Computing the Stretch of an Embedded Graph*

Sergio Cabello† Markus Chimani‡ Petr Hliněný §

November 17, 2013

Abstract

Let G be a graph embedded in an orientable surface Σ, possibly with edge weights, and denote by $\text{len}(\gamma)$ the length (the number of edges or the sum of the edge weights) of a cycle γ in G. The stretch of a graph embedded on a surface is the minimum of $\text{len}(\alpha)\cdot\text{len}(\beta)$ over all pairs of cycles α and β that cross exactly once. We provide two algorithms to compute the stretch of an embedded graph, each based on a different principle. The first algorithm is based on surgery and computes the stretch in time $O(g^4 n \log n)$ with high probability, or in time $O(g^4 n \log^2 n)$ in the worst case, where g is the genus of the surface Σ and n is the number of vertices in G. The second algorithm is based on using a short homology basis and computes the stretch in time $O(n^2 \log n + n^2 g + n g^3)$.

1 Introduction

Consider a graph G embedded on an orientable surface Σ of genus g. What can be said about the crossing number of G in the plane? Is it computable in polynomial time? If it is not, can we obtain a reasonable approximation in polynomial time? Unfortunately, Cabello and Mohar [3] show that the crossing number of such graphs is not computable in polynomial time, even when Σ is the torus. Djidjev and Vrt'o [4] show that the crossing number of G is upper bounded by $O(g\Delta n)$, where n is the number of vertices in G and Δ is the maximum degree of G. This is an improvement over the previous bound of $O(C^g \Delta n)$, for some constant C, by Böröczky, Pach and Tóth [1]. Under some mild assumptions about the density of the embedding of G, Hliněný and Chimani [9] give a $(3\cdot 2^{3g+2}\Delta^2)$-approximation algorithm for the crossing number of G. This last work is the main motivation for our research.

Hliněný and Chimani [9] define the ***stretch*** of an embedded graph G as

$$\texttt{str}(G) \;=\; \min\{\text{len}(\alpha)\cdot\text{len}(\beta) \mid \alpha \text{ and } \beta \text{ are cycles in } G \text{ that cross exactly once}\}.$$

*A preliminary version of this work was presented at XV Spanish Meeting on Computational Geometry in 2013.

†Department of Mathematics, IMFM, and Department of Mathematics, FMF, University of Ljubljana, Slovenia. Supported by the Slovenian Research Agency, program P1-0297, projects J1-4106 and L7-5459, and within the EUROCORES Programme EUROGIGA (project GReGAS) of the European Science Foundation.

‡Theoretical Computer Science, Department of Mathematics/Computer Science, Osnabrück University, Germany. This research was conducted while being funded by a Carl-Zeiss-Foundation juniorprofessorship, and partially supported by EUROCORES Programme EUROGIGA (project GraDR) of the European Science Foundation.

§Faculty of Informatics, Masaryk University. Brno, Czech Republic. Supported by EUROCORES Programme EUROGIGA (project GraDR) of the European Science Foundation, under Czech Science Foundation no. GIG/11/E023.

Here, $\text{len}(\alpha)$ denotes the number of edges in α and a ***cycle*** is a closed walk in a graph without repeated vertices. A precise definition of what "crossing exactly once" means is given in Section 2.2. The stretch plays a fundamental role in their analysis of the algorithm. Intuitively, a large value of stretch implies existence of a large toroidal grid minor in G, even if the density of the embedding is not large in traditional terms. The concept of stretch can be generalized to the case of positive edge-weighted graphs in a natural way: take $\text{len}(\alpha)$ to be the sum of the edge-weights along the cycle α. Henceforth we will assume this more general definition of stretch.

It is worth noting that, if two cycles α and β are crossing once, then they must both be (surface-)non-separating. That is, cutting the surface along α or β does not disconnect the surface. This is so because any cycle crosses a (surface-)separating cycle an even number of times. Thus, when computing the stretch, we can restrict our attention to pairs (α, β) of non-separating cycles.

In this paper we provide two algorithms to compute the stretch of an embedded graph. Let g denote the genus of the surface Σ and n the number of vertices in G. The first algorithm has a time bound of $O(g^4 n \log n)$ with high probability, or $O(g^4 n \log^2 n)$ in the worst case. The second algorithm has a time bound of $O(n^2 \log n + n^2 g + n g^3)$.

Overview of the approach. Let us provide an informal overview of the main ideas. We do not work directly with the concept of stretch, but use a detour through another concept: odd-stretch. The definition of odd-stretch resembles the definition of stretch, but we allow closed walks α and β, instead of just cycles, and allow an odd number of crossings, instead of exactly one crossing. It turns out that the stretch and odd-stretch of a graph is the same. However, working with the odd-stretch is easier because we only need to take care of the parity of crossings and, when constructing new closed walks via exchange arguments, we do not need to take care to construct cycles.

For the first algorithm we show the following recursive property of the stretch: it is defined either by the shortest non-separating cycle α^* and one other cycle crossing α^* exactly once, or by the stretch of the surface obtained by cutting along α^* and pasting disks. The eventual algorithm, given in Figure 2, is very simple. However, there is a fine point we have to take care of to obtain a polynomial-time algorithm. Repeatedly cutting along shortest non-separating cycles and pasting disks may give rise to an exponential growth in the size of the graphs: at each cut we make copies of the vertices along the cycle and thus the number of vertices may nearly double at each iteration. However, if at some iteration we get a shortest non-separating cycle with more vertices than the original graph, we can finish the recursive search. In this way we avoid the potentially exponential growth in the size of the graphs.

For the second algorithm we use (simple) properties of the homology group over $\mathbb{Z}_2$. We define a map $\texttt{cr}_2$ counting the (parity of the) number of crossings of 1-cycles. The key observations are that $\texttt{cr}_2$ is an invariant of homologous cycles and it is bilinear. It turns out that it is enough to restrict the search to cycles in a *shortest* set of cycles that generate the homology group. After this, it is just a matter of combining known results. The resulting algorithm is very simple to understand, provided that one is familiar with $\mathbb{Z}_2$-homology on surfaces.

2 Odd-stretch

We introduce the concept of odd-stretch, which is a generalization of stretch. We first discuss crossings for curves in general position and then crossings for closed walks in a

graph. Finally, we define the odd-stretch, discuss some of its properties and, eventually, show that the odd-stretch of a surface-embedded graph is the same as its stretch.

2.1 Crossings of curves in general position

Two curves C and C' on a surface Σ are in ***general position*** if they have a finite number of intersections and, at each intersection, they cross transversally. Formally, for every point x in the image of C or C', there exists an open neighborhood $N(x)$ of x such that:

(i) $N(x) \cap (C \cup C')$ is homeomorphic to an open straight-line segment, or

(ii) there is a homeomorphism h between $N(x) \cap (C \cup C')$ and two distinct straight-line open segments s_1 and s_2 that cross.

For two curves C and C' in general position, the ***set of crossings*** is $X(C, C') = C \cap C'$. These two curves ***cross*** k times if and only if $k = |X(C, C')|$. We will use the following (intuitive) fact: the number of crossings between two closed curves, modulo 2, is invariant under small perturbations of any or both of the two closed curves.

2.2 Crossings of closed walks

Observe that any walk α in G induces a corresponding curve on Σ. Generally, if it does not introduce any ambiguity, we may denote this curve by α as well.

Two closed walks α and β in G cross k times if and only if: there are arbitrarily small perturbations of α and β to general position that cross k times, and any small enough perturbation of α and β has at least k intersecting points. Moreover, we can always assume that the crossings of the perturbations occur in a neighborhood of the vertices. We denote by $\mathtt{cr}(\alpha, \beta)$ the number of crossings between α and β. We denote by $\mathtt{cr}_2(\alpha, \beta)$ the modulo 2 value of $\mathtt{cr}(\alpha, \beta)$.

For any closed walk α, the set of closed walks in G that cross α an odd number of times satisfies the so-called 3-path condition. The next lemma states this in an equivalent way for easier use later on. This Lemma 1 and Lemma 3, below, appear as claims in [9]; their proofs are in the full version. We prove them here to make the presentation self-contained and because we use a different language and perspective.

Lemma 1. *Let α and γ be two closed walks such that $\mathtt{cr}_2(\alpha, \gamma) = 1$. Let x and y be two vertices on γ and let π be some walk from x to y. Let γ' be the closed walk defined by concatenating $\gamma[y \to x]$ and π. Let γ'' be the closed walk defined by concatenating $\gamma[x \to y]$ and the reverse of π. Then either $\mathtt{cr}_2(\alpha, \gamma') = 1$ or $\mathtt{cr}_2(\alpha, \gamma'') = 1$.*

Proof. Consider a small perturbation C_α of α and a small perturbation C_γ of γ to general position such that C_α and C_γ cross an odd number of times. Take a small perturbation C_π of π such that: the endpoints of C_π are on $C_\gamma \setminus C_\alpha$, any subpath of C_π that excludes its endpoints is in general position with C_α and with C_γ. (Note that we do not have general position, according to our definition, because there are T-junctions around the endpoints of C_π.) See Figure 1, left.

We can combine pieces of the curves C_α, C_γ and C_π to construct small perturbations of γ' and γ''. Indeed, let us denote by x' and y' the endpoints of C_π such that x' is near x and y' is near y. Let us define $C_{\gamma'}$ as the concatenation of $C_\gamma[y' \to x']$ and C_π. Similarly, we define $C_{\gamma''}$ as the concatenation of $C_\gamma[x' \to y']$ and the reverse of C_π. See

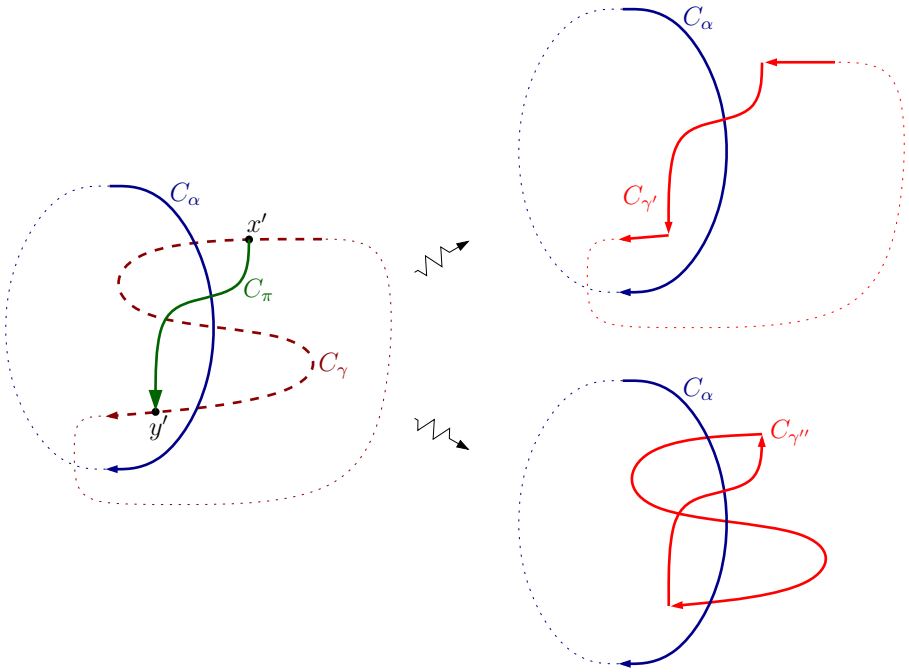

Figure 1: Figure for the proof of Lemma 1. Dotted parts are added to conceptually close the curves; they do not contribute crossings.

Figure 1, right. It is easy to see that $C_{\gamma'}$ is a small perturbation of γ' and $C_{\gamma''}$ is a small perturbation of γ''. Furthermore, by construction we have

$$|X(C_\alpha, C_{\gamma'})| + |X(C_\alpha, C_{\gamma''})| \;=\; |X(C_\alpha, C_\gamma)| + 2 \cdot |X(\alpha, C_\pi)|.$$

Since $|X(C_\alpha, C_\gamma)|$ is odd, either $|X(C_\alpha, C_{\gamma'})|$ or $|X(C_\alpha, C_{\gamma''})|$ is odd. It follows that either γ' or γ'' cross α an odd number of times. □

2.3 Odd-stretch of an embedded graph

The odd-stretch of an embedded graph G is

$$\texttt{oddstr}(G) \;=\; \min\{\text{len}(\alpha) \cdot \text{len}(\beta) \mid \alpha \text{ and } \beta \text{ are closed walks in } G \text{ with } \texttt{cr}_2(\alpha, \beta) = 1\}.$$

We remark the two main differences with the previous stretch: α and β iterate over closed walks, instead of cycles, and the curves can cross an odd number of times, instead of exactly once. A priori, the stretch and the odd-stretch of an embedded graph are different. A posteriori we will see that they are the same. We say that a pair (α, β) of closed walks in G defines $\texttt{oddstr}(G)$ if α and β cross an odd number of times and $\texttt{oddstr}(G) = \text{len}(\alpha) \cdot \text{len}(\beta)$.

Lemma 2. *Let (α^*, β^*) be two curves defining $\texttt{oddstr}(G)$. Then α^* and β^* are cycles.*

Proof. Assume, for the sake of contradiction, that β^* is not a cycle. The other case, when α^* is not a cycle, is symmetric. Let x be vertex that is repeated along β^*. We can then split β^* into two closed walks β' and β'' through x such that

$$\text{len}(\beta') < \text{len}(\beta^*) \quad \text{and} \quad \text{len}(\beta'') < \text{len}(\beta^*). \tag{1}$$

Because of Lemma 1 applied with $\alpha = \alpha^*$, $\gamma = \beta^*$, $x = y$, $\pi = x$, $\gamma' = \beta'$ and $\gamma'' = \beta''$, either β' or β'' cross α^* an odd number of times. This means that either (α^*, β') or (α^*, β'') is a pair of closed walks considered in the definition of $\texttt{oddstr}(G)$ and, because of (1), it has a smaller product of lengths than (α^*, β^*). In either case we reach a contradiction with the definition of (α^*, β^*). □

Lemma 3. *The odd-stretch of G and the stretch of G are the same.*

Proof. Clearly, the odd-stretch is not larger than the stretch. We next show the other inequality.

We want to work under the assumption that there is a unique shortest path between any pair of vertices of G. For each edge $e \in E(G)$, choose a value r_e and perturb the edge weight w_e by the amount $r_e \cdot \varepsilon$. Let G_ε be the resulting graph with perturbed edge weights. If we choose the values r_e, $e \in E(G)$, linearly independent over the rationals, then the graph G_ε has a unique shortest path between any two vertices for all sufficiently small $\varepsilon > 0$. If we show that $\texttt{str}(G_\varepsilon) = \texttt{oddstr}(G_\varepsilon)$ for all sufficiently small $\varepsilon > 0$, it follows by continuity that $\texttt{str}(G) = \texttt{oddstr}(G)$. Thus, we henceforth only need to consider the case where *there is a unique shortest path between any pair of vertices.*

Let (α^*, β^*) be a pair of closed walks that define the odd-stretch. Because of Lemma 2, α^* and β^* are cycles. We **claim** that α^* and β^* cross exactly once, and thus $\texttt{str}(G) = \texttt{oddstr}(G)$. Assume, for the sake of contradiction, that the cycles α^* and β^* cross more than once; so they cross at least three times. Let x and y be two distinct vertices of $V(\alpha^*) \cap V(\beta^*)$ where crossings of α^* and β^* occur, such that the intersection $\alpha^* \cap \beta^*$ does not contain a path connecting x to y. Let π be the shortest path in G from x to y. The path π is not contained in α^* or in β^*. Consider the case when π is not contained in β^*; the other case is symmetric. Since shortest paths are unique, we have

$$\text{len}(\pi) < \text{len}(\beta^*[x \to y]) \quad \text{and} \quad \text{len}(\pi) < \text{len}(\beta^*[y \to x]).$$

Consider the closed walk γ' obtained by concatenating $\beta^*[y \to x]$ with π and the closed walk γ'' obtained by concatenating $\beta^*[x \to y]$ with the reverse of π. (Note that γ' and γ'' are not necessarily cycles because π may cross β^*.) We have

$$\text{len}(\gamma') < \text{len}(\beta^*) \quad \text{and} \quad \text{len}(\gamma'') < \text{len}(\beta^*).$$

Moreover, because of Lemma 1, some $\gamma^* \in \{\gamma', \gamma''\}$ crosses α^* an odd number of times. Therefore the pair of closed walks (α^*, γ^*) contradicts the choice of (α^*, β^*). □

3 Algorithm using surgery

We will use the following two properties:

Lemma 4 ([2])**.** *Let G be a graph with m vertices embedded in a surface of genus g.*

- *We can compute a shortest non-separating cycle in time $O(g^2 m \log m)$ with high probability, or in time $O(g^2 m \log^2 m)$ in the worst case.*
- *For any given non-separating cycle α, we can compute a shortest cycle of G that crosses α exactly once in time $O(gm \log m)$ with high probability, or in time $O(gm \log^2 m)$ in the worst case.*

Lemma 5. *Let α be a shortest non-separating cycle in G. For any two vertices x and y on α, α contains a shortest path from x to y or from y to x.*

Algorithm ComputeStretchSurgery
Input: graph G embedded in a surface Σ of genus g
Output: stretch of G
1. $i \leftarrow 1$
2. $(G_1, \Sigma_1) \leftarrow (G, \Sigma)$
3. str $\leftarrow \infty$
4. **while** Σ_i not the sphere and $|V(G_i)| \leq g \cdot |V(G)|$ **do**
5. $\quad \alpha_i \leftarrow$ shortest non-separating cycle in G_i
6. $\quad \beta_i \leftarrow$ shortest cycle crossing α_i exactly once
7. $\quad$ str $\leftarrow \min\{\text{str}, \text{len}(\alpha_i) \cdot \text{len}(\beta_i)\}$
8. $\quad (G_{i+1}, \Sigma_{i+1}) \leftarrow$ cut (G_i, Σ_i) along α_i and attach disks to the boundaries
9. $\quad i \leftarrow i + 1$
10. **return** str

Figure 2: Algorithm ComputeStretchSurgery to compute the stretch factor of a graph embedded on an orientable surface.

The algorithm for computing the stretch of an embedded graph is given in Figure 2. We first discuss its time complexity and then its correctness.

Lemma 6. *Algorithm* ComputeStretchSurgery *has time complexity* $O(g^4 n \log n)$ *with high probability, or* $O(g^4 n \log^2 n)$ *in the worst case, where* n *is the number of vertices in* G.

Proof. Because of Lemma 4, each iteration of the while loop needs $O(g_i^2 n_i \log n_i)$ time whp, or $O(g_i^2 n_i \log^2 n_i)$ in the worst case, where n_i is the number of vertices in G_i and g_i is the genus of Σ_i. Since $n_i = O(gn)$ because of the condition for iterating the while loop and $g_i \leq g$, each iteration of the while loop takes $O(g^2(gn)\log(gn)) = O(g^3 n \log n)$ time whp, or $O(g^3 n \log^2 n)$ time in the worst case. Since there are at most g iterations of the while loop, the lemma follows. □

Lemma 7. *Let* α *be a shortest non-separating cycle and let* β *be a shortest cycle crossing* α *exactly once. Let* G' *be the embedded graph obtained from* G *by cutting along* α *and attaching a disk to the boundaries. The stretch of* G *is the minimum between* $\text{len}(\alpha)\cdot\text{len}(\beta)$ *and the stretch of* G'.

Proof. Let Σ be the surface where G is embedded and let Σ' be the surface where G' is embedded. Since any two closed curves of G' that cross an odd number of times in Σ' also cross an odd number of times in Σ, it is clear that

$$\texttt{str}(G) \;\leq\; \min\{\texttt{str}(G'), \text{len}(\alpha) \cdot \text{len}(\beta)\}.$$

Thus, we have to argue the other inequality. If (α, β) define the stretch of G, then this is obvious.

Let us assume that (α, β) do not define the stretch of G; it holds that $\texttt{str}(G) < \text{len}(\alpha) \cdot \text{len}(\beta)$. Let (γ^*, σ^*) be the pair of cycles that define the stretch of G. If there are several such pairs, we choose one such that $\texttt{cr}(\gamma^*, \alpha) + \texttt{cr}(\sigma^*, \alpha)$ is minimum. We distinguish 3 cases depending on the values of $\texttt{cr}(\gamma^*, \alpha)$ and $\texttt{cr}(\sigma^*, \alpha)$:

Case $\texttt{cr}(\gamma^*, \alpha) = \texttt{cr}(\sigma^*, \alpha) = 0$**.** In this case, γ^* and σ^* keep crossing in Σ', and thus $\texttt{str}(G) = \texttt{str}(G')$.

Case $\mathtt{cr}(\gamma^*, \alpha) = 1$ or $\mathtt{cr}(\sigma^*, \alpha) = 1$. This case cannot actually happen. Let us assume that $\mathtt{cr}(\gamma^*, \alpha) = 1$; the other case is symmetric. Since γ^* crosses α once and β is a shortest cycle crossing α once, we have $\mathrm{len}(\beta) \leq \mathrm{len}(\gamma^*)$. Using that α is a shortest non-separating cycle we would have

$$\mathtt{str}(G) \;=\; \mathrm{len}(\gamma^*) \cdot \mathrm{len}(\sigma^*) \;\geq\; \mathrm{len}(\beta) \cdot \mathrm{len}(\alpha).$$

Case $\mathtt{cr}(\gamma^*, \alpha) \geq 2$ or $\mathtt{cr}(\sigma^*, \alpha) \geq 2$. This case cannot actually happen. Let us assume that $\mathtt{cr}(\gamma^*, \alpha) \geq 2$; the other case is symmetric. Let x and y be two crossings of γ^* and α that are consecutive along α. Because of Lemma 5, α contains a shortest path between x and y. Let π denote this shortest path. We can use π and γ^* to construct two cycles γ' and γ'' that are not longer than γ^* and that cross α fewer times than γ^*. Because of Lemma 1, some $\tilde{\gamma} \in \{\gamma', \gamma''\}$ crosses σ^* an odd number of times. The pair $(\tilde{\gamma}, \sigma^*)$ contradicts the choice of (γ^*, σ^*).

This finishes all cases. (Note that the second and third cases are not mutually exclusive.) □

Lemma 7 shows correctness of the algorithm if the condition $|V(G_i)| \leq g \cdot |V(G)|$ is true for each $i = 1, \ldots, g$. We next argue why we can finish the search if at some iteration $|V(G_i)| > g \cdot |V(G)|$.

Lemma 8. *If, for some i, α_i has more than $|V(G)|$ vertices, then for any $\ell \geq i$*

$$\mathtt{str}(G) \;=\; \min\{\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \mid j = 1, \ldots, \ell - 1\}.$$

Proof. Assume, for the sake of this proof, that in the algorithm ComputeStretchSurgery we drop testing the condition $|V(G_i)| \leq g \cdot |V(G)|$. The algorithm then makes exactly g iterations and computes cycles α_j, β_j for each $j = 1, \ldots, g$. Because of Lemma 7 it holds

$$\mathtt{str}(G) \;=\; \min\{\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \mid j = 1, \ldots, g\}.$$

Assume that, at some iteration, the shortest non-separating cycle α_i in G_i, has more than $|V(G)|$ vertices. For each $k < i$, the cycle α_i corresponds to a closed walk W_k in the graph G_k. Moreover, the walk W_k does not cross the cycle α_k, for each $k < i$. Since α_i has more than $|V(G)|$ vertices, W_1 repeats some vertex of $G_1 = G$. This means that W_1 is not a cycle.

Let k be the maximum index, $1 \leq k < i$ such that W_k is *not a cycle* in G_k; thus W_{k+1} is a cycle in G_{k+1}. Since W_1 is not a cycle and W_i is a cycle, the index k is well defined. See Figure 3, left and center. Let v be a vertex of G_k that is repeated in W_k. Cutting G_k through α_k produces two copies α_k' and α_k'' of α_k. Let v' and v'' be the corresponding copies of v. We can form a closed walk W_k' in G_k by taking the subwalk of W_k from the first appearance of v until the second. This closed walk W_k' crosses α_k once. Therefore $\mathrm{len}(\beta_k) \leq \mathrm{len}(W_k') < \mathrm{len}(W_k) = \mathrm{len}(\alpha_i) \leq \mathrm{len}(\alpha_j)$ for each $j \geq i$. We conclude that, for each $j \geq i$,

$$\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \;>\; \mathrm{len}(\beta_k) \cdot \mathrm{len}(\alpha_j) \;\geq\; \mathrm{len}(\beta_k) \cdot \mathrm{len}(\alpha_k).$$

It follows that

$$\mathrm{len}(\alpha_k) \cdot \mathrm{len}(\beta_k) \;\leq\; \min\{\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \mid j = i, \ldots, g\}.$$

Since $k < i \leq \ell$ we conclude that

$$\begin{aligned} \mathtt{str}(G) \;&=\; \min\{\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \mid j = 1, \ldots, g\} \\ &=\; \min\{\mathrm{len}(\alpha_j) \cdot \mathrm{len}(\beta_j) \mid j = 1, \ldots, \ell - 1\}. \end{aligned}$$

□

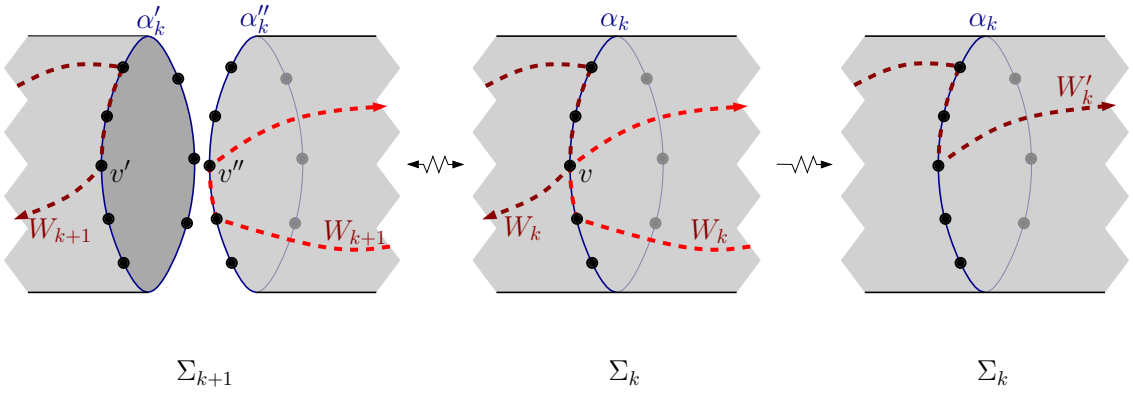

Figure 3: Figure for the proof of Lemma 8.

Theorem 9. *Let G be a graph with n vertices embedded in a surface of genus g. We can compute the stretch of G in time $O(g^4 n \log n)$ with high probability, or in time $O(g^4 n \log^2 n)$ in the worst case.*

Proof. The time bound follows from Lemma 6. We next argue the correctness of the algorithm. The while loop has the following simple invariant: at the start of iteration i the variable *str* stores the value $\min(\{\text{len}(\alpha_j) \cdot \text{len}(\beta_j) \mid j = 1, \ldots, i-1\} \cup \{\infty\})$. If $|V(G_i)| \leq g \cdot |V(G)|$ for each iteration, then the algorithm finishes with $i = g + 1$, the surface Σ_{g+1} is a sphere, and correctness follows from Lemma 7. Let us consider the other case, when at some iteration ℓ we have $|V(G_\ell)| > g \cdot |V(G)|$. Since there are at most g iterations, there must have been at least one iteration $i < \ell$ where the increase in vertices, given $|V(\alpha_i)|$, was larger than $|V(G)|$. Correctness then follows from Lemma 8. □

4 Algorithm using homology

We are going to describe an algorithm based on homology with time bound $O(n^2 \log n + n^2 g + n g^3)$. We start with a quick review of homology (over $\mathbb{Z}_2$), and describe the properties that we will use. For a comprehensive treatment, see Hatcher [8, Chapter 2] or Edelsbrunner and Harer [5, Chapter IV].

Homology of cycles. A ***1-chain*** is a formal sum of edges, $\alpha = \sum_{e \in E(G)} \alpha_e e$, where each $\alpha_e \in \mathbb{Z}_2$. We can identify each 1-chain α with the subset of edges $E(\alpha) = \{e \in E(G) \mid \alpha_e = 1\}$. The set of 1-chains forms a vector space of dimension $|E(G)|$, denoted by C_1. The sum in C_1 corresponds to the symmetric difference when looking at the corresponding subsets of edges. A 1-chain α is a ***1-cycle*** if $E(\alpha)$ has even degree in each vertex. The set of 1-cycles forms another vector space, denoted by Z_1. This vector space is sometimes called the *cycle space* and has dimension $|E(G)| - |V(G)| + 1$. Each face f of G defines a 1-cycle $\partial(f)$, which corresponds to the edges appearing in the facial walk of f exactly once. The subspace of Z_1 generated by $\{\partial(f) \mid f \text{ a face of } G\}$, is called the boundary space B_1. The (first) homology group H_1 is the quotient $\mathsf{Z}_1/\mathsf{B}_1$, which is also a vector space. Standard results in algebraic topology imply that the vector space H_1 has dimension $2g$, if G is (cellularly) embedded in an orientable surface of genus g. Each element of H_1, called a homology class, corresponds to a subset of 1-cycles. Two 1-cycles α and β are in the same homology class if there exist faces $f_1, \ldots, f_k$ such that $\alpha = \beta + \partial(f_1) + \cdots + \partial(f_k)$. We say that 1-cycles in the same homology class are ***homologous***. For any 1-cycle α, we use $[\alpha]$ to denote its homology class.

Each cycle α in G defines a 1-cycle in a natural way; with a slight abuse of notation, we also denote this 1-cycle by α. For each 1-cycle α in G, we can find cycles $\alpha_1, \ldots, \alpha_k$ such that $E(\alpha) = \bigcup_{i=1}^{k} E(\alpha_i)$. Thus, we can see each 1-cycle as the union of a finite subset of cycles. The length of a 1-cycle α, denoted by $\mathrm{len}(\alpha)$, is the sum of the weights of the edges in $E(\alpha)$, or $|E(\alpha)|$ if the graph is unweighted.

Greedy homology cycle basis. A ***greedy homology 1-cycle basis*** is a sequence $(\gamma_1, \ldots, \gamma_{2g})$ of 1-cycles with the following property: for each i, the 1-cycle γ_i is a shortest cycle such that the homology class $[\gamma_i]$ is linearly independent of the subspace spanned by $[\gamma_1], [\gamma_2], \ldots, [\gamma_{i-1}]$. We are implicitly using that H_1 has dimension $2g$. We note an easy property that is probably folklore.

Lemma 10. *Let $\Gamma = (\gamma_1, \ldots, \gamma_{2g})$ be a greedy homology 1-cycle basis. Then each γ_i in Γ is a cycle.*

Proof. Assume, for the sake of contradiction, that some γ_i is not a cycle. Let $\alpha_1, \ldots, \alpha_k$ be cycles such that $\gamma_i = \alpha_1 + \cdots + \alpha_k$. Since $[\gamma_i] = [\alpha_1] + \cdots + [\alpha_k]$ is linearly independent of $\{[\gamma_1], [\gamma_2], \ldots, [\gamma_{i-1}]\}$, some $[\alpha_j]$ is linearly independent of $\{[\gamma_1], [\gamma_2], \ldots, [\gamma_{i-1}]\}$. Such α_j is shorter than γ_i, which leads to a contradiction with the condition of Γ being a greedy homology 1-cycle basis. □

Erickson and Whittlesey [7] show how to compute a greedy homology 1-cycle basis in time $O(n^2 \log n + n^2 g + n g^3)$. See Erickson [6] for an overview of the techniques. We will use this as a subroutine in our algorithm.

Crossings of 1-cycles. We have defined $\mathtt{cr}_2(\cdot, \cdot)$ for closed walks. This definition can be extended to 1-cycles using bilinearity: for any 1-cycles α and β, we consider cycles $\alpha_1, \ldots, \alpha_k$ and $\beta_1, \ldots, \beta_\ell$ such that $E(\alpha) = \bigcup_{i=1}^{k} E(\alpha_i)$ and $E(\beta) = \bigcup_{j=1}^{\ell} E(\beta_j)$, and define

$$\mathtt{cr}_2(\alpha, \beta) \;=\; \sum_{i=1}^{k} \sum_{j=1}^{\ell} \mathtt{cr}_2(\alpha_i, \beta_j) \pmod 2.$$

One can see that this definition is independent of the choice of cycles α_i and β_j. Indeed, one can make a perturbed copy G' of G, so that G and G' are in general position. Then $\mathtt{cr}_2(\alpha, \beta)$ is the modulo 2 value of the number of crossings between $E(\alpha)$ and the copy of $E(\beta)$ in G'. Hence, it is clear that the definition is independent of the decomposition into cycles.

Next, we note that for any face f and any 1-cycle β we have $\mathtt{cr}_2(\partial f, \beta) = 0$. Using the bilinearity of $\mathtt{cr}_2$ we conclude that for any boundary 1-cycle $\alpha \in \mathsf{B}_1$ and any 1-cycle $\beta \in \mathsf{Z}_1$ we have $\mathtt{cr}_2(\alpha, \beta) = 0$. It follows that if α and α' are homologous 1-cycles, then $\mathtt{cr}_2(\alpha, \beta) = \mathtt{cr}_2(\alpha', \beta)$ for each $\beta \in \mathsf{Z}_1$. In particular, we can define the (parity of the) crossings between homology cycles $\mathtt{cr}_2 \colon \mathsf{H}_1 \times \mathsf{H}_1 \to \mathbb{Z}_2$ by $\mathtt{cr}_2([\alpha], [\beta]) = \mathtt{cr}_2(\alpha, \beta)$. For readers familiar with Algebraic Topology, it may be useful to mention that this is a special case of the topological intersection number defined using the cup product.

Algorithm. The eventual algorithm for computing the stretch of an embedded graph is given in Figure 4.

Lemma 11. *Let $\Gamma = (\gamma_1, \ldots, \gamma_{2g})$ be a greedy homology 1-cycle basis. Then*

$$\mathtt{oddstr}(G) \;=\; \min\{\mathrm{len}(\gamma_i) \cdot \mathrm{len}(\gamma_j) \mid \gamma_i \textit{ and } \gamma_j \textit{ in } \Gamma,\ \mathtt{cr}_2(\gamma_i, \gamma_j) = 1\}. \tag{2}$$

Algorithm ComputeStretchHomology
Input: graph G embedded in surface Σ of genus g
Output: stretch of G
1. $(\gamma_1, \ldots, \gamma_{2g}) \leftarrow$ greedy homology 1-cycle basis for G
2. $\text{str} \leftarrow \infty$
3. **for** $1 \le i < j \le 2g$ **do**
4. **if** $\texttt{cr}_2(\gamma_i, \gamma_j) = 1$ **then**
5. $\text{str} \leftarrow \min\{\text{str}, \text{len}(\gamma_i) \cdot \text{len}(\gamma_i)\}$
6. **return** str

Figure 4: Algorithm ComputeStretchHomology to compute the stretch factor of a graph embedded on an orientable surface.

Proof. Let R denote the value on the right hand side of (2). Since each γ_i in Γ is a cycle due to Lemma 10, we have $\texttt{oddstr}(G) \le R$. We have to argue the other inequality.

Let α and β be a pair of cycles attaining $\texttt{oddstr}(G)$. Because Γ is a homology 1-cycle basis, there are subsets of indices $I_\alpha, I_\beta \subseteq \{1, \ldots, 2g\}$ such that

$$[\alpha] = \sum_{i \in I_\alpha} [\gamma_i] \quad \text{and} \quad [\beta] = \sum_{j \in I_\beta} [\gamma_j].$$

Because Γ is a greedy homology 1-cycle basis, we have

$$\forall i \in I_\alpha : \ \text{len}(\gamma_i) \le \text{len}(\alpha) \quad \text{and} \quad \forall j \in I_\beta : \ \text{len}(\gamma_j) \le \text{len}(\beta).$$

Because of bilinearity of $\texttt{cr}$ we have

$$1 \ = \ \texttt{cr}_2([\alpha], [\beta]) \ = \ \sum_{i \in I_\alpha} \sum_{j \in I_\beta} \texttt{cr}_2([\gamma_i], [\gamma_j]).$$

Therefore there exists some $i \in I_\alpha$ and $j \in I_\beta$ such that $\texttt{cr}_2([\gamma_i], [\gamma_j]) = 1$. We conclude that

$$R \ \le \ \text{len}(\gamma_i) \cdot \text{len}(\gamma_j) \ \le \ \text{len}(\alpha) \cdot \text{len}(\beta) \ = \ \texttt{oddstr}(G).$$

□

Theorem 12. *Let G be a graph with n vertices embedded in a surface of genus g. We can compute the stretch of G in time $O(n^2 \log n + n^2 g + n g^3)$.*

Proof. Consider the algorithm ComputeStretchHomology. The algorithm correctly returns the stretch of G because of Lemma 11. We have to analyze its running time. Finding a greedy homology 1-cycle basis for G takes $O(n^2 \log n + n^2 g + n g^3)$ time [7]. We then have to check $O(g^2)$ times whether two cycles intersect an odd number of times. Each such test can be done in $O(n)$ time deforming one of the cycles, for example to lie in the vertex-face incidence graph. Thus all tests together take time $O(g^2 n)$. The result follows. □

Note that the running time of the algorithm is the running time to compute a greedy homology basis plus $O(g^2 n)$. Thus, an improvement over the result of Erickson and Whittlesey [7] readily implies an improvement in the time bound of this theorem.

Acknowledgments

We are grateful to Daniel Štefankovič for some earlier discussions.

References

[1] K. J. Börözky, J. Pach, and G. Tóth. Planar crossing numbers of graphs embeddable in another surface. *Int. J. Found. Comput. Sci.*, 17(5):1005–1016, 2006.

[2] S. Cabello, E. W. Chambers, and J. Erickson. Multiple-source shortest paths in embedded graphs. *SIAM J. Comput.*, 42(4):1542–1571, 2013.

[3] S. Cabello and B. Mohar. Adding one edge to planar graphs makes crossing number and 1-planarity hard. *SIAM J. Comput.*, 42(5):1803–1829, 2013.

[4] H. Djidjev and I. Vrt'o. Planar crossing numbers of graphs of bounded genus. *Discrete & Computational Geometry*, 48(2):393–415, 2012.

[5] H. Edelsbrunner and J. Harer. *Computational Topology, An Introduction.* American Mathematical Society, January 2010.

[6] J. Erickson. Combinatorial optimization of cycles and bases. In A. Zomorodian, editor, *Advances in Applied and Computational Topology*, volume 70 of *Proceedings of Symposia in Applied Mathematics*, page 195?228. AMS, 2012.

[7] J. Erickson and K. Whittlesey. Greedy optimal homotopy and homology generators. In *Proc. SODA 2005*, pages 1038–1046, 2005.

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

[9] P. Hliněný and M. Chimani. Approximating the crossing number of graphs embeddable in any orientable surface. In *Proc. SODA 2010*, pages 918–927, 2010.