Linear-Time Vertex-Connectivity for Graphs of Bounded Genus

Sergio Cabello
Faculty of Mathematics and Physics, University of Ljubljana, Slovenia
Institute of Mathematics, Physics and Mechanics, Ljubljana, Slovenia

Alexander Dobler
TU Wien, Austria

Gašper Fijavž
Faculty of Computer and Information Science, University of Ljubljana, Slovenia

Thekla Hamm
Department of Mathematics and Computer Science, TU Eindhoven, The Netherlands

Mirko H. Wagner
Theoretical Computer Science, Osnabrück University, Germany

Abstract

We provide a new linear-time algorithm for determining the vertex-connectivity of graphs with bounded genus. This generalizes and streamlines a linear-time algorithm for graphs with bounded crossing number which was recently obtained by Biedl, Bose and Murali [ESA 2024]. Compared to applying the even more recent fixed parameter linear-time algorithm for deciding bounded vertex-connectivity announced by Korhonen [STOC 2025] to graphs of bounded genus,our algorithm is far simpler, its correctness easier to establish, and it makes use of geometric ideas, as is natural for surface-embedded graphs.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Fixed parameter tractability

Keywords and phrases vertex-connectivity, graphs on surfaces, genus of a graph

Digital Object Identifier 10.4230/LIPIcs.ESA.2026.56

Funding *Sergio Cabello*: Funded in part by the Slovenian Research and Innovation Agency (P1-0297, N1-0218, N1-0285, J1-70045). Funded in part by the European Union (ERC, KARST, project number 101071836). Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.
Alexander Dobler: Vienna Science and Technology Fund (WWTF) grant [10.47379/ICT19035].

Acknowledgements This work was started during the Crossing Number Workshop 2024 that took place in Montserrat, Spain. We thank the organizers and participants for providing a fruitful research environment.

1 Introduction

The *vertex-connectivity* of a graph is the size of a smallest set of vertices whose removal results in at least two disconnected components or a single connected component consisting of only one vertex. It is used to measure the fault-tolerance or reliability of networks [8, 19] and also has important implications for graph algorithm design [1, 22, 24]. Until recently, most algorithms for computing vertex-connectivity on general graphs worked by solving a polynomial number of max-flow problems [18, 21]. In particular, an impressive recent result by Li, Nanongkai, Panigrahi, Saranurak, and Yingchareonthawornchai [29] gives a randomized (Monte Carlo) algorithm to compute the vertex-connectivity of arbitrary graphs

in near-linear time. This last result leverages the recent algorithm [11] to solve max-flow instances in near-linear time. For the special case of small vertex-connectivity, Korhonen announced a linear-time algorithm as a corollary of a more powerful algorithm for computing so called *k-lean tree decompositions* [28]. The full version of the paper [27] spans 110 pages and combines a breadth of technically complicated graph structural ideas.

In contrast, we give a far simpler algorithm for graphs embedded on a fixed surface, i.e. bounded-genus graphs to which Korhonen's result can, in principle, be applied, which continues and can contribute useful techniques to the research line on vertex-connectivity algorithms for drawn or embedded graphs. For planar graphs a linear-time algorithm can be achieved using the fact that minimal separating vertex sets correspond to cycles in a planar auxiliary extension of the instance and the existence of short cycles in planar graphs can be tested in linear time [14]. This makes beyond-planar graph classes a natural next target. Recently, a linear time algorithm for a subclass of 1-planar graphs was given [4]. Specifically, whenever graphs are given with 1-planar drawings in which no pair of crossing edges induces a matching, minimal separating vertex sets correspond to constant-diameter graphs in a planar auxiliary extension of the planarized instance. Their existence can once again be tested in linear time. This was generalized by Biedl, Bose, and Murali [2, 3] to graphs which are drawn in a way that each pair of crossing edges are connected via a path at constant dual-distance from the respective crossing (the authors introduce this measure as *ribbon radius*). This does not only include the aforementioned subclass of 1-planar graphs but also graphs of constant crossing number and optimal 2- or 3-planar graphs. In that work it was remarked that a linear-time algorithm for deciding the vertex-connectivity of toroidal graphs, i.e. graphs that can be embedded without crossings on a torus, cannot be achieved in a similar way. This is because there are simple examples in which minimal separating vertex sets can be arbitrarily far apart [3, Figure 7]. We (re-)obtain the following by a linear time algorithm that crucially makes use of a surface embeddability of an input:

▶ **Theorem 1.** *Let Σ be a fixed surface, orientable or not. There is a linear-time algorithm to compute the vertex-connectivity and a smallest vertex-separator of graphs embeddable in Σ.*

This result also generalizes the constant-crossing number case for which our approach streamlines the step of finding a minimal separating vertex set of constant size and diameter. In particular both [2] and [4] invoke Courcelle's theorem or dynamic programming on a constant-treewidth graph. They are only able to MSO-encode separating vertex sets that are local in the sense that they correspond to a *connected* constant-size subgraph in an auxiliary graph – this is where the constant ribbon radius becomes important. We instead rely on Bonsma's linear-time algorithm for constant-size subgraph isomorphism on constant genus graphs [5]. In addition to plain subgraph-isomorphism testing, we employ labels that encode the homology of cycles related to minimal vertex-separators in an auxiliary graph.

We can adapt our approach to the edge-connectivity, i.e. the minimum size of an edge set whose removal disconnects the graph, of simple graphs embedded on surfaces:

▶ **Theorem 2.** *Let Σ be a fixed surface, orientable or not. There is a linear-time algorithm to compute the edge-connectivity and a smallest edge-cut of simple graphs embeddable in Σ.*

Similarly as for vertex-connectivity, Korhonen's algorithm already implies a linear time algorithm for deciding the edge-connectivity for graphs of bounded genus, by essentially reducing edge-connectivity to vertex-connectivity. We remark, that our algorithm does not suffer from such a "detour"; the subgraphs we search for when deciding edge-connectivity are different and even simpler than those we use for vertex-connectivity. Turning to algorithms

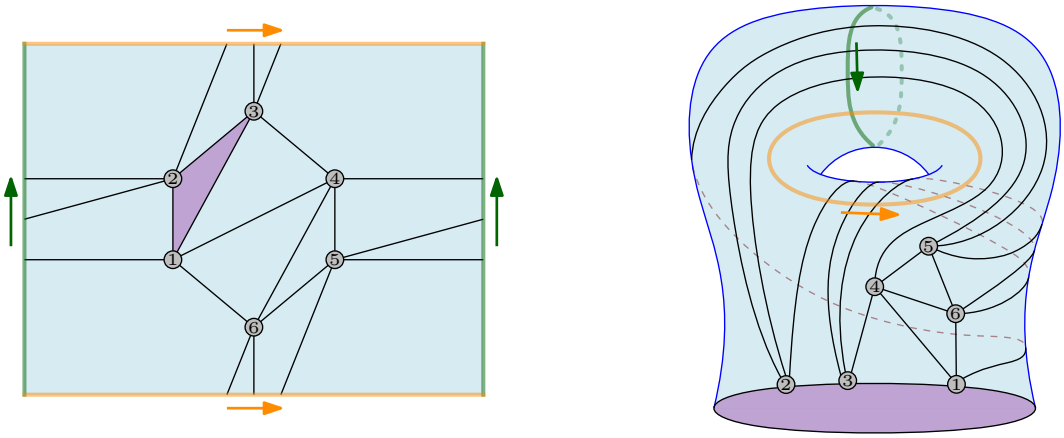

Figure 1 An embedding of K_6 in the torus. On the left side we have the embedding shown in a fundamental polygon; the torus is obtained by gluing the two orange boundaries along the indicated orientation and gluing the two green boundaries also along the indicated orientation. A triangular face is indicated in dark magenta. On the right side we have the same embedding (combinatorially equivalent, to be precise) displayed in a 3-dimensional drawing of the torus.

specifically designed for embedded graphs, the edge-connectivity of planar graphs can long be computed in linear time [14]. For graphs on surfaces, Erickson, Fox, and Lkhamsuren [15] gave a linear time algorithm for the minimum edge-cut for unweighted graphs embedded on a fixed, orientable surface and, recently, Chambers, Erickson, Fox, and Nayyeri [9] provided algorithms to compute in $O(n \log\log n)$ time the minimum edge-cut for *weighted* graphs embedded on a fixed, orientable surface.

Regarding the computation of edge-connectivity in general graphs, Karger [23] provides a randomized (Monte Carlo) algorithm that computes the edge-connectivity in near-linear time. Kawarabayashi and Thorup [26] provided the first deterministic near-linear-time algorithm; the running time has been further improved by Henzinger, Rao, and Wang [20].

Input. We assume that the input to our algorithms is a simple graph G embedded in Σ. An embedding of a graph is described combinatorially: for each vertex v of G, we give the circular order of the edges around v. See Figures 1 and 2 for examples. If the surface is orientable, this information suffices. If the surface is non-orientable, then we also have to give a signature along each edge, which essentially tells when we pass from one orientation to the other. Each face of an embedding is an open disk, but the same point may appear multiple times on its boundary. A face can be recovered in a combinatorial embedding by following the edges and at each vertex, "rotating" to the next incident edge. We refer the reader to some comprehensive treatments, such as [31]; see [12] for a survey.

However, for a fixed surface Σ there is no substantial difference whether we assume that the input is given as an abstract graph G *embeddable* in Σ or as a *combinatorial embedding* of G in Σ – we can test in linear time whether G can be embedded in Σ, and obtain a possible embedding, using an algorithm by Mohar [30]; see [25] for a follow-up improvement.

Technical overview

A *vertex-separator* in a graph G is a set $S \subset V(G)$ such that $G - S$ has at least two connected components or a single vertex. Computing the vertex-connectivity of G is equivalent to computing the size of a smallest vertex-separator in G. An *edge-cut* in a graph G is a set of

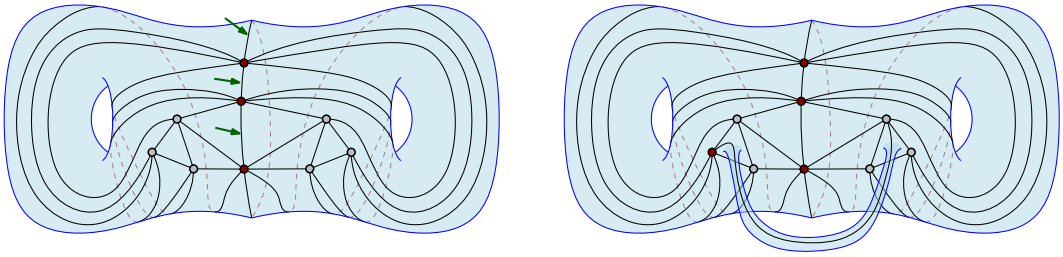

Figure 2 Left: An embedding in the double torus of the union of two K_6 with three vertices identified. It is obtained from two copies of the embedding in Figure 1 by cutting out the open, dark magenta faces and identifying their boundaries. The graph has a 3-vertex separator, indicated with red vertices, even if some edges between the identified vertices (green arrows) are deleted. Right: we can add an extra edge adding a new handle; this graph has a 4-vertex separator.

edges of the form $\delta(U) = \{uv \in E(G) \mid u \in U, v \in V(G) \setminus U\}$, where U is a subset of vertices such that $\emptyset \neq U \subsetneq V(G)$. It is easy to see that computing the edge-connectivity of G is equivalent to computing the size of a smallest edge-cut in G.

Through the paper, we assume that **Σ is a fixed surface of Euler genus g**, orientable or not. Using Euler's formula for graphs embedded on Σ, we can reduce the problem to deciding whether the input graph has an edge-cut or a vertex-separator of size k, for a given $k \leq 5$. In fact, the method we employ for $k \leq 5$ works for any constant k.

We use homology over $\mathbb{Z}_2$, a tool from algebraic topology. In a nutshell, it allows us to identify whether a family Γ of cycles in a graph embedded in Σ *splits* the surface: cutting the surface along all the cycles in Γ we get at least two connected subsurfaces (with boundary). We do not use homology for the input graph directly, but for some other derived graphs, such as the dual graph or a graph representing the incidences between vertices and faces.

We exploit that homology can be encoded locally. More precisely, we can preprocess a graph G embedded on Σ and attach bit-labels of length $O(g)$ to each edge of G such that, to identify the homology type of a family Γ of cycles in G, it suffices to look at the labels of the edges participating in Γ. In particular, we can detect whether a family Γ of cycles splits the surface Σ by looking at the labels attached to its edges. We call these bit-labels *annotations*.

Let us provide some intuition why the use of homology and derived graphs is natural. Let A be a set of edges or vertices such that $G - A$ has at least two connected components; set G_1 and G_2 such that $G_1 \cup G_2 = G - A$ and $G_1 \cap G_2 = \emptyset$. When we look at $G - A$ as embedded in Σ, there must be a minimal family Γ of closed curves on Σ that are disjoint from $G_1 \cup G_2$ and separate G_1 from G_2, that is, any curve connecting a vertex from G_1 to a vertex of G_2 intersects some curve of Γ. The closed curves of Γ can intersect the original graph G only in points inside A because $G_1 \cup G_1 \cup A = G$, and have to go through some points of A because G is connected. We can then perturb the curves of Γ so that each closed curve of Γ is contained in the dual graph, if A is a set of edges, or in the face-vertex incidence graph, if A is a set of vertices. Thus, we are looking for a family of cycles with few edges in some graph that is derived from the input graph G and the family should split the surface.

Consider first the case of deciding whether the given embedded graph G has an edge-cut of size k, for a given constant k. We use the dual graph G^*, which may have parallel edges. It is known, see for example [9], that a set A of edges in G is an edge cut of G if and only if the corresponding dual edges form a family of cycles with trivial homology; this is a condition on the annotations of the edges in the cycles. Therefore, we have reduced the problem to finding a subgraph in the dual graph that has k edges (some may be parallel), is a union of cycles, and has certain annotations on the edges.

In the case of vertex-connectivity we consider cycles in the vertex-face incidence graph. Again, we have to look for a family Γ of cycles with few edges overall and the correct annotations, so that Γ splits the surface. However, now we cannot guarantee that each of the subsurfaces contains at least one vertex. This issue does not show up for edge cuts because any set of cycles in the dual graph that splits the surface contains at least one face of the dual graph in each subsurface, which corresponds to a vertex of the primal graph.

To ensure that we indeed have at least one vertex in each subsurface we consider patterns that consist of two parts. The first part uses homology annotations and takes care that we are indeed splitting the surface into two subsurfaces; we encode this using what we call *separator graph*. The second part makes sure that on each of the subsurfaces there is at least one vertex; we call this part a *patch*. The patch and separator graph share some vertices, and the patterns overall size is $O(gk)$. We also use labels on the vertices of the derived graphs to distinguish their different roles. For convenience, we do not use the vertex-face incidence graph directly, but a graph derived from it called the polyhedral extension.

Summarizing, we reduce the problem of finding an edge-cut or a vertex-separator of size k in G to the problem of finding a certain pattern in a graph derived from G that has vertex and edge labels. The pattern has size $O(gk)$ and, for each pattern, we have to consider the $2^{O(gk)}$ different annotations of the edges and filter out those that do not lead to a solution.

To test whether a pattern (with labels) appears in the derived graph we adapt a result of Bonsma [6]. Since the original algorithm is done for counting or reporting patterns and does not consider labels, we have to discuss how to adapt it to our setting.

2 Preliminaries

For a natural number d, we denote by $[d]$ the set of natural numbers from 1 through d.

An embedding of G in surface Σ determines pairs v, v' of *consecutive* neighbors of a vertex u. In this case we say the edges uv and uv' form an *angle*. More generally, we say that two vertices v_0 and v_ℓ adjacent to u are at *cyclic distance* ℓ around u, if there exists a sequence of vertices $Q = (v_0, v_1, \ldots, v_\ell)$ of neighbors of v, so that for every $i \in \{0, \ldots, \ell - 1\}$ the vertices v_i and v_{i+1} are consecutive around u, and Q is the shortest possible such sequence.

Let G be a graph embedded in a surface Σ of Euler genus g. Let $F(G)$ be the set of faces defined by the embedding. Euler's formula for Σ tells that $|V(G)| - |E(G)| + |F(G)| = 2 - g$.

Homology and annotations. Let G be a graph embedded in Σ. A subgraph H of G is *even* (or *Eulerian*) if each vertex has even degree in H. An Eulerian subgraph is the union of edge-disjoint cycles. Let $\Sigma \setminus\setminus H$ be the surface (with boundary) obtained from Σ by cutting along the edges of H. We say that H is separating if and only if $\Sigma \setminus\setminus H$ consists of at least two surfaces (with boundary). Moreover, H is minimally separating when no Eulerian subgraph of H is separating.

We will make a black-box use of homology over $\mathbb{Z}_2$; we explain now the properties that will be used. An *annotation* is a function $a\colon E(G) \to (\mathbb{Z}_2)^g$ with the following properties:

- the annotation of a subgraph H, denoted by $a(H)$, is defined by $\sum_{e \in E(H)} a(e)$, where the sum is done componentwise in $\mathbb{Z}_2$;
- for each even subgraph H that is minimally separating, we have $a(H) = 0$; in particular, if H is the boundary of a face, then $a(H) = 0$;
- for each even subgraph H, if $a(H) = 0$, then H is separating.

The annotations as a standalone object were used in [7] and they are a natural object to work with homology. Observe that the (first) $\mathbb{Z}_2$-homology group of Σ is a $\mathbb{Z}_2$-vector space of dimension g, and $a(e)$ is a binary vector in a fixed, ordered basis of the $\mathbb{Z}_2$-homology space.

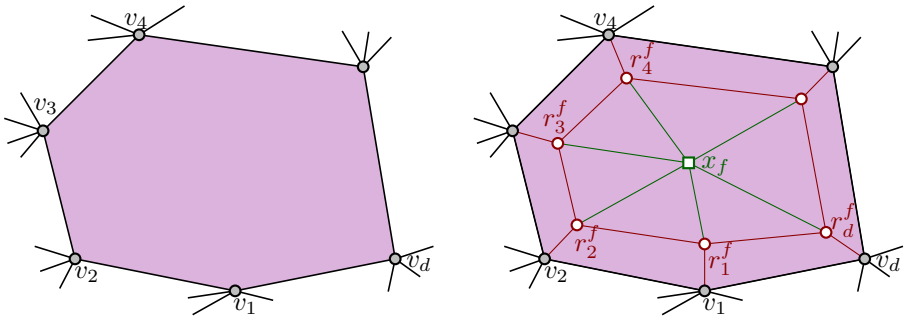

Figure 3 Construction of the polyhedral extension of an embedded graph – the point of view of a face, colored dark magenta. Note that some of the vertices in the facial walk may be the same.

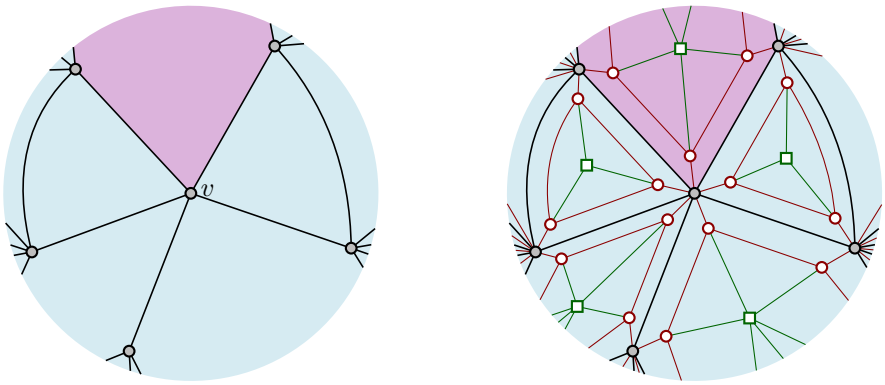

Figure 4 Construction of the polyhedral extension of an embedded graph – the point of view of a vertex v. Note that all the vertices of G in the figure have to be distinct because the graph has no parallel edges. However, some face nodes may be the same!

Their computation in $O(gn)$ time for graphs with n edges embedded on a surface of genus g has been discussed for example in [17, Section 4], where it did not get any particular name, in [16, Section 3] under the name of homology signatures, or in [10, Section 3.3] under the name of crossing vector. See [9, Section 3] for a more recent exposition.

Polyhedral extension. Let $F(G)$ denote the set of faces of G. Now let us describe a *polyhedral extension* $G^{\boxtimes}$ of G, which is also a graph embedded in Σ. We perform the following operation on every face $f \in F(G)$. Let d be the length of f and $v_1, v_2, \ldots, v_d$ be its facial walk. Note that the vertices $v_1, v_2, \ldots, v_d$ may not necessarily be distinct. We insert in f a d-cycle $R_f = r_1^f, r_2^f, \ldots, r_d^f$, an additional central vertex x_f in the interior of R_f, and for every $i \in [d]$ also add the edges $v_i r_i^f$ and $r_i^f x_f$. See Figures 3 and 4.

Clearly, the resulting graph $G^{\boxtimes}$ is embedded in Σ and has no parallel edges if G has no parallel edges. We will call vertices of G *vertex nodes* of $G^{\boxtimes}$; central vertices x_f will be called *face nodes*, and the vertices of cycles added to faces will be called *ring nodes*. We observe that each ring node has degree 4, and every face of $G^{\boxtimes}$ is either a triangle (if it contains a face node) or a quadrangle (if it contains an edge of G). Note that $G \subset G^{\boxtimes}$. We can directly compute an annotation of $G^{\boxtimes}$, and then its restriction to $E(G)$ has to be an annotation of G. We summarize:

▶ **Lemma 3.** *For a given graph G embedded in Σ, we can compute in linear time the polyhedral extension $G^{\boxtimes}$ and an annotation of $G^{\boxtimes}$.*

3 Finding labeled subgraphs

Let G be a graph and let $\mathcal{L}$ be a finite set (of labels). A *labeling* of G is a mapping $\ell_G : X_G \to \mathcal{L}$, where $X_G \subseteq V(G) \cup E(G)$. Notice that we allow both vertex and edge labels and that a labeling can be partial, i.e. not having all vertices and edges in its domain. Let G be a target graph, H a pattern graph, and let ℓ_G and ℓ_H be their respective labelings. A *labeled subgraph isomorphism* of (H, ℓ_H) in (G, ℓ_G) is a map $\phi\colon V(H) \to V(G)$ such that:

- ϕ is injective;
- for all $u, v \in V(H)$ if $uv \in E(H)$ then also $\phi(u)\phi(v) \in E(G)$;
- for all $u \in V(H) \cap X_H$ we have $\phi(u) \in X_G$ and $\ell_H(u) = \ell_G(\phi(u))$; and
- for all $uv \in E(H) \cap X_H$ we have $\phi(u)\phi(v) \in X_G$ and $\ell_H(uv) = \ell_G(\phi(u)\phi(v))$.

If a graph G is unlabeled, we can treat it as a labeled graph with empty labeling $\ell_\emptyset$. Note that assigning label 0 to a vertex/edge x is meaningful and not equivalent to an empty label.

▶ **Lemma 4.** *Let (H, ℓ_H) be a labeled pattern graph, let (G, ℓ_G) be a labeled target graph, and assume that the labels of H and G are in $[L]$. Assume that G is embedded in the surface Σ. In $O(f(|H|, g) \cdot |G| \log L)$ time we can find a labeled subgraph isomorphism of (H, ℓ_H) in (G, ℓ_G), or report that no such isomorphism exist.*

Proof. We use (a slightly modified version of) Bonsma [6, Thm. 20]. The original version of that theorem finds not only one, but all isomorphic subgraphs and the time needed reflects this. As the number of isomorphic subgraphs may be $\Omega(|V|^{|H|})$ and we are only interested in a single isomorphic subgraph, we modify the algorithm to never count higher than one. Furthermore, there are no labels in the original theorem, but adherence to labels can be enforced quite easily. These modifications will be done in Lemma 5. ◀

▶ **Lemma 5** (Bonsma's Algorithm (Thm. 20 of [5]) but finding only one labeled subgraph isomorphism)**.** *Let g be a constant. Let (G, ℓ_G) be a simple labeled graph on n vertices of Euler genus at most g, and let (P, ℓ_P) be a labeled graph on k vertices that is not necessarily connected. In time $2^{O(k)} \cdot O(n)$, we can find a labeled subgraph isomorphism of (P, ℓ_P) in (G, ℓ_G) or determine that no such isomorphism exists.*

Proof. Similar to Bonsma's approach described in [5], we first decide whether such a subgraph exists and then we generate it. Let P consist of c components $P^1, \ldots, P^c$. For $S \subseteq [c]$, the subgraph of P induced by the components with labels in S is denoted by P^S.

We start with a subroutine: For the next paragraph we fix a subgraph $G' \subset G$ with eccentricity $O(k)$, i.e., the maximum distance between any two nodes in G' is in $O(k)$, and some $S \subseteq [c]$ and look for subgraph isomorphisms of P^S in G'. This is done by dynamic programming over a *rooted branch decomposition*. A rooted branch decomposition (T, μ) of G' consists of a rooted binary tree T and a bijection μ between the edges of G' and the leaves of T. For a tree edge $e \in E(T)$, let G_e be the subgraph of G induced by the subtree of T rooted at e. The *middle set* $mid(e)$ of a tree edge $e \in E(T)$ consists of the vertices that the subgraphs G_f and $G_{f'}$ have in common, where $f, f' \in E(T)$ are the root edges of the children of e. The *width of* T is the maximum over the cardinality of its edges' middle sets. Bonsma [5, Sec. 5] introduces a kind of branch decompositions, called *surface split decompositions*, with width $O(g \cdot k)$ and linear construction time for G'. In the dynamic programming approach [5, Sec. 3] there is a table $\mathcal{T}_e$ for every edge $e \in E(T)$, which stores information about possible subgraph isomorphisms of subgraphs of P^S in G_e. Such a table $\mathcal{T}_e$ is built by combining every pair of compatible entries from the tables of the two children of e and then eliminating equivalent entries. The size of such a table is exponential in the width of T and the size of

P^S [5, Sec. 4]. In Bonsma, the entries only store how many (potentially partial) subgraph isomorphisms there are and how the vertices that are relevant for calculating the table of the parent tree edge are involved, but not the actual isomorphisms. It is done this way because the number of partial subgraph isomorphisms at this stage is potentially exponential both in the overall number of subgraph isomorphisms and the size of G'. Because we are interested in finding a single subgraph isomorphism and each entry already has size $\Omega(g \cdot k)$, we can store an arbitrary isomorphism without any extra time or space. Similarly, adherence to the labels can be easily enforced at this point without any extra time. Bonsma furthermore employed colors, but they are only needed to correctly count, so we omit them here. The dynamic programming takes $8^w \cdot 2^{|V(P^S)|} \cdot O(n') = 2^{O(k)} \cdot O(n')$ time in total to find a subgraph isomorphism of P^S in G' or determine that no such isomorphism exists, where $n' = |V(G')|$ and w is the width of the branch decomposition that is being used [5, Thm. 10].

The subroutine is then used in the main algorithm ([5, Sec. 6] extending [32, 13]), which uses dynamic programming again: We choose an arbitrary vertex $r \in V(G)$ and construct a BFS tree T_{BFS} that is rooted at r. Let G_i^j denote the subgraph of G induced by all vertices that have distance between i and j from r, i.e. the vertices with depth between i and j in T_{BFS}. Let $\text{DPC}_i^j(S)$ denote whether there is a subgraph isomorphism of P^S in $G' = G_i^j$. Notice that the subgraph G_i^{i+k} includes every vertex that has a distance of at most k from any vertex of height i and whose height is at least i. Thus, it suffices to compute $\text{DPC}_i^j(S)$ for every $0 \leq i \leq j \leq |T_{\text{BFS}}|$ with $j - i < k$ using the subroutine described above. There are 2^c choices for S and every vertex of G appears in G_i^j for at most k^2 choices of i, j, so the total running time of this stage is $2^c \cdot k^2 \cdot 2^{O(k)} \cdot O(n) = 2^{O(k)} \cdot O(n)$.

For every $j \in [|T_{\text{BFS}}|]$ and $S \subseteq [c]$ let $\text{DPT}^j(S)$ denote a subgraph isomorphism of P^S in G_0^j if it exists and $\emptyset$ otherwise. We calculate $\text{DPT}^j(S)$ as $\text{DPT}^{j-1}(S)$ or $\text{DPT}^{j-x-1}(S_1) \cup \text{DPC}_{j-x+1}^j(S_2)$ for some $x \in [k]$ and some partition of S into S_1, S_2 where $S_2 \neq \emptyset$ if any of these sets is nonempty and $\emptyset$ otherwise. Then $\text{DPT}^{|T_{\text{BFS}}|}([c])$ holds a subgraph isomorphism of P in G if it exists. This can be done in $2^{O(k)} \cdot O(n)$ time as well, as we need to consider $|T_{\text{BFS}}| \in O(n)$ many values of j and the number of partitions of $[c]$ into S_1, S_2, and $[c] \setminus (S_1 \cup S_2)$, and the number of choices of x is bounded by $2^{O(k)}$.

In [5], $\text{DPC}_i^j(S)$ and $\text{DPT}^j(S)$ are numbers of subgraph isomorphisms and in order to generate the subgraph isomorphisms at this point the subroutines that contributed to the overall number of isomorphisms would be rerun, but this time actually generating the subgraph isomorphisms and not just counting them. ◀

Union of consistent labeled graphs. Let G_1 and G_2 be simple graphs and let $\iota \colon U_{G_1} \to U_{G_2}$ be a bijection between vertex sets $U_{G_1} \subseteq V(G_1)$ and $U_{G_2} \subseteq V(G_2)$. Let $G_1 \cup_\iota G_2$, called the *ι-union of G_1 and G_2*, be the simple graph obtained from the disjoint union of graphs G_1 and G_2 by identifying all pairs of vertices $(x, \iota(x))$, where $x \in U_{G_1}$. As graphs G_1, G_2, and $G_1 \cup_\iota G_2$ are all simple, we can say that ι also identifies pairs of edges $(xy, \iota(x)\iota(y))$, provided $x, y \in U(G_1)$, $xy \in E(G_1)$, and $\iota(x)\iota(y) \in E(G_2)$.

Note that we can still identify pairs of vertices which are adjacent in G_1, yet their images are not adjacent in G_2, or vice versa.

We can extend the notion of the ι-union to labeled graphs. Assume that (G_1, ℓ_1) and (G_2, ℓ_2) are simple labeled graphs such that X_1 and X_2 are the supports of ℓ_1 and ℓ_2, respectively. We say that $\iota : U_{G_1} \to U_{G_2}$ is *consistent* with labelings ℓ_1 and ℓ_2 if

- for every $u \in V(G_1) \cap X_1$ we have $\iota(u) \notin X_2$ or $\ell_1(u) = \ell_2(\iota(u))$, and
- for every $u, v \in V(G_1)$ where $uv \in E(G_1)$ and $\iota(u)\iota(v) \in E(G_2)$, we have $uv \notin X_1$ or $\iota(u)\iota(v) \notin X_2$ or $\ell_1(uv) = \ell_2(\iota(u)\iota(v))$.

If ι is consistent, then $(G_1 \cup_\iota G_2, \ell_1 \cup_\iota \ell_2)$ is a labeled graph where the labels of its vertices/edges are induced by the labels in ℓ_1 and ℓ_2. Note that a vertex in $(G_1 \cup_\iota G_2, \ell_1 \cup_\iota \ell_2)$ is unlabeled if it is obtained by identification of two unlabeled vertices from (G_1, ℓ_1) and (G_2, ℓ_2), respectively. The same is true for edges of $(G_1 \cup_\iota G_2, \ell_1 \cup_\iota \ell_2)$.

4 Structure around vertex-separators

The purpose of this section is to show the following.

▶ **Theorem 6.** *Let Σ be a fixed surface, orientable or not, and let $k \in \{3, 4, 5\}$. For any given k-connected graph G embedded in Σ, we can compute in linear time a labeling ℓ of its polyhedral extension $G^{\boxtimes}$ and a sequence of labeled graphs (patterns) $(H_1, \ell_1), \ldots, (H_r, \ell_r)$, where $r = O(1)$, so that G is not $(k+1)$-connected (i.e. G has a k-separator) if and only if there exists a labeled subgraph isomorphism of (H_i, ℓ_i) in $(G^{\boxtimes}, \ell)$ for some $i \in [r]$.*

First, we use Lemma 3 to compute in linear time the polyhedral extension $G^{\boxtimes}$ and a (homology) annotation $a\colon E(G^{\boxtimes}) \to (\mathbb{Z}_2)^{2g}$. We then define the labeling ℓ of $G^{\boxtimes}$ by setting $\ell(e) = a(e)$ for every $e \in E(G^{\boxtimes})$, while for $v \in V(G^{\boxtimes})$, we set $\ell(v) = 1$ for every vertex node, $\ell(v) = 2$ for every ring node, and $\ell(v) = 3$ for every face node. This finishes the description of the labeling ℓ of $G^{\boxtimes}$ for Theorem 6. The proof that there are the patterns claimed in Theorem 6 spans the rest of this section.

Separator graph. Given a (not necessarily proper) q-coloring of edges $c : E(G) \to \{1, \ldots, q\}$, we say that the edges e, e' form a *mixed angle* if they form an angle and $c(e) \neq c(e')$.

Take a k-connected graph G embedded in Σ. Let us assume that G is not $k+1$-connected and not a complete graph. Then G has a minimal k-separator $S \subseteq V(G)$. Let $K_1, K_2, \ldots, K_q$ be the components of $G - S$. Let $c : E(G) \to \{1, \ldots, q\}$ be a coloring of edges of G with q colors where $c(e) = i$ if e has an endvertex in K_i, and for edges with both endvertices in S we choose their colors so that the number of mixed angles is cumulatively as small as possible.

The coloring c induces a q-coloring of faces of the polyhedral extension $c : F(G^{\boxtimes}) \to \{1, \ldots, q\}$ using the following principle: if $f^{\square}$ is a quadrangular face of $G^{\boxtimes}$, and e is the only edge in $f^{\square} \cap E(G)$ then we set $c(f^{\square}) = c(e)$, and if $f^{\triangle}$ is a triangular face adjacent to a quadrangular face $f^{\square}$ we set $c(f^{\triangle}) = c(f^{\square})$.

As S is a (minimal) k-separator, there exists a pair of vertices v_1, v_2 which lie in different components of $G - S$. Without loss of generality we may assume that $v_1 \in V(K_1)$ and $v_2 \in V(K_2)$. We will now keep relabeling the faces of $G^{\boxtimes}$ according to the following rule:

> if there exists adjacent faces f, f' with $c(f) = i$ for $i \in 1, 2$ and $c(f') = j \in \{3, \ldots, q\}$, then adjust c by globally replacing label j with label i.

The *separator graph* $S^{\boxtimes}$ is a subgraph of $G^{\boxtimes}$ spanned by the edges of $E(G^{\boxtimes})$ which lie on the boundary of the region whose faces have color 1. We also define the corresponding partition of $G - S$ into two parts G_1 and G_2, where G_1 is the subgraph of $G - S$ induced by the vertices lying on a face of label 1, and G_2 is the subgraph $G - S - V(G_1)$.

▶ **Proposition 7.** *The separator graph $S^{\boxtimes}$ satisfies the following properties:*

(S1) *Every edge of $S^{\boxtimes}$ is incident with a ring node,*

(S2) *every ring node $r \in V(S^{\boxtimes})$ has degree 2 (in $S^{\boxtimes}$),*

(S3) *$S^{\boxtimes}$ is Eulerian (i.e. its vertices have even degree) and is therefore obtained as a union of edge-disjoint cycles.*

Proof. By construction if adjacent faces $f, f' \in F(G^{\boxtimes})$ satisfy $c(f) = 1$ and $c(f') \neq 1$, then they have the same length and their common edge is incident with a ring node, hence (S1). This also implies that at most two edges incident with a ring node belong to $S^{\boxtimes}$ which is what (S2) states.

For every node of $G^{\boxtimes}$ we can observe the number of transitions between a face of color 1 and a face whose color is not 1 along its local rotation in the embedding, which can only be an even number. This establishes (S3). ◀

Now $(G^{\boxtimes}, \ell)$ is a labeled graph, and restricting ℓ to structures of $S^{\boxtimes}$ yields a *labeled separator graph* $(S^{\boxtimes}, \ell)$.

▶ **Proposition 8.** *The labeled separator graph $(S^{\boxtimes}, \ell)$ also satisfies*
(S4) *$(S^{\boxtimes}, \ell)$ is* nullhomologous*, i.e. $\sum_{e \in E(S^{\boxtimes})} \ell(e) = 0$.*

Proof. It suffices to observe that every face of $G^{\boxtimes}$ is nullhomologous and that $\sum_{e \in E(S^{\boxtimes})} \ell(e)$ equals the sum of sums of labels over all faces of color 1. ◀

Reversing the direction

In what follows, we shall reverse the direction. Assuming we are given a k-connected graph G embedded in surface Σ, we would like to test whether G contains a k-separator.

We shall first establish that, assuming G contains a separator S of order k, there exists a pair of vertices in different parts of $G - S$ which are fairly close in the embedding.

Next we will build our patterns (H_i, ℓ_i) as unions of a separator graph (with prescribed labels) and a planar *patch* whose edges are not labeled but the planar structure is encoded by the labeling of its vertices. We shall also argue that given an embedded graph G and its induced polyhedral extension $G^{\boxtimes}$ equipped with a labeling ℓ, if there exist a labeled subgraph isomorphism ϕ of a pattern (H_i, ℓ_i) in $(G^{\boxtimes}, \ell)$, then G contains a separator S of order k.

Finally, we will show that there is a bound on the number of patterns (in terms of the target connectivity k and genus g) which will conclude the proof.

Separating a pair of vertices. Let $s \in S$. We will argue that s has neighbors from two different parts so that their cyclic distance around s is small.

▶ **Lemma 9.** *Let s be a vertex in a minimal k-separator $S \subseteq V(G)$. Then there exists a pair of vertices v_1, v_2, where $v_1 \in V(G_1)$ and $v_2 \in V(G_2)$, both adjacent to s, so that the cyclic distance between v_1 and v_2 around s is at most $\lceil k/2 \rceil \leq 3$.*

Proof. Fix a cyclic ordering of neighbors of s. As s has a neighbor in both G_1 and G_2, there exist at least two ordered pairs of vertices (x, y) neighboring s, so that all intermediate vertices in the ordering of neighbors of s from x to y belong to the separator S.

Assuming the lemma is false, for every such pair we have at least $\lceil k/2 \rceil$ intermediate vertices from S. This is absurd, as $S \setminus \{s\}$ would then contain at least k vertices. ◀

▶ **Lemma 10.** *Given a surface Σ and target connectivity $k \in \{3, 4, 5\}$ there exists a (computable) function $f' = f'(k, g)$ so that the number of possible edge-minimal labeled separator graphs in Σ with at most k vertex nodes is at most $f'(k, g)$.*

Proof. In order to establish Lemma 10 we will first argue that the order (i.e. the number of vertices and edges) of a minimal separator graph $S^{\boxtimes}$ is bounded. If $S^{\boxtimes}$ is an arbitrary separator graph, then by (S3) $S^{\boxtimes}$ is an edge-disjoint union of simple cycles, $S^{\boxtimes} = C_1 \cup C_2 \cup$

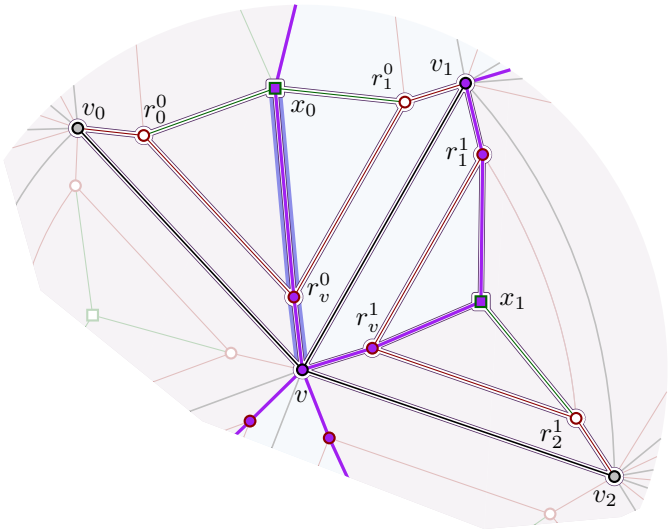

Figure 5 A 2-patch in $G^{\boxtimes}$ (outlined); $S^{\boxtimes}$ is violet, and a root is highlighted blue.

$C_3 \cup \ldots \cup C_j$. By (S1) and (S2) in each cycle C_i every other vertex is a ring node, and the remaining vertices alternate between vertex nodes and face nodes. As every vertex node of C_i belongs to S and $|S| \leq 5$, we have that the length of every cycle C_i is at most 20.

Pick an arbitrary vertex node $s \in V(S^{\boxtimes})$. By Lemma 9 there exist a pair of vertex nodes v_1, v_2, where $v_1 \in V(G_1)$ and $v_2 \in V(G_2)$ which are at cyclic distance at most 3 around s. This short angle $\angle v_1 s v_2$ between edges sv_1 and sv_2 contains $d \leq 3$ ring nodes $r_1^f, \ldots, r_d^f$ of $G^{\boxtimes}$ so that also the edges $sr_1^f, \ldots, sr_d^f$ emanate from s into said angle. We may without loss of generality assume that $V(S^{\boxtimes}) \cap \{r_1^f, \ldots, r_d^f\}$ is contained in $C_1 \cup C_2 \cup C_3$.

Now if the number of cycles j is at least $g + 4$ (where g is the dimension of the first homology group of Σ considered as a binary vector space), then there exists a strict subset of indices $I \subsetneq \{4, \ldots, j\}$ so that $\bigcup_{i \in I \cup \{1,2,3\}} C_i$ is nullhomologous in $(S^{\boxtimes}, \ell)$ and it still separates v_1 and v_2. This contradicts the edge-minimality of $S^{\boxtimes}$. We therefore have $j \leq g+3$. This yields a bound on the size of $S^{\boxtimes}$.

Finally, for $(S^{\boxtimes}, \ell)$ is a labelled graph, there is a bounded number of possible labels and a bounded number of labeled edges. This concludes the proof. ◀

Patches. Let $q \in \{1, 2, 3\}$. A *q-patch* $P = (V_P, E_P)$ is a graph defined as follows:

$$V_P := \{v, v_0, v_1, \ldots, v_q\} \cup \{x_0, \ldots, x_{q-1}\} \cup \bigcup_{j \in [q-1]} \{r_j^j,\, r_v^j,\, r_{j+1}^j\},$$

$$E_P := \{vv_q\} \cup \bigcup_{j \in [q-1]} \{x_j r_j^j,\ x_j r_v^j,\ x_j r_{j+1}^j,\ r_v^j v,\ v v_j,\ v_j r_j^j,\ r_j^j r_v^j,\ r_v^j r_{j+1}^j,\ r_{j+1}^j v_{j+1}\}$$

The vertices v_0 and v_q are called *peripheral vertices* of a q-patch and v is called its *center*.

Next we describe a vertex-labeling ℓ_P of a q-patch. The vertex v and vertices v_j are called the vertex nodes of a patch and are labeled with 1, vertices r_v^j, r_j^j, and r_{j+1}^j are called the ring nodes of a patch and are labeled with 2, and vertices x_j are called the face nodes of a patch and are labeled with 3. See Figure 5 for an example of a 2-patch.

If (P, ℓ_P) is a labeled q-patch, then its root R_P is an arbitrary directed path of length 2 starting at v and ending in a face node. There are q choices for a root in a q-patch.

Identifications and patterns. Let $(S^{\boxtimes}, \ell_S)$ be a labeled separator graph. Pick an arbitrary vertex node $s \in V(S^{\boxtimes})$ and let d be its degree. The root $R_{S^{\boxtimes}}$ of $S^{\boxtimes}$ (with the choice of s being implicit) is an arbitrary path of length 2 starting at s and ending in a face node of G. Observe that if $\deg(s) = d$ a separator graph $S^{\boxtimes}$ has d distinct choices for its root.

Let (P, ℓ_P) be a labeled patch with root R_P and $(S^{\boxtimes}, \ell_S)$ be a labeled separator graph with root R_S. Let ι_0 be the label preserving bijection from $V(R_S)$ to $V(R_P)$. As no edge in (P, ℓ_P) is labeled, the mapping ι_0 is consistent with both labelings and the labeled graph $(S^{\boxtimes} \cup_{\iota_0} P, \ell_S \cup_{\iota_0} \ell_P)$ is called the *free identification* of $(S^{\boxtimes}, \ell_S)$ and (P, ℓ_P).

We have obtained a free identification of a separator graph and a patch by pasting a root of a patch on a root of the separator graph, in the process we have identified exactly three pairs of vertices and two pairs of edges. But we have no control whether some other vertex from the separator graph and a vertex from a patch also coincide – we have to consider all possibilities – yet we can assume that peripheral vertex nodes of a patch are not identified with vertex nodes of a separator graph.

Let ι be the label preserving injection from a subset of vertices of $S^{\boxtimes}$ to $V(P)$ which is an extension of ι_0 (which is a mapping from R_S to R_P). A *pattern* (H, ℓ) is the ι-union $(S^{\boxtimes} \cup_\iota P, \ell_S \cup_\iota \ell_P)$ if

(PT1) for every ring node $r \in V(S^{\boxtimes})$ if r is in the domain of ι, then so are both its neighbors v_r and x_f[1], and $\iota(v_r)$ and $\iota(x_f)$ are neighbors of $\iota(r)$ in P,

(PT2) none of the peripheral vertex nodes of P are in the image of ι and all the remaining vertex nodes of P are in the image of ι, and

(PT3) an odd number (exactly one or all three) of ring nodes adjacent with the center v of P are in the image of ι.

(PT1) implies that if a ring node r of the separator graph $S^{\boxtimes}$ is identified with a ring node $\iota(r)$ of the patch P, then also the adjacent non-ring vertices and the edges from r to them are identified with the corresponding structures in the patch. Observe that these identifications are uniquely determined, as $\iota(r)$, a ring node in the patch, has a unique vertex node neighbor and a unique face node neighbor.

(PT2) implies that we use the smallest possible patch. If, apart from the peripheral vertex nodes v_0 and v_q, an additional vertex node v_x of P is not identified with a vertex node of a separator graph, then v_x could replace either v_0 or v_q using a strictly smaller patch.

Finally (PT3) is to ensure that peripheral vertex nodes of the patch will be separated by removing the vertex nodes of the separator graph.

Fix $k \in \{3, 4, 5\}$. Given a graph G embedded in Σ, let $G^{\boxtimes}$ be its polyhedral extension and $(G^{\boxtimes}, \ell)$ a labelled graph whose labeling is induced by a homology annotation of G.

By Lemma 10 there exist at most $f(g, k)$ different labeled separator graphs, each having exactly k-vertex nodes, and by construction there is a bounded number of q-patches as $q \in \{1, 2, 3\}$. Choosing a root in both the separator graph and the patch can be done in a bounded number of ways as their respective orders are of bounded size. Similarly, there exists a bounded number of extensions of identifications of vertices of a separator graph and a patch which satisfy (PT1), (PT2), and (PT3), therefore the number of patterns is bounded.

Choose a pattern $(S^{\boxtimes} \cup_\iota P, \ell_S \cup_\iota \ell_P)$ and inductively assume that G has no $k-1$ separator. If there exists a labeled subgraph isomorphism $\phi : (S^{\boxtimes} \cup_\iota P, \ell_S \cup_\iota \ell_P) \to (G^{\boxtimes}, \ell)$ we claim that the ϕ-images of vertex nodes of $S^{\boxtimes}$ separate the ϕ-images of peripheral vertex nodes v_0 and v_q of P. As $\phi(S^{\boxtimes})$ is nullhomologous, it separates the surface Σ into several components.

[1] $r = r^j_{j'}$ with $j \in [q-1]$, $j' \in \{j, v, j+1\}$ and $x^r = x^j$ and $v_r = v_{j'}$ if $j' \neq v$ and otherwise $v_r = v$

But a curve starting at $\phi(v_0)$ traversing the ϕ-images of 4-faces of P and ending in $\phi(v_q)$ crosses $\phi(S^{\boxtimes})$ an odd number of times by (P3), implying that $\phi(v_0)$ and $\phi(v_q)$ are separated by $V(G) \cap \phi(S^{\boxtimes})$ which is of order k.

To summarize, we prepare

1. all possible labelled separator graphs (there is bound on their number by Lemma 10),
2. all possible q-patches for $q = 1, 2, 3$ (technically we need 3-patches only for testing 5-connectivity, see Lemma 9),
3. choosing a root in a labeled separator graph and a root in a patch to produce a labeled free identification, and
4. from it construct a pattern with additional identifications of vertices from a separator graph and a patch (and as these structures are bounded in size there is a bounded number of possible patterns that stem from a single free identification).

Now if there exists a labeled subgraph isomorphism ϕ of (H_i, ℓ_i) in $(G^{\boxtimes}, \ell)$, then the ϕ-image of the set of non-peripheral vertex nodes from H_i is a separator in G, separating the ϕ-images of peripheral vertex nodes. And vice versa, if G contains a minimal k-separator S, then a choice of two vertices by Lemma 9 together with S forms an appropriate pattern.

This concludes the proof of Theorem 6.

5 Putting it all together

First, we show that it suffices to look at small connectivity, and then prove our main result.

▶ **Lemma 11.** *Let G be a graph embedded in a fixed surface Σ. If G is 6-connected, then we can compute its vertex-connectivity and report a smallest vertex-separator in linear time.*

Proof. By Euler's formula we have $|E(G)| \leq 3|V(G)| - 6 + 3g$, which in turn implies that the average degree of G is $\frac{2|E(G)|}{|V(G)|} \leq 6 + \frac{6g-12}{|V(G)|}$. If G has at most $6g - 12$ vertices, then it has constant size and we compute the vertex-connectivity in constant time. If G has more than $6g - 12$ vertices, then its average degree is bounded by $6 + \frac{6g-12}{|V(G)|} < 7$, which implies that it has a vertex v of degree at most 6. The neighbours of v form a vertex-separator of size at most 6 and, since G is 6-connected, this vertex-separator has minimum size. ◀

Proof of Theorem 1. Let G be the given graph embedded in the fixed surface Σ. Standard algorithms can be used to check whether G has vertex-connectivity 1 or 2. Then we check for $k = 3, 4, 5$ whether G has a k-separator. We finish as soon as we find a k-separator. If we do not find any k separator for $k \leq 5$, then G is 6-connected and we invoke Lemma 11. To check whether G has a vertex-separator of size k, for a given $k \in \{3, 4, 5\}$, we use Theorem 6 to generate labels and patterns, and use Lemma 4 to test whether any of the labeled patterns has a labeled subgraph isomorphism in $(G^{\boxtimes}, \ell)$. The result follows. ◀

Edge-connectivity

The case of edge-connectivity is easier.

Proof of Theorem 2. For simple graphs, Lemma 11 also applies to edge-connectivity. Thus, we only need to solve the problem of checking whether G has an edge-cut of size k for a given $k \in \{1, \ldots, 5\}$. We compute the dual graph G^*, with its natural embedding in Σ, and a (homology) annotation a of G^*. This takes linear time, as mentioned in Section 2.

As mentioned before, and used for example in [9], the graph G has an edge-cut of size k if and only if the dual graph G^* has an even subgraph with k edges and annotation 0. We generate all pairs (H_i, ℓ_i) where H_i is a (multi)graph with k edges and ℓ_i is a mapping $\ell_i \colon E(H_i) \to (\mathbb{Z}_2)^g$ that add up to 0. Then, G has an edge-cut of size k if and only if there

is a labeled subgraph isomorphism from at least one of the pairs (H_i, ℓ_i) in (G^*, a). Since the number of possible pairs (H_i, ℓ_i) to consider is $O(1) \cdot O(2^{9^k}) = O(1)$, there are at most $2^9 = O(1)$ many different labels, and testing each labeled pattern takes $O(n)$ time because of Lemma 4, the statement of Theorem 2 follows. ◀

References

1 András A. Benczúr and David R. Karger. Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. In *Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing, Philadelphia, Pennsylvania, USA, May 22-24, 1996*, STOC '96, pages 47–55, New York, NY, USA, 1996. ACM. doi:10.1145/237814.237827.

2 Therese Biedl, Prosenjit Bose, and Karthik Murali. A parameterized algorithm for vertex and edge connectivity of embedded graphs. In *32nd Annual European Symposium on Algorithms, ESA 2024*, volume 308 of *LIPIcs*, pages 24:1–24:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.ESA.2024.24.

3 Therese Biedl, Prosenjit Bose, and Karthik Murali. A parameterized algorithm for vertex and edge connectivity of embedded graphs. *CoRR*, abs/2407.00586, 2024. doi:10.48550/arXiv.2407.00586.

4 Therese Biedl and Karthik Murali. On computing vertex connectivity of 1-planar graphs. *Algorithmica*, 88(1):4, 2026. doi:10.1007/S00453-025-01355-3.

5 Paul Bonsma. Surface split decompositions and subgraph isomorphism in graphs on surfaces. doi:10.48550/arXiv.1109.4554.

6 Paul S. Bonsma. Surface split decompositions and subgraph isomorphism in graphs on surfaces. In *29th International Symposium on Theoretical Aspects of Computer Science, STACS 2012*, volume 14 of *LIPIcs*, pages 531–542. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012. Full version available at http://arxiv.org/abs/1109.4554. doi:10.4230/LIPIcs.STACS.2012.531.

7 Oleksiy Busaryev, Sergio Cabello, Chao Chen, Tamal K. Dey, and Yusu Wang. Annotating simplices with a homology basis and its applications. In *13th Scandinavian Symposium and Workshops on Algorithm Theory, SWAT 2012*, volume 7357 of *Lecture Notes in Computer Science*, pages 189–200. Springer, 2012. Full version available at https://arxiv.org/abs/1107.3793. doi:10.1007/978-3-642-31155-0_17.

8 Tanmoy Chakraborty, Julia Chuzhoy, and Sanjeev Khanna. Network design for vertex connectivity. In Cynthia Dwork, editor, *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, STOC '08, pages 167–176, New York, NY, USA, 2008. ACM. doi:10.1145/1374376.1374403.

9 Erin W. Chambers, Jeff Erickson, Kyle Fox, and Amir Nayyeri. Minimum cuts in surface graphs. *SIAM J. Comput.*, 52(1):156–195, 2023. doi:10.1137/19M1291820.

10 Erin W. Chambers, Jeff Erickson, and Amir Nayyeri. Minimum cuts and shortest homologous cycles. In *Proceedings of the 25th ACM Symposium on Computational Geometry, SoCG 2009*, pages 377–385. ACM, 2009. doi:10.1145/1542362.1542426.

11 Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. *J. ACM*, 72(3):19:1–19:103, 2025. doi:10.1145/3728631.

12 Éric Colin de Verdière. Computational topology of graphs on surfaces. In Jacob E. Goodman, Joseph O'Rourke, and Csaba D. Tóth, editors, *Handbook of Discrete and Computational Geometry*, chapter 23. CRC Press LLC, third edition, 2017.

13 Frederic Dorn. Planar subgraph isomorphism revisited. In Jean-Yves Marion and Thomas Schwentick, editors, *27th International Symposium on Theoretical Aspects of Computer Science, STACS 2010*, volume 5 of *LIPIcs*, pages 263–274. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010. doi:10.4230/LIPIcs.STACS.2010.2460.

14 David Eppstein. Subgraph isomorphism in planar graphs and related problems. *J. Graph Algorithms Appl.*, 3(3):1–27, 1999. doi:10.7155/JGAA.00014.

15 Jeff Erickson, Kyle Fox, and Luvsandondov Lkhamsuren. Holiest minimum-cost paths and flows in surface graphs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1319–1332, Los Angeles CA USA, June 2018. ACM. doi:10.1145/3188745.3188904.

16 Jeff Erickson and Amir Nayyeri. Minimum cuts and shortest non-separating cycles via homology covers. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011*, pages 1166–1176. SIAM, 2011. doi:10.1137/1.9781611973082.88.

17 Jeff Erickson and Kim Whittlesey. Greedy optimal homotopy and homology generators. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2005*, pages 1038–1046. SIAM, 2005. doi:10.5555/1070432.1070581.

18 Shimon Even and R. Endre Tarjan. Network flow and testing graph connectivity. *SIAM Journal on Computing*, 4(4):507–518, 1975. doi:10.1137/0204043.

19 Nicola Galesi, Fariba Ranjbar, and Michele Zito. Vertex-connectivity for node failure identification in boolean network tomography. *Inf. Process. Lett.*, 184:106450, 2024. doi:10.1016/J.IPL.2023.106450.

20 Monika Henzinger, Satish Rao, and Di Wang. Local flow partitioning for faster edge connectivity. *SIAM J. Comput.*, 49(1):1–36, 2020. doi:10.1137/18M1180335.

21 Monika R. Henzinger, Satish Rao, and Harold N. Gabow. Computing vertex connectivity: New bounds from old techniques. *Journal of Algorithms*, 34(2):222–250, 2000. doi:10.1006/jagm.1999.1055.

22 David R. Karger. Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, 24(2):383–413, 1999. doi:10.1287/moor.24.2.383.

23 David R. Karger. Minimum cuts in near-linear time. *J. ACM*, 47(1):46–76, 2000. doi:10.1145/331605.331608.

24 David R. Karger and Matthew S. Levine. Finding maximum flows in undirected graphs seems easier than bipartite matching. In *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing, Dallas, Texas, USA, May 23-26, 1998*, STOC '98, pages 69–78, New York, NY, USA, 1998. ACM. doi:10.1145/276698.276714.

25 Ken-ichi Kawarabayashi, Bojan Mohar, and Bruce A. Reed. A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In *Proc. of the 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008*, pages 771–780. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.53.

26 Ken-ichi Kawarabayashi and Mikkel Thorup. Deterministic edge connectivity in near-linear time. *J. ACM*, 66(1):4:1–4:50, 2019. doi:10.1145/3274663.

27 Tuukka Korhonen. Linear-time algorithms for k-edge-connected components, k-lean tree decompositions, and more. *CoRR*, abs/2411.02658, 2024. doi:10.48550/arXiv.2411.02658.

28 Tuukka Korhonen. Linear-time algorithms for k-edge-connected components, k-lean tree decompositions, and more. In Michal Koucký and Nikhil Bansal, editors, *Proceedings of the 57th Annual ACM Symposium on Theory of Computing, STOC 2025, Prague, Czechia, June 23-27, 2025*, pages 111–119. ACM, 2025. doi:10.1145/3717823.3718123.

29 Jason Li, Danupon Nanongkai, Debmalya Panigrahi, Thatchaphol Saranurak, and Sorrachai Yingchareonthawornchai. Vertex connectivity in poly-logarithmic max-flows. *J. ACM*, 72(4):28:1–28:34, 2025. doi:10.1145/3743670.

30 Bojan Mohar. A linear time algorithm for embedding graphs in an arbitrary surface. *SIAM J. Discret. Math.*, 12(1):6–26, 1999. doi:10.1137/S089548019529248X.

31 Bojan Mohar and Carsten Thomassen. *Graphs on Surfaces*. Johns Hopkins University Press, 2001. URL: http://jhupbooks.press.jhu.edu/ecom/MasterServlet/GetItemDetailsHandler?iN=9780801866890&qty=1&source=2&viewMode=3&loggedIN=false&JavaScript=y.

32 Hisao Tamaki. A linear time heuristic for the branch-decomposition of planar graphs. In Giuseppe Di Battista and Uri Zwick, editors, *11th Annual European Symposium, ESA 2003*, volume 2832 of *Lecture Notes in Computer Science*, pages 765–775. Springer, 2003. doi:10.1007/978-3-540-39658-1_68.