champaign.

computes the shortest *essential* cycle. However, as we show in Section 3, this claim is incorrect, in part because the set of essential cycles does not have the 3-path property of Thomassen [35, 31]. We give a weaker characterization of shortest essential cycles that leads directly to a cubic-time algorithm. In Section 4, we improve the running time to $O(n^2 \log n)$, matching the running time of Erickson and Har-Peled's algorithm for shortest non-contractible cycles. For surfaces with constant genus and a constant number of boundary cycles, we show in Section 5 that the running time can be improved to $O(n \log n)$, again matching the fastest algorithms known for shortest non-contractible cycles [29, 5]. The correctness of this faster algorithm relies on a convexity property of shortest essential cycles that may be of independent interest.

2 Notation and Definitions

We begin with some definitions from topology [24, 36] and topological graph theory [31].

Surface, paths, cycles, and homotopy. A *surface* (more formally, a 2-*manifold with boundaries*) is a connected Hausdorff topological space that is locally homeomorphic to a plane or a closed halfplane. The points with neighborhood homeomorphic to the closed halfplane comprise the *boundary* of the surface; the boundary of a compact surface is homeomorphic to the disjoint union of circles. A surface is *orientable* if it has no subset homeomorphic to the Möbius strip; with a few explicit exceptions, our results apply only to orientable surfaces.[1]

In any topological space Σ, a *path* is a continuous function $\alpha\colon [0,1] \to \Sigma$. An *arc* is a path whose endpoints lie on the boundary of Σ, and a *loop* is a path whose endpoints coincide. The *concatenation* $\alpha \cdot \beta$ of two paths α and β with $\alpha(1) = \beta(0)$ is defined by setting $(\alpha \cdot \beta)(t) = \alpha(2t)$ for all $t \le 1/2$, and $(\alpha \cdot \beta)(t) = \beta(1-2t)$ for all $t \ge 1/2$. The *reversal* $\overline{\alpha}$ of a path α is defined by setting $\overline{\alpha}(t) = \alpha(1-t)$ for all t. A *cycle* is a continuous function $\gamma\colon S^1 \to \Sigma$, where S^1 is the circle $\mathbb{R}/\mathbb{Z}$.

A path or cycle on a surface Σ is *simple* if it is injective; at the risk of confusing the reader, we sometimes use the same symbol to refer to both a simple path or cycle (a function) and its image (a subset of Σ). Two simple paths α and β on a surface *intersect transversely* at a point p if there is a homeomorphism from an open neighborhood B of p to the plane, such that $\alpha \cap B$ and $\beta \cap B$ are mapped to a pair of orthogonal lines. Two simple paths α and β *cross* if, after contracting each component of $\alpha \cap \beta$ to a point, the remaining paths intersect transversely at some point. Equivalently, two (sufficiently tame) simple paths do *not* cross if and only if they can be made disjoint by an arbitrarily small perturbation. In particular, two simple paths that intersect only at their endpoints do not cross.

A *homotopy* between two paths α and β with the same endpoints is a continuous function $h\colon [0,1] \times [0,1] \to \Sigma$ such that $h(0,t) = \alpha(t)$ and $h(1,t) = \beta(t)$ for all t, and $h(s,0) = \alpha(0) = \beta(0)$ and $h(s,1) = \alpha(1) = \beta(1)$ for all s. Two paths α and β are *homotopic* (written $\alpha \simeq \beta$) if there is a homotopy from one to the other. Two cycles γ and δ are *(freely) homotopic* if there is a continuous function $h : [0,1] \times S^1 \to \Sigma$ such that $h(0,t) = \gamma(t)$ and $h(1,t) = \delta(t)$ for all t.

A loop or a cycle on a surface Σ is *contractible* if it is (freely) homotopic to a point in Σ. A simple cycle γ in Σ is *separating* if $\Sigma \setminus \gamma$ has more than one connected component. A cycle in Σ is *essential* if it is simple, non-contractible, and not homotopic to a boundary cycle of Σ.

Following earlier work [6, 7], we write $\Sigma \,✄\, \alpha$ to denote the surface obtained by *cutting* Σ along a simple arc or cycle α; each point of α becomes a pair of boundary points in $\Sigma \,✄\, \alpha$. (We suggest the pronunciation 'snip' for the symbol ✄.)

[1] In particular, Lemma 4.1 and its dependents require orientability, and the algorithms in Section 5 rely on previous results that assume orientability [5, 8].

Combinatorial surfaces. An *embedding* of a graph G on a surface Σ maps the vertices of G to distinct points in Σ and edges of G to paths in Σ that are disjoint except at common endpoints. The *faces* of the embedding are maximal subsets of Σ that are disjoint from the image of the graph. An embedding is *cellular* if every face is homeomorphic to an open disk; in particular, each boundary cycle is covered by a cycle of edges in G. Any cellular embedding onto an orientable surface can be represented combinatorially by a *rotation system*, which consists of a cyclic permutation π_v of the edges incident to each vertex v. If e is an edge incident to vertex v, then the cyclic sequence $e, \pi_v(e), \pi_v(\pi_v(e)), \ldots$ is called the *clockwise ordering* around v.

The input to our problem is a *combinatorial surface* [10, 12, 18, 30], which is an abstract topological surface M together with an edge-weighted graph G cellularly embedded on M. In the combinatorial surface model, we only consider paths and cycles in M that arise as walks in G; in particular, paths and cycles may traverse the same edge of G multiple times. The length of a path or cycle is the sum of the weights of its edges, counted with appropriate multiplicity. A path or cycle in a combinatorial surface is *essentially simple* if contains no crossing subpaths, or equivalently, if an arbitrarily small perturbation is simple (injective). Essential cycles on combinatorial surfaces may be only essentially simple.

The *dual graph* G^* is defined with respect to the combinatorial surface $\Sigma^\bullet = (M^\bullet, G)$ obtained from $\Sigma = (M, G)$ by gluing a disk to each boundary cycle of M. Specifically, the dual graph G^* has a vertex for every face in G (including the disks glued to the boundaries of Σ) and an edge for each edge in G (including edges along the boundary of Σ). The presence of a boundary cycle in M is recorded by setting a bit in the corresponding face of the graph G; the dual graph G^* stores the same information in the vertices.[2]

For any subgraph $F = (U, D)$ of $G = (V, E)$, we write $G \setminus F$ to denote the edge-complement $(V, E \setminus D)$. Also, for any subgraph F of G, we abuse notation by writing F^* to denote the corresponding subgraph of G^*; every edge in F^* is the dual of a unique edge in F. In particular, we have the identity $(G \setminus F)^* = G^* \setminus F^*$.

If α is an essentially simple curve in a combinatorial surface Σ, the cut surface $\Sigma \mathbin{\not\!\!\!\!\!\>\mathrm{X}} \alpha$ is obtained by duplicating the vertices and edges of α with appropriate multiplicity. In the resulting surface $\Sigma \mathbin{\not\!\!\!\!\!\>\mathrm{X}} \alpha$, some edges may lie on two different boundary cycles (or even twice on the same boundary cycle), either because they appeared more than once in α, or because they appeared both in α and on the boundary of Σ.

It is sometimes helpful to view the graph G as a continuous metric space, so that distances and shortest paths between points in the interior of edges are well-defined. Specifically, any edge with weight ℓ is isometric to the real interval $[0, \ell]$.

Shortest paths and useful cycles. Without loss of generality, we assume that any two vertices in the graph are joined by a unique shortest path. A standard perturbation technique like the Isolation Lemma [32] can be used to enforce this assumption if necessary. Our assumption implies that the intersection of any two shortest paths is either empty or a single common subpath. Unlike previous works on combinatorial surfaces [10, 12, 18, 19, 30], we also assume without loss of generality that each edge in G is a shortest path between its endpoints. This assumption can be enforced, without changing the shortest essential cycle, by bisecting each 'long' edge at its midpoint.

For any source vertex s, we let T_s denote the tree of shortest paths in G from s to every other vertex. The *cut locus* with respect to s is $(G \setminus T_s)^*$, the subgraph of G^* containing all edges *not* dual to edges of T_s. For a surface of genus g, Euler's formula implies that the cut locus is a tree with $2g$ extra edges [16]. The *reduced cut locus* R_s^* is the graph obtained by repeatedly removing vertices of degree 1 from the cut locus [19].

[2]Our definition of the dual graph differs slightly from the definition used by Erickson and Colin de Verdière [12].

For a fixed graph G, let $\sigma(u,v)$ denote the shortest path from vertex u to vertex v. For any vertex s and edge uv, let $\gamma(s,uv)$ denote the oriented cycle formed by concatenating the shortest path $\sigma(s,u)$, the edge uv, and the shortest path $\sigma(v,s)$. Finally, for any edges st and uv, let $\gamma(st,uv)$ denote the directed cycle $ts \cdot \sigma(s,u) \cdot uv \cdot \sigma(v,t)$. In particular, $\gamma(st,uv)$ and $\gamma(st,vu)$ are different cycles.

3 Antipodal Edges

Fix an orientable combinatorial surface Σ. A set $\mathcal{C}$ of cycles in Σ has the *3-path property* if the following condition holds: For any paths α, β, and σ in Σ with the same endpoints, if the cycle $\alpha \cdot \overline{\beta}$ is in $\mathcal{C}$, then at least one of the cycles $\beta \cdot \overline{\sigma}$ and $\sigma \cdot \overline{\alpha}$ is also in $\mathcal{C}$.

Thomassen [35, 31] defined the 3-path property and proved that the set of non-contractible cycles has the 3-path property. It follows that for any vertex x on the shortest non-contractible cycle γ, the cycle can be decomposed into two equal-length shortest paths from x to its furthest point x' along γ. (If shortest paths are unique, the furthest point x' lies in the interior of an edge.) This observation implies that the shortest non-contractible cycle can be computed in $O(n^3)$ time, by combining Dijkstra's shortest path algorithm with a linear-time test for contractibility [35, 7]. By interleaving the contractibility test with Dijkstra's algorithm, Erickson and Har-Peled [18] improved the running time of this algorithm to $O(n^2 \log n)$. Faster algorithms are known for surfaces with small genus and few boundaries [7, 4, 29, 5]; see Section 5.

Erickson and Har-Peled claimed that a similar algorithm computes the shortest *essential* cycle, but their algorithm is incorrect, in part because the set of essential cycles does not have the 3-path property [11]. The shortest essential loop through a point x is not necessarily composed of two shortest paths from x. Moreover, in the combinatorial setting, the overall shortest essential cycle cannot necessarily be split into two equal-length shortest paths that share a *vertex*. The following counterexample establishes these claims, along with the necessity of considering essentially simple cycles.

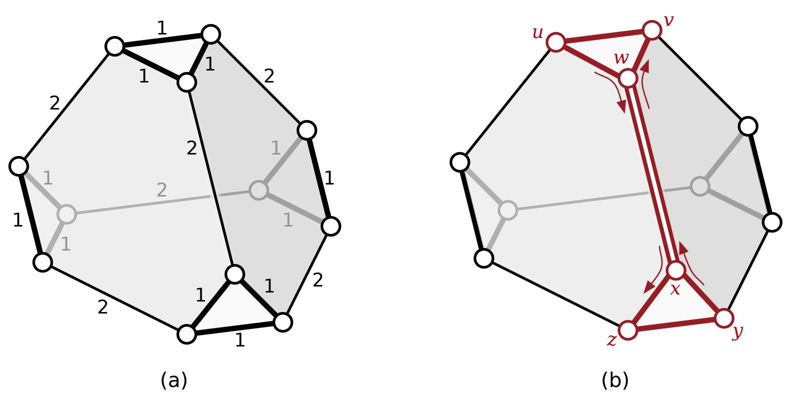

Figure 1. A counterexample for Erickson and Har-Peled's algorithms to compute shortest essential loops and cycles. (a) A combinatorial surface with genus 0 and four triangular boundary cycles. (b) A shortest essential cycle in this surface.

Consider the combinatorial surface Σ with genus zero and four boundaries illustrated in Figure 1(a). This surface can be constructed by deleting a small neighborhood of each vertex from the boundary of a regular tetrahedron. Each face of Σ is an irregular hexagon; each boundary edge has length 1; and each non-boundary edge has length 2. The shortest essential cycles in this surface have length 10; each shortest essential cycle has the form $\gamma = uv \cdot vw \cdot wx \cdot xy \cdot yz \cdot zx \cdot xw \cdot wu$, where wx is a non-boundary edge between two clockwise boundary cycles uvw and xyz. See Figure 1(b). This cycle is not simple, because it traverses the edge wx twice, but it is essentially simple. (The related cycle

$uv \cdot vw \cdot wx \cdot xz \cdot zy \cdot yx \cdot xw \cdot wu$ traverses the same edges as γ, with the same multiplicity, but it is not essentially simple, because the subpaths $vw \cdot wx \cdot xz$ and $yx \cdot xw \cdot wu$ cross.) Vertices u and y are antipodal on γ, but the the shortest path $uw \cdot wx \cdot xy$ is not a subpath of γ.

However, we can prove a weaker structural result. Like the 3-path condition, the following lemmas apply not only to combinatorial surfaces, but more generally to any surface endowed with a metric.

Lemma 3.1. *Let γ be the shortest essential cycle in a (not necessarily combinatorial or orientable) surface Σ, and let x and z be arbitrary points in γ. There is a shortest path from x to z that does not cross γ.*

Proof: Suppose to the contrary that every shortest path from x to z crosses γ. Let $\sigma(x,z)$ be a shortest path from x to z. It suffices to consider the case where $\sigma(x,z)$ crosses γ exactly once; let y be a point in $\gamma \cap \sigma$ such that the subpaths $\sigma(x,y)$ and $\sigma(y,z)$ do not cross γ. Decompose γ into three paths $\gamma(x,y) \cdot \gamma(y,z) \cdot \gamma(z,x)$, and observe that none of these three paths is a shortest path. See Figure 2.

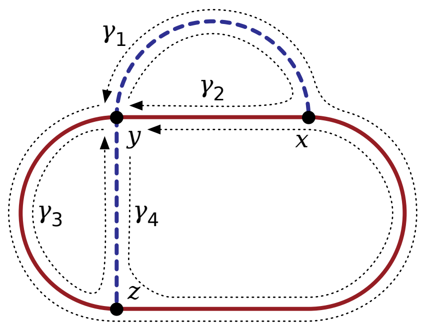

Figure 2. A shortest path σ (dashed) crossing a shortest essential cycle γ (solid).

Now consider the following essentially simple cycles, expressed as loops based at y:

$$\gamma_1 := \gamma(y,z) \cdot \gamma(z,x) \cdot \sigma(x,y), \qquad \gamma_2 := \overline{\sigma(x,y)} \cdot \gamma(x,y),$$
$$\gamma_3 := \gamma(y,z) \cdot \overline{\sigma(y,z)} \qquad \gamma_4 := \sigma(y,z) \cdot \gamma(z,x) \cdot \gamma(x,y).$$

Each cycle γ_i is shorter than γ and is therefore inessential. Each cycle γ_i is contractible in $\Sigma^\bullet$, because every essentially simple cycle in Σ that is noncontractible in $\Sigma^\bullet$ is essential in Σ. The shortest essential cycle γ is homotopic (in Σ) to the cycles $\gamma_1 \cdot \gamma_2$ and $\gamma_3 \cdot \gamma_4$; thus, each cycle γ_i is noncontractible in Σ. It follows that each cycle γ_i is homotopic to a distinct boundary cycle. (Moreover, the surface Σ must have genus zero and exactly four boundaries.)

We conclude that the essentially simple cycle $\gamma_2 \cdot \overline{\gamma_3}$ is essential. But this is impossible, because $\gamma_2 \cdot \overline{\gamma_3} = \overline{\sigma(x,y)} \cdot \gamma(x,y) \cdot \sigma(y,z) \cdot \overline{\gamma(y,z)}$ is shorter than the shortest essential cycle γ. □

Lemma 3.2. *Let γ be the shortest essential cycle in a (not necessarily combinatorial or orientable) surface. There is a point $x \in \gamma$ such that γ consists of two equal-length shortest paths from x to its farthest point on γ.*

Proof: Let x be an arbitrary point of γ; let x' be the point furthest from x along γ; and let α and β be the two paths from x to x' that comprise γ. Clearly α and β have the same length. Suppose α and β are not shortest paths. Let σ be a shortest path from x to x' that does not cross γ, as guaranteed by the previous lemma. The loops $\gamma' = \alpha \cdot \overline{\sigma}$ and $\gamma'' = \sigma \cdot \overline{\beta}$ are both shorter than γ, so neither of them is essential. However, because σ does not cross γ, both of these loops are essentially simple. Thus, $\gamma' \not\simeq \gamma$ and $\gamma'' \not\simeq \gamma$. On the other hand, neither γ' nor γ'' is contractible, because $\gamma \simeq \gamma' \cdot \gamma''$. Thus, both γ'

and γ'' are freely homotopic to boundary cycles, which implies that γ must bound a pair of pants. (In particular, if the surface has at most one boundary, the proof is complete.)

Reparametrize the cycle γ as a loop $\gamma : [0,1] \to M$ with basepoint $\gamma(0) = \gamma(1) = x$, such that for all t, the point on γ furthest from $\gamma(t)$ is $\gamma(t \pm 1/2)$. For each t, let α_t and β_t denote the two paths from $\gamma(t)$ to $\gamma(t \pm 1/2)$ in γ, and let σ_t denote a shortest path from $\gamma(t)$ to $\gamma(t \pm 1/2)$ that does not cross γ.

We now use a continuity argument to show that α_t and β_t must be shortest paths for some t. For the sake of argument, suppose α_t and β_t are never shortest paths. By our previous argument, for all t, the loops $\alpha_t \cdot \overline{\sigma_t}$ and $\sigma_t \cdot \overline{\beta_t}$ are freely homotopic to boundary cycles.

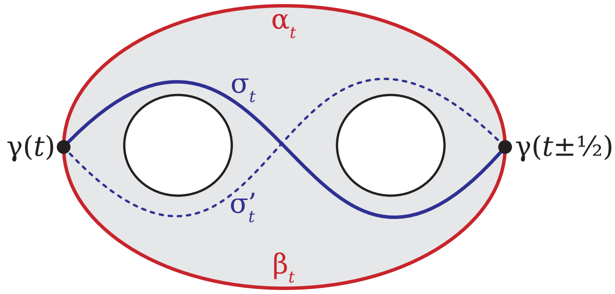

Figure 3. Splitting a pair of pants.

Let P denote the pair of pants bounded by γ. If every path σ_t lies inside P, then for some t, there must be two shortest paths σ_t and σ'_t from $\gamma(t)$ to $\gamma(t \pm 1/2)$ that are not homotopic in P; see Figure 3. If σ'_t does not separate the two boundary circles, then either $\alpha_t \cdot \overline{\sigma_t}$ or $\sigma_t \cdot \overline{\beta_t}$ is an essential cycle shorter than γ, which is a contradiction. Otherwise, σ_t and σ'_t must cross. By switching paths at the crossing, we obtain a third shortest path σ''_t that does not separate the two boundary circles, and again we obtain a contradiction.

On the other hand, if σ_t is not always inside P, then for some t, there are two non-homotopic shortest paths σ_t and σ'_t, one inside P and the other outside. By our earlier argument, the complement of P must be another pair of pants, whose legs are separated by σ'_t. It follows that $\sigma'_t \cdot \overline{\sigma_t}$ is an essential cycle shorter than γ, which is a contradiction. □

Lemma 3.2 immediately implies the following characterization of shortest essential cycles on combinatorial surfaces. Recall that $\gamma(st, uv)$ denotes the cycle $ts \cdot \sigma(s,u) \cdot uv \cdot \sigma(v,t)$.

Theorem 3.3 (Antipodal edges). *In any combinatorial surface Σ, there are two edges uv and st such that $\gamma(st, uv)$ is the shortest essential cycle in Σ.*

The following lemma characterizes the *essentially simple* cycles of the form $\gamma(st, uv)$.

Lemma 3.4. *For any edges st and uv, the cycle $\gamma(st, uv)$ is essentially simple if and only if the shortest paths $\sigma(s,u)$ and $\sigma(v,t)$ do not cross.*

Proof: If $\gamma(st, uv)$ is essentially simple then clearly $\sigma(s,u)$ and $\sigma(v,t)$ do not cross.

On the other hand, suppose $\sigma(s,u)$ and $\sigma(v,t)$ do not cross. Assume that $\sigma(s,u)$ and $\sigma(v,t)$ intersect, since otherwise $\gamma(st, uv)$ is simple. Uniqueness of shortest paths implies that $\tau = \sigma(s,u) \cap \sigma(v,t)$ is a simple path.

Suppose uv is an edge in the shortest path tree T_s. If u is an ancestor of v in T_s, then $\sigma(s,v) = \sigma(s,u) \cdot uv$, which implies that $\gamma(st, uv) = ts \cdot \sigma(s,v) \cdot \sigma(v,t) = \gamma(v, ts)$. On the other hand, if v is an ancestor of u in T_s, then $\gamma(st, uv)$ is the concatenation of $\gamma(v, ts)$ with the spur $vu \cdot uv$. In both cases,

$\gamma(st, uv)$ is essentially simple. Symmetric arguments imply that $\gamma(st, uv)$ is essentially simple if $uv \in T_t$ or $st \in T_u \cup T_v$.

Now suppose $uv \notin T_s \cup T_t$ and $st \notin T_u \cup T_v$. Then vertices t and v do not lie on the shortest path $\sigma(s, u)$; otherwise, either st or uv would not be a shortest path. Similarly, vertices s and u do not lie on the shortest path $\sigma(v, t)$. Thus, each of the vertices s, t, u, and v appears exactly once in $\gamma(st, uv)$, and therefore none of these four vertices lie on the common path τ. Because the paths $\sigma(s, u)$ and $\sigma(v, t)$ do not cross, we can perturb $\sigma(s, u)$ and $\sigma(v, t)$ within an arbitrarily small neighborhood of τ so that they become disjoint. This neighborhood avoids st and uv, so the perturbation removes all self-intersections from $\gamma(st, uv)$. We conclude that $\gamma(st, uv)$ is essentially simple. □

We emphasize that Lemma 3.4 requires our assumption that every edge in G is a shortest path between its endpoints. Figure 4 shows a graph in which the edge st is not the shortest path from s to t. The cycle $\gamma(st, uv)$ crosses itself at vertex s, and thus is not essentially simple, even though the shortest paths $\sigma(s, u)$ and $\sigma(v, t)$ do not cross. We can enforce our assumption by bisecting the long edge st into two equal-length edges sx and xt. In the resulting graph, the cycle $\gamma(sx, uv)$ is essentially simple, because the shortest paths $\sigma(x, v)$ and $\sigma(s, u)$ do not cross, and the cycle $\gamma(xt, uv)$ is not essentially simple, because the shortest paths $\sigma(t, v)$ and $\sigma(x, u)$ do cross.

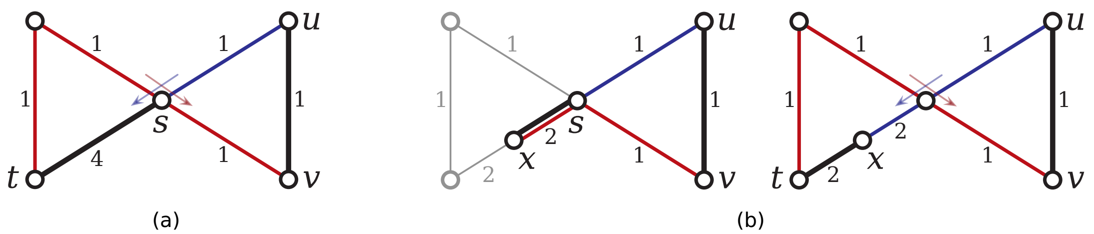

Figure 4. (a) Lemma 3.4 may be false if some edge is not a shortest path. (b) Bisecting long edges restores Lemma 3.4.

We are now ready to describe our algorithm for computing the shortest essential cycle. Our algorithm relies on the following subroutine.

Lemma 3.5. *Given two shortest paths σ and τ with distinct endpoints, we can determine whether they cross in $O(n)$ time.*

Proof: Color the vertices of $\sigma \setminus \tau$ red, $\tau \setminus \sigma$ blue and $\sigma \cap \tau$ purple. If there are no purple vertices, then σ and τ are disjoint and therefore do not cross. Otherwise the uniqueness of shortest paths implies that $\sigma \cap \tau$ is a single path; let x and y denote the endpoints of this path. If either x or y is an endpoint of either σ or τ, then the paths do not cross. Otherwise, there are two cases to consider. If there is only one purple vertex $x = y$, then σ and τ cross if and only if the red and blue neighbors of x alternate in cyclic order. Otherwise, σ and τ cross if the cyclic orders of red, blue, and purple neighbors of x and y are the same.

We can color the vertices of σ and τ and identify x and y in $O(n)$ time by simply traversing the paths. The cyclic order of the neighbors of x and y can be obtained in $O(n)$ time from the embedding. Thus, the total running time is $O(n)$. □

Theorem 3.6. *The shortest essential cycle in a combinatorial surface can be computed in $O(n^3)$ time.*

Proof: We begin by computing the shortest-path tree T_s for every vertex s using, for example, Dijkstra's algorithm [15]. Then for every pair of edges st and uv, we check whether the cycles $\gamma(st, uv)$ and $\gamma(st, vu)$ are essential as described below. Finally, we return the shortest candidate cycle that is found to be essential.

Given a candidate cycle $\gamma(st, uv)$, we first check whether it is essentially simple using Lemma 3.5. If the cycle is essentially simple, we then perform simultaneous depth-first searches on both sides of the cycle, following the strategy of Thomassen [35] and Erickson and Har-Peled [18]. If the two searches meet, the cycle is non-separating and therefore essential. Otherwise, we compute the Euler characteristic of the smaller component by a depth-first traversal. The cycle is essential if and only if neither component has Euler characteristic 1 (a disk) or 0 (an annulus).

Constructing n shortest path trees requires $O(n^2 \log n)$ time, and we test each of the $O(n^2)$ candidate cycles in $O(n)$ time, so the overall running time of our algorithm is $O(n^3)$. □

4 Faster Algorithms

In this section, we improve the brute-force algorithm described in the previous section. The key ingredient in our improvement is a preprocessing phase that allows us to determine in *constant* time whether a given cycle $\gamma(st, uv)$ is essentially simple, and if so, whether the cycle is essential. We first describe the improvement for genus-zero surfaces, then its generalization to higher-genus surfaces.

4.1 Genus Zero

Fix an orientable combinatorial surface Σ with genus 0 and b boundary cycles. We can assume that $b \geq 4$, since otherwise there are no essential cycles in Σ.

We first describe our data structure for fast simplicity queries. We call two edges st and uv *opposing* if $uv \notin T_s \cup T_t$ and $st \notin T_u \cup T_v$. The proof of Lemma 3.4 implies that if st and uv are *not* opposing edges, then $\gamma(st, uv)$ must be essentially simple.

So suppose st and uv are opposing edges. Vertices t and u do not lie on the shortest path $\sigma(s, v)$, and vertices t and v do not lie on $\sigma(s, u)$. It follows that the shortest paths $\sigma(s, u)$, $\sigma(s, v)$, and $\sigma(s, t) = st$ have a well-defined orientation around s, either clockwise or counterclockwise, induced by the rotation system of Σ. Let Σ' be the combinatorial surface obtained by contracting all but the final edges of $\sigma(s, u)$ and $\sigma(s, v)$. If edges st, su, and sv appear in clockwise order around s in Σ', we say that the triple (t, u, v) *is oriented clockwise around* s; otherwise, (t, u, v) *is oriented counterclockwise around* s. The orientations of (s, u, v) around t, (s, t, v) around u, and (s, t, u) around v are defined similarly.

Lemma 4.1. *Let st and uv be opposing edges in an orientable combinatorial surface of genus 0. The cycle $\gamma(st, uv)$ is **not** essentially simple if and only if the orientations of (t, u, v) around s, (s, u, v) around t, (s, t, v) around u, and (s, t, u) around v are either all clockwise or all counterclockwise.*

Proof: The lemma follows from a brute-force enumeration of all possible embeddings of the complete graph K_4 onto the sphere (or the plane) with vertices s, t, u, v and 'edges' st, uv, $\sigma(s, u)$, $\sigma(s, v)$, $\sigma(t, u)$, and $\sigma(t, v)$.

Two shortest paths with a common vertex cannot cross, and no path can cross a single edge. If both $\sigma(s, u)$ and $\sigma(t, v)$ cross and $\sigma(s, v)$ and $\sigma(t, u)$ cross, the Jordan curve theorem implies that one of those pairs of shortest paths must cross more than once, which is impossible. Thus, there are exactly three possible crossing patterns, illustrated in Figure 5: (1) $\sigma(s, v)$ and $\sigma(t, u)$ cross, but $\sigma(s, u)$ and $\sigma(t, v)$ do not; (2) $\sigma(s, u)$ and $\sigma(t, v)$ cross, but $\sigma(s, v)$ and $\sigma(t, u)$ do not; and (3) neither pair of shortest paths crosses. By Lemma 3.4, only the second crossing pattern implies that $\gamma(st, uv)$ is not essentially simple. Each crossing pattern is realized by exactly two orientations of the four vertex triples. In particular, the second crossing pattern occurs exactly for the orientations listed in the statement of the lemma. We omit further tedious details. □

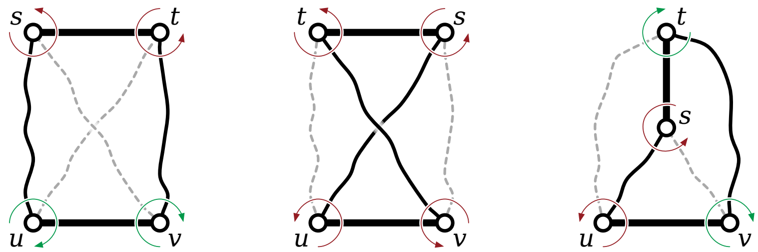

Figure 5. The only possible configurations of s,t,u,v, up to homotopy and reflection. The bold cycle is $\gamma(st,uv)$.

Lemma 4.2. *After $O(n^2)$ preprocessing time, we can determine in $O(1)$ time whether any cycle $\gamma(st,uv)$ is essentially simple, on an orientable combinatorial surface of genus 0.*

Proof: In the preprocessing phase, we compute the shortest path tree T_s for each vertex s, using the linear-time algorithm of Henzinger *et al.* [25]. We then rank the nodes in each shortest path tree T_s by a *clockwise preorder traversal*. We select an arbitrary child x of s and use the local clockwise ordering around s to linearly order the neighbors of s starting from x; we visit the subtrees of s in this linear order. For each vertex $y \neq s$, we visit the neighbors of y starting with the successor of the parent of y in the local clockwise ordering about y. We rank the nodes of T_s by their first appearance in this traversal; thus, a node visited earlier has a lower rank than a node visited later. We store the ranks in an array indexed by the vertices: $rank[s,v]$ is the clockwise preorder rank of v in T_s. Finally, for any opposing edges st and uv, the triple (t,u,v) is oriented clockwise around s if and only if

$$\begin{aligned} &rank[s,t] < rank[s,u] < rank[s,v], \text{ or} \\ &rank[s,u] < rank[s,v] < rank[s,t], \text{ or} \\ &rank[s,v] < rank[s,t] < rank[s,u]. \end{aligned}$$

Now given any pair of edges st and uv, we can easily determine whether they are opposing in constant time; if not, the cycle $\gamma(st,uv)$ must be essentially simple. If st and uv are opposing, we can determine whether $\gamma(st,uv)$ is essentially simple, in constant time, by computing the orientation of each triple of vertices around the fourth vertex and applying Lemma 4.1. □

Once we know that a cycle $\gamma(st,uv)$ is essentially simple, we check whether it is essential by computing the number of boundary cycles on one side. For any essentially simple *directed* cycle γ in Σ, let $b(\gamma)$ denote the number of boundary cycles in the component of $\Sigma \not\!\!\!\!\times \gamma$ lying to the right of γ; the cycle γ is essential if and only if $2 \le b(\gamma) \le b-2$. A second preprocessing phase allows us to compute $b(\gamma(st,uv))$ in constant time.

Lemma 4.3. *After $O(n^2)$ preprocessing time, we can determine in $O(1)$ time whether any essentially simple cycle $\gamma(st,uv)$ is essential, on an orientable combinatorial surface of genus 0.*

Proof: Fix a vertex s. Recall that we have already computed the shortest path tree T_s, and that boundary cycles are represented by marked vertices in the dual graph G^*. Fix an arbitrary unmarked dual vertex r^* as the root of the dual spanning tree $(G \setminus T_s)^*$. For any dual vertex f^*, let $\beta(s,f^*)$ denote the number of marked dual vertices in the subtree rooted at f^*. We can compute $\beta(s,f^*)$ for all dual vertices f^* in $O(n)$ time by a simple depth-first traversal of the dual spanning tree. We store these values in yet another array.

Now consider any cycle $\gamma(s,uv)$. If uv is an edge in T_s, then $\gamma(s,uv)$ is just a doubled shortest path, so $b(\gamma(s,uv)) = 0$. Otherwise, let f be the face adjacent to uv whose dual vertex f^* is further from the root r^* of the dual spanning tree. If f lies to the right of the directed edge uv, then $b(\gamma(s,uv)) = \beta(s,f^*)$; otherwise, $b(\gamma(s,uv)) = b - \beta(s,f^*)$. Thus, after preprocessing, we can compute $b(\gamma(s,uv))$ in constant time.

Finally, any essentially simple cycle $\gamma(st,uv)$ can be obtained by concatenating the cycles $\gamma(v,ts)$ and $\gamma(s,uv)$ and canceling the common shortest path $\sigma(v,s)$. If the regions to the right of $\gamma(v,ts)$ and $\gamma(s,uv)$ are disjoint, then $b(\gamma(st,uv)) = b(\gamma(v,ts)) + b(\gamma(s,uv))$; otherwise, $b(\gamma(st,uv)) = b(\gamma(v,ts)) + b(\gamma(s,uv)) - b$. In both cases, we have the identity

$$b(\gamma(st,uv)) = b(\gamma(v,ts)) + b(\gamma(s,uv)) \bmod b.$$

Thus, we can determine whether $\gamma(st,uv)$ is essential in constant time. □

We conclude:

Theorem 4.4. *The shortest essential cycle in an orientable combinatorial surface of genus* 0 *can be computed in* $O(n^2)$ *time.*

4.2 Positive Genus

Now suppose the input surface Σ has genus $g > 0$ and $b \geq 0$ boundary cycles. Recall that $\Sigma^\bullet$ is obtained from Σ by gluing a disk to each boundary cycle.

Lemma 4.5. *Suppose* $\gamma(st,uv)$ *is the shortest essential cycle in* Σ*. Either* $\gamma(st,uv)$ *is also the shortest non-contractible cycle in* $\Sigma^\bullet$*, or the cycles* $\gamma(s,uv)$*,* $\gamma(t,uv)$*,* $\gamma(u,st)$*, and* $\gamma(v,st)$ *are all contractible in* $\Sigma^\bullet$*.*

Proof: Every non-contractible cycle in $\Sigma^\bullet$ is essential in Σ. The cycle $\gamma(s,uv)$ cannot be longer than $\gamma(st,uv)$. Thus, if $\gamma(s,uv)$ is non-contractible in $\Sigma^\bullet$, it must have the same length as $\gamma(st,uv)$. The uniqueness of shortest paths then implies that $\gamma(s,uv)$ and $\gamma(st,uv)$ are in fact the same cycle. Thus, $\gamma(st,uv)$ is the shortest essential cycle in Σ and is non-contractible in $\Sigma^\bullet$, which implies that it must be the shortest non-contractible cycle in $\Sigma^\bullet$. Similar arguments apply if any of the other three cycles is non-contractible in $\Sigma^\bullet$. □

Say that cycle $\gamma(st,uv)$ is *reducible* if all the cycles $\gamma(s,uv)$, $\gamma(t,uv)$, $\gamma(u,st)$, and $\gamma(v,st)$ are all contractible in $\Sigma^\bullet$. Lemma 4.5 immediately implies that either the shortest essential cycle $\gamma(st,uv)$ is non-contractible in $\Sigma^\bullet$, or $\gamma(st,uv)$ is the shortest essential *reducible* cycle.

Our algorithm to find the shortest essential reducible cycle in Σ requires only a few changes from the genus-zero case. As before, our algorithm preprocesses the graph so that we can quickly determine whether a cycle $\gamma(st,uv)$ is both essentially simple and essential; however, we now only have to test reducible cycles.

Lemma 4.6. *After* $O(n^2 \log n)$ *preprocessing time, we can determine in* $O(1)$ *time whether any cycle* $\gamma(st,uv)$ *is both reducible and essential, on any orientable combinatorial surface.*

Proof: For any cycle γ that is contractible in $\Sigma^\bullet$, let $b(\gamma)$ denote the number of holes *in the genus-*0 *component* of $\Sigma \not\!\!\!\text{✂} \gamma$. Recall that the *reduced cut locus* of a vertex s is the dual subgraph R_s^* obtained by repeatedly removing all vertices of degree 1 from the cut locus $(G \setminus T_s)^*$.

In the preprocessing phase, we preform the following computations for each vertex s. First, we compute the shortest path tree T_s using Dijkstra's algorithm [15]. Next, we compute the reduced cut locus R_s^* and its complementary forest $F_s^* = (G \setminus T_s)^* \setminus R_s^*$, as well as the clockwise preorder ranks of the vertices of T_s. Finally, for every edge uv such that $(uv)^* \in F_s^*$ (so the cycle $\gamma(s,uv)$ is contractible in $\Sigma^\bullet$), we compute $b(\gamma(s,uv))$ as follows. We define the root of each tree in F_s^* to be the unique vertex that is also in the reduced cut locus R_s^*. For each dual node f^* in F_s^*, let $\beta(s,f^*)$ denote the number of marked dual vertices (representing boundary cycles) in the subtree of F_s^* rooted at f^*; we can compute all these values by a straightforward depth-first search of F_s^*. For each edge uv, let $f(s,uv)$ denote the face adjacent to uv whose dual vertex $f(s,uv)^*$ lies further from the root of its component of F_s^*. Observe that $b(\gamma(s,uv)) = \beta(s,f(s,uv)^*)$. We also store whether the face $f(s,uv)$, and thus the genus-zero component of Σ ✂ $\gamma(s,uv)$, lies to the right or left of the oriented edge uv. The preprocessing time for each vertex s is dominated by Dijkstra's algorithm, which runs in $O(n\log n)$ time; everything else takes $O(n)$ time.

For any vertex s and any edge uv, the cycle $\gamma(s,uv)$ is contractible in $\Sigma^\bullet$ if and only if the dual edge $(uv)^*$ is not in the reduced cut locus R_s^*. Thus, to test whether a cycle $\gamma(st,uv)$ is reducible in constant time, we simply verify that the edges st or uv do not lie in the reduced cut loci of s, t, u, and v.

Now suppose $\gamma(st,uv)$ is reducible. Let H be the union of the shortest paths st, uv, $\sigma(s,u)$, $\sigma(s,v)$, $\sigma(t,u)$, and $\sigma(t,v)$; tedious case analysis implies that at most one component of Σ ✂ H is not a disk. Thus, the proof of Lemma 4.1 applies verbatim to *reducible* cycles $\gamma(st,uv)$ on higher-genus surfaces. It follows that we can test whether a reducible cycle $\gamma(st,uv)$ is essentially simple in constant time exactly as in the genus-zero case: first check whether st and uv are opposing, and if so, compute the orientation of each triple of vertices around the fourth using clockwise preorder ranks.

If $\gamma(st,uv)$ is both reducible and essentially simple, we compute $b(\gamma(st,uv))$ in constant time as follows. If faces $f(s,uv)$ and $f(u,st)$ lie on the same side of cycles $\gamma(s,uv)$ and $\gamma(v,ts)$, then $b(\gamma(st,uv)) = b(\gamma(s,uv)) + b(\gamma(v,st))$. Otherwise, the genus-zero components of Σ ✂ $\gamma(s,uv)$ and Σ ✂ $\gamma(v,st)$ must be nested, so $b(\gamma(st,uv)) = |b(\gamma(s,uv)) - b(\gamma(v,ts))|$. Finally, $\gamma(st,uv)$ is essential if and only if $b(\gamma(st,uv)) \geq 2$, because the other component of Σ ✂ $\gamma(st,uv)$ has positive genus. □

This lemma immediately implies that we can compute the shortest reducible essential cycle γ in $O(n^2\log n)$ time. We can also compute the shortest non-contractible cycle $\gamma^\bullet$ in $\Sigma^\bullet$ in $O(n^2\log n)$ time, using the algorithm of Erickson and Har-Peled [18]. The shortest essential cycle in Σ is the shorter of γ and $\gamma^\bullet$. This concludes the proof of our main result.

Theorem 4.7. *The shortest essential cycle in an orientable combinatorial surface can be computed in $O(n^2\log n)$ time.*

If $g = O(n^{1-\varepsilon})$ for some constant $\varepsilon > 0$, we can improve the running time to $O(n^2)$ by substituting the shortest-path algorithm of Henzinger *et al.* [25] for Dijkstra's algorithm. This exactly matches the performance of Erickson and Har-Peled's algorithm to compute shortest non-contractible cycles [18].

5 Small Genus and Few Boundaries

In this section, we describe how to compute the shortest essential cycle in $O(b^4 n\log n)$ time for an combinatorial surface of genus 0, or in $O((b^2+g^3)n\log n)$ time when the genus g is positive. The first algorithm improves our earlier result whenever $b = o(n^{1/4})$; the second algorithm is faster whenever $g = o(n^{1/3})$ and $b = o(n^{1/2})$.

In fact, our faster algorithm is a special case of a more general result. An essentially simple cycle γ is *k-inessential* if it separates the underlying surface into two components, at least one of which has genus zero and at most k boundaries (including γ), and *k-essential* if it is essentially simple but not k-inessential. Thus, 1-essential is the same as non-contractible, and 2-essential is the same as essential.

Here we describe algorithms to compute the shortest k-essential cycle in $O(\text{poly}(g, b) \cdot n \log n)$ time for any fixed integer k.[3]

As in Section 4, we first describe our algorithm for genus-zero surfaces and then generalize to the higher genus case. Surprisingly, genus-zero surfaces present more difficulties than higher-genus surfaces, and surfaces with few boundaries are more difficult than surfaces with many boundaries. In fact, the most difficult input for our original problem ($k = 2$) is a surface with genus zero and four boundaries; see Figure 1!

5.1 Constant Genus and Number of Boundaries

If the parameters g and b are *constants*, then for any k, we can compute the shortest k-essential cycle in $O(n \log n)$ time using a recent algorithm of Chambers *et al.* [8, Theorem 6.1]. The input to their algorithm is a combinatorial surface Σ and a set of pairs $S = \{(g_1, b_1), (g_2, b_2), \ldots, (g_r, b_r)\}$. An essentially simple cycle γ is *allowed by S* if it separates Σ into two components, at least one of which has genus g_i and b_i boundaries (including γ), for some pair $(g_i, b_i) \in S$. The algorithm of Chambers *et al.* computes the shortest cycle allowed by any set S in $(g + b)^{O(g+b)} n \log n$ time.

Theorem 5.1. *For any integer k, the shortest k-essential cycle in an orientable combinatorial surface with genus g and b boundaries can be computed in $(g + b)^{O(g+b)} n \log n$ time.*

Proof: On a genus-0 surface, a cycle is k-essential if and only if it is allowed by the set $\{(0, b') \mid k+1 \le b' \le b-k+1\}$. (In particular, the surface has *no* k-essential cycles if $b < 2k$.) Thus, the shortest k-essential cycle can be computed in time $b^{O(b)} n \log n$.

On a surface Σ with positive genus, a cycle is essential if and only if it is either non-contractible in $\Sigma^\bullet$ or allowed by the set $S = \{(0, b') \mid k+1 \le b' \le b+1\}$. (In particular, if $b < k$, every essential cycle in Σ is non-contractible in $\Sigma^\bullet$.) The shortest non-contractible cycle in $\Sigma^\bullet$ can be computed in time $g^{O(g)} n \log n$ using an algorithm of Kutz [29],[4] and the shortest cycle allowed by S can be computed in time $(g + b)^{O(g+b)} n \log n$. The shortest k-essential cycle is the shorter of these two cycles. □

5.2 Convexity

To improve the running time of our algorithm when g and b are large, we rely on another relaxation of Thomassen's 3-path condition. A subset X of a (not necessarily combinatorial) surface Σ is *convex* if every shortest path in Σ between two points in X lies entirely in X. This definition is consistent with both Harary and Nieminen's definition of *geodesic convexity* in graphs [3, 22] and the standard definition of *total convexity* in Riemannian geometry [1]. In particular, a *cycle* is convex if and only if no shortest path crosses the cycle more than once; moreover, a convex cycle must be simple (not just essentially simple). An immediate consequence of the 3-path condition, first observed by Cabello and Mohar [7] and then later used by several authors [4, 5, 29], is that the shortest non-contractible and non-separating cycles are convex. Erickson and Whittlesey proved that every cycle in the minimum-length homology basis is convex, or in their terminology, *tight* [19].

Non-separating cycles are k-essential for every integer k. Thus, if the shortest essential cycle γ is non-separating, it is the shortest non-separating cycle, which implies that it is convex. However, if the shortest k-essential cycle is separating, it may not be convex (unless $k = 1$); see Figure 1(b). The

[3] It is unclear whether our earlier near-quadratic algorithms can be extended to find shortest k-essential cycles for any $k > 2$. In particular, Lemma 3.2 does not directly generalize: There may be no pair of antipodal points on the shortest 3-essential cycle that splits it into two equal-length shortest paths.

[4] The time for this step can be improved to $O(g^3 n \log n)$ using the more recent algorithm of Cabello and Chambers [5], but this improvement will not change the final running time.

following lemma shows that if the shortest essential cycle is non-convex, then it separates the surface into two components with useful structure.

Lemma 5.2. *Let γ be the shortest k-essential cycle in a (not necessarily combinatorial or orientable) surface Σ. Suppose γ is separating, and let Σ^+ and Σ^- denote the components of $\Sigma \,\text{✂}\, \gamma$. If Σ^+ is non-convex, then Σ^- has genus 0 and at most $2k-1$ boundaries (including γ).*

Proof: Suppose Σ^+ (and therefore also γ) is not convex. Let p and q be points in Σ^+ whose shortest path $\sigma(p,q)$ does not lie entirely in Σ^+. Without loss of generality, we can assume that $\sigma(p,q)$ lies entirely in Σ^- and intersects γ only at its endpoints.

The cycles $\gamma' = \sigma(p,q)\cdot\gamma(q,p)$ and $\gamma'' = \sigma(q,p)\cdot\gamma(q,p)$ are boundaries of the surface $\Sigma^- \,\text{✂}\, \gamma$. Uniqueness of shortest paths implies that both γ' and γ'' are shorter than γ and therefore k-inessential. In particular, both γ' and γ'' are contractible in $\Sigma^\bullet$, which implies that γ is also contractible in $\Sigma^\bullet$. In other words, Σ^- has genus 0.

It follows that $\Sigma^- \,\text{✂}\, \sigma(p,q)$ has two genus-0 components, one bounded by γ' and the other bounded by γ''. Each of these components has at most k boundaries. We conclude that Σ^- has at most $2k-1$ boundaries. □

For the special case $k=1$, Lemma 5.2 simply restates the observation that the shortest non-contractible (1-essential) cycle is convex. (If Σ^+ and Σ^- are convex, then their intersection γ is convex as well. If the shortest non-contractible cycle is non-separating, it is the shortest non-separating cycle and is therefore convex.) The special case $k=2$ is also implicit in the proof of Lemmas 3.1 and 3.2: If the shortest (2-)essential cycle is not convex, then it must be the boundary of a pair of pants.

5.3 Genus Zero, Many Boundaries

Suppose the input surface Σ has genus 0. If $b \le 4k-4$, we can compute the shortest k-essential cycle in $b^{O(b)} n \log n = k^{O(k)} n \log n$ time by Theorem 5.1. So let us assume that $b \ge 4k-3$.

Our approach is to find two disjoint sets of k boundary cycles on opposite sides of the shortest k-essential cycle, by brute force enumeration, and compute the shortest cycle separating those two sets. Specifically, we enumerate all possible pairs $\Delta = \{\delta_1, \dots, \delta_k\}$ and $\Lambda = \{\lambda_1, \dots, \lambda_k\}$ of disjoint sets, each containing k boundary cycles from Σ. For each pair of sets Δ and Λ, we compute a certain cycle $\gamma(\Delta,\Lambda)$ that separates Δ and Λ. Finally, we return the shortest cycle $\gamma(\Delta,\Lambda)$, over all pairs (Δ,Λ), as the shortest k-essential cycle.

For any set Δ of k boundaries, we define a forest $F(\Delta)$ of shortest paths that connect Δ as follows. Shrink each boundary in Δ to a single representative vertex. (Equivalently, change the weights of the edges of boundaries in Δ to 0, and choose an arbitrary vertex on each boundary to represent that boundary.) Next, construct the minimum spanning tree T of those k representative vertices in the shortest-path metric. Each edge of T is a shortest path in Σ between two boundaries in Δ; let $F(\Delta)$ be the union of those $k-1$ shortest paths. Erickson and Har-Peled [18, Lemma 6.4] show that $F(\Delta)$ can be constructed in $O(n \log n)$ time.

Now fix two disjoint sets Δ and Λ, each containing k boundaries. Let $\Sigma(\Delta,\Lambda)$ denote the surface obtained by gluing disks to all boundaries of Σ outside the set $\Delta \cup \Lambda$ and cutting along the forest $F(\Delta)$. This surface has exactly $k+1$ boundaries: all the boundaries in Λ, plus one new boundary δ^*. We define $\gamma(\Delta,\Lambda)$ to be the shortest cycle in $\Sigma(\Delta,\Lambda)$ that is homotopic to the new boundary δ^*. This cycle can be computed in $O(n \log n)$ time (with no hidden dependence on g, b, or k) using an algorithm of Cabello *et al.* [6, Lemma 6].

Our algorithm examines $\binom{b}{k}\binom{b-k}{k} = O(b^{2k})$ pairs of k-element subsets of boundaries, and computes a k-essential cycle separating each pair in $O(n \log n)$ time. Thus, the overall running time of our algorithm is $O(b^{2k} n \log n)$.

The correctness of our algorithm follows from Lemma 5.2. Let Δ and Λ be arbitrary sets of k boundaries on opposite sides of the shortest k-essential cycle γ. Then γ is also the shortest cycle separating Δ and Λ. Because $b \geq 4k - 3$, the surface $\Sigma \,\text{✂}\, \gamma$ has at least $4k - 1$ boundaries (including both copies of γ), so at least one of its components must have at least $2k$ boundaries. So Lemma 5.2 implies that at least one component of $\Sigma \,\text{✂}\, \gamma$ is convex, which implies that either no shortest path in $F(\Delta)$ crosses γ, or no shortest path in $F(\Lambda)$ crosses γ. Our algorithm tries both possibilities.

Theorem 5.3. *For any integer k, the shortest k-essential cycle in an orientable combinatorial surface with genus 0 and b boundaries can be computed in $O((b^{2k} + k^{O(k)})\, n \log n)$ time.*

In the special case $k = 2$, we only need to fall back on the slower algorithm described by Theorem 5.1 when $b = 4$, and the forest $F(\Delta)$ is just a single shortest path.

Corollary 5.4. *The shortest essential cycle in an orientable combinatorial surface with genus 0 and b boundaries can be computed in $O(b^4 n \log n)$ time.*

5.4 Positive Genus

Finally, suppose Σ has positive genus. A cycle γ is k-essential if and only if it is either non-contractible in $\Sigma^\bullet$ or separates Σ into two components, one of which has genus 0 and at least $k + 1$ boundaries (including γ). We compute the shortest non-contractible cycle in $\Sigma^\bullet$ in $O(g^3 n \log n)$ time, using the recent algorithm of Cabello and Chambers [5]. To compute the shortest cycle of the second type, we use the following simplification of our genus-0 algorithm. The shorter of these two cycles is the shortest k-essential cycle.

Our algorithm enumerates all $\binom{b}{k} = O(b^k)$ subsets Δ of k boundary cycles in Σ. For each set Δ, we first compute the surface $\Sigma(\Delta)$ obtained from Σ by gluing disks on every boundary outside Δ and then cutting along the shortest-path forest $F(\Delta)$. This surface has genus g and exactly one boundary, and we can compute it in $O(n \log n)$ time, just as in the genus-0 case [18]. We then compute the shortest cycle $\gamma(\Delta)$ in $\Sigma(\Delta)$ homotopic to its single boundary, in $O(n \log n)$ time [6]; this cycle is clearly k-essential in the original surface Σ. Finally, we return the shortest cycle $\gamma(\Delta)$.

The correctness of our algorithm again follows from Lemma 5.2. If the shortest k-essential cycle γ is contractible in Σ, then the genus-zero component of $\Sigma \,\text{✂}\, \gamma$ must be convex. Thus, for any set Δ of k boundaries in this component, the shortest-path forest $F(\Delta)$ does not cross γ.

Unlike the genus-0 case, we never need to fall back on the slower algorithm described in Theorem 5.1.

Theorem 5.5. *The shortest k-essential cycle in an orientable combinatorial surface with genus $g > 0$ and b boundaries can be computed in $O((b^k + g^3)\, n \log n)$ time.*

Corollary 5.6. *The shortest essential cycle in an orientable combinatorial surface with genus $g > 0$ and b boundaries can be computed in $O((b^2 + g^3)\, n \log n)$ time.*

Acknowledgments. Thanks to Éric Colin de Verdière [11] for finding the error in [18] and other useful discussions. We also thank Sergio Cabello and anonymous reviewers for their very helpful and detailed comments on previous versions of the paper.

References

[1] M. Berger. *A Panoramic View of Riemannian Geometry*. Springer, 2003.

[2] G. Borradaile, J. R. Lee, and A. Sidiropoulos. Randomly removing g handles at once. *Proc. 25th Ann. Symp. Comput. Geom.*, 371–376, 2009.

[3] F. Buckley and F. Harary. *Distance in Graphs*. Addison-Wesley, 1990.

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

[5] S. Cabello and E. W. Chambers. Multiple source shortest paths in a genus g graph. *Proc. 18th Ann. ACM-SIAM Symp. Discrete Algorithms*, 89–97, 2007.

[6] S. Cabello, M. DeVos, J. Erickson, and B. Mohar. Finding one tight cycle. *Proc. 19th Ann. ACM-SIAM Symp. Discrete Algorithms*, 527–531, 2008.

[7] S. Cabello and B. Mohar. Finding shortest non-separating and non-contractible cycles for topologically embedded graphs. *Discrete Comput. Geom.* 37:213–235, 2007.

[8] E. W. Chambers, É. Colin de Verdière, J. Erickson, F. Lazarus, and K. Whittlesey. Splitting (complicated) surfaces is hard. *Comput. Geom. Theory Appl.* 41(1–2):94–110, 2008.

[9] E. W. Chambers, J. Erickson, and A. Nayyeri. Minimum cuts and shortest homologous cycles. *Proc. 25th Ann. ACM Symp. Comput. Geom.*, 377–385, 2009.

[10] É. Colin de Verdière. *Shortening of curves and decomposition of surfaces*. Ph.D. thesis, Université Paris 7, December 2003. ⟨http://www.di.ens.fr/users/colin/textes/these.html.en⟩.

[11] É. Colin de Verdière. Personal communication, 2004.

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

[13] É. Colin de Verdière and F. Lazarus. Optimal pants decompositions and shortest homotopic cycles on an orientable surface. *J. ACM* 54(4), 2007.

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

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

[16] D. Eppstein. Dynamic generators of topologically embedded graphs. *Proc. 15th Ann. ACM-SIAM Symp. Discrete Algorithms*, 599–608, 2004.

[17] D. Eppstein. Squarepants in a tree: Sum of subtree clustering and hyperbolic pants decomposition. *Proc. 18th ACM-SIAM Symp. Discrete Algorithms*, 29–38, 2007.

[18] J. Erickson and S. Har-Peled. Optimally cutting a surface into a disk. *Discrete Comput. Geom.* 31(1):37–59, 2004.

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

[20] B. Farb and D. Margalit. A primer on mapping class groups. Preprint, version 3.1, June 1, 2009. ⟨http://www.math.utah.edu/~margalit/primer/⟩.

[21] I. Guskov and Z. Wood. Topological noise removal. *Proc. Graphics Interface*, 19–26, 2001.

[22] F. Harary and J. Nieminen. Convexity in graphs. *J. Diff. Geom.* 16(2):185–190, 1981.

[23] J. L. Harer and R. C. Penner. *Combinatorics of Train Tracks*. Annals of Math. Studies 125. Princeton University Press, 1992.

[24] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2001.

[25] M. R. Henzinger, P. Klein, S. Rao, and S. Subramanian. Faster shortest-path algorithms for planar graphs. *J. Comput. Syst. Sci.* 55(1):3–23, 1997.

[26] P. Indyk and A. Sidiropoulos. Probabilistic embeddings of bounded genus graphs into planar graphs. *Proc. 23rd Ann. ACM Symp. Comput. Geom.*, 204–209, 2007.

[27] K. Kawarabayashi and B. Mohar. Graph and map isomorphism and all polyhedral embeddings in linear time. *Proc. 40th Ann. ACM Symp. Theory Comput.*, 471–480, 2008.

[28] K. Kawarabayashi and B. Reed. Computing crossing number in linear time. *Proc. 39th Ann. ACM Symp. Theory Comput.*, 382–390, 2007.

[29] M. Kutz. Computing shortest non-trivial cycles on orientable surfaces of bounded genus in almost linear time. *Proc. 22nd Ann. ACM Symp. Comput. Geom.*, 430–438, 2006.

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

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

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

[33] S.-H. Poon and S. Thite. Pants decomposition of the punctured plane. *Proc. 22nd European Workshop Comput. Geom.*, 99–102, 2006. arXiv:cs.CG/0602080.

[34] M. Seppälä and T. Sorvali. *Geometry of Riemann Surfaces and Teichmüller Spaces*. North-Holland Mathematics Studies 169. North-Holland, 1992.

[35] C. Thomassen. Embeddings of graphs with no short noncontractible cycles. *J. Comb. Theory Ser. B* 48(2):155–177, 1990.

[36] A. Zomorodian. *Topology and Computing*. Cambridge University Press, 2005.